Unsupervised Learning : Model-guided and Model-agnostic Approaches

Sudipto Mukherjee

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

Doctor of Philosophy

University of Washington

2020

Reading Committee:

Sreeram Kannan, Chair

Mari Ostendorf

Sewoong Oh

Program Authorized to Offer Degree:
Electrical and Computer Engineering

University of Washington

Abstract

Unsupervised Learning : Model-guided and Model-agnostic Approaches

Sudipto Mukherjee

Chair of the Supervisory Committee:
Sreeram Kannan
Electrical and Computer Engineering

"Most of human and animal learning is unsupervised learning. If intelligence was a cake, unsupervised learning would be the cake, supervised learning would be the icing on the cake, and reinforcement learning would be the cherry on the cake. We know how to make the icing and the cherry, but we don't know how to make the cake. We need to solve the unsupervised learning problem before we can even think of getting to true AI."

- Yann LeCun, Professor, New York University.

Unsupervised learning is the branch of machine learning that is aimed at learning patterns from data without labels. Supervised learning with millions of labels for image classification had driven the modern deep learning revolution in the past few years. Deep neural networks have exceeded human performance at this specific task. But the requirement of such large amounts of labeled data for these models makes one skeptical about the generalization of such intelligence to myriad tasks. While training a neural network to classify images of a cat, one might wonder : *Do humans really need hundreds of images to differentiate a cat from an elephant ? Or is there some underlying principle that can be rendered useful by a machine in its race to match human intelligence ?* Unsupervised learning unveils the potential of machine learning algorithms beyond empirical risk minimization and extend them to learning non-trivial representations of the data. At the core of such learning are two

distinct principles - model-agnostic representation learning and model-guided inference. The goal of this thesis is to extend the present literature on unsupervised learning through design of novel unsupervised algorithms for clustering, information estimation and model-guided inference.

Our journey starts with one of the simplest, yet most fundamental unsupervised learning problem, namely clustering. We explore how modern generative principles such as Generative Adversarial Learning (GAN) can be used to cluster diverse types of data. Even though auto-encoders had been used for clustering in the past, clustering using GANs was unexplored prior to this work. ClusterGAN modifies the vanilla GAN architecture to enable embedding of data in the latent space where cluster structure is revealed. It also improves the generation ability of vanilla GANs by segregating a complex multi-modal distribution into simpler components.

Recently, information-theoretic quantities such as mutual information and cross-entropy have been used to regularize unsupervised representation learning and improve clustering. Estimation of such quantities is another fundamental problem in unsupervised learning, which is related to the broader statistical problem of estimating functionals of probability density. We design an estimator, CCMI, for mutual information estimation using classifier likelihood ratio in an unsupervised manner and demonstrate its suitability for high dimensional real-valued information estimation. The conditional variant of this quantity, conditional mutual information (CMI), is also estimated and applied to conditional independence testing.

The above approaches to unsupervised learning do not assume any model for the data-generation and learn it implicitly from data. However, in many real-world problems, one has domain knowledge about the data-generation process. Utilizing such domain knowledge can help to further reduce data complexity and abandon the need for deep learning models. It also imparts interpretability to the learning process. We apply such learning techniques to a specific phenomenon in genomics, known as segmental duplication. The problem can

be formulated as either a (a) low-rank matrix completion or a (b) robust signed community detection based on suitable assumptions on the data. We design algorithms for resolving segmental duplication in genomes under these two formulations.

Finally, we explore another application in natural language understanding where unsupervised and supervised approaches blend gracefully. This also illustrates a situation where labeled data could be difficult to obtain and an unsupervised solution may be used.

DEDICATION

To Almighty God,
on whose blessing
the sun shines,
the wind blows,
and who sets the stage
for birth and death.

ACKNOWLEDGMENTS

As the common saying goes, "A journey of a thousand miles begins with a single step", the single step for me started when I joined graduate school at the University of Washington. First of all, I would like to thank my advisor, Professor Sreeram Kannan, for giving me the opportunity to work under his guidance. He taught me how to distill a research problem into smaller components and study each small component meticulously before analyzing the complicated system in its entirety. Even though it sometimes took me more time to grasp the fundamentals of a research discussion, he was extremely patient and explained the concept to me multiple times. He always ensured that I received sufficient funding throughout my Ph.D. years, failing which I would have had to take the next flight back home. I hope, over the years, I have picked up some of his strengths, of not giving up easily on tough problems and to persevere even in hard times.

I would like to thank my other committee members, Professor Sewoong Oh, Professor Mari Ostendorf and Professor Archis Ghate for agreeing to be in my dissertation committee. Their feedback for improvement, as well as general discussion on some pitfalls helped me think more deeply about those topics and address any shortcomings. The University of Washington provided the ideal ambiance to learn and grow. I consider myself extremely fortunate to be able to attend the courses of Professor Maryam Fazel, Professor Sham Kakade, Professor Pedro Domingos and Professor Noah Smith. Maryam's course om Convex Optimization provided me the first research idea of using low rank matrix completion for resolving segmental duplication. Her course on Optimization Algorithms further equipped me with the basics of alternating projected gradient descent, which I used in my research. Sham's course strengthened my basics of Machine Learning. Finally, it was Noah Smith who introduced me

to the amazing field of Natural Language Processing (and which since then has fascinated me to the extent that it will possibly be my bread and butter for at least the next 3 years).

I extend my gratitude to my collaborators Dr. Mark Chaisson, Dr. Eugene Lin, Dr. Himanshu Asnani and Dr. Karthikeyan Shanmugam. Mark introduced me to the problem of segmental duplication and I wrote my first Ph.D. research paper with him. Working alongside him helped me improve my coding skills. Thanks to Himanshu and Karthikeyan for helpful discussion on estimators, independence testing and causality. I had the wonderful opportunity of spending two summers at Microsoft Corporation as an intern. I am thankful to my mentors and managers (Sebastian, Ahmed, Subhabrata, Marcello, Ke Jiang) for sharing their expertise with me and empowering me to strive towards excellence. My undergraduate research advisors, Dr. Xiaoyi Jiang, Dr. Ananda Shankar Chowdhury and Dr. Swagatam Das, fostered in me a keen interest in research from an early age.

I convey my regards to the administrative staff at UW ECE, specially Brenda Larson. I would like to thank my lab-mates Shunfu Mao, Yihan Jiang, Arman Rahimzamani and Bowen Xue for constantly being by my side throughout these years. Thanks also to my friends and seniors Nilanjana Laha, Sumit Mukherjee, Sachin Mehta, Hossein Hosseini, Michael Driscoll, Rohan Patidar, Swati Padmanabhan, Navonil de Sarkar, Anindita Chatterjee, Gourab Chatterjee, Jagori Saha, Bindita Chowdhuri and Rahul Mallik. Special thanks to my apartment-mates Dennis Leung and Avijit Hazra who have been supportive and kind. Avijit is a role-model to me as a researcher. He taught me how to define novel research problems, a skill that has helped me grow as a researcher.

Last but not least, I convey my sincere gratitude to my family members and close friends back at home. Thanks to my brothers Nilotpal and Subhodip, my aunt Jayasree and uncles, my grandparents, my friends Nilavra, Jaydeep, Suryadipta and Ishita. Words are not enough to convey my gratitude to my parents Jnan Prakash and Jhuma Mukherjee, whose faith in my abilities and unconditional affection has shaped my life.

TABLE OF CONTENTS

Chapter 1

INTRODUCTION

1.1 Background

In the pursuit of artificial intelligence, machine learning algorithms have sought to achieve (and oftentimes exceed) human-level performance at various tasks. As modern deep learning methods obtained unprecedented success in tasks such as image classification, natural language translation, speech recognition and so on, there was a need to dive deep into these success stories and understand the differences with human learning. One striking difference was the need for millions of labeled (supervised) data points. The initial successes of deep learning hinged upon the availability of large labeled datasets on which deep neural architectures could be trained using back-propagation to minimize the empirical risk. For certain specific tasks such as image classification, labeled data was made available through years of crowd-sourcing. Equipped with the high-bandwidth parallel computing of GPUs and a massive labeled ImageNet dataset, deep learning methods outperformed traditional feature based computer vision algorithms for image classification. Nevertheless, the need for such supervision portrayed a severe limitation of machine learning algorithms as a general recipe for solving data science problems.

Unsupervised learning algorithms, on the contrary, aim to find patterns in the data without using labeled information. At the core of many such approaches is the underlying assumption that there is a low dimensional semantic space $\mathcal{Z}$ in which distinct patterns are observed. The data generation then follows a unique process for different problems (say through a function $\mathcal{G}(\cdot)$) to finally give rise to the space $\mathcal{X}$ of points. This general abstraction $\mathcal{X} = \mathcal{G}(\mathcal{Z})$ is helpful to study these problems and lay out the various assumptions explicitly. As a concrete example, we can consider Principal Component Analysis (PCA) in the light

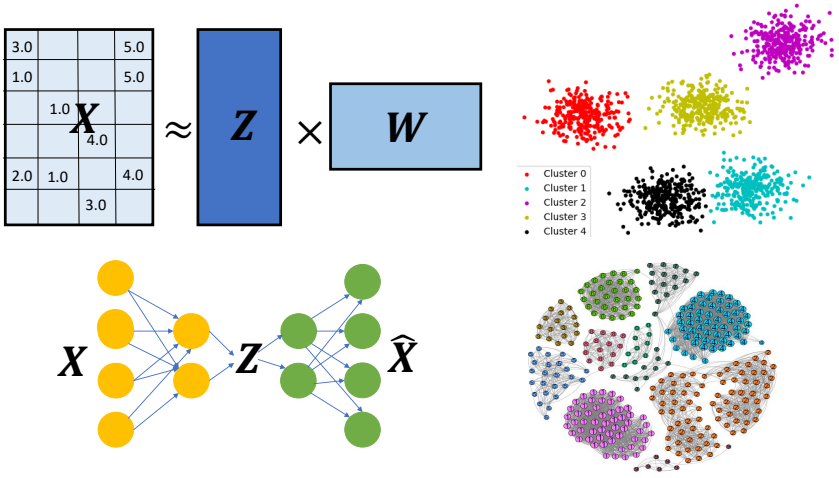


Figure 1.1: Examples of unsupervised learning problems studied in this thesis : Matrix completion, Clustering, Latent factor representation learning, community detection.

of latent factor model. The generative process can be expressed as $X = ZW^T$, where W is the orthogonal projection matrix. The assumption here is that the semantic space is a low-dimensional linear subspace. The concept can be extended to probabilistic PCA, factor analysis as well as deep representation learning methods such as Auto-Encoders and Generative Adversarial Networks (GANs).

Despite having a common abstraction, these models vastly differ in their formulations with regards to:

i. distribution of the latent variable

ii. the type of generative function used (linear or non-linear)

iii. the loss function involved in training the model viz. adversarial loss, l_1 loss, etc.

iv. assumptions on the data; while PCA assumes a linear subspace of latent variables, auto-encoders are general enough to incorporate non-linear embedding.

There may be additional constraints such as sparsity or disentanglement in the latent space of $\mathcal{Z}$. These constraints are aimed at incorporating structure to the problem in order to ease the learning process. An auxiliary benefit is that it often imparts interpretability to the model. We can understand in what way a particular latent component affects the high-dimensional data distribution.

There are two broad categories of solution strategy that can be adopted to solve these unsupervised representation learning problems:

(a) **Model-guided**

(b) **Model-agnostic**

The key distinction comes from whether there is an explicit mathematical expression for the generative function $\mathcal{G}$. For model-guided approaches, we assume this explicit form connecting $\mathcal{X}$ and $\mathcal{Z}$ is known to us. In addition, the noise distribution for the generative process has to be considered. Given the explicit function $\mathcal{G}(\cdot)$, the distribution of latent and noise variables, one can define a loss using classical statistical inference (for example - maximizing log-likelihood, parameter estimation through method of moments or a combination of both). In model-agnostic methods, we consider the generative function $\mathcal{G}(\cdot)$ to be parameterized by a neural network and no explicit functional form is assumed. The loss in such cases is defined through the end goals of the application. For instance, we may want to match the pixel values of original and decoded images, or ensure real and generated distribution have a minimal distance in probability space.

This raises the obvious question of whether there are some clear benefits of one approach over the other. One advantage of the model-guided approach is that it can embed our beliefs about the physical phenomena directly into the problem. In data-scarce situations and those in which the underlying physical phenomena is well-understood, this is a preferred solution strategy. Oftentimes the noise model or some parameter estimation step may have mismatch. In such case, robust strategies need to be designed accounting for outliers in the

data. However, there are plenty of real-world problems where the data generative process is not well understood, such as generation of images from real vectors or natural language from thought. In such situations, model-agnostic strategy offers a clear advantage. The main goal of this thesis is to solve multiple unsupervised learning problems and observe how these two strategies are at play.

1.2 Main Contributions

An unsupervised learning problem of widespread interest across diverse disciplines of data science is *clustering*. Clustering provides the first insight about hidden patterns in a data by separating commonalities in the data into distinct groups (clusters). While there has been extensive research on clustering, k-means has remained the most popular algorithm once the data is reduced to a low-dimensional manifold. At the same time, advancements in representation learning has given rise to powerful representation learning networks, GANs [56]. The first question of this thesis arises from an attempt to use Generative Adversarial Networks for clustering.

Question 1: *Can we cluster in the latent space of Generative Adversarial Networks ?*

We answer this question in the affirmative. Traditional priors such as Gaussian latent variable fail to cluster in the latent space and we resort to a discrete-continuous latent vector to achieve the clustering. Unlike GANs, an Encoder is an additional component of *ClusterGAN* architecture to ensure the reverse mapping preserves distance in the latent space as well. *ClusterGAN* outperforms recent auto-encoder based algorithms as well as classical spectral and agglomerative clustering and non-negative matrix factorization.

Although ClusterGAN is able to cluster data belonging to diverse modalities such as images, point-cloud and single-cell RNA sequencing, it is not robust to semantic transformations in the data. By semantic transformations, we mean high-level attributes such as structure, class-specific features which are not captured by low-level pixel patches. For instance, an insect and a frog may be placed in an identical green background; yet the images differ in the animal class. This makes it difficult to cluster more diverse datasets such as

natural images of scenes, where intra-cluster variations are significant. Recent research has focused on mutual information as a regularizer to preserve semantic information between latent vectors and high-dimensional data [63]. A paradigm shift in mutual information estimation was proposed in [9] where the authors used lower bounds of mutual information to estimate it. We find this approach to have drawbacks, such as training instabilities and estimates diverging in high dimensions. A change in number of hidden layers or optimization algorithm leads to the estimate diverging to infinity. This is undesirable since we do not know apriori which architecture and learning rate would work best for which dataset. We refer to this issue as instability in the context of neural estimator training. The gradients used to train such estimator also has biased. This leads to our second question.

Question 2*a*: *Can we design an estimator for mutual information estimation in high dimensions that is stable (with regards to training) and avoid biased gradients in training ?*

Question 2*b*: *Does the estimator extend to conditional mutual information estimation ?*

To achieve this, we propose a new approach for mutual information estimation based on classifier likelihood-ratio. It deviates from the state-of-the-art framework of optimizing for a lower bound directly whereby biased gradients were produced.

Moreover, the estimator can be easily extended for conditional mutual information estimation.

These previous two solutions are model-agnostic in that they do not assume an explicit form of how the data is generated from latent variables. Our last two problems are defined for a concrete downstream task. We first address a biology application of resolving segmental duplications in genomes. Segmental duplications are regions in the human genome which are very similar to each other, yet have distinct variations. One can think of this is a special type of clustering problem with missing data. We raise some modeling questions in this application.

Question 3*a*: *Can we formulate the problem of resolving segmental duplication as a low-rank matrix completion ?*

Question 3*b*: *With the assumption of single-nucleotide variants, does the problem reduce to*

signed community detection ? How do we determine communities in signed networks ?

We show that the problem can be successfully formulated as *matrix completion over discrete entries*. This is different from traditional matrix completion and needs additional algorithmic steps to solve it. The second question, in fact, leads us to design a novel algorithm for detecting signed communities in graphs and understand the fundamental limits of this problem.

Finally, we study another application in natural language generation where unsupervised and supervised approaches blend together. This problem demonstrates a way in which unsupervised learning may be used for extracting relevant information in the absence of labeled data. The application domain is email, a primary form of communication in both personal and enterprise settings. We define a new problem of task-focused summarizarion in emails; in particular, generation of To-Do items from emails.

Question 4: *How can we combine unsupervised task-focused sentence extraction with a sequence-to-sequence generation ?*

The solution is a two-stage framework. The first stage uses matching (either lexical key phrases or semantic concepts) to determine which sentences in the email are informative with regards to completing a task promised by the user. It is difficult to obtain such labels for each and every sentence in an email. So, unsupervised learning is used for this stage.

1.3 Organization

The rest of the thesis is organized into multiple chapters, each chapter focusing on one of the aforementioned questions. Each chapter is written in a self-sufficient manner so that the reader can dive directly into any one of them.

1. Chapter 2 introduces the problem of clustering using GAN and proposes the ClusterGAN architecture. There are various interesting results other than the clustering performance itself. We find the interpolation to be preserved in the latent space of ClusterGAN. The image generation quality is also improved by this new architecture.

2. Chapter 3 studies the problem of mutual information and conditional mutual information estimation. En route finding the best estimator, we investigate several alternatives. The conditional mutual information estimator is also applied for conditional independence testing.

   The last two chapters might be more appealing to the reader interested in applications of unsupervised learning to diverse problem domains.

3. The biological application of resolving segmental duplication is presented in Chapter 4. We draw inspiration from low-rank matrix completion and design novel algorithms to solve it. The alternate formulation of signed community detection is also studied here. Perhaps the most interesting portion is applying these algorithms to sequence multiple strains of HIV-Virus.

4. Chapter 5 defines the problem of task-focused summarization and introduces the new problem of To-Do item generation from emails. It has a component of new data generation through crowd-sourcing as well as combining multiple blocks to form a working pipeline for a real-world problem.

5. We conclude the thesis in Chapter 6 and discuss future research directions arising from this thesis.

Chapter 2

CLUSTERGAN : LATENT SPACE CLUSTERING IN GENERATIVE ADVERSARIAL NETWORKS

Generative Adversarial networks (GANs) have obtained remarkable success in many unsupervised learning tasks and unarguably, clustering is an important unsupervised learning problem. While one can potentially exploit the latent-space back-projection in GANs to cluster, we demonstrate that the cluster structure is not retained in the GAN latent space. In this Chapter, we propose ClusterGAN[1] as a new mechanism for clustering using GANs. By sampling latent variables from a mixture of one-hot encoded variables and continuous latent variables, coupled with an inverse network (which projects the data to the latent space) trained jointly with a clustering specific loss, we are able to achieve clustering in the latent space. Our results show a remarkable phenomenon that GANs can preserve latent space interpolation across categories, even though the discriminator is never exposed to such vectors. We compare our results with various clustering baselines and demonstrate superior performance on both synthetic and real datasets.

2.1 Introduction

2.1.1 Motivation

Representation learning enables machine learning models to decipher underlying semantics in data and disentangle hidden factors of variation. These powerful representations have made it possible to transfer knowledge across various tasks. But what makes one representation better than another ? [10] mentioned several general-purpose priors that are not dependent on the downstream task, but appear as commonalities in good representations. One of the

[1]This chapter is based on joint work with Himanshu Asnani, Eugene Lin and Sreeram Kannan [106].

general-purpose priors of representation learning that is ubiquitous across data intensive domains is clustering. Clustering has been extensively studied in unsupervised learning with multifarious approaches seeking efficient algorithms [111], problem specific distance metrics [155], validation [61] and the like. Even though the main focus of clustering has been to separate out the original data into classes, it would be even nicer if such clustering was obtained along with dimensionality reduction where the real data actually seems to come from a lower dimensional manifold.

In recent times, much of unsupervised learning is driven by deep generative approaches, the two most prominent being Variational Autoencoder (VAE) [75] and Generative Adversarial Network (GAN) [56]. The popularity of generative models themselves is hinged upon the ability of these models to capture high dimensional probability distributions, imputation of missing data and dealing with multimodal outputs. Both GAN and VAE aim to match the real data distribution (VAE using an explicit approximation of maximum likelihood and GAN through implicit sampling), and simulataneously provide a mapping from a latent space $\mathcal{Z}$ to the input space $\mathcal{X}$. The latent space of GANs not only provides dimensionality reduction, but also gives rise to novel applications. Perturbations in the latent space could be used to determine adversarial examples that further help build robust classifiers [66]. Compressed sensing using GANs [16] relies on finding a latent vector that minimizes the reconstruction error for the measurements. Generative compression is yet another application involving $\mathcal{Z}$ [130]. One of the most fascinating outcomes of the GAN training is the interpolation in the latent space. Simple vector arithmetic properties emerge which when manipulated lead to changes in the semantic qualities of the generated images [123]. This differentiates GANs from traditional dimensionality reduction techniques [98] [96] which lack interpretability. One potential application that demands such a property is clustering of cell types in genomics [92]. GANs provide a means to understand the change in high-dimensional gene expression as one traverses from one cell type (i.e., cluster) to another in the latent space. Here, it is critical to have both clustering as well as good interpretability and interpolation ability. This brings us to the principal motivation of this work: ***Can we design a GAN***

training methodology that clusters in the latent space?

2.1.2 Prior Art

Deep learning approaches have been used for dimensionality reduction starting with variants of the autoencoder such as the stacked denoising autoencoders [152], sparse autoencoder [31] and deep CCA [3]. Architectures for deep unsupervised subspace clustering have also been built on the encoder-decoder framework [68]. Recent works have addressed this problem of joint clustering and dimensionality reduction in autoencoders. [156] solved this problem by initializing the cluster centroids and the embedding with a stacked autoencoder. Then they use alternating optimization to improve the clustering and report state-of-the-art results in both clustering accuracy and speed on real datasets. The clustering algorithm is referred to as DEC in their paper. Since K-means is often the most widely used algorithm for clustering, [158] improved upon DEC by introducing a modified cost function that incorporates the K-means loss. They optimized the non-convex objective using alternating SGD to obtain an embedding that is amenable to K-means clustering. Their algorithm DCN was shown to outperform all standard clustering methods on a range of datasets. It is interesting to note that the vanilla autoencoder by itself did not explicitly have any clustering objective. But it could be improved to achieve this end by careful algorithmic design. Since GANs have outperformed autoencoders in generating high fidelty samples, we had a strong intuition in favour of the powerful latent representations of GAN providing improved clustering performance also.

Interpretable representation learning in the latent space has been investigated for GANs in the seminal work of [27]. The authors trained a GAN with an additional term in the loss that seeks to maximize the mutual information between a subset of the generator's noise variables and the generated output. The key goal of InfoGAN is to create interpretable and disentangled latent variables. While InfoGAN does employ discrete latent variables, it is not specifically designed for clustering. In fact, a key distinction between ClusterGAN and InfoGAN is the encoder design. InfoGAN shares the weights of the information network with

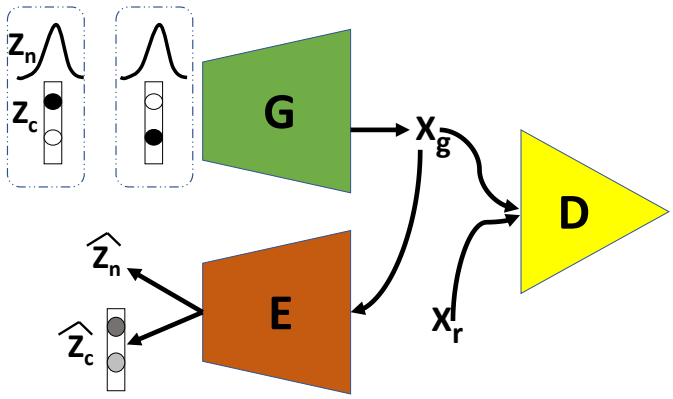


Figure 2.1: ClusterGAN Architecture

the discriminator and the lower layers of the information network change weights during both discriminator training and information objective training. However, in ClusterGAN, the encoder network is separate and only caters to the cycle loss term. In this Chapter, we show that our proposed architecture is superior to InfoGAN for clustering. The other prominent family of generative models, VAE, has the additional advantage of having an inference network, the encoder, which is jointly learnt during training. This enables mapping from $\mathcal{X}$ to $\mathcal{Z}$ that could potentially preserve cluster structure by suitable algorithmic design. Unfortunately, no such inference mechanism exists in GANs, let alone the possibility of clustering in the latent space. To bridge the gap between VAE and GAN, various methods such as Adversarially Learned Inference (ALI) [41], Bidirectional Generative Adversarial Networks (BiGAN) [38] have introduced an inference network which is trained to match the joint distributions of (x, z) learnt by the encoder $\mathcal{E}$ and decoder $\mathcal{G}$ networks. Typically, the reconstruction in ALI/BiGAN is poor as there is no deterministic pointwise matching between x and $\mathcal{G}(\mathcal{E}(x))$ involved in the training. Architectures such as Wasserstein Autoencoder [150], Adversarial Autoencoder [97], which depart from the traditional GAN framework, also have an encoder as part of the network. So this led us to consider a formulation using an Encoder which could *both reduce the cycle loss as well as aid in clustering.*

2.1.3 Main Contributions

To the best of our knowledge, this is the first work that addresses the problem of clustering in the latent space of GAN. The main contributions of this Chapter can be summarized as follows:

- We show that even though the GAN latent variable preserves information about the observed data, the latent points are smoothly scattered based on the latent distribution leading to no observable clusters.

- We propose three main algorithmic ideas in ClusterGAN in order to remedy this situation.

    1. We utilize a **mixture of discrete and continuous** latent variables in order to create a non-smooth geometry in the latent space.
    2. We propose a **novel backpropagation algorithm** accommodating the discrete-continuous mixture, as well as an **explicit inverse-mapping network** to obtain the latent variables given the data points, since the problem is non-convex.
    3. We propose to jointly train the GAN along with the inverse-mapping network with **a clustering-specific loss** so that the distance geometry in the projected space reflects the distance-geometry of the variables.

- We compare ClusterGAN and other possible GAN based clustering algorithms, such as InfoGAN, along with multiple clustering baselines on varied datasets. This demonstrates the superior performance of ClusterGAN for the clustering task.

- We demonstrate that ClusterGAN surprisingly retains good interpolation across the different classes (encoded using one-hot latent variables), even though the discriminator is never exposed to such samples.

The formulation is general enough to provide a *meta* framework that incorporates the additional desirable property of clustering in GAN training.

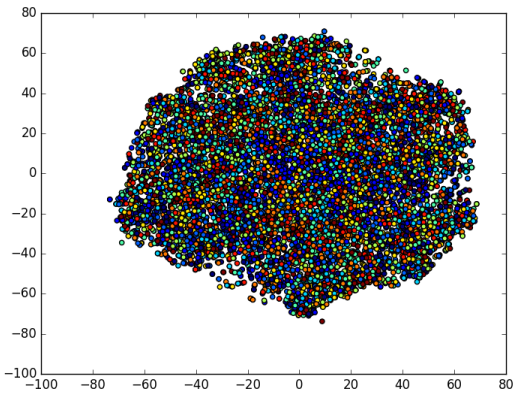
2.2 Discrete-Continuous Prior

2.2.1 Background

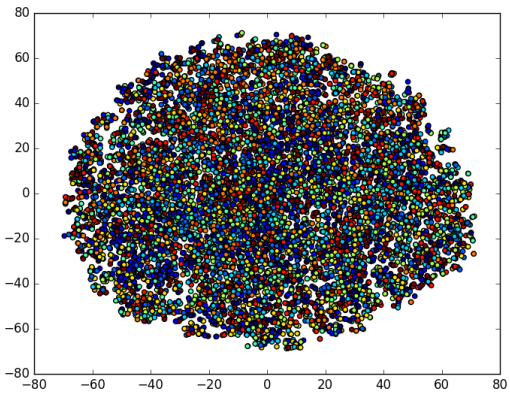
Generative adversarial networks consist of two components, the generator $\mathcal{G}$ and the discriminator $\mathcal{D}$. Both $\mathcal{G}$ and $\mathcal{D}$ are usually implemented as neural networks parameterized by Θ_G and Θ_D respectively. The generator can also be considered to be a mapping from latent space to the data space which we denote as $\mathcal{G} : \mathcal{Z} \mapsto \mathcal{X}$. The discrimator defines a mapping from the data space to a real value which can correspond to the probability of the sample being real, $\mathcal{D} : \mathcal{X} \mapsto \mathbb{R}$. The GAN training sets up a two player game between $\mathcal{G}$ and $\mathcal{D}$, which is defined by the minimax objective : $\min_{\Theta_G} \max_{\Theta_D} \mathbf{E}_{x\sim\mathbb{P}_x^r}\, q(\mathcal{D}(x)) + \mathbf{E}_{z\sim\mathbb{P}_z}\, q(1 - \mathcal{D}(\mathcal{G}(z)))$, where $\mathbb{P}_x^r$ is the distribution of real data samples, $\mathbb{P}_z$ is the prior noise distribution on the latent space and $q(.)$ is the quality function. For vanilla GAN, $q(x) = \log x$, and for Wasserstein GAN (WGAN) $q(x) = x$. We also denote the distribution of generated samples x_g as $\mathbb{P}_x^g$. The discriminator and the generator are optimized alternatively so that at the end of training $\mathbb{P}_x^g$ matches $\mathbb{P}_x^r$.

2.2.2 Vanilla GAN does not cluster well in the latent space

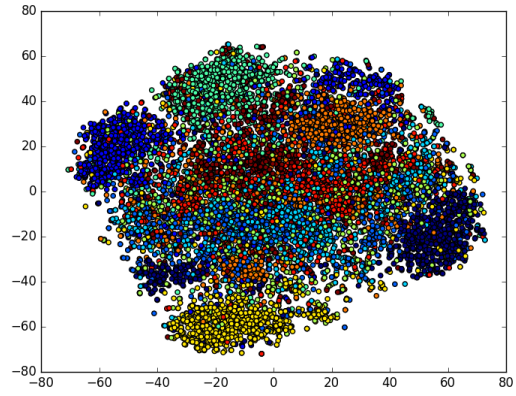
One possible way to cluster using a GAN is to back-propagate the data into the latent space (using back-propogation decoding [93]) and cluster the latent space. However, this method usually leads to very bad results (see Fig. 2.2 for clustering results on MNIST). The key reason is that, if indeed, back-propagation succeeds, then the back-projected data distribution should look similar to the latent space distribution, which is typically chosen to be a Gaussian or uniform distribution, and we cannot expect to cluster in that space. Thus even though the latent space may contain full information about the data, the distance geometry in the latent space does not reflect the inherent clustering. In [60], the authors



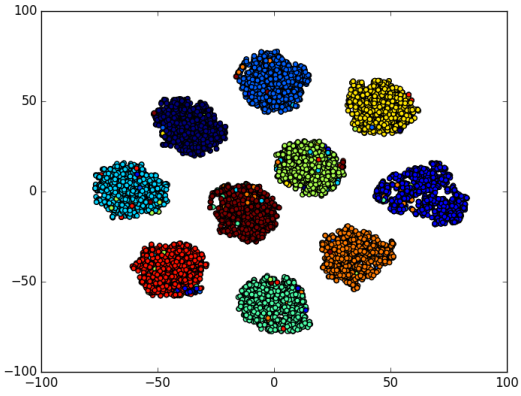
(a) $z \sim$ Uniform



(b) $z \sim$ Normal



(c) $z \sim$ Gaussian Mix



(d) $z \sim (z_n, z_c)$

Figure 2.2: TSNE visualization of latent vectors for GANs trained with different priors on MNIST.

sampled from a Gaussian mixture prior and obtained diverse samples even in limited data regimes. However, even GANs with a Gaussian mixture failed to cluster, as shown in 2.2(c). As observed by the authors of DeLiGAN, Gaussian components tend to 'crowd' and become redundant. Lifting the space using categorical variables could solve the problem effectively. But continuity in latent space is traditionally viewed to be a pre-requisite for the objective of good interpolation. In other words, interpolation seems to be at loggerheads with the clustering objective. We demonstrate in this Chapter how ClusterGAN can obtain **good interpolation and good clustering** simultaneously.

2.2.3 Sampling from Discrete-Continuous Mixtures

In ClusterGAN, we sample from a prior that consists of normal random variables cascaded with one-hot encoded vectors. To be more precise $z = (z_n, z_c), z_n \sim \mathcal{N}(0, \sigma^2 I_{d_n}), z_c = e_k, k \sim \mathcal{U}\{1, 2, \ldots, K\}$, e_k is the k^{th} elementary vector in $\mathbb{R}^K$ and K is the number of clusters in the data. In addition, we need to choose σ in such a way that the one-hot vector provides sufficient signal to the GAN training that leads to each mode only generating samples from a corresponding class in the original data. To be more precise, we chose $\sigma = 0.10$ in all our experiments so that each dimension of the normal latent variables, $z_{n,j} \in (-0.6, 0.6) << 1.0 \; \forall j$ with high probability. Small variances σ are chosen to ensure the clusters in $\mathcal{Z}$ space are separated. Hence this prior naturally enables us to design an algorithm that clusters in the latent space.

2.2.4 Modified Backpropagation Based Decoding

Previous works [33] [93] have explored solving an optimization problem in z to recover the latent vectors, $z^* = \arg\min_z \mathcal{L}(\mathcal{G}(z), x) + \lambda \|z\|_p$, where $\mathcal{L}$ is some suitable loss function and $\|\cdot\|_p$ denotes the norm. This approach is insufficient for clustering with traditional latent priors even if backpropagation was lossless and recovered accurate latent vectors. To make the situation worse, the optimization problem above is non-convex in z ($\mathcal{G}$ being implemented as a neural network) and can obtain different embeddings in the $\mathcal{Z}$ space based

on initialization. Some of the approaches to address this issue could be multiple restarts with different initialiations to obtain z^*, or stochastic clipping of z at each iteration step. None of these lead to clustering, since they do not address the root problem of sampling from separated manifolds in $\mathcal{Z}$. But our sampling procedure naturally gives way to such an algorithm. We use $\mathcal{L}(\mathcal{G}(z), x) = \|\mathcal{G}(z) - x\|_1$. Since we sample from a normal distrubution, we use the regularizer $\|z_n\|_2^2$, penalizing only the normal variables. We use K restarts, each sampling z_c from a different one-hot component and optimize with respect to only the normal variables, keeping z_c fixed. Adam [74] is used for the updates during Backprop decoding. Formally, Algorithm 1 summarizes the approach.

Input: Real sampler x, Generator function $\mathcal{G}$, Number of Clusters K,
Regularization parameter λ, Adam iterations τ

Output: Latent embedding z^*

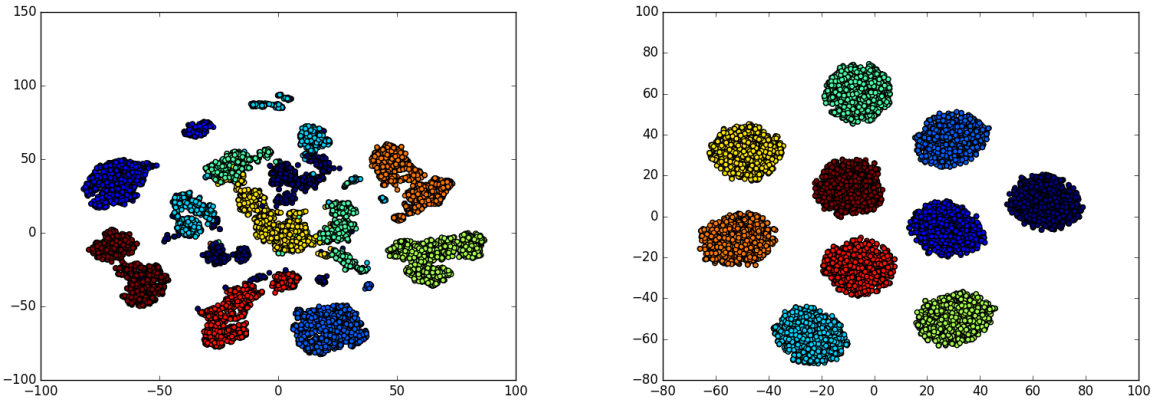
for $k \in \{1, 2, \ldots K\}$ **do**

- Sample $z_n^0 \sim \mathcal{N}(0, \sigma^2 I_{d_n})$
- Initialization $z_k^0 \leftarrow (z_n^0, e_k)$ (e_k is k^{th} elementary unit vector in K dimensions)
- **for** $t \in \{1, 2, \ldots \tau\}$ **do**
  - Obtain the gradient of loss function $g \leftarrow \nabla_{z_n} \left(\|\mathcal{G}(z_k^{t-1}) - x\|_1 + \lambda \|z_n^{t-1}\|_2 \right)$
  - Update z_n^t using g with Adam iteration to minimize loss.
  - Clipping of z_n^t, i.e., $z_n^t \leftarrow \mathcal{P}_{[-0.6,0.6]}(z_n^t)$
  - $z_k^t \leftarrow (z_n^t, e_k)$
- **end**
- Update z^* if z_k^τ has lowest loss obtained so far.

end

return z^*

Algorithm 1: DECODE_LATENT



(a) Non-linear generator with $z \sim \mathcal{N}(0, I)$ (b) Linear generator with z one-hot encoded

Figure 2.3: TSNE visualization of latent vectors. Linear Generator recovers clusters, suggesting that representation power is not a bottleneck.

2.2.5 Linear Generator clusters perfectly

The following lemma suggests that with discrete-continuous mixtures, we need only linear generation to generate mixture of Gaussians in the generated space.

Lemma 1. *Clustering with only z_n cannot recover a mixture of gaussian data in the linearly generated space. Further $\exists$ a linear $G(\cdot)$ mapping discrete-continuous mixtures to a mixture of Gaussians.*

Proof. If latent space has only the continuous part, $z_n \sim \mathcal{N}(0, \sigma^2 I_{d_n})$, then by the linearity property, any linear generation can only produce Gaussian in the generated space. Now we show there exists a $G(\cdot)$ mapping discrete-continuous mixtures to the generate data $X \sim \mathcal{N}(\mu_\omega, \sigma^2 I_{d_n})$, where $\omega \sim \mathcal{U}\{1, 2, \ldots, K\}$ (K is the number of mixtures). This is possible if we let $z_n \sim \mathcal{N}(0, \sigma^2 I_{d_n}), z_c = e_k, k \sim \mathcal{U}\{1, 2, \ldots, K\}$ and $G(z_n, z_c) = z_n + Az_c$, $A = diag[\mu_1, \cdots, \mu_K]$ being a $K \times K$ diagonal matrix with diagonal entries as the means μ_i. □

To illustrate this lemma, and hence the drawback of traditional priors $\mathbb{P}_z$ for clustering, we performed a simple experiment. The real samples are drawn from a mixture of 10 Gaussians in $\mathbb{R}^{100}$. The means of the Gaussians are sampled from $\mathcal{U}(-0.3, 0.3)^{100}$ and the variance of each component is fixed at $\sigma = 0.12$. We trained a GAN with $z \sim \mathcal{N}(0, I_{10})$ where the generator is a multi-layer perceptron with two hidden layers of 256 units each. For comparison, we also trained a GAN with z sampled from one-hot encoded normal vectors, the dimension of categorical variable being 10. The generator for this GAN consisted of a linear mapping $W \in \mathbb{R}^{100\times 10}$, such that $x = Wz$. After training, the latent vectors are recovered using Algorithm 1 for the linear generator, and 10 restarts with random initializations for the non-linear generator. Even for this toy setup, the linear generator perfectly clustered the latent vectors (Acc. = 1.0, NMI = 1.0, ARI = 1.0), but the non-linear generator performed poorly (Acc. = 0.73, NMI = 0.75, ARI = 0.60) (Figure 2.3). The situation becomes worse for real datasets such as MNIST when we trained a GAN using latent vectors drawn from uniform, normal or a mixture of Gaussians. None of these configurations succeeded in clustering in the latent space as shown in Figure 2.2.

2.2.6 Separate Modes for distinct classes in the data

It was surprising to find that trained in a purely unsupervised manner with the true number of clusters known, without additional loss terms, each one-hot encoded component generated points from a specific class in the original data. For instance, $z = (z_n, e_k)$ generated a particular digit $\pi(k)$ in MNIST, for multiple samplings of $z_n \sim \mathcal{N}(0, \sigma^2 I_{d_n})$ (π denotes a permutation). This was a necessary first step for the success of Algorithm 1. We also quantitatively evaluated the modes learnt by the GAN by using a supervised classifier for MNIST. The supervised classifier had a test accuracy of 99.2%, so it had high reliability of distinguishing the digits. We sample from a mode k and generate a digit x_g. It is then classified by the classifier as $\hat{y}$. From this pair $(k, \hat{y})$, we can map each mode to a digit and compute the accuracy of digit $\hat{y}$ being generated from mode k. This is denoted as Mode Accuracy. Each digit sample x_r with label y can be decoded in the latent space by Algorithm

1 to obtain z. Now z can be used to generate x_g, which when passed through the classifier gives the label $\hat{y}$. The pair $(y, \hat{y})$ must be equal in the ideal case and this accuracy is denoted as Reconstruction Accuracy. Finally, all the mappings of points in the same class in $\mathcal{X}$ space should have the same one-hot encoding when embedded in $\mathcal{Z}$ space. This defines the Cluster Accuracy. This metholodgy can be extended to quantitatively evaluate mode generation for other datasets also, provided there is a reliable classifier. For MNIST, we obtained Mode Accuracy of 0.97, Reconstruction Accuracy of 0.96 and Cluster Accuracy of 0.95. Some of the modes in Fashion-MNIST and MNIST are shown in Figures 2.4 and 2.5, respectively. Supplementary materials contain the images from all modes in these two datasets.

Figure 2.4: Fashion items generated from distinct modes : Fashion-MNIST

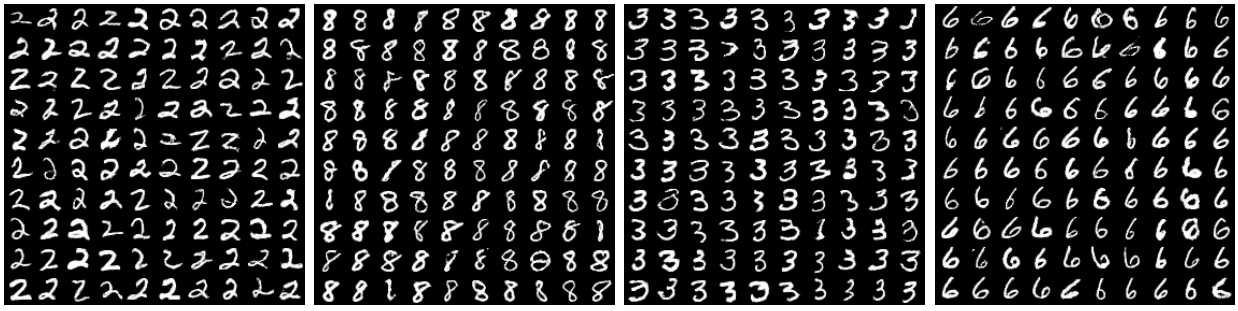


Figure 2.5: Digits generated from distinct modes : MNIST

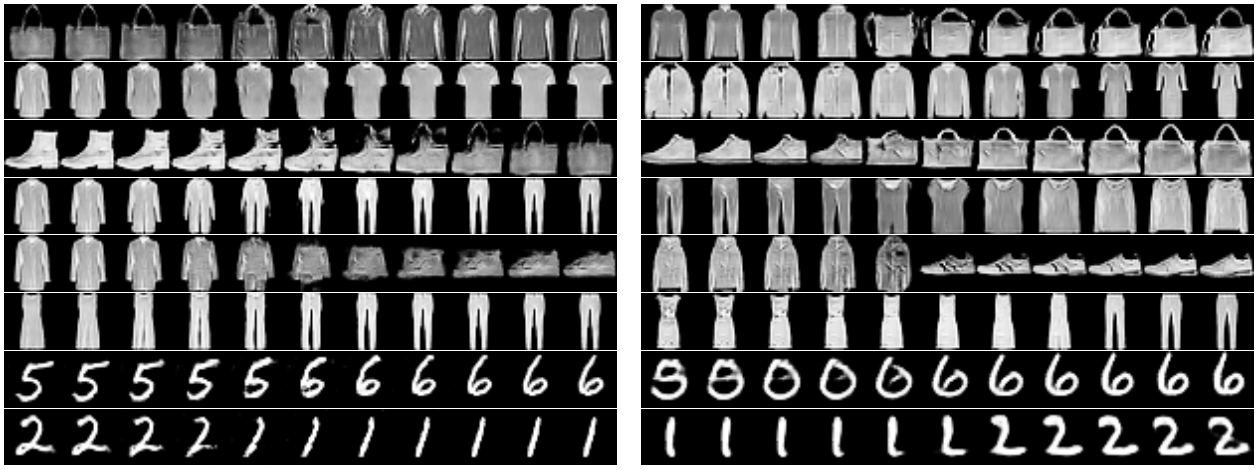


Figure 2.6: Comparison of Latent Space Interpolation : ClusterGAN (left) and vanilla WGAN (right)

2.2.7 Interpolation in latent space is preserved

The latent space in a traditional GAN with Gaussian latent distribution enforces that different classes are continuously scattered in the latent space, allowing nice inter-class interpolation, which is a key strength of GANs. In ClusterGAN, the latent vector z_c is sampled with a one-hot distribution and in order to interpolate across the classes, we will have to sample from a convex combination on the one-hot vector. While these vectors have never been sampled during the training process, we surprisingly observed very smooth inter-class interpolation in ClusterGAN. To demonstrate interpolation, we fixed the z_n in two latent vectors with different z_c components, say $z_c^{(1)}$ and $z_c^{(2)}$ and interpolated with the one-hot encoded part to give rise to new latent vectors $z = (z_n, \mu z_c^{(1)} + (1-\mu) z_c^{(2)}), \mu \in [0, 1]$. As Figure 2.6 illustrates, we observed a nice transition from one digit to another as well as across different classes in FashionMNIST. This demonstrates that ClusterGAN learns a very smooth manifold even on the untrained directions of the discrete-continuous distribution. We also show interpolations from a vanilla GAN trained with Gaussian prior as reference.

Input: Functions $\mathcal{G}$, $\mathcal{D}$ and $\mathcal{E}$, Regularization parameters β_n, β_c, learning rate η, parameters Θ_G^t, Θ_E^t

Output: $\Theta_G^{(t+1)}, \Theta_E^{(t+1)}$

Sample $z^{(i)}{}_{i=1}^{m}$ from $\mathbb{P}^z$, $z = (z_n, z_c)$

$$g_{\Theta_G} \leftarrow \nabla_{\Theta_G}\left(-\sum_{i=1}^{m} q(\mathcal{D}(\mathcal{G}(z^{(i)})) + \beta_n \sum_{i=1}^{m} \|z_n^{(i)} - \mathcal{E}(\mathcal{G}(z_n^{(i)}))\|_2^2 + \beta_c \sum_{i=1}^{m} \mathcal{H}(z_c^{(i)}, \mathcal{E}(\mathcal{G}(z_c^{(i)})))\right)$$

$$g_{\Theta_E} \leftarrow \nabla_{\Theta_E}\left(\beta_n \sum_{i=1}^{m} \|z_n^{(i)} - \mathcal{E}(\mathcal{G}(z_n^{(i)}))\|_2^2 + \beta_c \sum_{i=1}^{m} \mathcal{H}(z_c^{(i)}, \mathcal{E}(\mathcal{G}(z_c^{(i)})))\right)$$

Update Θ_G using $(g_{\Theta_G}, \Theta_G^t)$ with Adam ; similarly for Θ_E.

return Θ_G, Θ_E

Algorithm 2: UPDATE_PARAM

2.3 ClusterGAN

Even though the above approach enables the GAN to cluster in the latent space, it may be able to perform even better if we had a clustering specific loss term in the minimax objective. For MNIST, digit strokes correspond well to the category in the data. But for more complicated datasets, we need to enforce structure in the GAN training. One way to ensure that is to enforce precise recovery of the latent vector. We therefore introduce an encoder $\mathcal{E} : \mathcal{X} \mapsto \mathcal{Z}$, a neural network parameterized by Θ_E. The GAN objective now takes the following form:

$$\min_{\Theta_G, \Theta_E} \max_{\Theta_D} \mathop{\mathbf{E}}_{x \sim \mathbb{P}_x^r} q(\mathcal{D}(x)) + \mathop{\mathbf{E}}_{z \sim \mathbb{P}_z} q(1 - \mathcal{D}(\mathcal{G}(z))) + \beta_n \mathop{\mathbf{E}}_{z \sim \mathbb{P}_z} \|z_n - \mathcal{E}(\mathcal{G}(z_n))\|_2^2 + \beta_c \mathop{\mathbf{E}}_{z \sim \mathbb{P}_z} \mathcal{H}(z_c, \mathcal{E}(\mathcal{G}(z_c)) \quad (2.1)$$

where $\mathcal{H}(.,.)$ is the cross-entropy loss. The relative magnitudes of the regularization coeficients β_n and β_c enable a flexible choice to vary the importance of preserving the discrete and continuous portions of the latent code. One could imagine other variations of the regularization that map $\mathcal{E}(\mathcal{G}(z))$ to be close to the centroid of the respective cluster, for in-

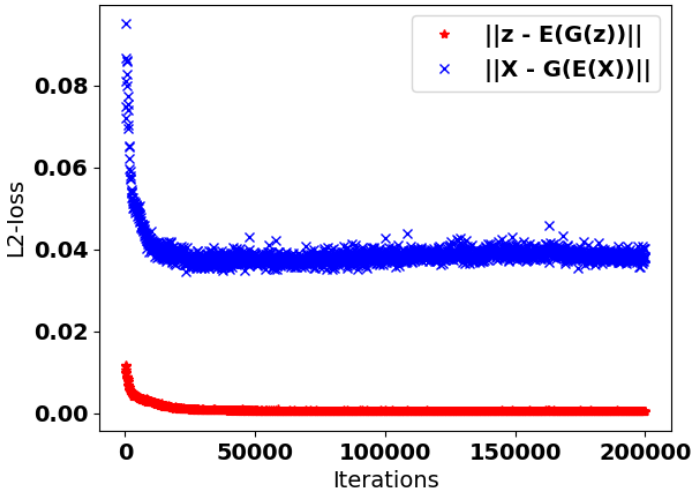


Figure 2.7: Decrease of Cycle Loss with iterations in MNIST. The mean square L2-distance $||x - \mathcal{G}(\mathcal{E}(x))||$ was 0.038 and $||z - \mathcal{E}(\mathcal{G}(z))||$ was 0.0004. Mean distances were computed on a test batch not used in training.

stance $\|\mathcal{E}(\mathcal{G}(z^{(i)})) - \mu^{c(i)}\|_2^2$, in similar spirit as K-Means. The GAN training in this approach involves jointly updating the parameters of Θ_G and Θ_E (Algorithm 2).

As shown in Figure 2.7, in our architecture, both x is close to $\mathcal{G}(\mathcal{E}(x))$ and z is close to $\mathcal{E}(\mathcal{G}(z))$. Even though our architecture is optimizing for one type of cycle loss, both losses are small. The loss optimized for is even smaller.

2.4 Experiments

2.4.1 Datasets

Synthetic Data The data is generated from a mixture of Gaussians with 4 components in 2D, which constitutes the $\mathcal{Z}$ space. We generated 2500 points from each Gaussian. The $\mathcal{X}$ space is obtained by a non-linear transformation : $x = f(\mathbf{U} \cdot f(\mathbf{W}z))$, where $\mathbf{W} \in \mathbb{R}^{10\times 2}, \mathbf{U} \in \mathbb{R}^{100\times 10}$ with $W_{i,j} \sim \mathcal{N}(0,1), U_{i,j} \sim \mathcal{N}(0,1)$. $f(\cdot)$ is the sigmoid function to introduce non-

linearity.

MNIST It consists of 70k images of digits ranging from 0 to 9. Each data sample is a 28×28 greyscale image. We used the DCGAN with conv-deconv layers, batch normalization and leaky relu activations, the details of which are available in the Supplementary material.

Fashion-MNIST (10 and 5 classes) This dataset has the same number of images with the same image size as MNIST, but it is fairly more complicated. Instead of digits, it consists of various types of fashion products. Supervised methods achieve lower accuracy than MNIST on this dataset. For training a GAN, we used the same architecture as MNIST for this dataset. We also merged some categories which were similar to form a separate 5-class dataset. The five groups were as follows : {Tshirt/Top, Dress}, {Trouser}, {Pullover, Coat, Shirt}, {Bag}, {Sandal, Sneaker, Ankle Boot}.

10x_73k Even though GANs have achieved unprecedented success in generating realistic images, it is not clear whether they can be equally effective for other types of data. In this experiment, we trained a GAN to cluster cell types from a single cell RNA-seq counts matrix. Moreover, computer vision might have ample supply of labelled images, obtaining labels for some fields, for instance biology, is extremely costly and laborious. Thus, unsupervised clustering of data is truly a necessity for this domain. The dataset consists of RNA-transcript counts of 73233 data points belonging to 8 different cell types [163]. To reduce the dimension of the data, we selected 720 highest variance genes across the cells. The entries of the counts matrix $\mathbf{C}$ are first transformed as $\log_2(1 + C_{ij})$ and then divided by the maximum entry of the transformation to obtain values in the range of $[0, 1]$. One of the major challenges in this data is sparsity. Even after sub-selection of genes based on variance, the data matrix was close to 40% zero entries.

Pendigits It is a very different dataset that consists of a time series of $\{x_t, y_t\}_{t=1}^{T}$ coordinates. The points are sampled as writers write digits on a pressure sensitive tablet. The total number of datapoints is 10992, and consists of 10 classes, each for a digit. It provided a unique challenge of training GANs for point cloud data.

For all our experiments in this Chapter, we used an improved variant (WGAN-GP) which

includes a gradient penalty [58]. Using cross-validation for selecting hyperparameters is not an option in purely unsupervised problems due to absence of labels. We adapted standard architectures for the datasets [27] and avoided data specific tuning as much as possible. Some choices of regularization parameters $\lambda = 10$, $\beta_n = 10$, $\beta_r = 10$ worked well across all datasets.

2.4.2 Evaluation

Since clustering is an unsupervised problem, we ensured that all the algorithms are oblivious to the true labels unlike a supervised framework like conditional GAN [100]. We compared ClusterGAN with other possible GAN based clustering approaches we could conceive.

Algorithm 1 + K-Means is denoted as "GAN with bp". To assign a cluster to points using InfoGAN, we used $\arg\max_c \mathbb{P}(c \mid x)$ as an inferred cluster label for x. Further, the features $\phi(x)$ in the last layer of the Discriminator could contain some class-specific discriminating features for clustering. So we used Kmeans on $\phi(x)$ to cluster, denoted as "GAN with Disc. ϕ". We also included clustering results from Non-negative matrix Factorization (NMF) [83] and Aggolomerative Clustering (AGGLO) [162]. AGGLO with Euclidean affinity score and ward linkage gave best results. NMF had both l-1 and l-2 regularization, initialized with Non-negative Double SVD and used KL-divergence loss. We reported normalized mutual information (NMI), adjusted Rand index (ARI), and clustering purity (ACC). Since DCN has been shown to outperform various deep-learning based clustering algorithms, we reported its metrics from the paper [158] for MNIST and Pendigits. We found DCN to be very sensitive to hyperparameter choice, architecture and learning rates and could not obtain reasonable results from it on the other datasets. But we outperformed DCN results on MNIST and Pendigits dataset[2]. The cluster accuracy of VAE+GMM for MNIST is also reported from [87].

Since clustering metrics do not reveal the quality of generated samples from a GAN, we report the Frechet Inception Distance (FID) [62] for the image datasets. We found that

[2]For all baselines and GAN variants, Table 2.1 reports metrics for the model with best validation purity from 5 runs.

Dataset	Algorithm	ACC	NMI	ARI
Synthetic	ClusterGAN	0.99	0.99	0.99
	Info-GAN	0.88	0.75	0.74
	GAN with bp	0.95	0.85	0.88
	GAN with Disc. ϕ	0.99	0.98	0.98
	AGGLO.	0.99	0.99	0.99
	NMF	0.98	0.96	0.97
MNIST	ClusterGAN	0.95	0.89	0.89
	Info-GAN	0.87	0.84	0.81
	GAN with bp	0.95	0.90	0.89
	GAN with Disc. ϕ	0.70	0.62	0.52
	DCN	0.83	0.81	0.75
	VAE+GMM	0.77	-	-
	AGGLO.	0.64	0.65	0.46
	NMF	0.56	0.45	0.36
Fashion-10	ClusterGAN	0.63	0.64	0.50
	Info-GAN	0.61	0.59	0.44
	GAN with bp	0.56	0.53	0.37
	GAN with Disc. ϕ	0.43	0.37	0.23
	AGGLO.	0.55	0.57	0.37
	NMF	0.50	0.51	0.34
Fashion-5	ClusterGAN	0.73	0.59	0.48
	Info-GAN	0.67	0.55	0.42
	GAN with bp	0.73	0.54	0.45
	GAN with Disc. ϕ	0.67	0.49	0.40
	AGGLO.	0.66	0.52	0.36
	NMF	0.67	0.48	0.40
10x_73k	ClusterGAN	0.81	0.73	0.67
	Info-GAN	0.62	0.58	0.43
	GAN with bp	0.65	0.59	0.45
	GAN with Disc. ϕ	0.33	0.17	0.07
	AGGLO.	0.63	0.58	0.40
	NMF	0.71	0.69	0.53
Pendigits	ClusterGAN	0.77	0.73	0.65
	Info-GAN	0.72	0.73	0.61
	GAN with bp	0.76	0.71	0.63
	GAN with Disc. ϕ	0.65	0.57	0.45
	DCN	0.72	0.69	0.56
	AGGLO.	0.70	0.69	0.52
	NMF	0.67	0.58	0.45

Table 2.1: Comparison of clustering metrics across datasets

Dataset	**Algorithm**			
	Cluster GAN	WGAN (Normal)	WGAN (One-Hot)	Info GAN
MNIST	0.81	0.88	0.94	1.88
Fashion	0.91	0.95	6.14	11.04

Table 2.2: Comparison of Frechet Inception Distance (FID) (Lower distance is better)

Dataset : MNIST, Algorithm : ClusterGAN				
ACC				
K = 7	**K = 9**	**K = 10**	**K = 11**	**K = 13**
0.60	0.84	0.95	0.90	0.84

Table 2.3: Robustness to Cluster Number K

ClusterGAN achives good clustering without compromising sample quality as shown in Table 2.2.

In all datasets, we provided the true number of clusters to all algorithms. In addition, for MNIST, Table 2.3 provides the clustering performance of ClusterGAN as number of clusters is varied. Overestimates do not severely hurt ClusterGAN; but underestimate does.

2.4.3 Scalability to Large Number of Clusters

We ran ClusterGAN on Coil-20 ($N = 1440$, $K = 20$) and Coil-100 ($N = 7200, K = 100$) datasets, where N is the number of Data points. ClusterGAN could obtain good clusters even with such high value of K. These data sets were particularly difficult for GAN training with only a few thousand data points. Yet, we found similar behavior as MNIST / Fashion-MNIST emerging here as well. Distinct modes generated distinct 3D-objects along with

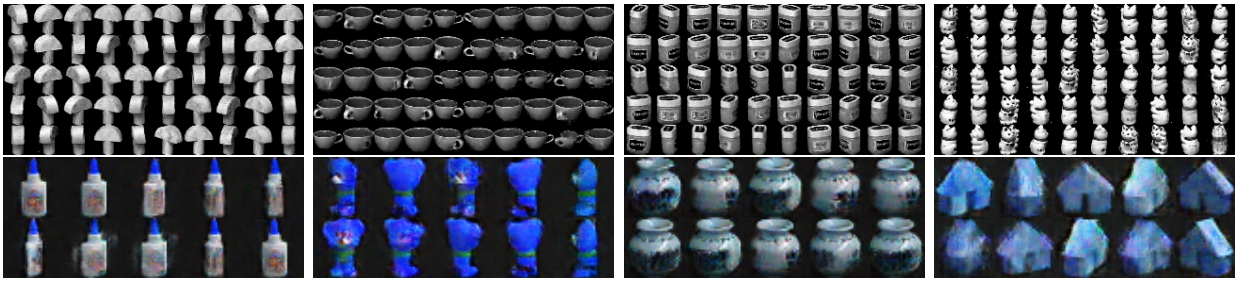


Figure 2.8: Scalability of ClusterGAN to large number of clusters : Modes of Coil-20 (above) and Coil-100 (below)

rotations as shown in Figure 2.8.

2.5 Recent architectures and extensions

Our findings are reinforced by state-of-the-art GAN architectures, which deviate from using traditional uniform or Gaussian priors. We explore in this Section the similarities of these recent advances and study them in the light of ClusterGAN.

As shown in Figure 2.9, recent state-of-the-art GAN architectures advocate for multi-modal prior. In StyleGAN [73], the Gaussian noise distribution is first morphed by a deep fully-connected layer before being fed to the subsequent generator layers for *styling*. Even though this might alleviate some issues, by distributing the Gaussian mass using a non-linear transformation, the discreteness is lacking. uMM-GAN [137] suggests improving StyleGAN further by selecting one of K latent variables (in a weighted combination). The authors show how this discreteness aids in learning a multi-modal real distribution, since each component now needs to learn a relatively simple uni-modal distribution. Similar to ClusterGAN, uMM-GAN also improves the FID scores compared to StyleGAN.

2.6 Preserving Semantic Features: Information meets Clustering

We also ran ClusterGAN on CIFAR-10, which is a dataset with considerable intra-class variability. For CIFAR-10, the modes of ClusterGAN generate images based on a commonality

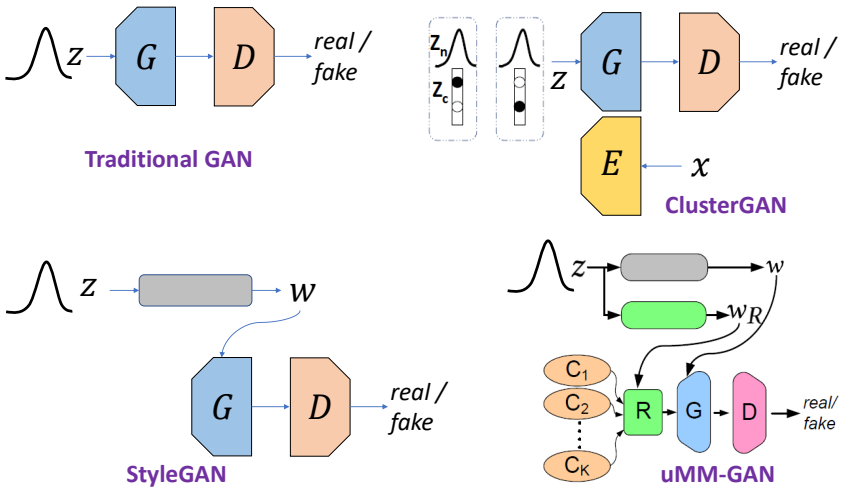


Figure 2.9: Comparison of traditional GAN, ClusterGAN, StyleGAN and uMM-GAN. Unlike traditional GAN, recent advances in GAN training advocate for multi-modal priors.

(like blue background or white background), which may not be in correspondence with the labels. Blue background is present across images in airplane, bird, ship categories. Left to itself, a purely unsupervised model learns a representation that most easily separates the images into categories; but there is no impulsion to enforce higher order semantics of the labels. We believe that maximizing mutual information between an intermediate convolutional layer and the generated image would preserve the high-level semantics.

In [63], the authors are able to achieve significant improvement in clustering performance by using mutual information as a regularizer. In fact, their framework leads to unsupervised representations whose performance on downstream tasks is comparable to supervised learning! Adversarial training is used along with mutual information maximization between local patches in images and the learned high-level representation. In yet another work [69], the authors maximize the mutual information between the classified label on an original image and its transformed variant. With multiple works using mutual information for clustering, the obvious question we asked was: *What are the tools available for high-dimensional mutual information estimation ?* This leads us to the next Chapter in the thesis.

Figure 2.10: Modes of CIFAR-10 pick up features that easily segregate categories, but may not correspond to dataset labels. Blue background (left) and predominant white background (right).

Chapter 3

CCMI : CLASSIFIER BASED CONDITIONAL MUTUAL INFORMATION ESTIMATION

In this Chapter[1] we explore the available estimators for mutual information (MI), a fundamental measure used in data science to capture dependence between variables. In the previous Chapter, we saw how maximizing the mutual information between semantic representations and local features in images can lead to better clustering. MI extends beyond this particular application of clustering. It is a metric that appears in the expression for capacity of a noisy channel, as a bottleneck objective in representation learning and as a test for Bayesian network learning. While we set out to study MI estimators, we found the problem of conditional mutual information (CMI) estimation as a sub-problem.

Conditional Mutual Information (CMI) is a measure of conditional dependence between random variables X and Y, given another random variable Z. It can be used to quantify conditional dependence among variables in many data-driven inference problems such as graphical models, causal learning, feature selection and time-series analysis. While k-nearest neighbor (kNN) based estimators as well as kernel-based methods have been widely used for CMI estimation, they suffer severely from the curse of dimensionality. In this chapter, we leverage advances in classifiers and generative models to design methods for MI estimation and extend it for CMI estimation. Specifically, we introduce an estimator for KL-Divergence based on the likelihood ratio by training a classifier to distinguish the observed joint distribution from the product distribution. We then show how to construct several CMI estimators using this basic divergence estimator by drawing ideas from conditional generative models.

[1]This Chapter is based on joint work with Himanshu Asnani and Sreeram Kannan [105]. An extension of this work was studied in a joint work with additional collaborators Arnab Kumar Mondal, Arnab Bhattacharjee and Prathosh AP in [103].

We demonstrate that the estimates from our proposed approaches do not degrade in performance with increasing dimension and obtain significant improvement over the widely used KSG estimator. Finally, as an application of accurate CMI estimation, we use our best estimator for conditional independence testing and achieve superior performance than the state-of-the-art tester on both simulated and real data-sets.

3.1 Introduction

Mutual information (MI) is a fundamental information theoretic quantity that captures the dependence two random variables. Conditional mutual information (CMI) extends the nice properties of mutual information (MI) in conditional settings. For two random variables X and Y, the mutual information is defined as

$$I(X;Y) = \iint p(x,y) \log \frac{p(x,y)}{p(x)p(y)} dxdy$$

Similarly, for three continuous random variables, X, Y and Z, the conditional mutual information is defined as:

$$I(X;Y|Z) = \iiint p(x,y,z) \log \frac{p(x,y,z)}{p(x,z)p(y|z)} dxdydz$$

assuming that the distributions admit the respective densities $p(\cdot)$. One of the striking features of MI and CMI is that they can capture non-linear dependencies between the variables. In scenarios where Pearson correlation is zero even when the two random variables are dependent, mutual information can recover the truth. This is illustrated in Figure 3.1. Likewise, in the sense of conditional independence for the case of three random variables X,Y and Z, conditional mutual information provides strong guarantees, i.e., $X \perp Y|Z \iff I(X;Y|Z) = 0$.

The conditional setting is even more interesting as dependence between X and Y can potentially change based on how they are connected to the conditioning variable. For instance, consider a simple Markov chain where $X \to Z \to Y$. Here, $X \perp Y|Z$. But a slightly different relation $X \to Z \leftarrow Y$ has $X \not\perp Y|Z$, even though X and Y may be independent as a pair. It is a well known fact in Bayesian networks that a node is independent of its

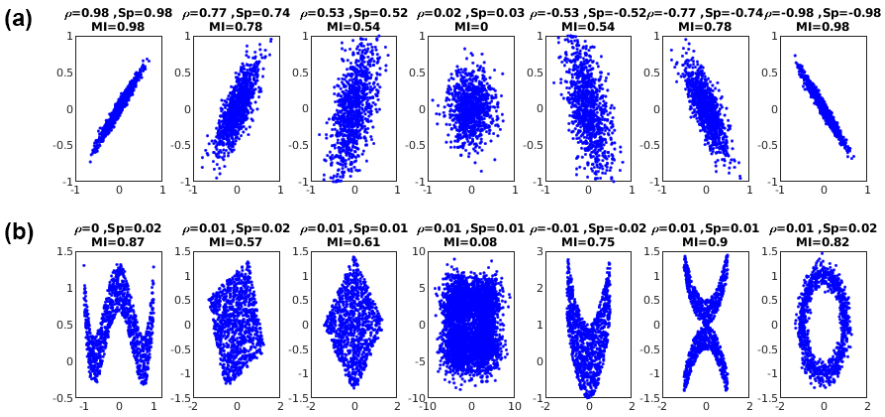


Figure 3.1: Comparison of Pearson Correlation (ρ), Spearman Correlation (Sp) and Mutual Information (MI) for linear and non-linear dependence between two random variables. As shown, only MI is able to capture the dependence when it truly exists. Courtesy [160]

non-descendants given its parents. CMI goes beyond stating whether the pair (X, Y) is conditionally dependent or not. It also provides a quantitative strength of dependence.

3.1.1 Prior Art

The literature is replete with works aimed at applying CMI for data-driven knowledge discovery. [47] used CMI for fast binary feature selection to improve classification accuracy. [94] improved non-rigid image registration by using CMI as a similarity measure instead of global mutual information. CMI has been used to infer gene-regulatory networks [89] or protein modulation [55] from gene expression data. Causal discovery [88] [64] [151] is yet another application area of CMI estimation.

Despite its wide-spread use, estimation of conditional mutual information remains a challenge. One naive method may be to estimate the joint and conditional densities from data and plug it into the expression for CMI. But density estimation is not sample efficient and is often more difficult than estimating the quantities directly. The most widely used technique expresses CMI in terms of appropriate arithmetic of differential entropy estimators

(referred to here as ΣH estimator): $I(X;Y|Z) = h(X,Z) + h(Y,Z) - h(Z) - h(X,Y,Z)$, where $h(X) = -\int_{\mathcal{X}} p(x) \log p(x)\, dx$ is known as the differential entropy.

The differential entropy estimation problem has been studied extensively by [8] [110] [99] [84] [85] [145] [142] and can be estimated either based on kernel-density [71] [52] or k-nearest-neighbor estimates [146] [70][115] [78][141] [143]. Building on top of k-nearest-neighbor estimates and breaking the paradigm of ΣH estimation, a coupled estimator (which we address henceforth as KSG) was proposed by [79]. It generalizes to mutual information, conditional mutual information as well as for other multivariate information measures, including estimation in scenarios when the distribution can be mixed [128] [49][51] [53][151] [125].

The kNN approach has the advantage that it can naturally adapt to the data density and does not require extensive tuning of kernel band-widths. However, all these approaches suffer from the curse of dimensionality and are unable to scale well with dimensions. Moreover, [50] showed that exponentially many samples are required (as MI grows) for the accurate estimation using kNN based estimators. This brings us to the central motivation of this work : ***Can we propose estimators for conditional mutual information that estimate well even in high dimensions ?***

3.1.2 Our Contribution

In this Chapter, we explore various ways of estimating CMI by leveraging tools from classifiers and generative models. To the best of our knowledge, this is the first work that deviates from the framework of kNN and kernel based CMI estimation and introduces neural networks to solve this problem.

The main contributions of the Chapter can be summarized as follows :

Classifier Based MI Estimation: We propose a novel KL-divergence estimator based on classifier two-sample approach that is more stable and performs superior to the recent neural methods [9].

Divergence Based CMI Estimation: We express CMI as the KL-divergence between two distributions $p_{xyz} = p(z)p(x|z)p(y|x,z)$ and $q_{xyz} = p(z)p(x|z)p(y|z)$, and explore candidate

generators for obtaining samples from $q(\cdot)$. The CMI estimate is then obtained from the divergence estimator.

Difference Based CMI Estimation: Using the improved MI estimates, and the difference relation $I(X;Y|Z) = I(X;YZ) - I(X;Z)$, we show that estimating CMI using a difference of two MI estimates performs best among several other proposed methods such as divergence based CMI estimation and KSG.

Improved Performance in High Dimensions: On both linear and non-linear data-sets, all our estimators perform significantly better than KSG. Surprisingly, our estimators perform well even for dimensions as high as 100, while KSG fails to obtain reasonable estimates even beyond 5 dimensions.

Improved Performance in Conditional Independence Testing: As an application of CMI estimation, we use our best estimator for conditional independence testing (CIT) and obtain improved performance compared to the state-of-the-art CIT tester on both synthetic and real data-sets.

3.2 Estimation of Conditional Mutual Information

The CMI estimation problem from finite samples can be stated as follows. Let us consider three random variables X, Y, $Z \sim p(x, y, z)$, where $p(x, y, z)$ is the joint distribution. Let the dimensions of the random variables be d_x, d_y and d_z respectively. We are given n samples $\{(x_i, y_i, z_i)\}_{i=1}^{n}$ drawn i.i.d from $p(x, y, z)$. So $x_i \in \mathbb{R}^{d_x}, y_i \in \mathbb{R}^{d_y}$ and $z_i \in \mathbb{R}^{d_z}$. The goal is to estimate $I(X;Y|Z)$ from these n samples.

3.2.1 Divergence Based CMI Estimation

Definition 1. *The Kullback-Leibler (KL) divergence between two distributions $p(\cdot)$ and $q(\cdot)$ is given as :*

$$D_{KL}(p||q) = \int p(x) \log \frac{p(x)}{q(x)}\, dx$$

Definition 2. *Conditional Mutual Information (CMI) can be expressed as a KL-divergence*

between two distributions $p(x, y, z)$ *and* $q(x, y, z) = p(x, z)p(y|z)$*, i.e.,*

$$I(X; Y|Z) = D_{KL}(p(x, y, z)||p(x, z)p(y|z))$$

The definition of CMI as a KL-divergence naturally leads to the question : *Can we estimate CMI using an estimator for divergence ?* However, the problem is still non-trivial since we are only given samples from $p(x, y, z)$ and the divergence estimator would also require samples from $p(x, z)p(y|z)$. This further boils down to whether we can learn the distribution $p(y|z)$.

Generative Models

We now explore various techniques to learn the conditional distribution $p(y|z)$ given samples $\sim p(x, y, z)$. This problem is fundamentally different from drawing independent samples from the marginals $p(x)$ and $p(y)$, given the joint $p(x, y)$. In this simpler setting, we can simply permute the data to obtain $\{x_i, y_{\pi(i)}\}_{i=1}^{n}$ (π denotes a permutation, $\pi(i) \neq i$). This would emulate samples drawn from $q(x, y) = p(x)p(y)$. But, such a permutation scheme does not work for $p(x, y, z)$ since it would destroy the dependence between X and Z. The problem is solved using recent advances in generative models which aim to learn an unknown underlying distribution from samples.

Conditional Generative Adversarial Network (CGAN): There exist extensions of the basic GAN framework [56] in conditional settings, CGAN [100]. Once trained, the CGAN can then generate samples from the generator network as $y = \mathcal{G}(s, z), s \sim p(s), z \sim p(z)$.

Conditional Variational Autoencoder (CVAE): Similar to CGAN, the conditional setting, CVAE [75] [144], aims to maximize the conditional log-likelihood. The input to the decoder network is the value of z and the latent vector s sampled from standard Gaussian. The decoder Q gives the conditional mean and conditional variance (parametric functions of s and z) from which y is then sampled.

k**NN based permutation**: A simpler algorithm for generating the conditional $p(y|z)$ is to permute data values where $z_i \approx z_j$. Such methods are popular in conditional independence

testing literature [136] [39]. For a given point $\{x_i, y_i, z_i\}$, we find the k-nearest neighbor of z_i. Let us say it is z_j with the corresponding data point as $\{x_j, y_j, z_j\}$. Then $\{x_i, y_j, z_i\}$ is a sample from $q(x, y, z)$.

Now that we have outlined multiple techniques for estimating $p(y|z)$, we next proceed to the problem of estimating KL-divergence.

Divergence Estimation

Recently, [9] proposed a neural network based estimator of mutual information (MINE) by utilizing lower bounds on KL-divergence. Since MI is a special case of KL-divergence, their neural estimator can be extended for divergence estimation as well. The estimator can be trained using back-propagation and was shown to out-perform traditional methods for MI estimation. The core idea of MINE is cradled in a dual representation of KL-divergence. The two main lower bounds used by MINE are stated below.

Definition 3. *The Donsker-Varadhan representation expresses KL-divergence as a supremum over functions,*

$$D_{KL}(p||q) = \sup_{f \in \mathcal{F}} \mathop{\mathbb{E}}_{x \sim p} [f(x)] - \log(\mathop{\mathbb{E}}_{x \sim q} [\exp(f(x))]) \tag{3.1}$$

where the function class $\mathcal{F}$ includes those functions that lead to finite values of the expectations.

Definition 4. *The f-divergence bound gives a lower bound on the KL-divergence:*

$$D_{KL}(p||q) \geq \sup_{f \in \mathcal{F}} \mathop{\mathbb{E}}_{x \sim p} [f(x)] - \mathop{\mathbb{E}}_{x \sim q} [\exp(f(x) - 1)] \tag{3.2}$$

MINE uses a neural network f_θ to represent the function class $\mathcal{F}$ and uses gradient descent to maximize the RHS in the above bounds.

Even though this framework is flexible and straight-forward to apply, it presents several practical limitations. The estimation is very sensitive to choices of hyper-parameters (hidden-units/layers) and training steps (batch size, learning rate). We found the optimization

process to be unstable and to diverge at high dimensions (Section : Experimental Results). Our findings resonate those by [121] in which the authors found the networks difficult to tune even in toy problems.

3.2.2 Difference Based CMI Estimation

Another seemingly simple approach to estimate CMI could be to express it as a difference of two mutual information terms by invoking the chain rule, i.e.: $I(X;Y|Z) = I(X;Y,Z) - I(X;Z)$. As stated before, since mutual information is a special case of KL-divergence, viz. $I(X;Y) = D_{KL}(p(x,y)||p(x)p(y))$, this again calls for a stable, scalable, sample efficient KL-divergence estimator as we present in the next Section.

3.3 Classifier Based MI Estimation

In their seminal work on independence testing, [95] introduced classifier two-sample test to distinguish between samples coming from two unknown distributions p and q. The idea was also adopted for conditional independence testing by [136]. The basic principle is to train a binary classifier by labeling samples $x \sim p$ as 1 and those coming from $x \sim q$ as 0, and to test the null hypothesis $\mathcal{H}_0 : p = q$. Under the null, the accuracy of the binary classifier will be close to 0.5. It will be away from 0.5 under the alternative. The accuracy of the binary classifier can then be carefully used to define P-values for the test.

We propose to use the classier two-sample principle for estimating the likelihood ratio $\frac{p(x,y)}{p(x)p(y)}$. While existing literature has instances of using the likelihood ratio for MI estimation, the algorithms to estimate the likelihood ratio are quite different from ours. Both [149] [112] formulate the likelihood ratio estimation as a convex relaxation by leveraging the Legendre-Fenchel duality. But performance of the methods depend on the choice of suitable kernels and would suffer from the same disadvantages as mentioned in the Introduction.

3.3.1 Problem Formulation

Given n i.i.d samples $\{x_i^p\}_{i=1}^n, x_i^p \sim p(x)$ and m i.i.d samples $\{x_j^q\}_{j=1}^m, x_j^q \sim q(x)$, we want to estimate $D_{KL}(p||q)$. We label the points drawn from $p(\cdot)$ as $y = 1$ and those from $q(\cdot)$ as $y = 0$. A binary classifier is then trained on this supervised classification task. Let the prediction for a point l by the classifier is γ_l where $\gamma_l = Pr(y = 1|x_l)$ (Pr denotes probability). Then the point-wise likelihood ratio for data point l is given by $\mathcal{L}(x_l) = \frac{\gamma_l}{1-\gamma_l}$.

The following Proposition is elementary and has already been observed in [9](Proof of Theorem 4). We restate it here for completeness and quick reference.

Proposition 1. *The optimal function in Donsker-Varadhan representation* (3.1) *is the one that computes the point-wise log-likelihood ratio, i.e,* $f^*(x) = \log \frac{p(x)}{q(x)} \, \forall \, x$, *(assuming* $p(x) = 0$, *where-ever* $q(x) = 0$*).*

Based on Proposition 1, the next step is to substitute the estimates of point-wise likelihood ratio in (3.1) to obtain an estimate of KL-divergence.

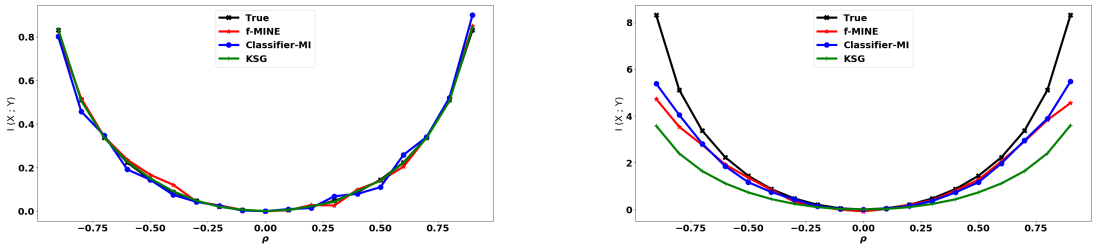
$$\hat{D}_{KL}(p||q) = \frac{1}{n} \sum_{i=1}^{n} \log \mathcal{L}(x_i^p) - \log \left(\frac{1}{m} \sum_{j=1}^{m} \mathcal{L}(x_j^q) \right) \tag{3.3}$$

We obtain an estimate of mutual information from (3.3) as

$$\hat{I}_n(X;Y) = \hat{D}_{KL}(p(x,y)||p(x)p(y))$$

3.3.2 Probability Calibration

The estimation of likelihood ratio from classifier predictions $Pr(y = 1|x)$ hinges on the fact that the classifier is well-calibrated. As a rule of thumb, classifiers trained directly on the cross entropy loss are well-calibrated. But boosted decision trees would introduce distortions in the likelihood-ratio estimates. There is an extensive literature devoted to obtaining better calibrated classifiers that can be used to improve the estimation further [80] [113, 59]. We experimented with Gradient Boosted Decision Trees and multi-layer perceptron trained on



(a) $d_x = d_y = 1$ (b) $d_x = d_y = 10$

Figure 3.2: Mutual Information Estimation of Correlated Gaussians : In this setting, X and Y have independent co-ordinates, with (X_i, Y_i) $\forall\, i$ being correlated Gaussians with correlation coefficient ρ. $I^*(X;Y) = -\frac{1}{2} d_x \log(1-\rho^2)$

the log-loss in our algorithms. Multi-layer perceptron gave better estimates and so is used in all the experiments.

Even though logistic regression is well-calibrated and might seem to be an attractive candidate for classification in sparse sample regimes, we show that linear classifiers cannot be used to estimate D_{KL} by two-sample approach. For this, we consider the simple setting of estimating mutual information of two correlated Gaussian random variables as a counter-example.

Lemma 2. *A linear classifier with marginal features fails the classifier Two sample MI estimation.*

Proof. Consider two correlated Gaussians in 2 dimensions $(X_1, X_2) \sim \mathcal{N}\left(0, M = \left(\begin{smallmatrix} 1 & \rho \\ \rho & 1 \end{smallmatrix}\right)\right)$, where ρ is the Pearson correlation. The marginals are standard Gaussans $X_i \sim \mathcal{N}(0,1)$. Suppose we are trying to estimate the mutual information $D_{KL}(p(x_1,x_2)||p(x_1)p(x_2))$. The classifier decision boundary would seek to find $Pr(y=1|x_1,x_2) > Pr(y=0|x_1,x_2)$, thus $p(x_1,x_2) > p(x_1)p(x_2) => x_1 x_2 > \frac{1}{2\rho}\log(1-\rho^2)$ □

The decision boundary is a rectangular hyperbola. Here the classifier would return 0.5 as

prediction for either class (leading to $\hat{D}_{KL} = 0$), even when X_1 and X_2 are highly correlated and the mutual information is high.

We use the Classifier two-sample estimator to first compute the mutual information of two correlated Gaussians [9]. This setting also provides us a way to choose reasonable hyper-parameters that are used throughout in all the synthetic experiments. We also plot the estimates of f-MINE and KSG to ensure we are able to make them work in simple settings. In the toy setting $d_x = 1$, all estimators accurately estimate $I(X;Y)$ as shown in Figure 3.2.

3.3.3 Modular Approach to CMI Estimation

Our classifier based divergence estimator does not encounter an optimization problem involving exponentials. MINE optimizing (3.1) has biased gradients while that based on (3.2) is a weaker lower bound [9]. On the contrary, our classifier is trained on cross-entropy loss which has unbiased gradients. Furthermore, we plug in the likelihood ratio estimates into the tighter Donsker-Varadhan bound, thereby, achieving the best of both worlds. Equipped with a KL-divergence estimator, we can now couple it with the generators or use the expression of CMI as a difference of two MIs (which we address from now as MI-Diff.). Algorithm 3 describes the CMI estimation by tying together the generator and Classifier block. For MI-Diff., function block "Classifier-D_{KL}" in Algorithm 3 has to be used twice : once for estimating $I(X;Y,Z)$ and another for $I(X;Z)$. For mutual information, $\mathcal{D}_q$ in "Classifier-D_{KL}" is obtained by permuting the samples of $p(\cdot)$.

For the Classifier coupled with a generator, the generated distribution $g(y|z)$ may deviate from the target distribution $p(y|z)$ - introducing a different kind of bias. The following Lemma suggests how such a bias can be corrected by subtracting the KL divergence of the sub-tuple (Y,Z) from the divergence of the entire triple (X,Y,Z). We note that such a clean relationship is not true for general divergence measures, and indeed require more sophisticated conditions for the total-variation metric [135].

Lemma 3 (Bias Cancellation)**.** *The estimation error due to incorrect generated distribution $g(y|z)$ can be accounted for using the following relation :*

Input: Dataset $\mathcal{D} = \{x_i, y_i, z_i\}_{i=1}^{n}$, number of outer boot-strap iterations B, Inner iterations T, clipping constant τ.

Output: CMI estimatate $\hat{I}(X;Y|Z)$

for $b \in \{1, 2, \dots B\}$ **do**

Permute the points in dataset $\mathcal{D}$ to obtain $\mathcal{D}^{\pi}$.

Split $\mathcal{D}^{\pi}$ equally into two parts $\mathcal{D}_{\text{class,joint}} = \{x_i, y_i, z_i\}_{i=1}^{n/2}$ and $\mathcal{D}_{\text{gen}} = \{x_i, y_i, z_i\}_{i=n/2}^{n}$.

Train the generator $\mathcal{G}(\cdot)$ on $\mathcal{D}_{\text{gen}}$.

Generate the marginal data-set using points $y_i' = \mathcal{G}(z_i) \,\forall\, z_i \in \mathcal{D}_{\text{class,joint}}(:, Z)$.

$\mathcal{D}_{\text{class,marg}} = \{x_i, y_i', z_i\}_{i=1}^{n/2}$

$\hat{I}_b(X;Y|Z) = \text{Classifier\_D}_{\text{KL}}(\mathcal{D}_{\text{class,joint}}, \mathcal{D}_{\text{class,marg}}, T, \tau)$

end

return $\frac{1}{B}\sum_b \hat{I}_b(X;Y|Z)$

Function $\text{Classifier\_D}_{\text{KL}}(\mathcal{D}_p, \mathcal{D}_q, T, \tau)$**:**

Label points $u \in \mathcal{D}_p$ as $l = 1$ and $v \in \mathcal{D}_q$ as $l = 0$.

for $t \in \{1, 2, \dots T\}$ **do**

$\mathcal{D}_p^{\text{train}}, \mathcal{D}_p^{\text{eval}} \leftarrow \text{SPLIT\_TEST\_TRAIN}(\mathcal{D}_p)$.

$\mathcal{D}_q^{\text{train}}, \mathcal{D}_q^{\text{eval}} \leftarrow \text{SPLIT\_TEST\_TRAIN}(\mathcal{D}_q)$

Train classifier $\mathcal{C}$ on $\{\mathcal{D}_p^{\text{train}}, \vec{1}\}, \{\mathcal{D}_q^{\text{train}}, \vec{0}\}$

Obtain classifier predictions $Pr(l = 1|w) \,\forall\, w \in \mathcal{D}_p^{\text{eval}} \cup \mathcal{D}_q^{\text{eval}}$, and clip to $[\tau, 1-\tau]$.

$$\hat{D}_{KL}^{t}(p||q) \leftarrow \frac{1}{|\mathcal{D}_p^{\text{eval}}|} \sum_{u \in \mathcal{D}_p^{\text{eval}}} \log \frac{Pr(l=1|u)}{1-Pr(l=1|u)} - \log \left(\frac{1}{|\mathcal{D}_q^{\text{eval}}|} \sum_{v \in \mathcal{D}_q^{\text{eval}}} \exp \log \frac{Pr(l=1|v)}{1-Pr(l=1|v)} \right)$$

end

return $\hat{D}_{KL}(p||q) = \frac{1}{\tau}\sum_t \hat{D}_{KL}^{t}(p||q)$

Algorithm 3: Generator + Classifier

$$D_{KL}(p(x,y,z)||p(x,z)p(y|z)) = D_{KL}(p(x,y,z)||p(x,z)g(y|z)) - D_{KL}(p(y,z)||p(z)g(y|z))$$

3.4 Experimental Results

In this Section, we compare the performance of various estimators on the CMI estimation task. We used the Classifier based divergence estimator and MINE in our experiments. [9] had two MINE variants, namely Donsker-varadhan (DV) MINE and f-MINE. The f-MINE has unbiased gradients and we found it to have similar performance as DV-MINE, albeit with lower variance. So we used f-MINE in all our experiments.

The "Generator"+"Divergence estimator" notation will be used to denote the various estimators. For instance, if we use CVAE for the generation and couple it with f-MINE, we denote the estimator as CVAE+f-MINE. When coupled with the Classifier based Divergence block, it will be denoted as CVAE+Classifier. For MI-Diff. we represent it similarly as MI-Diff.+"Divergence estimator".

We compare our estimators with the widely used KSG estimator.[2] For f-MINE, we used the code provided to us by the author [9]. The same hyper-parameter setting is used in all our synthetic data-sets for all estimators (including generators and divergence blocks). For KSG, we vary $k \in \{3, 5, 10\}$ and report the results for the best k for each data-set.

3.4.1 Linear Relations

We start with the simple setting where the three random variables X, Y, Z are related in a linear fashion. We consider the two linear models in Table 3.1, where $\mathcal{U}(-0.5, 0.5)^{d_z}$ means that each co-ordinate of Z is drawn i.i.d from a uniform distribution between -0.5 and 0.5. Similar notation is used for the Gaussian : $\mathcal{N}(0,1)^{d_z}$. Z_1 is the first dimension of Z. We

[2]The implementation of CMI estimator in Non-parametric Entropy Estimation Toolbox (`https://github.com/gregversteeg/NPEET`) is used.

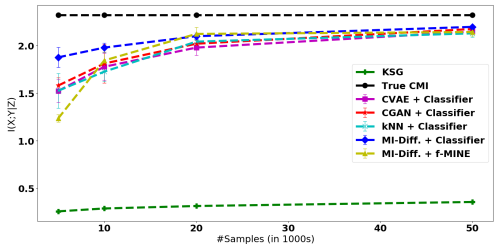
Table 3.1: Linear Models

Model I	**Model II**
$X \sim \mathcal{N}(0,1)$	$X \sim \mathcal{N}(0,1)$
$Z \sim \mathcal{U}(-0.5, 0.5)^{d_z}$	$Z \sim \mathcal{N}(0,1)^{d_z}$
	$U = w^T Z, \|w\|_1 = 1$
$\epsilon \sim \mathcal{N}(Z_1, \sigma_\epsilon^2)$	$\epsilon \sim \mathcal{N}(U, \sigma_\epsilon^2)$
$Y \sim X + \epsilon$	$Y \sim X + \epsilon$

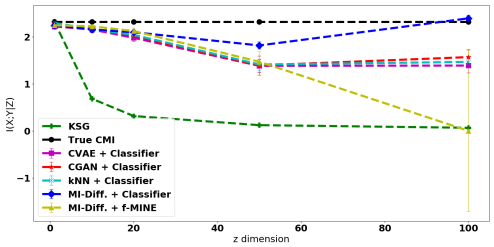
used $\sigma_\epsilon = 0.1$ and obtained the constant unit norm random vector w from $\mathcal{N}(0, I_{d_z})$. w is kept constant for all points during data-set preparation.

As common in literature on causal discovery and independence testing [136] [39], the dimension of X and Y is kept as 1, while d_z can scale. Our estimators are general enough to accommodate multi-dimensional X and Y, where we consider a concatenated vector $X = (X_1, X_2, \ldots, X_{d_x})$ and $Y = (Y_1, Y_2, \ldots, Y_{d_y})$. This has applications in learning interactions between Modules in Bayesian networks [134] or dependence between group variables [45] [117] such as distinct functional groups of proteins/genes instead of individual entities. Both the linear models are representative of problems encountered in Graphical models and independence testing literature. In Model I, the conditioning set can go on increasing with independent variables $\{Z_k\}_{k=2}^{d_z}$, while Y only depends on Z_1. In Model II, we have the variables in the conditioning set combining linearly to produce Y. It is also easy to obtain the ground truth CMI value in such models by numerical integration.

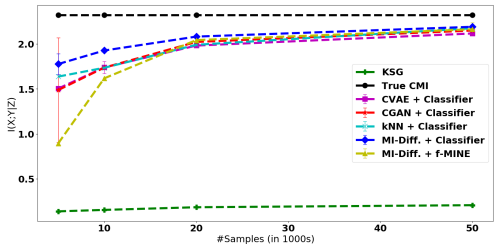
For both these models, we generate data-sets with varying number of samples n and varying dimension d_z to study their effect on estimator performance. The sample size is varied as $n \in \{5000, 10000, 20000, 50000\}$ keeping d_z fixed at 20. We also vary $d_z \in \{1, 10, 20, 50, 100\}$,



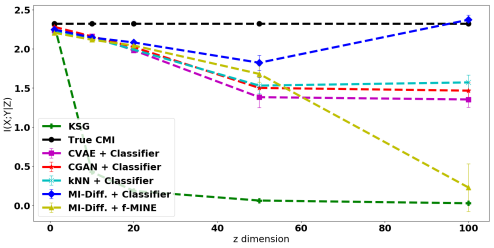
(a) Model I : Variation with n, $d_z = 20$



(b) Model I : Variation with d_z, $n = 20,000$



(c) Model II : Variation with n, $d_z = 20$



(d) Model II : Variation with d_z, $n = 20,000$

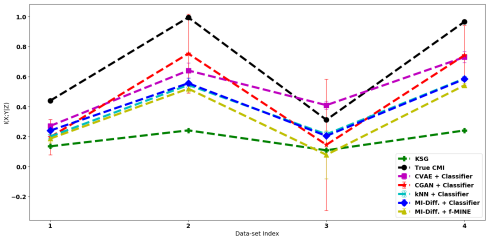
Figure 3.3: CMI Estimation in Linear models : We study the effect of various estimators as either number of samples n or dimension d_z is varied. MI-Diff.+Classifier performs the best among our estimators, while all our proposed estimators improve the estimation significantly over KSG. Average of 10 runs is plotted. Error bars depict 1 standard deviation from mean. (Best viewed in color)

keeping sample size fixed at $n = 20000$.

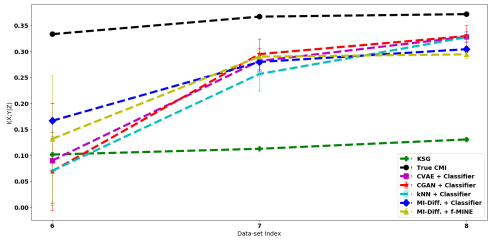
Several observations stand out from the experiments: (1) KSG estimates are accurate at very low dimension but drastically fall with increasing d_z even when the conditioning variables are completely independent and do not influence X and Y (Model-I). (2) Increasing the sample size does not improve KSG estimates once the dimension is kept moderate (even 20!). The dimension issue is more acute than sample scarcity. (3) The estimates from f-MINE have greater deviation from the truth at low sample sizes. At high dimensions, the

instability is clearly portrayed when the estimate suddenly goes negative (Truncated to 0.0 to maintain the scale of the plot). (4) All our estimators using Classifier are able to obtain reasonable estimates even at dimensions as high as 100, with MI-Diff.+Classifier performing the best.

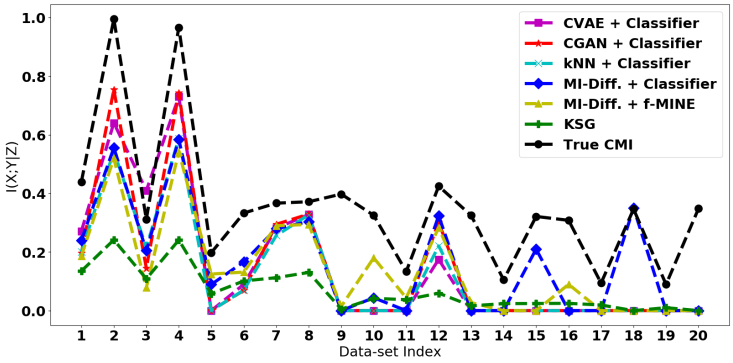
3.4.2 Non-Linear Relations



(a) Non-linear Model : Number of samples increase with Data-index, $d_z = 10$ (fixed)



(b) Non-linear Model : Number of samples increase with Data-index, $d_z = 20$ (fixed)



(c) Non-linear Models (All 20 data-sets)

Figure 3.4: On non-linear data-sets, a similar trend is observed. KSG under-estimates $I^*(X;Y|Z)$, while our estimators track it closely. Average over 10 runs is plotted. (Best viewed in color)

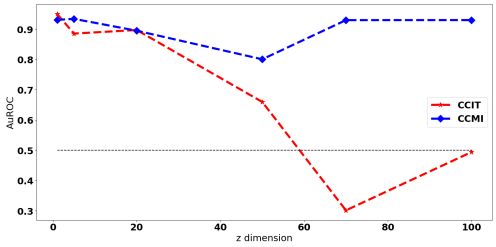
Here, we study models where the underlying relations between X, Y and Z are non-linear. Let $Z \sim \mathcal{N}(\mathbb{1}, I_{d_z}), X = f_1(\eta_1), Y = f_2(A_{zy}Z + A_{xy}X + \eta_2)$. f_1 and f_2 are non-linear bounded functions drawn uniformly at random from $\{cos(\cdot), tanh(\cdot), \exp(-|\cdot|)\}$ for each data-set. A_{zy} is a random vector whose entries are drawn $\mathcal{N}(0,1)$ and normalized to have unit norm. The vector once generated is kept fixed for a particular data-set. We have the setting where $d_x = d_y = 1$ and d_z can scale. A_{xy} is then a constant. We used $A_{xy} = 2$ in our simulations. The noise variables η_1, η_2 are drawn i.i.d $\mathcal{N}(0, \sigma_\epsilon^2)$, $\sigma_\epsilon^2 = 0.1$.

We vary $n \in \{5000, 10000, 20000, 50000\}$ across each dimension d_z. The dimension d_z itself is then varied as $\{10, 20, 50, 100, 200\}$ giving rise to 20 data-sets. Data-index 1 has $n = 5000, d_z = 10$, data-index 2 has $n = 10000, d_z = 10$ and so on until data-index 20 with $n = 50000, d_z = 200$.

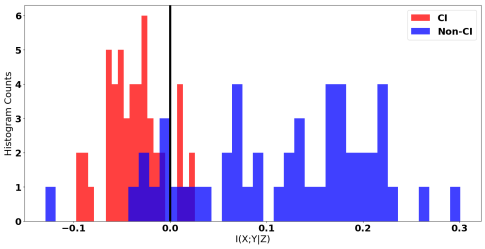
Obtaining Ground Truth $I^*(X;Y|Z)$: Since it is not possible to obtain the ground truth CMI value in such complicated settings using a closed form expression, we resort to using the relation $I(X;Y|Z) = I(X;Y|U)$ where $U = A_{zy}Z$. The dependence of Y on Z can be completely captured once U is given. But, U has dimension 1 and can be estimated accurately using KSG. We generate 50000 samples separately for each data-set to estimate $I(X;Y|U)$ and use it as the ground truth. We observed similar behavior (as in Linear models) for our estimators in the Non-linear setting.

(1) KSG continues to have low estimates even though in this setup the true CMI values are themselves low (< 1.0). (2) Up to $d_z = 20$, we find all our estimators closely tracking $I^*(X;Y|Z)$. But in higher dimensions, they fail to perform accurately. (3) MI-Diff. + Classifier is again the best estimator. (It is able to recover the true CMI value for data-index 18 where $d_z = 200$ and $n = 10000$).

From the above experiments, we found MI-Diff.+Classifier to be the most accurate and stable estimator. We use this combination for our downstream applications and henceforth refer to it as CCMI.



(a) CCIT performance degrades with increasing dz; CCMI retains high AuROC score even at dz = 100.



(b) Estimates for CI data-sets are ≤ 0 and those for non-CI are > 0 at $d_z = 100$. Thresholding CMI estimates at 0 yields Precision = 0.86, Recall = 0.84.

Figure 3.5: Conditional Independence Testing in Post Non-linear Synthetic Data-set

3.5 Application to Conditional Independence Testing

As a testimony to accurate CMI estimation, we apply CCMI to the problem of Conditional Independence Testing(CIT). Here, we are given samples from two distributions $p(x, y, z)$ and $q(x, y, z) = p(x, z)p(y|z)$. The hypothesis testing in CIT is to distinguish the null $\mathcal{H}_0 : X \perp Y|Z$ from the alternative $\mathcal{H}_1 : X \not\perp Y|Z$.

We seek to design a CIT tester using CMI estimation by using the fact that $I(X;Y|Z) = 0 \iff X \perp Y|Z$. A simple approach would be to reject the null if $I(X;Y|Z) > 0$ and accept it otherwise. The CMI estimates can serve as a proxy for the P-value. CIT testing based on CMI Estimation has been studied by [128], where the author uses KSG for CMI estimation and use k-NN based permutation to generate a P-value. The P-value is computed as the fraction of permuted data-sets where the CMI estimate is $\geq$ that of the original data-set. The same approach can be adopted for CCMI to obtain a P-value. But since we report the AuROC (Area under the Receiver Operating Characteristic curve), CMI estimates suffice.

3.5.1 Post Non-linear Noise : Synthetic Data

In this experiment, we generate data based on the post non-linear noise model similar to [136]. As before, $d_x = d_y = 1$ and d_z can scale in dimension. The data is generated using the follow model.

$$Z \sim \mathcal{N}(\mathbb{1}, I_{d_z}),\ X = cos(a_x Z + \eta_1)$$
$$Y = \begin{cases} cos(b_y Z + \eta_2) & \text{if} X \not\perp Y | Z \\ cos(cX + b_y Z + \eta_2) & \text{if } X \perp Y | Z \end{cases}$$

The entries of random vectors(matrices if $d_x, d_y > 1$) a_x and b_y are drawn $\sim \mathcal{U}(0, 1)$ and the vectors are normalized to have unit norm, i.e., $\|a\|_2 = 1, \|b\|_2 = 1$. $c \sim \mathcal{U}[0, 2]$. This is different from the implementation in [136] where the constant is $c = 2$ in all data-sets. But by varying c, we obtain a tougher problem where the true CMI value can be quite low for a dependent data-set and the tester is required to separate it correctly from an independent data-set.

a_x, b_y and c are kept constant for generating points for a single data-set and are varied across data-sets. We vary $d_z \in \{1, 5, 20, 50, 70, 100\}$ and simulate 100 data-sets for each dimension. The number of samples is $n = 5000$ in each data-set. Our algorithm is compared with the state-of-the-art CIT tester in [136], known as CCIT. We used the implementation provided by the authors and ran CCIT with $B = 50$ bootstraps [3]. For each data-set, an AuROC value is obtained. Figure 3.5 shows the mean AuROC values from 5 runs for both the testers as d_z varies. While both algorithms perform accurately upto $d_z = 20$, the performance of CCIT starts to degrade beyond 20 dimensions. Beyond 50 dimensions, it performs close to random guessing. CCMI retains its superior performance even at $d_z = 100$, obtaining a mean AuROC value of 0.93.

Since AuROC metric finds best performance by varying thresholds, it is not clear what precision and recall is obtained from CCMI when we threshold the CCMI estimate at 0

[3] `https://github.com/rajatsen91/CCIT`

(and reject or accept the null based on it). So, for $d_z = 100$ we plotted the histogram of CMI estimates separately for CI and non-CI data-sets. Figure 3.5b shows that there a clear demarcation of CMI estimates between the two data-set categories and choosing the threshold as 0.0 gave the precision as 0.86 and recall as 0.84.

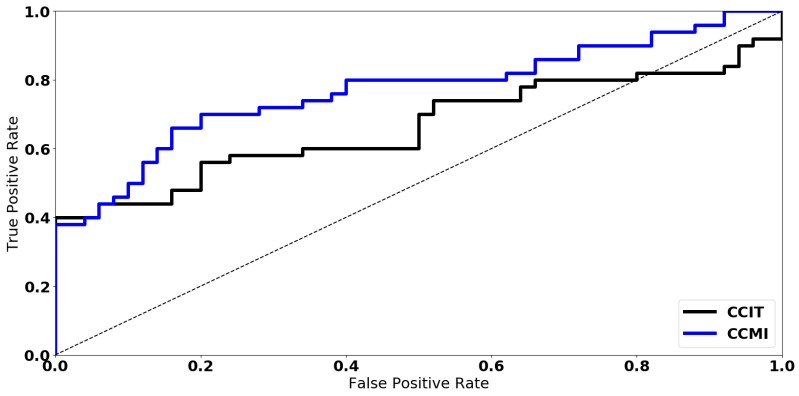


Figure 3.6: AuROC Curves : Flow-Cytometry Data-set. CCIT obtains a mean AuROC score of 0.6665, while CCMI out-performs with mean of 0.7569.

3.5.2 Flow Cytometry : Real Data

Flow Cytometry is a methodology to obtain the expression levels of proteins. The expressions of multiple proteins are obtained with different interventions and stimulatory signals. Such a multivariate data can be then used to build a Bayesian network by answering questions such as : *Is Protein A independent of Protein B given Protein C?*

To extend our estimator beyond simulated settings, we use CMI estimation to test for conditional independence in the protein network data used in [136]. The dataset consists of flow cytometry measurements of 11 phosphorylated proteins and phospholipids. The consensus graph in [129] is used as the ground truth. We obtained 50 CI and 50 non-CI relations from the Bayesian network. The basic philosophy used is that a protein X is

independent of all other proteins Y in the network given its parents, children and parents of children. Moreover, in the case of non-CI, we notice that a direct edge between X and Y would never render them conditionally independent. So the conditioning set Z can be chosen at random from other proteins. These two settings are used to obtain the CI and non-CI data-sets. The number of samples in each data-set is only 853 and the dimension of Z varies from 5 to 7.

Until now we had used the same hyper-parameter for CCMI across all data-sets, sample sizes and dimensions. For Flow-Cytometry data, we reduce the number of hidden units of the Classifier, keeping every other hyper-parameter the same (since the number of samples is too small). CCMI is compared with CCIT on the real data and the mean AuROC curves from 5 runs is plotted in Figure 3.6. The superior performance of CCMI over CCIT is retained in sparse data regime.

3.6 Additional Results

3.6.1 Calibration Curve

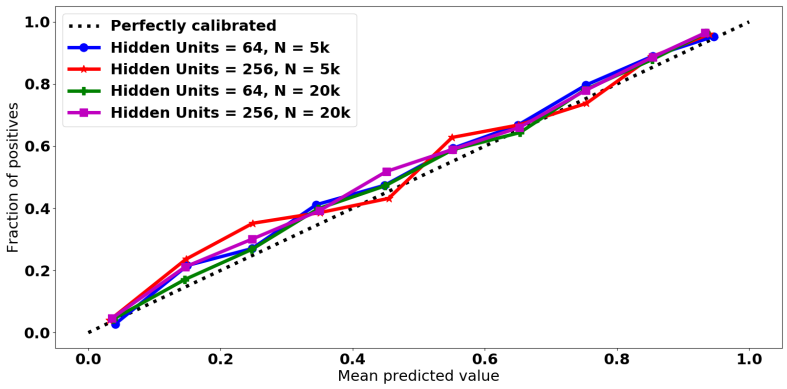
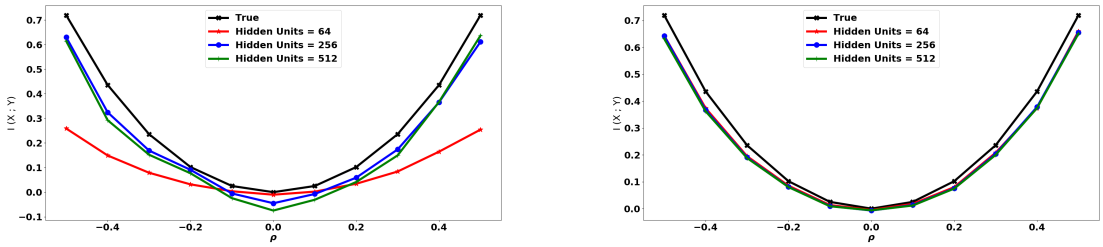
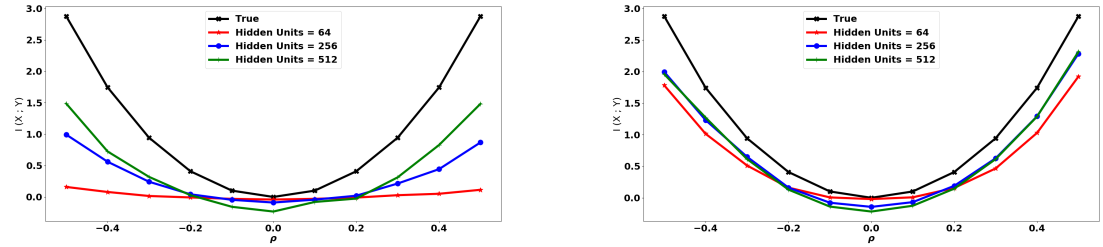


Figure 3.7: Calibrated Classifiers : We find that our classifiers trained with $L2$-regularization and two hidden layers are well-calibrated. The calibration is obtained for MI Estimation of Correlated Gaussians with $d_x = 10, \rho = 0.5$



(a) $d_x = d_y = 5, N = 500$,

(b) $d_x = d_y = 5, N = 5000$



(c) $d_x = d_y = 20, N = 500$

(d) $d_x = d_y = 20, N = 5000$

Figure 3.8: The Donsker-Varadhan Representation provides a lower bound of the true MI. For each hyper-paramter choice, the estimates lie below $I^*(X;Y)$. An optimal estimator would return the maximum estimate from multiple hyper-parameter choices for a given data-set. Estimates are plotted for Correlation Gaussians introduced in Figure 3.2.

While [113] showed that neural networks for binary classification produce well-calibrated outputs. the authors in [59] found miscalibration in deep networks with batch-normalization and no L2 regularization. In our experiments, the classifier is shallow, consisting of only 2 layers with relatively small number of hidden units. There is no batch-normalization or dropout used. Instead, we use $L2$-regularization which was shown in [59] to be favorable for calibration. Figure 3.7 shows that our classifiers are well-calibrated.

3.6.2 Choosing Optimal Hyper-parameter

The Donsker-Varadhan representation 3.1 is a lower bound on the true MI estimate (which is the supremum over all functions). So, for any classifier parameter, the plug-in estimate value computed on the test samples will be less than or equal to the true value $I(X;Y)$ with high probability (Theorem 2). We illustrate this using estimation of MI for Correlated Gaussians in Figure 3.8. The estimated value lies below the true values of MI. Thus, the optimal hyper-parameter is the one that returns the maximum value of MI estimate on a validation set.

Once we have this block that returns the maximum MI estimate after searching over hyper-parameters, CMI estimate in CCMI is the difference of two MI estimates, calling this block twice.

3.7 Theoretical Properties of CCMI

In this Section, we explore some of the theoretical properties of CCMI. Let the samples $x_i \sim p(x)$ be labeled as $l = 1$ and $x_j \sim q(x)$ be labeled as $l = 0$. Let $Pr(l = 1) = Pr(l = 0) = 0.5$. The positive label probability for a given point x is denoted as $\gamma(x) = Pr(l = 1|x)$. When the prediction is from a classifier with parameter θ, then it is denoted as $\gamma_\theta(x)$. The argument x of γ is dropped when it is understood from the context.

The following assumptions are used throughout this Section.

- Assumption (A1) : The underlying data distributions $p(\cdot)$ and $q(\cdot)$ admit densities in a compact subset $\mathcal{X} \subset \mathbb{R}^{d_x}$.

- Assumption (A2) : $\exists\ \alpha, \beta > 0$, such that $\alpha \leq p(x), q(x) \leq \beta\ \forall\, x$.

- Assumption (A3) : We clip predictions in algorithm such that $\gamma(x) \in [\tau, 1-\tau]\, \forall\, x$, with $0 < \tau \leq \alpha/(\alpha+\beta)$.

- Assumption (A4) : The classifier class $\mathcal{C}_\theta$ is parameterized by θ in some compact

domain $\Theta \subset \mathbb{R}^h$. $\exists$ constant K, such that $\|\theta\| \leq K$ and the output of the classifier is L-Lipschitz with respect to parameters θ.

Notation and Computation Procedure

- In the case of mutual information estimation $I(U;V)$, $x \in \mathbb{R}^{d_u+d_v}$ represents the concatenated data point (u, v). To be precise, $p(x) = p(u, v)$ and $q(x) = p(u)p(v)$.

- In the proofs below, we need to compute the Lipschitz constant for various functions. The general procedure for those computations are as follows.

$$|\phi(x) - \phi(y)| \leq L_\phi |x - y|$$

We compute L_ϕ using $\sup_z |\phi'(z)|, z \in \text{domain}(\phi)$. The functions encountered in the proofs are continuous, differentiable and have bounded domains.

- The binary-cross entropy loss estimated from n samples is

$$\text{BCE}_n(\gamma) = -\left(\frac{1}{n}\sum_i l_i \log \gamma(x_i) + (1 - l_i)\log(1 - \gamma(x_i))\right) \tag{3.4}$$

When computed on the train samples (resp. test samples), it is denoted as $\text{BCE}_n^{\text{ERM}}(\gamma)$ (resp. $\text{BCE}_n(\gamma)$). The population mean over the joint distribution of data and labels is

$$\text{BCE}(\gamma) = -\left(\mathbb{E}_{XL} L \log \gamma(X) + (1 - L)\log(1 - \gamma(X))\right) \tag{3.5}$$

- The estimate of MI from n test samples for classifier parameter $\hat{\theta}$ is given by

$$I_n^{\gamma_{\hat{\theta}}} = \frac{1}{n}\sum_{i=1}^{n} \log \frac{\gamma_{\hat{\theta}(x_i)}}{1 - \gamma_{\hat{\theta}(x_i)}} - \log\left(\frac{1}{n}\sum_{j=1}^{n} \frac{\gamma_{\hat{\theta}}(x_j)}{1 - \gamma_{\hat{\theta}}(x_j)}\right)$$

The population estimate for classifier parameter $\hat{\theta}$ is given by

$$I^{\gamma_{\hat{\theta}}} = \mathop{\mathbb{E}}_{x\sim p} \log \frac{\gamma_{\hat{\theta}(x)}}{1 - \gamma_{\hat{\theta}(x)}} - \log\left(\mathop{\mathbb{E}}_{x\sim q} \frac{\gamma_{\hat{\theta}}(x)}{1 - \gamma_{\hat{\theta}}(x)}\right)$$

Theorem 1. *Classifier-MI is consistent, i.e., given* $\epsilon, \delta > 0, \exists\, n \in \mathbb{N}$*, such that with probability at least* $1 - \delta$*, we have*

$$|I_n^{\gamma_{\hat{\theta}}}(U;V) - I(U;V)| \leq \epsilon$$

Intuition of Proof

The classifier is trained to minimize the empirical risk on the train set and obtains the minimizer as $\hat{\theta}$. From generalization bound of classifier, this loss value ($\text{BCE}(\gamma_{\hat{\theta}})$) on the test set is close to the loss obtained by the best optimizer in the classifier family ($\text{BCE}(\gamma_{\tilde{\theta}})$), which itself is close to the loss from global optimizer γ^* (viz. $\text{BCE}(\gamma^*)$) by Universal Function Approximation Theorem of neural-networks.

The BCE loss is strongly convex in γ. γ links BCE to $I(\cdot\,;\,\cdot)$, i.e., $|\text{BCE}_n(\gamma_{\hat{\theta}}) - \text{BCE}(\gamma^*)| \leq \epsilon' \implies \|\gamma_{\hat{\theta}} - \gamma^*\|_1 \leq \eta \implies |\hat{I}_n(U;V) - I(U;V)| \leq \epsilon$.

Lemma 4 (Likelihood-Ratio from Cross-Entropy Loss)**.** *The point-wise minimizer of binary cross-entropy loss* $\gamma^*(x)$ *is related to the likelihood ratio as* $\frac{\gamma^*(x)}{1-\gamma^*(x)} = \frac{p(x)}{q(x)}$*, where* $\gamma^*(x) = Pr(l = 1|x)$ *and* l *is the label of point* x*.*

Proof. The binary cross entropy loss as a function of gamma is defined in (3.5). Now,

$$\begin{aligned}\mathbb{E}_{XL} L \log \gamma(X) &= \sum_{x,l} p(x,l) l \log \gamma(x) = \sum_{x,l=1} p(x|l=1)p(l=1)\log\gamma(x) + 0 \\ &= \frac{1}{2}\sum_x p(x)\log\gamma(x)\end{aligned}$$

Similarly,

$$\mathbb{E}_{XL}(1-L)\log(1-\gamma(X)) = \frac{1}{2}\sum_x q(x)\log(1-\gamma(x))$$

Using these in the expression for $\text{BCE}(\gamma)$, we obtain

$$\text{BCE}(\gamma) = -\frac{1}{2}\left(\sum_{x\in\mathcal{X}} p(x)\log\gamma(x) + q(x)\log(1-\gamma(x))\right)$$

The point-wise minimizer γ^* of $\text{BCE}(\gamma)$ gives $\frac{\gamma^*(x)}{1-\gamma^*(x)} = \frac{p(x)}{q(x)}$. □

Lemma 5 (Function Approximation)**.** *Given* $\epsilon' > 0$, $\exists\, \tilde{\theta} \in \Theta$ *such that*

$$\mathrm{BCE}(\gamma_{\tilde{\theta}}) \leq \mathrm{BCE}(\gamma^*) + \frac{\epsilon'}{2}$$

Proof. The last layer of the neural network being sigmoid (followed by clipping to $[\tau, 1-\tau]$) ensures that the outputs are bounded. So by the Universal Function Approximation Theorem for multi-layer feed-forward neural networks [65], $\exists$ parameter $\tilde{\theta}$ such that $|\gamma^* - \gamma_{\tilde{\theta}}| \leq \epsilon'' \,\forall\, x$, where $\gamma_{\tilde{\theta}}$ is the estimated classifier prediction function with parameter $\tilde{\theta}$. So,

$$|\mathrm{BCE}(\gamma_{\tilde{\theta}}) - \mathrm{BCE}(\gamma^*)| \leq \frac{1}{\tau}\epsilon''$$

since log is Lipshitz continuous with constant $\frac{1}{\tau}$. Choose $\epsilon'' = \frac{\epsilon'\tau}{2}$ to complete the proof.

□

Lemma 6 (Generalization)**.** *Given* $\epsilon', \delta > 0$, $\forall\, n \geq \frac{18M^2}{\epsilon'^2}(h\log(96KL\sqrt{d}/\epsilon') + \log(2/\delta))$, *such that with probability at least* $1-\delta$, *we have*

$$\mathrm{BCE}_n(\gamma_{\hat{\theta}}) \leq \mathrm{BCE}(\gamma_{\tilde{\theta}}) + \frac{\epsilon'}{2}$$

Proof. Let $\hat{\theta} \leftarrow \arg\min_{\theta} \mathrm{BCE}_n^{\mathrm{ERM}}(\gamma_\theta)$.

From Hoeffding's inequality,

$$Pr\left(|\mathrm{BCE}_n^{\mathrm{ERM}}(\gamma_\theta) - \mathrm{BCE}(\gamma_\theta)| \geq \mu\right) \leq 2\exp\left(\frac{-2n\mu^2}{M^2}\right)$$

where $M = \log\left(\frac{1-\tau}{\tau}\right)$.

Similarly, for the test samples,

$$Pr\left(|\mathrm{BCE}_n(\gamma_\theta) - \mathrm{BCE}(\gamma_\theta)| \geq \mu\right) \leq 2\exp\left(\frac{-2n\mu^2}{M^2}\right) \tag{3.6}$$

We want this to hold for all parameters $\theta \in \Theta$. This is obtained using the covering number of the compact domain $\Theta \subset \mathbb{R}^h$. We use small balls $B_r(\theta_j)$ of radius r centered at θ_j

so that $\Theta \subset \cup_j B_r(\theta_j)$ The covering number $\kappa(\Theta, r)$ is finite as Θ is compact and is bounded as

$$\kappa(\Theta, r) \leq \left(\frac{2K\sqrt{h}}{r}\right)^h$$

Using the union bound on these finite hypotheses,

$$Pr\left(\max_\theta |\mathrm{BCE}_n^{\mathrm{ERM}}(\gamma_\theta) - \mathrm{BCE}(\gamma_\theta)| \geq \mu\right) \leq 2\kappa(\Theta, r)\exp\left(\frac{-2n\mu^2}{M^2}\right) \tag{3.7}$$

Choose $r = \frac{\mu}{8L}$ [102]. Solving for number of samples n with $2\kappa(\Theta, r)\exp\left(\frac{-2n\mu^2}{M^2}\right) \leq \delta$, we obtain $n \geq \frac{M^2}{2\mu^2}(h\log(16KL\sqrt{d}/\mu) + \log(2/\delta))$.

So for $n \geq \frac{M^2}{2\mu^2}(h\log(16KL\sqrt{d}/\mu) + \log(2/\delta))$, with probability at least $1-\delta$,

$$\begin{aligned}\mathrm{BCE}_n(\gamma_{\hat{\theta}}) &\overset{(a)}{\leq} \mathrm{BCE}(\gamma_{\hat{\theta}}) + \mu \overset{(b)}{\leq} \mathrm{BCE}_n^{\mathrm{ERM}}(\gamma_{\hat{\theta}}) + 2\mu \\ &\overset{(c)}{\leq} \mathrm{BCE}_n^{\mathrm{ERM}}(\gamma_{\tilde{\theta}}) + 2\mu \overset{(d)}{\leq} \mathrm{BCE}(\gamma_{\tilde{\theta}}) + 3\mu\end{aligned}$$

(a) follows from (3.6). (b) and (d) follow from (3.7). (c) is due to the fact that $\hat{\theta}$ is the minimizer of train loss. Choosing $\mu = \epsilon'/6$ completes the proof. □

Lemma 7 (Convergence to minimizer)**.** *Given* $\epsilon' > 0$, $\exists\, \eta \left(= (1-\tau)\sqrt{\frac{2\lambda(\mathcal{X})\epsilon'}{\alpha}}\right) > 0$ *such that whenever* $\mathrm{BCE}(\gamma_\theta) - \mathrm{BCE}(\gamma^*) \leq \epsilon'$*, we have*

$$\|\vec{\gamma_\theta} - \vec{\gamma^*}\|_1 \leq \eta$$

where $\vec{\gamma} = [\gamma(x)]_{x\in\mathcal{X}}$ *and* $\lambda(\mathcal{X})$ *is the measure of compact set* $\mathcal{X} \subset \mathbb{R}^{d_x}$*.*

Proof.

$$\mathrm{BCE}(\gamma) = -\frac{1}{2}\left(\sum_{x\in\mathcal{X}} p(x)\log\gamma(x) + q(x)\log(1-\gamma(x))\right)$$

is α'-strongly convex as a function of $\vec{\gamma}$ under Assumption (A2), where $\alpha' = \frac{\alpha}{(1-\tau)^2}$. So $\forall\gamma\,, \frac{\partial^2 \mathrm{BCE}}{\partial\gamma(x_k)\partial\gamma(x_l)} \geq \alpha'$ for $k = l$ and 0 otherwise. Using the Taylor expansion for strongly convex functions, we have

$$\mathrm{BCE}(\vec{\gamma_\theta}) \geq \mathrm{BCE}(\vec{\gamma^*}) + \langle\nabla\mathrm{BCE}(\vec{\gamma^*}), \vec{\gamma_\theta} - \vec{\gamma^*}\rangle + \frac{\alpha'}{2}\|\vec{\gamma_\theta} - \vec{\gamma^*}\|_2^2$$

Since $\vec{\gamma^*}$ is the minimizer, $\nabla \text{BCE}(\vec{\gamma^*}) = 0$. So,

$$\|\vec{\gamma^*} - \vec{\gamma_\theta}\|_2 \leq (1-\tau)\sqrt{\frac{2}{\alpha}\left(\text{BCE}(\vec{\gamma_\theta}) - \text{BCE}(\vec{\gamma^*})\right)} \leq (1-\tau)\sqrt{\frac{2}{\alpha}\epsilon'}$$

From Holder's inequality in finite measure space,

$$\|\vec{\gamma^*} - \vec{\gamma_\theta}\|_1 \leq \sqrt{\lambda(\mathcal{X})}\|\vec{\gamma^*} - \vec{\gamma_\theta}\|_2 \leq (1-\tau)\sqrt{\frac{2}{\alpha}\lambda(\mathcal{X})\epsilon'} = \eta$$

□

Lemma 8 (Estimation from Samples)**.** *Given $\epsilon > 0$, for any classifier with parameter $\theta \in \Theta$, $\exists\, n \in \mathbb{N}$ such that with probability 1,*

$$|I_n^{\gamma_\theta}(U;V) - I^{\gamma_\theta}(U;V)| \leq \frac{\epsilon}{2}$$

Proof. We denote the empirical estimates as $\mathop{\mathbb{E}}_{x\sim p_n}(\cdot)$ and $\mathop{\mathbb{E}}_{x\sim q_n}(\cdot)$ respectively. The proof essentially relies on the empirical mean of functions of independent random variables converging to the true mean. More specifically, we consider the functions $f^\theta(x) = \log\frac{\gamma^\theta(x)}{1-\gamma^\theta(x)}$ and $g^\theta(x) = \frac{\gamma^\theta(x)}{1-\gamma^\theta(x)}$. Since $\gamma(x) \in [\tau, 1-\tau]$, both $f(x)$ and $g(x)$ are bounded. ($f \in [\log\frac{\tau}{1-\tau}, \log\frac{1-\tau}{\tau}]$ and $g \in [\frac{\tau}{1-\tau}, \frac{1-\tau}{\tau}]$). Functions of independent random variables are independent. Also, since the functions are bounded, they have finite mean and variance. Invoking the law of large numbers, $\exists\, n \geq n_1'(\epsilon)$ such that with probability 1

$$|\mathop{\mathbb{E}}_{x\sim p_n} f^\theta - \mathop{\mathbb{E}}_{x\sim p} f^\theta| \leq \frac{\epsilon}{4} \tag{3.8}$$

and $\exists\, n \geq n_2'(\epsilon)$ such that with probability 1

$$|\mathop{\mathbb{E}}_{x\sim q_n} g^\theta - \mathop{\mathbb{E}}_{x\sim q} g^\theta| \leq \frac{\epsilon\tau}{4(1-\tau)} \tag{3.9}$$

Then, for $n \geq \max(n_1'(\epsilon), n_2'(\epsilon))$, we have with probability 1

$$\begin{aligned}
&|I_n^{\gamma_\theta}(U;V) - I^{\gamma_\theta}(U;V)| \\
&\leq |\mathop{\mathbb{E}}_{x\sim p_n} f^\theta - \mathop{\mathbb{E}}_{x\sim p} f^\theta| + |\log \mathop{\mathbb{E}}_{x\sim q_n} g^\theta - \log \mathop{\mathbb{E}}_{x\sim q} g^\theta| \\
&\leq |\mathop{\mathbb{E}}_{x\sim p_n} f^\theta - \mathop{\mathbb{E}}_{x\sim p} f^\theta| + \frac{1-\tau}{\tau}|\mathop{\mathbb{E}}_{x\sim q_n} g^\theta - \mathop{\mathbb{E}}_{x\sim q} g^\theta| \\
&= \frac{\epsilon}{4} + \frac{\epsilon}{4} = \frac{\epsilon}{2}
\end{aligned}$$

where in the last inequality, we use the Lipschitz constant for log with the bounded function g as argument. □

Proof of Theorem 1

Using Proposition 1, $I^{\gamma^*}(U;V) = I(U;V)$, where γ^* is the unique global minimizer of BCE(γ).

The empirical risk minimizer of BCE loss is $\hat{\theta}$. For a rich enough class Θ and large enough samples n, Lemma 6 and Lemma 5 combine to give $\text{BCE}_n(\gamma_{\hat{\theta}}) - \text{BCE}(\gamma^*) \leq \epsilon'$. Applying Lemma 7 with $\epsilon' = \frac{\alpha}{8\lambda(\mathcal{X})}\left(\frac{\eta}{\beta(1-\tau}\right)^2$, we have $\|\vec{\gamma^*} - \vec{\gamma}_{\hat{\theta}}\|_1 \leq \frac{\eta}{2\beta}$. This further implies that

$$\mathop{\mathbb{E}}_{x\sim p} |\gamma^* - \hat{\gamma}_{\hat{\theta}}| \leq \frac{\eta}{2} \tag{3.10}$$

and

$$\mathop{\mathbb{E}}_{x\sim q} |\gamma^* - \hat{\gamma}_{\hat{\theta}}| \leq \frac{\eta}{2} \tag{3.11}$$

We now compute the Lipschitz constant for $f = \log \frac{\gamma}{1-\gamma}$ as a function of γ, which links the classifier predictions to Donsker-Varadhan representation.

$$|f^* - \hat{f}^{\hat{\theta}}| = |\log \frac{\gamma^*}{1-\gamma^*} - \log \frac{\hat{\gamma}_{\hat{\theta}}}{1-\hat{\gamma}_{\hat{\theta}}}| \leq \frac{1}{\tau^2}|\gamma^* - \hat{\gamma}_{\hat{\theta}}|$$

and

$$|e^{f^*} - e^{\hat{f}^{\hat{\theta}}}| = |\frac{\gamma^*}{1-\gamma^*} - \frac{\hat{\gamma}_{\hat{\theta}}}{1-\hat{\gamma}_{\hat{\theta}}}| \leq \frac{1}{\tau^2}|\gamma^* - \hat{\gamma}_{\hat{\theta}}|$$

For $\gamma \in [\tau, 1-\tau]$, the function $f \in [\log \frac{\tau}{1-\tau}, \log \frac{1-\tau}{\tau}]$ is continuous and bounded with Lipschitz constant $\frac{1}{\tau^2}$. So, using (3.10) and (3.11),

$$\mathop{\mathbb{E}}_{x\sim p}|f^* - \hat{f}^{\hat{\theta}}| \leq \frac{1}{\tau^2}\frac{\eta}{2} \text{ and } \mathop{\mathbb{E}}_{x\sim q}|e^{f^*} - e^{\hat{f}^{\hat{\theta}}}| \leq \frac{1}{\tau^2}\frac{\eta}{2}$$

Finally, from the Donsker-Varadhan representation 3.1,

$$\begin{aligned}
&|I(U;V) - I^{\gamma_{\hat{\theta}}}(U;V)| \leq |\mathop{\mathbb{E}}_{x\sim p} f^* - \mathop{\mathbb{E}}_{x\sim p}\hat{f}^{\hat{\theta}}| + \\
&|\log \mathop{\mathbb{E}}_{x\sim q} e^{f^*} - \log \mathop{\mathbb{E}}_{x\sim q} e^{\hat{f}^{\hat{\theta}}}| \\
&\leq \mathop{\mathbb{E}}_{x\sim p}|f^* - \hat{f}^{\hat{\theta}}| + \mathop{\mathbb{E}}_{x\sim q}|e^{f^*} - e^{\hat{f}^{\hat{\theta}}}| \\
&= \frac{\eta}{2\tau^2} + \frac{\eta}{2\tau^2} = \frac{\eta}{\tau^2}
\end{aligned} \tag{3.12}$$

where we use the inequality $\log(t) \leq t - 1$ coupled with the fact that $\mathop{\mathbb{E}}_{x\sim q} e^{f^*} = 1$. Given $\epsilon > 0$, we choose $\eta = \tau^2\frac{\epsilon}{2}$.

To complete the proof, we combine the above result (3.12) with Lemma 8 using Triangle Inequality,

$$\begin{aligned}
&|I_n^{\gamma_{\hat{\theta}}}(U;V) - I(U;V)| \\
\leq &|I_n^{\gamma_{\hat{\theta}}}(U;V) - I^{\gamma_{\hat{\theta}}}(U;V)| + |I^{\gamma_{\hat{\theta}}}(U;V) - I(U;V)| \\
&\frac{\epsilon}{2} + \frac{\epsilon}{2} = \epsilon
\end{aligned}$$

Corollary 1. *CCMI is consistent.*

Proof. For each individual MI estimation, we can obtain the classifier parameter θ_1(resp. θ_2) $\in \Theta$ such that Theorem 1 holds with approximation accuracy $\epsilon/2$. So, $\exists n \geq n_1(\epsilon/2)$ such that with probability at least $1 - \delta$

$$|\hat{I}_n^{\gamma_{\theta_1}}(X;YZ) - I(X;YZ)| \leq \frac{\epsilon}{2}$$

and $n \geq n_2(\epsilon/2)$ such that with probability at least $1 - \delta$

$$|\hat{I}_n^{\gamma_{\theta_2}}(X;Z) - I(X;Z)| \leq \frac{\epsilon}{2}$$

Using Triangle inequality, for $n \geq \max(n_1, n_2)$, with probability at least $1 - \delta$, we have

$$\begin{aligned}
&|\hat{I}_n(X;Y|Z) - I(X;Y|Z)| \\
&= |\hat{I}_n^{\gamma_{\theta_1}}(X;Y,Z) - \hat{I}_n^{\gamma_{\theta_2}}(X;Z) - I(X;Y,Z) + I(X;Z)| \\
&\leq |\hat{I}_n^{\gamma_{\theta_1}}(X;Y,Z) - I(X;Y,Z)| + |\hat{I}_n^{\gamma_{\theta_2}}(X;Z) - I(X;Z)| \\
&\leq \frac{\epsilon}{2} + \frac{\epsilon}{2} = \epsilon
\end{aligned}$$

□

The following Theorem shows that even for a small number of samples, the produced MI estimate is a true lower bound on mutual information value with high probability.

Theorem 2. *The finite sample estimate from Classifier-MI is a lower bound on the true MI value with high probability, i.e., given n test samples and the trained classifier parameter $\hat{\theta}$, we have for $\epsilon > 0$*

$$Pr(I(U;V) + \epsilon \geq I_n^{\gamma_{\hat{\theta}}}(U;V)) \geq 1 - 2\exp(-Cn)$$

where C is some constant independent of n and the dimension of the data.

Proof.

$$I(U;V) = \max_{\gamma} I^{\gamma}(U;V)) \geq \max_{\theta} I^{\gamma_\theta}(U;V)) \geq I^{\gamma_{\hat{\theta}}}(U;V))$$

We apply one-sided Hoeffding's inequality to (3.8) and (3.9) with given $\epsilon > 0$,

$$Pr(\mathop{\mathbb{E}}_{x\sim p_n} f^{\hat{\theta}} - \mathop{\mathbb{E}}_{x\sim p} f^{\hat{\theta}} \leq \frac{\epsilon}{2}) \geq 1 - \exp\left(-\frac{n\epsilon^2}{8(\log((1-\tau)/\tau))^2}\right) = 1 - \exp(-C_1 n\epsilon^2)$$

$$Pr\left(\mathop{\mathbb{E}}_{x\sim p} g^{\hat{\theta}} - \mathop{\mathbb{E}}_{x\sim p_n} g^{\hat{\theta}} \leq \frac{\epsilon\tau}{2(1-\tau)}\right) \geq 1 - \exp\left(-\frac{n\epsilon^2}{2}\left(\frac{\tau}{1-\tau}\right)^4\right) = 1 - \exp(-C_2 n\epsilon^2)$$

$$Pr\left(I_n^{\gamma_{\hat{\theta}}}(U;V)) \leq I^{\gamma_{\hat{\theta}}}(U;V)) + \epsilon\right) \geq 1 - 2\exp(-Cn)$$

where $C = \epsilon^2 \min(C_1, C_2)$.

□

Chapter 4

LATENT FACTOR MODELS : APPLICATIONS IN GENOMICS

In this Chapter[1], we investigate model-guided unsupervised learning where the data generation steps in the application domain provide prior knowledge. We apply unsupervised learning approaches to a pivotal problem in computational genomics, namely sequencing duplicated regions of the genome. In the process, we develop algorithms that can be applied to a broad class of problems in machine learning such as matrix completion over a discrete alphabet or community detection in a signed network in the presence of outliers. Our approaches hinge on utilizing the appropriate latent factor model. This line of algorithmic thinking is different from model-agnostic representation learning explored in the previous chapters.

While the rise of single-molecule sequencing systems has enabled an unprecedented rise in the ability to assemble complex regions of the genome, long segmental duplications in the genome still remain a challenging frontier in assembly. Segmental duplications are at the same time both gene rich and prone to large structural rearrangements, making the resolution of their sequences important in medical and evolutionary studies. Duplicated sequences that are collapsed in mammalian *de novo* assemblies are rarely identical; after a sequence is duplicated, it begins to acquire *paralog specific variants*. This is different from exact repeats which may be duplicated in large numbers and interspersed throughout the human genome. Segmental duplications on the other hand are low copy duplications. Their copy number is also known to vary across individuals and such variations have been associated with diseases of genomic origin including schizophrenia and autism. In this chapter, we

[1]This Chapter is based on joint work with Mark Chaisson, Sreeram Kannan and Evan Eichler [24]. Special thanks to Mitchell R. Vollger for providing the real dataset on human chromosome segments.

study the problem of resolving the variations in multicopy long-segmental duplications by developing and utilizing algorithms for *polyploid phasing*. In *haplotype phasing* there are two unknown segments of a DNA which differ at multiple positions, known as *variant* sites. In *polyploid phasing*, there are more than two segments that need to be resolved.

4.1 Introduction

Advances in single-molecule sequencing (SMS) by Pacific Biosciences (Menlo Park, CA), and Oxford Nanopore (Cambridge, UK) have recently enabled the assembly of draft *de novo* mammalian genomes [138, 57] nearing the quality of the original release of the human genome. The goal of *de novo* fragment assembly is to estimate the sequence of a genome given overlaps of relatively short sequencing reads, and is a well-studied problem. While there are multiple formulations of the fragment assembly problem [108, 120], the common challenge is that repeats in the genome longer than the length of sequenced DNA fragments make a unique reconstruction of the genome impossible [119]. Reads produced by SMS are advantageous for *de novo* assembly because the read length is at least two orders of magnitude greater than other high-throughput sequencing methods, so that genome order may be uniquely resolved when repeats are small.

SMS reads are characterized by a raw read accuracy between 75% and 90% with read lengths that follow a log-normal distribution. Initial development in *de novo* assembly of SMS reads focused on efficient methods to detect overlaps between long but noisy reads [109, 12]. Consistent with information theory [104], regions of genomes without sufficiently long repeats are contiguously assembled [77] with SMS reads. A type of repeat not well represented in human and other mammalian *de novo* SMS assemblies are *segmental duplications*: sequences 1 to 400kbp in length that are duplicated with at least 90% identity [43]. Comparing an SMS-based assembly of a Yoruban individual [147] to the human reference (GRCh38) reveals that only 64.2% of known segmentally duplicated bases in the human genome are present in the assembly. Segmental duplications are at the same time both gene rich and prone to large structural rearrangements [139], making the resolution of their sequences important in

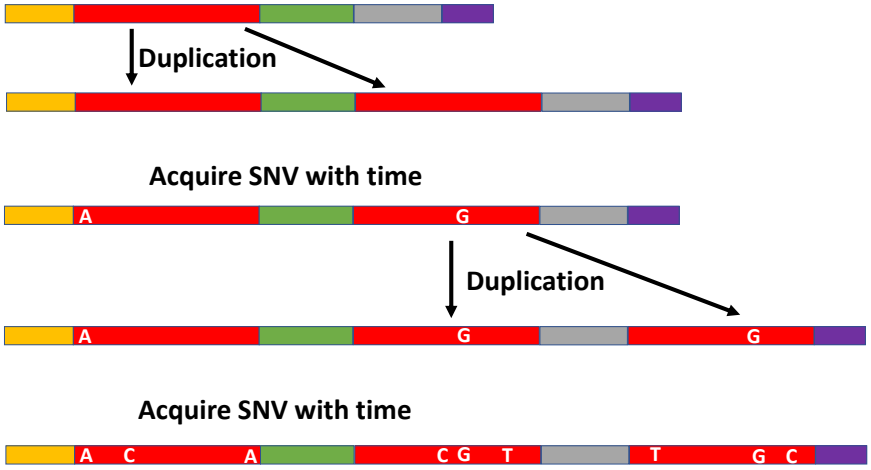


Figure 4.1: Ancestral genome undergoes segmental duplication and acquires single nucleotide variants over time. The duplicated *red* regions are known as paralogs and the unique variants in them (nucleotide bases shown in *white*) are known as paralog specific variants.

medical and evolutionary studies.

A DNA assembly algorithm seeks to recover the entire sequence, but often has to settle with *contigs* instead. Contigs are sub-sequences of the DNA recovered by assembly algorithms to high enough precision. Due to the low raw-read accuracy of SMS sequences, reads from different duplication paralogs are frequently merged together into the same sequence in an assembly. As a result, mammalian assemblies of SMS reads contain large contigs with correctly resolved unique sequence, and shorter contigs containing the collapse of multiple copies of a duplication into one sequence. Segmental duplications pose a barrier to this grand goal of assembling the entire DNA sequence.

Duplicated sequences that are collapsed in mammalian *de novo* assemblies are rarely identical; after a sequence is duplicated, it begins to acquire *paralog specific variants* (PSVs): single-nucleotide variants that distinguish different duplication paralogs. We use the terms paralog specific variants and single nucleotide variants (SNVs) interchangeably in the text. To put this in an evolutionary context, sequences that have duplicated after the human-

chimpanzee divergence (6 million years ago) have acquired up to roughly one PSV per thousand bases [36]. Although the ultimate goal of *de novo* assembly is to completely resolve the sequence of a genome, an intermediate goal is to resolve the individual sequences that are collapsed in the assembly. We propose resolving sequences by estimating the number of duplications collapsed into an individual sequence in an assembly, and determining the PSVs belonging to each duplication. Figure 4.1 shows how an ancestral genome undergoes segmental duplication and acquires PSV sites over time.

If there are more than two copies of a duplication that are collapsed, one may assume that the consensus sequence represents the ancestral sequence of a duplication before PSVs are acquired. Given S segmental-duplication paralogs of the same length containing V variants, one may represent all paralogs as a $S \times V$ matrix P with entries in $\{0, 1\}$, where each entry $P(i, j)$ is 0 if the repeat paralog i is in the ancestral state at site j, or 1 if it is a site that has mutated to a PSV. The set of N reads from all repeat paralogs may be aligned to the consensus sequence, and represented as an $N \times V$ read-fragment matrix X with entries in $\{0, 1, -\}$ corresponding to ancestral, variant, or absent (since reads only give information about certain positions).

The goal is to reconstruct the paralog matrix P given only the read matrix X, where there are also sequencing errors creating erroneous entries in X. Let us assume that the error probability is ϵ at any position, i.e., with probability $1 - \epsilon$, the location is read correctly and with probability ϵ, the location is read incorrectly (0 is read as a 1 and vice-versa).

For $S = 2$, this problem is identical to haplotype phasing of a diploid genome [82, 86, 7, 118]. Defining a read conflict as two overlapping reads that are non-gap and disagreeing at a site, haplotype phasing with error-free reads may be determined by grouping all conflict-free reads. To handle sequencing errors, a common formulation for haplotype phasing is Minimal Error Correction (MEC), where a minimal number of base changes are applied to reads so that they may be partitioned into two conflict-free sets. For $S = 2$, there has also been an exact information theoretic characterization of when it is possible to phase the genome correctly [140, 29], along with efficient algorithms. This is based on connections to a problem

called "community detection" [48] where the goal is to cluster users into communities based on positive or negative interactions between individuals.

When $S > 2$ this corresponds to the much less studied problem of *polyploid phasing*, which was dicussed in pioneering work by Aguiar and Istrail [2]. Beginning with Hapcompass [2], there has been some work on polyploid phasing using algorithms based on branch-and-extend [11], belief propagation [122] and semi-definite programming [34]. In a recent theoretical work [15], the hardness of optimizing the MEC for $S > 2$ has also been proven, indicating that algorithms for this problem need to be necessarily approximate or tailored to some assumptions. A major drawback of existing works is that they consider only $S = 3, 4$ and none have been developed, optimized or tested for the high polyploidy that is encountered in segmental duplications, where S can be potentially larger than 10. This issue is compounded in our problem by the presence of higher error-rates common in single-molecule sequencing reads, rather than the low-error rates in Illumina sequencers. Thus algorithms that are robust to the high error rates and can handle the high poly-ploidy are imperative in solving the segmental duplication problem, and in this paper, we will design such algorithms.

In particular, we propose two algorithms for solving the problem. The first approach is based on a discrete matrix completion paradigm where the goal is maximize the likelihood of the observed data given the underlying haplotypes. The second approach is based on a correlation-clustering framework with an inherent assumption that each haplotype has a paralog-specific variant (which holds in many types of segmental duplications). By performing detailed simulations, we demonstrate the superior performance of the proposed algorithms over existing algorithms, especially in the high ploidy regime. We show that the former algorithm has the highest likelihood estimate and better performance than existing algorithms, the latter algorithm can indeed return the correct answer on a larger number of datasets than existing algorithms, due to a stronger regularization. We demonstrate the superior practical performance of these algorithms on simulated datasets. We measure the likelihood score as well as reconstruction accuracy, i.e., what fraction of the reads are clustered correctly. In both the performance metrics, we find that our algorithms dominate

existing algorithms on more than 93% of the datasets. We also show that our correlation-clustering algorithm can reconstruct on an average 7.3 haplotypes in 10-copy duplication data-sets whereas existing algorithms reconstruct less than 1.5 copies.

4.2 Haplotype phasing via Discrete Matrix Completion

4.2.1 A probabilistic model

In order to represent the matrices in real-valued arithmetic, we adopt the following mapping: $\{0, 1, -\} \to \{-1, 1, 0\}$, i.e., we represent the ancestral allele as -1, variant as 1 and undisclosed locations as 0. To model the read matrix X, we first consider an idealized matrix M, which does not contain any noise nor does it contain any undisclosed position. If read n is sampled from the s-th paralog, then the n-th row of this matrix M is given by the s-th row of the paralog matrix, i.e., $M_{n.} = P_{s.}$. Figure 4.2 shows how the read data matrix is obtained from the underlying paralog matrix. The disclosed locations of the matrix are represented by a set Ω which comprises of the set of tuples (n, v) where read n contains information about variant v. Given M and Ω, the matrix X is not a deterministic function since there are independent read errors, which convert a 1 into a -1 with probability ϵ and vice versa. The probability of observing X given M and Ω is therefore given as follows,

$$\begin{aligned}
\log \mathbb{P}(X \mid M, \Omega) &= \sum_{(n,v)\in\Omega} \log \mathbb{P}(X_{n,v} \mid M, \Omega) \\
&= \sum_{(n,v)\in\Omega} \log\left((1-\epsilon)\mathbb{1}_{X_{n,v}=M_{n,v}}\right) + \log\left(\epsilon \mathbb{1}_{X_{n,v}\neq M_{n,v}}\right) \\
&= d_H(X, M) * \log(\epsilon) + (NV - d_H(X, M)) * \log(1-\epsilon) \\
&= -d_H(X, M) * \log(\frac{1-\epsilon}{\epsilon}) + (NV) * \log(1-\epsilon),
\end{aligned}$$

where $d_H(X, M)$ is the hamming distance between the two matrices X and M in the locations Ω, i.e., where $X \neq 0$. Different haplotype assembly algorithms have sought to minimize varied objective criteria in order to obtain the correct clustering of reads belonging to the

respective haplotypes [132]. Some of the noteworthy objectives are minimum edge removal (MER), minimum SNP removal (MSR) and minimum error correction (MEC). The quantity $d_H(X, M)$ is called as the error criterion, and and in our approach, maximizing the likelihood is equivalent to minimizing this error criterion referred to as MEC.

We observe that the ideal matrix M has repeated rows, since all rows sampled from the same paralog are identical. This implies that the matrix M has low-rank, and the observed matrix X is a noisy observation of this low-rank matrix. Indeed the matrix M can be factorized as the product of two matrices $M = A \cdot B$, where $A \in \mathbb{R}^{N \times S}$ with $A_{ij} \in \{0, 1\}\ \forall i, j$ and $B \in \mathbb{R}^{S \times V}$ with $B_{ij} \in \{-1, 1\}\ \forall i, j$.Each tow of A is an elementary vector of length S denoting which paralog the read is from and matrix B is identical to P (represented in $\{-1, 1\}$).

The observed matrix X is a noisy partial observation of a low-rank matrix M, and the goal is to reconstruct the matrices A and B given X. If each read spanned the entire segmental duplication, the problem would be trivial, since similar reads can be grouped together and taking a consensus inside clusters reveals the segmental duplications. The difficulty is posed by the fact that read lengths are much smaller and do not span all variant positions.

Each read only provides partial phasing information. The resulting X matrix is thus sparse, and our goal can be formally stated as follows:

$$\text{argmin}_{A,B} d_H(X, A \cdot B). \tag{4.1}$$

Real-valued versions of this problem has received much attention and is called the matrix completion problem. While this problem has a rich history, there is a significant difference in our setting, since the matrices A and B have structure (i.e., A has only elementary row vectors and B has binary entries) and the matrix X is ternary. We therefore have to develop new algorithms that exploit the discrete structure of the problem.

The problem of finding missing entries in a matrix arises in diverse research domains. One of the most illustrative examples is the Netflix challenge where users rate a small fraction of movies at random and the task is to predict user preferences for an unrated movie; a

$$P = \begin{pmatrix} C & A & C & G & T & G & G & C & T & A & G \\ C & A & T & C & T & C & A & C & G & C & G \\ C & C & C & C & C & G & G & C & G & A & A \end{pmatrix}$$

$$\downarrow$$

$$\mathrm{X} = \begin{pmatrix} C & A & C & G & . & . & . & . & . & . & . \\ . & . & C & G & T & G & G & C & T & . & . \\ . & . & . & . & . & . & G & C & T & A & G \\ C & A & T & C & T & . & . & . & . & . & . \\ . & . & T & C & T & C & . & . & . & . & . \\ . & . & . & . & . & C & A & C & G & C & G \\ C & C & C & C & C & . & . & . & . & . & . \\ . & . & . & C & C & G & G & C & G & A & A \end{pmatrix}$$

Figure 4.2: The paralog matrix P and the read matrix obtained from it. We illustrate a noiseless case here for simplicity.

key assumption in this domain is that the true matrix of preferences is low-rank. While low-rank matrix-completion problem is known to be NP-Hard, there are methods that can give provably correct reconstruction under probabilistic rather than worst-case assumptions [19, 127]. Popular techniques for this problem include convex relaxation of the rank to nuclear norm [127], singular value thresholding [18] and alternating minimization [67], all of which have theoretical guarantees as well. The key difference between these works and our problem is that they consider real-valued matrix-completion, whereas, in this paper, we adapt and extend the algorithms to the discrete setting inherent to the phasing problem.

In a recent paper [17], Cai *et al.* formulate haplotype phasing as a low rank matrix completion problem and uses structure constrained alternating minimization for obtaining the haplotypes. In the paper, they demonstrated improved performance over HapCompass for diploid and simulated polyploid data (with $S = 3, 4$). We show in this paper that while that method has good performance with small S, the performance starts deteriorating with higher S. The main reason for the deteriorating performance is the inability of the algorithm to exploit the discrete structure of the problem (for example, the algorithm does not use the

fact that the B matrix is binary, instead treating it as a real-valued matrix). We alleviate this problem in the present paper by proposing an algorithm that explicitly exploits this fact.

Input: : Noisy incomplete Matrix X, Rank Estimate S

Initialize $A_{init} \in \mathbb{R}^{N \times S}$ and $B_{init} \in \mathbb{R}^{S \times V}$ with sign corrected SVD.

$e \leftarrow$ Error rate

$k \leftarrow S$

while $k \geq 2$ *and MEC Score decreases* **do**

$B_{\text{est}} \leftarrow \text{RealMatCom}(A_{\text{init}}, B_{\text{init}}, X, k)$

$A_{\text{est}}, B_{\text{est}} \leftarrow \text{DiscreteMatCom}(B_{\text{est}}, X, e)$

Choose the best segment based on individual scores

$A_{\text{init}} \leftarrow A_{\text{est}}$

$B_{\text{init}} \leftarrow B_{\text{est}}$

$k \leftarrow k - 1$

end

return *Estimated Haplotypes* B_{est}

Algorithm 4: Iterative Matrix Completion

4.2.2 Iterative Two Stage Matrix Completion

Our problem stated in (4.1) is a hard combinatorial problem. While one can design alternating minimization based techniques for this problem, where A and B are optimized alternatively while keeping the other variable fixed. While such methods monotonically increase likelihood, they are not guaranteed to find the global optimum of the problem and display high sensitivity to initial conditions. The key idea in our approach is to first neglect the discrete nature of our problem, and view it as a real-valued matrix completion problem. We then "round" the results obtained from this real valued matrix completion to obtain a feasible solution for the discrete problem. This rounded solution then becomes the initial

`RealMatCom`$(A_{\text{init}}, B_{\text{init}}, X, k)$

$A \leftarrow A_{\text{init}}$

$B \leftarrow B_{\text{init}}$

while *stopping criterion not satisfied* **do**

  Minimize A using projected gradient descent

  Minimize $B_{1:k}$ using projected gradient descent

end

return sign(B)

Algorithm 5: Real Valued Matrix Completion

value of a discrete matrix completion routine designed based on the alternative minimization technique. While this method already has superior performance compared to existing approaches, we found that in the regime when the ploidy is high, the algorithm is able to extract some dominant haplotypes correctly while being incorrect on the other haplotypes. In order to overcome this barrier, in iteration i, we only fix the best $i-1$ haplotypes based on the current MEC, and optimize for the rest. A schematic representation of this algorithm is depicted in Fig. 4.3, and the detailed pseudocode is in Algorithm 4, Algorithm 5 and Algorithm 6.

A standard approach in combinatorial optimization is to relax the integrality constraints in the problem in order to get a real-valued optimization problem, and then to round the obtained results to get a feasible solution. We follow a similar approach here by relaxing our discrete problem to a continuous optimization problem, and along with it, we relax the objective too. Instead of optimizing according to the Hamming distance objective with the discrete constraints on A, B (see (4.1)), we instead minimize the Frobenius norm of the difference while at the same time assuming that A and B are real valued.

`DiscreteMatCom`(B_{est}, X, e)

while *MEC Score decreases* **do**

  for *each row* i *of* X **do**

    for *each segment* s *of* B_{est} **do**

      $d(i,s) \leftarrow$ Hamming Distance of X_i and $B_{\text{est},s}$ for known entries

      $W_i \leftarrow$ Window size of revealed entries of X_i

      $A_{\text{est},is} \leftarrow (1-e)^{W_i - d(i,s)} \cdot e^{d(i,s)}$

    end

    Update overall MEC score and score for each individual segment

    Normalize $A_{\text{est},i}$ to be a probability distribution

  end

  Initialize $B_{\text{est},new} \in \mathbb{R}^{S \times V}$ with zeros

  for *each row* i *of* X **do**

    $B_{\text{est,new}} \leftarrow B_{\text{est,new}} + \mathcal{P}_{\Omega}(A_{est,i}^{T} \cdot X_i)$

  end

  $B_{\text{est}} \leftarrow \text{sign}(B_{\text{est,new}})$

end

return $A_{\text{est}}, B_{\text{est}}$

Algorithm 6: Discrete Valued Matrix Completion

The noisy low rank matrix completion can be formally stated as an optimization problem.

$$\min_{A,B} \frac{1}{2}\|\mathcal{P}_\Omega(A \cdot B - X)\|_F^2$$

The objective function is a squared sum of errors over all the known entries of X. $\mathcal{P}_\Omega(\cdot)$ is the projection operator and Ω is the set of known incides of X. So, $\mathcal{P}_\Omega(Z_{ij}) = Z_{ij}$ if $(i, j) \in \Omega$ and 0 otherwise. While we relax the integrality constraints of the problem, we assume the following linear constraints to hold.

$$0 \leq A_{ij} \leq 1 \quad \forall\, i \in [N], j \in [S] \tag{4.2}$$

$$-1 \leq B_{ij} \leq 1 \quad \forall\, i \in [S], j \in [V] \tag{4.3}$$

Since the optimization is over unknown matrices A and B in a product form, the problem is non-convex. However, alternating minimization algorithms are known to have guaranteed reconstruction performance in certain regimes [67] and therefore we resort to using such algorithms. Thus we first solve the optimization over A, keeping B fixed, which makes the problem convex in A and vice-versa.

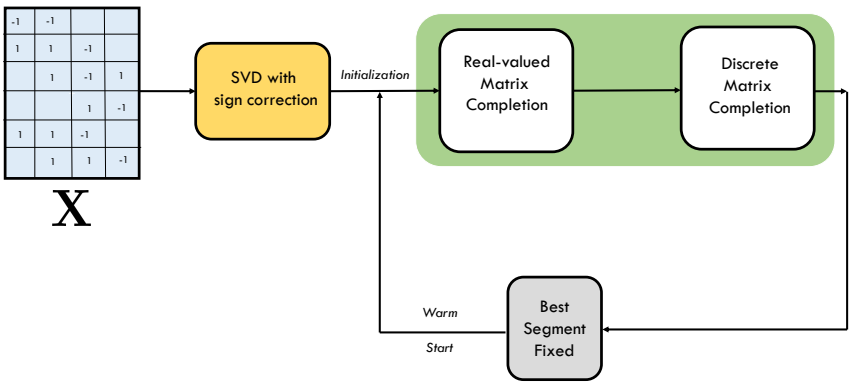


Figure 4.3: The initialization and iterative workflow for DMP.

4.2.3 ***Projected Gradient Descent***

The alternating minimization for our problem therefore can be stated as follows:

$$\min_{A} \frac{1}{2}\|\mathcal{P}_{\Omega}(A \cdot B - X)\|_F^2$$
$$s.t. \quad 0 \leq A_{ij} \leq 1 \; \forall \, i, j$$

and similarly

$$\min_{B} \frac{1}{2}\|\mathcal{P}_{\Omega}(A \cdot B - X)\|_F^2$$
$$s.t. \quad -1 \leq B_{ij} \leq 1 \; \forall \, i, j$$

To incorporate the constraints on the variables, we use a projected gradient descent to minimize each of the convex formulations.

4.2.4 ***Initialization***

Since the overall problem is non-convex, it is required to choose a suitable initialization for better performance. Prior theoretical results [67] suggest taking the S singular vectors of $\mathcal{P}_{\Omega}(X)$ as the initial guess for A and B. While this is a reasonable initialization, the signs of the singular vectors obtained from SVD decomposition may not be consistent with our problem since we require the entries of A to be strictly non-negative. We note that the signs of the singular vectors can be swapped without affecting the SVD. Therefore, in our algorithm, in order to ensure this sign consistency, we reverse the signs of certain rows of B to ensure that all columns of A have a positive sum.

$$\mathcal{P}_{\Omega}(X) = U \cdot \Sigma \cdot V^T \quad \Gamma = \text{sign}(\mathbb{1}^T U) \quad A_{\text{init}} = U * \text{diag}(\Gamma) \quad B_{\text{init}} = (V * \text{diag}(\Gamma))^T$$

For details of the projected gradient descent, we refer the reader to Appendix I.

4.2.5 ***Discrete Matrix Completion***

We round the output of the real-valued matrix completion to satisfy the discrete constraints of the A and B and utilize this to run a discrete alternating minimization algorithm to solve

(4.1). The optimization of A given a fixed B is easy to solve: the basic idea is to assign each read to the segment which minimizes the hamming distance with the read. To optimize B given a fixed A, we find the consensus of all the reads which are informative about a given position. In our algorithm, instead of having A to be a hard decision of which segment a given read belongs to, each row i of A encodes the probability that read i belongs to segment j. Therefore, while optimizing over B, we utilize the weighted consensus rather than the plain consensus of the read assignments. We refer the reader to Algorithm 6 for a detailed description of the algorithm.

4.2.6 ***Choosing the best segment and Effective Rank Reduction***

As pointed out earlier, the algorithm as stated above works well with small polyploid instances; however, in the presence of higher ploidy, the algorithm returns only the top few haplotypes correctly. For example, consider the cascading topology of repeats in Figure 2, it is easier to resolve segment 7 but the other segments are more easily confused. Therefore, we propose an iterative algorithm, where in each iteration, the best haplotype is fixed and then the algorithm is run to optimize over possibilities of the other haplotypes. Thus in order to do matrix completion with S haplotypes, the algorithm is iterated over $S-1$ times. Such algorithms have a precedent even in real-valued matrix completion, for example, stagewise alternating minimization is shown to have better theoretical guarantees in [67]. In our implementation, at iteration i, the best $i-1$ haplotypes are chosen as the ones which have minimum hamming distance from their assigned reads.

4.3 ***Haplotype phasing with correlation clustering***

One limitation of the MEC objective function and therefore of the discrete matrix completion algorithm is that the ploidy must be known *a priori* or estimated. Since the MEC objective itself decreases monotonically with ploidy, it is not possible to estimate the ploidy using the MEC objective. This can be potentially remedied using regularized alternatives that account for model complexity like AIC, BIC or MDL. We propose an alternative algorithm

here that can jointly estimate the ploidy while estimating the haplotypes themselves. This algorithm is based on a key assumption, distinct from the assumptions of the discrete matrix completion problem: that each of the haplotypes have uniquely identifying variants. While this assumption is stronger, it can lead to stronger regularization of the problem by restricting the search space and therefore leads to better estimates, especially when the ploidy is high.

The basic idea of the algorithm is the following: each locus is represented as a vertex and reads that straddle multiple vertices create edges between the vertices that have either positive or negative weight based on whether reads share the variant or not. The goal is then to cluster the nodes into groups which share the same variant, with each cluster representing a haplotype and each locus (node) in the cluster representing a haplotype-specific variant.

To formally define our algorithm, we begin with an alternative formulation for polyploid phasing through *correlation clustering* [6], with the premise that a metric defines how similar or dissimilar two objects are, and clusters maximize the amount of similarity within each cluster and dissimilarity between clusters. Importantly, in correlation clustering the number of clusters is discovered as a result of clustering and not as a parameter.

We use an augmented form of the SNP conflict graph $\mathcal{G}_{\mathcal{S}}$ introduced in [82], denoted $\mathcal{G}_{\text{PSV}} = (V, E), E = \{E^+, E^-\}$. The construction of $\mathcal{G}_{\text{PSV}}$ requires the fragment matrix $\mathcal{M}$, and some data-dependent parameters: the expected range of coverage per haplotype c_{min} and c_{max}, and a distance d that is the maximum distance reads are expected to overlap variants. A vertex exists for each of the columns (sites) in the fragment matrix M, connected by an edge $(u, v) \in E^+$ if u and v are overlapped by between c_{min} and c_{max} reads that are variant (e.g., 1) at both sites, or an edge $(u, v) \in E^-$ if the sites corresponding u and v are within d bases and $(u, v) \notin E^+$. A weight $W(u, v)$ is assigned to each edge.

Correlation clustering on $\mathcal{G}_{\text{PSV}}$ corresponds to finding clusters $C = c_1, \ldots, c_n$ that minimize the sum of negative edges within each cluster plus the sum of weights of positive edges between clusters:
$\text{Score}_{\text{CC}} = \sum_{c_i} (\sum_{(u,v) \in c_i, (u,v) \in E^-} w(u, v) + \sum_{(u \in c_i, v \notin c_i), (u,v) \in E^+} w(u, v))$. Each cluster defines a set of sites that belong to a haplotype. This was shown to be APX-hard [35, 25, 44], and

approximations based on linear programming (LP) were described in [35, 25]. We developed an implementation of the LP approach that was successful at clustering smaller datasets, however the number of constraints grows with $|E|^2$, and $|E|$ grows by p^2v^2, for ploidy p and number of paralog specific variants v, which requires excessive resources for larger datasets.

To evaluate correlation clustering on larger datasets, we developed a simple randomized heuristic to search for clusters that provide acceptable values for Score_{CC} that follows the steps:

1. Define clusters likely to represent repeat paralogs through a random search.

2. Merge clusters with sufficient overlap, and assign nodes to unique clusters.

3. Optimize clusters by swapping vertices from adjacent clusters.

Define the *neighbor similarity* $\text{Sim}(u, v)$ of two vertices to be the number of neighbors shared between u and v connected by edges in E^+, and $\text{Score}(V, E, c)$ to be the Score_{CC} of a single cluster c assuming all vertices $V \setminus c$ are in a separate cluster. The first step is a method that defines clusters by iteratively adding vertices neighboring a cluster as long as the neighbor similarity is sufficient and addition of the vertex decreases Score_{CC}, and is described in 7.

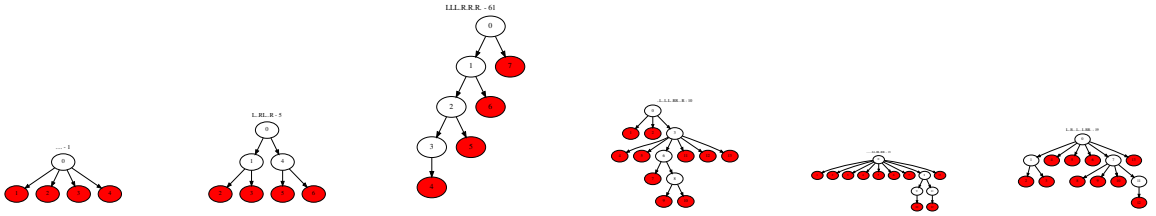
Given parameters for neighbor similarity s, a maximal number of search iterations (*max search it*) and swap iterations (*max swap it*), and fraction cluster overlap f^{ovp}, the method FindCluster is used to find a set of clusters C by first initializing $C = \emptyset$, and iteratively selecting a vertex $v_i \notin C$, and adding the result of $\text{FindCluster}(V, v_i, E, s)$ to C until C contains all vertices in V or max_it iterations are reached. The resulting clusters in C are not disjoint, and so any cluster c_i with a fraction of vertices overlapping with a cluster $c_j > f^{\text{ovp}}$ is first merged into c_j, then remaining vertices belonging to more than one cluster are assigned to the largest cluster for which they are a member. Finally, the clusters are further optimized by selecting edges $(u, v) \in E^+$ where $u \in c_i$ and $v \in c_j$ and swapping u and v if this improves Score_{CC} for up to max_swap_it iterations.

```
FindCluster(V, v_i, E, s)
c ← v_i
repeat
    forall v ∈ c do
        forall n ∈ Neighbors(v) ∉ c do
            if Sim(v, n) ≥ s and Score(V, E, c ∪ n) < Score(V, E, c) then
                c ← c ∪ n
            end
        end
    end
until c has not grown
return c
```

Algorithm 7: Find cluster

4.4 Results - CC Vs DMP

We benchmarked our methods on a dataset of simulated collapsed segmental duplications. Starting with an ancestral sequence, sequences are duplicated according to a specified tree topology T and mutation rate r, where each child node is a copy of a parent node mutated at a rate of r random SNV mutations per base. To capture the complexity of evolution, we used two classes of trees: 12 simulations from well defined topologies such as flat, bifurcating, and cascading, and 50 simulations from random tree topologies that have 10 child nodes, for which examples are shown in Figure 4.4. The mutation rate was varied from between 0.01, 0.005, and 0.001, and 0.0005 mutations per base to simulate various ages of duplication. For each set of duplications we simulated 50× read coverage using the Alchemy SMS read simulator [23], a model based simulator that emulates a sequencing run by Pacific Biosciences, and mapped reads back to the ancestral sequence. PSV sites are detected as sites that contain between 25 and 60 non-ancestral bases.



Flat Bifurcating Cascading Random - 1 Random - 2 Random 3

Figure 4.4: Examples of topologies of duplication simulations. In total there are 12 structured trees and 50 random topologies. The divergence between any two simulated duplications is given by the mutation rate $r\times$ the shortest path between the duplications in the tree.

For each of the simulated topologies and mutation rates, we evaluated the Discrete Matrix Completion for Phasing (DMP), Correlation Clustering (CC), and Structure Constrained Gradient Descent (SCGD). The SCGD method has been shown to outperform other previously developed methods methods in polyploid phasing [17].

For each haplotype we count the number of reads in the haplotype that are shared with the reads simulated in each duplication, and define a *matching* statistic as the sum of number of reads in the maximally matched duplication divided by the total number of reads. This statistic ranges between 1 for perfect reconstruction of haplotypes down to a $1/p$ when all of them are collapsed into a single reconstructed haplotype. The results are shown in Figure 4.5. CC had the greatest matching score 67.7% of the datasets, DMP 26.1% of the datasets, and SCGD on 6.1% of the datasets. The CC method exploits the assumption that each position has only one single variant, thereby resulting in stronger regularization. Even though the likelihood score is somewhat lower for CC method than other methods, it is able to fit the data more accurately. The other methods DMP and SCGD are unable to exploit this assumption and therefore overfit more severely to the data. The DMP method is sensitive to the initialization conditions for B_{est}, and so we used a solution derived by CC as initial conditions for DMP. We measured improvements on this combination (CC+DMP) relative

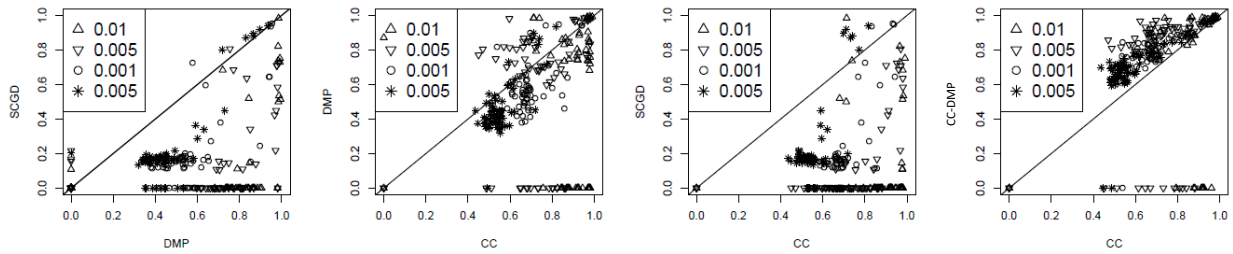


Figure 4.5: Matching statistics for the SCGD, DMP, CC and CC-DMP methods. A perfect reconstruction of haplotypes shows a score of 1, while a random assignment will score 1/ ploidy. Higher matching score is better.

to DMP and CC for matching score. For 220 of the 224 simulations where both CC and CC+DMP had a solution, we observed had a greater matching score in CC+DMP.

We also measure a more stringent quality of reconstruction accuracy: we ask for which fraction of the true haplotypes is there a reconstructed cluster into which 90% of the correct reads are assigned. Formally, for each simulated duplication we determined which haplotype had the most reads overlapping with the reads simulated from that duplication, and counted how many such haplotypes had at least 90% of the reads from that haplotype reciprocally assigned to that duplication. This gives an indication of the number of copies of a segmental duplication that would be correctly assembled given the phased haplotypes. For duplications of ploidy 10, the CC method resolves on average 7.0 copies of each duplication, whereas the DMP and SCGD methods resolve on average 3.0 and 0.03 copies, respectively. These fractions denote how many out of 10 duplicates (on average across the datasets) are resolved by each of these algorithms.

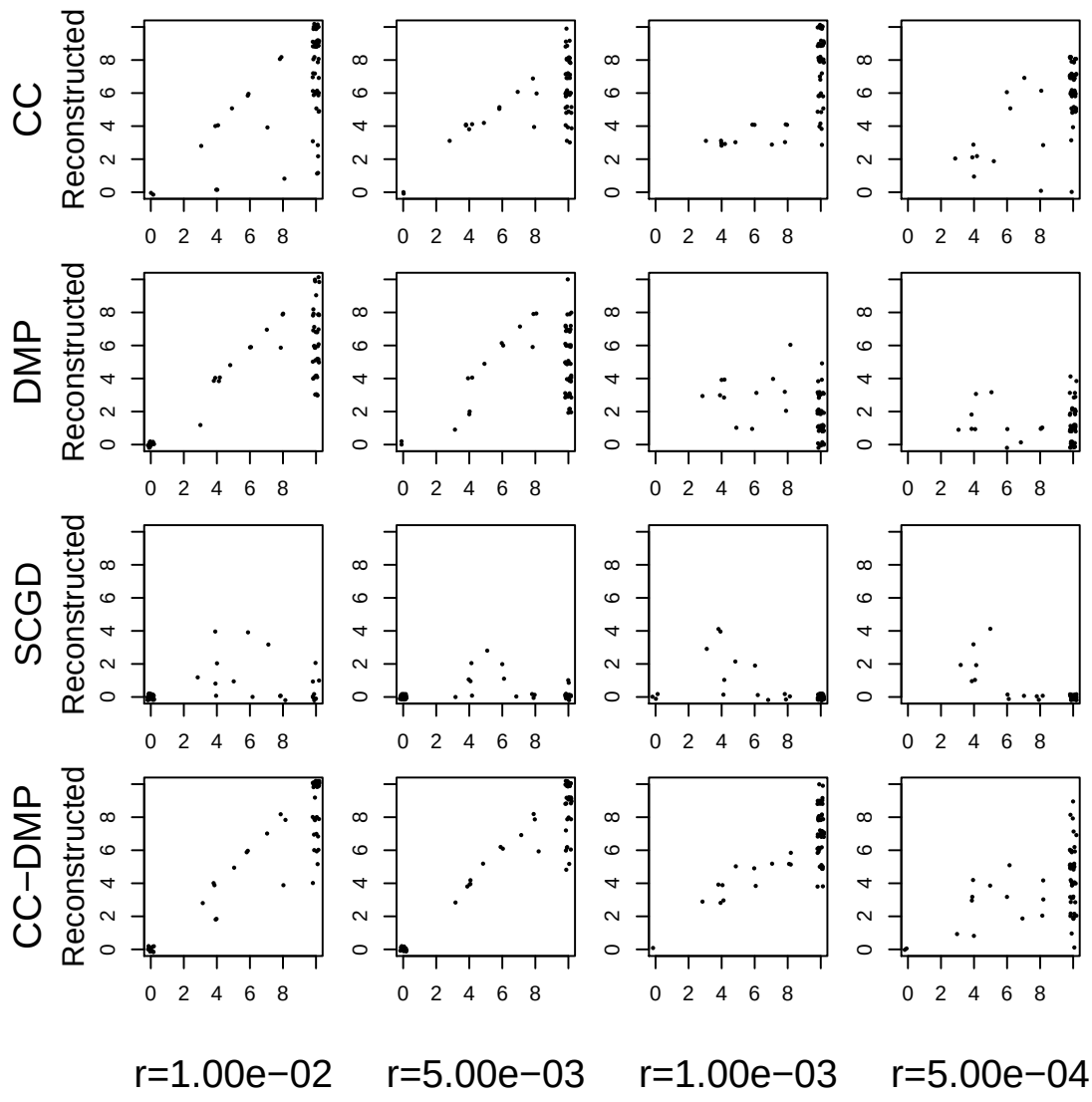
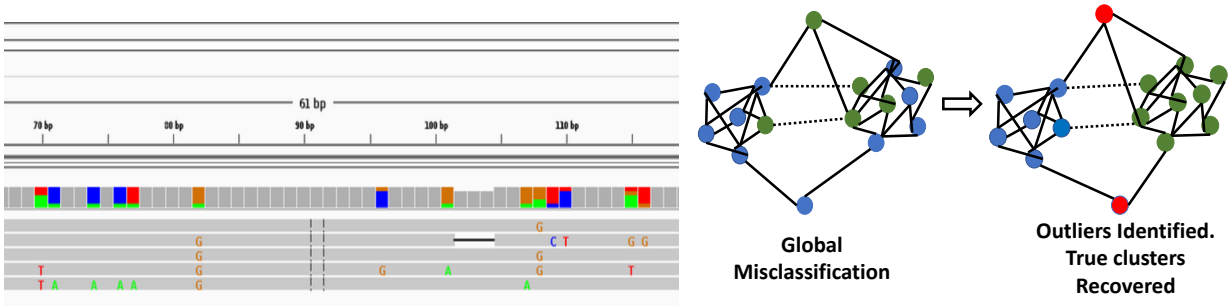


Figure 4.6: Correctly assembled haplotypes for the SCGD, DMP, CC and CC-DMP methods. For each simulated dataset, a point is added according to the number of duplications simulated, and number of correctly phased genotypes according to the 90% read similarity cutoff. The ideal performance is the $y = x$ line and the higher the better.

4.5 Robust Signed Community Detection : Accounting for Regularization

In this Section, we deviate from the usual framework of haplotype phasing [1] [17] [42] and proceed with the aim of separating out the reads belonging to distinct paralogs motivated by [24]. By taking consensus from the separated reads, we can then recover each individual paralog accurately. This approach has distinct advantages. Besides being scalable, it is able to recover any structural variation in the paralogs and performs well even when the number of duplications is high.



(a) Snapshot of alignment to reference showing PSV and non-PSV sites

(b) Signed community detection with outliers

4.5.1 Problem Formulation

To resolve the duplicated region, the reads are first aligned to a reference genome say of length G. Let K be the number of duplicated segments. We define PSV sites to be those positions in the reference where exactly one of the duplicated segments $k \in \{1, 2, \dots K\}$ contains an allele different from the reference and every other segment has the reference allele. Let $n\,(< G)$ be the total number of PSV sites. We denote PSV sites $[z_i]_{1 \le i \le n}$ to be the nodes of the graph, each belonging to one of K communities, based on which segment contains the variant allele. Thus, two nodes i and j belong to the same community if $z_i = z_j$; otherwise they belong to different communities. Since the long reads span multiple PSV sites, they provide noisy pairwise measurements y_{ij} between two sites. We denote the reference

allele as '+1', variant allele as '−1' and '0' for position not spanned by read, leading to a read data matrix $\mathcal{X} \in \mathbb{R}^{r \times n}$ with entries $\mathcal{X}_{ij} \in \{-1, +1, 0\}$. By combining the noisy partial information provided by r reads, we want to infer $z_i \, \forall \, i$. Previous approaches to haplotype phasing consisting of only two segments have considered representing z_i as either $+1$ or -1 (corresponding to $k = 1$ or $k = 2$ respectively) and then solving the maximum likelihood estimation for global reconstruction [30]. Such approaches do not trivially extend to higher ploidy. Instead, we need to transform z_i to a higher dimensional vector representation in a lifted space. Let $\mathbf{x_i} \in \mathbb{R}^K$ be the K-dimensional vector representing z_i. Moreover, $\mathbf{x_i}$ is a shifted and rescaled unit vector. If $z_i = k$, then $x_{ik} = -1$ and $x_{ij} = +1 \, \forall \, j \neq k$.

The goal is to assign values to nodes i and j so as to maximize the probability of $\mathbf{x_i}$ and $\mathbf{x_j}$ belonging to the same communities (or different communities) $\forall \, (i, j)$ based on the pairwise noisy measurements y_{ij} (where in a noiseless regime $y_{ij} = +1$, if $z_i = z_j$ and is -1 if $z_i \neq z_j$). We can define a likelihood matrix $\mathbf{L}_{ij} \in \mathbb{R}^{K \times K}$ for the lifted space where $(\mathbf{L}_{ij})_{\alpha,\beta} = \mathbb{P}(y_{ij} \mid z_i = a, z_j = b)$, $a, b \in \{1, 2, \dots K\}$. We can now state the optimization problem as

$$\begin{aligned} \underset{\mathbf{x}}{\text{minimize}} \quad & \mathbf{x}^T \mathbf{L} \mathbf{x} \\ \text{subject to} \quad & \mathbf{x_i} \in \mathbf{1} - 2 \cdot \{\mathbf{e}_1, \dots \mathbf{e}_K\}; \; i = 1, \dots, n \end{aligned} \tag{4.4}$$

where $\mathbf{e_j}$ is the unit vector and $\mathbf{1}$ is the vector of all ones, $\mathbf{x} \in \mathbb{R}^{nK \times 1}$.

If the contraints were real-valued, then the solution would be the largest eigenvector and is computed through the power method. As a natural extension, 4.4 is solved by performing power iteration steps and projecting $x_i^t \, \forall \, i$ onto a simplex at each step. This framework is motivated from [28] where the authors seek to recover a joint discrete alignment from pairwise modulo measurements. Our problem has less information since we do not have modulo measurements $y_{ij} = z_i - z_j \mod K$. The only information provided by reads is whether $z_i = z_j$ or not. Furthermore, the initialization framework provided in [28] does not work for community detection and needs to be correctly determined in order to enter the basin of attraction of the non-convex problem.

CC often overestimates the number of segments and separates reads into mutiple disjoint

groups for a single true underlying segment, specially in high duplication regimes with low mutation rates. These separated reads when grouped together do not span the entire segment and so cannot recover it completely. A reconstruction score favoring only the correctly separated reads for an overestimated number of recovered segments is unable to capture this acute deficiency. We believed that this is due to the heuristic nature of the algorithm and resorted to developing algorithms that embodied strong theoretical guarantees. Since it is difficult to detect PSV sites accurately with increase in K, we further incorporated robustness in our framework to address the issue with large fraction of outlier nodes.

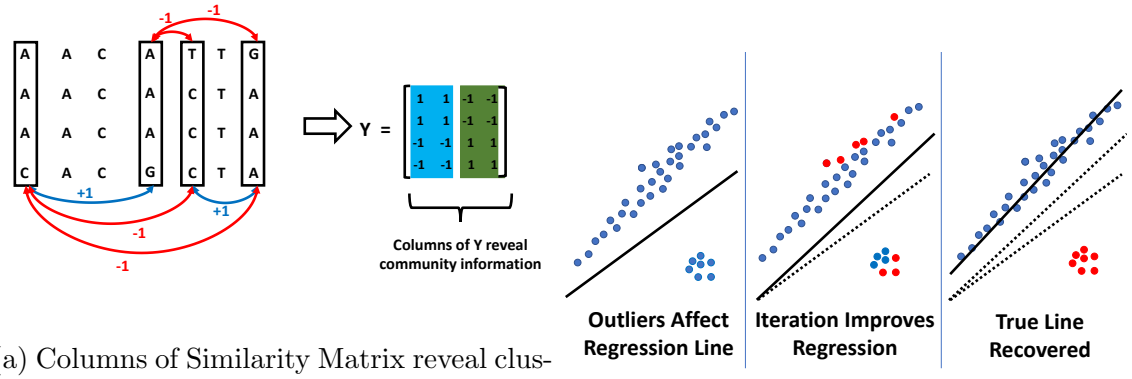
4.5.2 Similarity Matrix Computation

Unlike a community detection setting (where we might obtain the signs of edges by measuring y_{ij} explicitly), here we need to infer y_{ij} from read data matrix $\mathcal{X}$ based on pairwise counts of reference and variant (alternate) alleles. For each pair of sites, we obtain four counts $N_{\mathrm{r,r}}, N_{\mathrm{r,a}}, N_{\mathrm{a,r}}$ and $N_{\mathrm{a,a}}$, where 'r' denotes reference and 'a' is alternate. From this data, we can compute l_+ and l_-, the log likelihoods of $Y_{ij} = +1$ and -1 respectively as a function of ϵ (the substitution error rate) given the pairwise counts. If $l_+ - l_- > 1$, we infer $y_{ij} = +1$ and vice-versa. If the log likelihood difference is less than 1 or there is no pairwise count information available for any pair of sites, then we set $y_{ij} = 0$.

A simple calculation leads to

$$l_+ = \log \mathbb{P}(N_{\mathrm{r,r}}, N_{\mathrm{r,a}}, N_{\mathrm{a,r}}, N_{\mathrm{a,a}} \mid y_{ij} = +1) = N_{\mathrm{r,r}} \cdot \log\left(\frac{1}{K}(1-\epsilon)^2 + \frac{K-1}{K}\epsilon^2\right) + (N_{\mathrm{r,a}} + N_{\mathrm{a,r}}) \cdot \log\left(\epsilon(1-\epsilon)\right) + N_{\mathrm{a,a}} \cdot \log\left(\frac{1}{K}\epsilon^2 + \frac{K-1}{K}(1-\epsilon)^2\right)$$

$$l_- = \log \mathbb{P}(N_{\mathrm{r,r}}, N_{\mathrm{r,a}}, N_{\mathrm{a,r}}, N_{\mathrm{a,a}} \mid y_{ij} = -1) = N_{\mathrm{r,r}} \cdot \log\left(\frac{2}{K}\epsilon(1-\epsilon) + \frac{K-2}{2}(1-\epsilon)^2\right) + (N_{\mathrm{r,a}} + N_{\mathrm{a,r}}) \cdot \log\left(\frac{1}{K}((1-\epsilon)^2 + \epsilon^2) + \frac{K-2}{2}\epsilon(1-\epsilon)\right) + N_{\mathrm{a,a}} \cdot \log\left(\frac{2}{K}\epsilon(1-\epsilon) + \frac{K-2}{2}(1-\epsilon)^2\right)$$



(a) Columns of Similarity Matrix reveal cluster structure

(b) Robust Regression Iterations

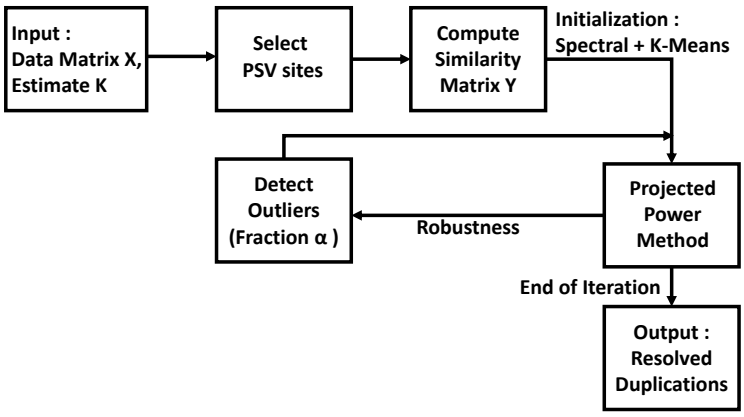


Figure 4.9: FlowChart : Resolving Segmental Duplications using Robust PPM

4.5.3 Initialization and Robustness

The optimization problem 4.4 being non-convex requires suitable initialization to reach the global minima. For the global alignment problem in [28], any random column of the low rank approximation of $\mathbf{L}$ suffices to provide a good estimate for convergence. A similar analysis of $\mathbb{E}(\mathbf{L})$ reveals that information from all K columns of $\mathbb{E}(\mathbf{L})$ need to be incorporated in order to determine the community membership of a node. We also observe that in the absence of modulo information, the likelihood matrix L contains the same node relationships as captured by similarity matrix Y. So, we obtain the K largest eigen-vectors of $\mathbf{Y}$, stack

them together in a matrix $\tilde{\mathbf{V}} \in \mathbf{R}^{n \times K}$ and then perform K-means clustering of the rows of $\tilde{\mathbf{V}}$, treating each row as a data point, to obtain $z_i^0 \, \forall i$. We could have performed clustering directly on $\mathbf{Y}$, but the eigenvectors have a denoising effect considering the fact that $\mathbf{Y}$ could be quite sparse. Once the nodes are clustered, we obtain x_i^0 from $z_i^0 \, \forall i$, using the encoding explained previously. This gives the initial vector $\mathbf{x}^0$.

Since the duplicated regions of the reference have more reads aligned to them compared to unique regions of the genome, a coverage plot provides estimate of the number of segments K. Also, we could choose the PSV sites either by performing a likelihood ratio test (PPM_LRT) or by thresholding the fraction of alternate alleles at a site (PPM_frac) (since a PSV site would contain around $\frac{1}{K}$ of alternate alleles). As $1/K$ approaches the error rate ϵ, it becomes difficult to choose the sites accurately and a large fraction of outlier nodes α may be present. This adversely affects global reconstruction. To improve performance in high duplication regions, we provide an iterative robustness scheme motivated by robust regression [13] [157] [159]. Initially, we start with all the nodes. At each subsequent step, we evaluate a score $\mathcal{S} = \sum_j Y_{ij} x_i^T x_j$ for all nodes. The fraction $1 - \alpha$ of nodes with highest score is considered for reconstruction in next iteration. This fraction is not known apriori and can be estimated based on specific problem domain. This setting occurs in social networks also such as fake profiles in Facebook or apolitical individuals in an electoral college. Presence of large fraction of outliers can lead to misclassification by PPM. We illustrate some plots in Figure 4.12 for random corruption model where vanilla PPM (output of first iteration) has high error that is reduced by incorporating robustness. The details of the algorithm are provided in pseudocode 8.

4.6 Experiments on Simulated Data : CC Vs Robust PPM

We first compared the two algorithms with an extensive simulation based study. By communicating with the authors of [24], we obtained the datasets consisting of duplicated sequences that have evolved from some standard topologies such as flat, bifurcating, cascading along with some random trees. The datasets have varied number of segments ranging from 3 to

Input: Noisy incomplete matrix $\mathcal{X}$, Estimated number of segments K, error rate ϵ, outlier fraction α.

Output: Recovered segments $\mathcal{R}$

Obtain pairwise counts $N_{\mathrm{r,r}}, N_{\mathrm{r,a}}, N_{\mathrm{a,r}}$ and $N_{\mathrm{a,a}}\, \forall\, (i,j)$.

Compute similarity matrix $\mathbf{Y}$.

Assign $\tilde{\mathbf{Y}} \leftarrow Y, \text{true\_node} \leftarrow [n], \text{adv\_node} \leftarrow \phi$.

for $t \in \{1, 2, \dots \tau\}$ **do**

$\tilde{\mathbf{V}} = [\tilde{\mathbf{v_1}}, \tilde{\mathbf{v_2}} \dots \tilde{\mathbf{v_K}}]$, largest eigenvectors of $\tilde{\mathbf{Y}}$ stacked together.

$\mathbf{z}^0 \leftarrow$ k-means clustering of $\tilde{\mathbf{V}}$, treating each row as a data-point.

$\mathbf{x}^0 \leftarrow \mathcal{E}(\mathbf{z}^0)$, encoding of $z_i^0\, \forall i$ to scaled and shifted unit vector.

Run projected power iterations with $\mathbf{x}^0$ as initialization. Returns $\mathbf{z}$.

for *all* $i \in$ adv_node **do**

Assign it to the cluster with maximum score i.e.,

$z_i \leftarrow \underset{c}{\operatorname{argmin}} \sum_{j \neq i} \mathbb{1}_{\{z_j = c\}} Y_{ij} - \mathbb{1}_{\{z_j \neq c\}} Y_{ij}$

end

for *all* $i \in \{\text{true\_node} \cup \text{adv\_node}\}$ **do**

$\mathcal{S}(i) \leftarrow \sum_{j \neq i} \mathbb{1}_{\{z_i = z_j\}} Y_{ij} - \mathbb{1}_{\{z_i \neq z_j\}} Y_{ij}$

end

Sort nodes $i \in [n]$ in decreasing order of $\mathcal{S}$.

Update sets true_node as indices of the $(1-\alpha)n$ highest score nodes.

adv_node $\leftarrow [n] -$ true_node

$\tilde{\mathbf{Y}} \leftarrow (\mathbf{Y})_{[\text{true\_node}] \times [\text{true\_node}]}$

end

$\mathcal{R} \leftarrow [\mathcal{R}_i \mid \mathbf{e}_{z_i}$ if $i \in$ true_node, else $\mathbf{0}$ if $i \in$ adv_node$]_{1 \leq i \leq n}$.

return $\mathcal{R}$

Algorithm 8: ROBUST_PPM

	Metric m-10x					Metric m-90p				
	Avg	r				Avg.	r			
Algorithm	**10 copy**	**0.0005**	**0.001**	**0.005**	**0.01**	**10 copy**	**0.0005**	**0.001**	**0.005**	**0.01**
CC	2.53	0.26	0.46	1.70	7.70	1.80	0.00	0.00	0.58	6.62
PPM_LRT	4.87	0.22	3.72	7.78	7.74	3.48	0.00	0.02	6.62	7.28
PPM_frac	5.22	0.10	3.68	8.48	8.62	3.83	0.00	0.00	7.52	7.80
Robust PPM	5.45	0.20	3.60	9.08	8.90	4.02	0.00	0.00	8.04	8.04

Table 4.1: Comparison of Algorithm Reconstruction. Best Results in blue.

10 and mutation rates of $0.0001, 0.001, 0.005$ and 0.01. We compared the performance of our algorithm variants with CC on 248 datasets. Since CC outputs higher number of segments that the truth, MEC is not an appropriate criterion for comparison. The end goal for both these algorithms is to separate out the reads belonging to distinct paralogs. Let $\hat{K}$ be the reconstructed number of segments. After reconstruction, we assign each read to the reconstructed duplication i to which it is closest in Hamming distance. This leads to a clustering matrix $\mathcal{C} \in \mathbf{N}^{\hat{K} \times K}$, where $\mathcal{C}_{ij}$ denotes the number of reads that originated from true segment j, but were assigned to reconstructed segment i. We then evaluate how many of the true segments can be reconstructed based on two metrics m-$10x$ and m-$90p$. m-$10x$ counts number of entires (i, j) of $\mathcal{C}$ which are 10-times higher than all other values along row i or column j. m-$90p$ is a more stringent metric that requires (i, j) of $\mathcal{C}$ to have 90% of the reads assigned along that row i or column j. Since CC has no provision to utilize the knowledge of true number of segments, it often outputs more than the truth, some of them being fragmented partial portions of the truth. We observed that for higher duplications, CC is unable to provide a reconstructed segment spanning the true segment in its entirety, but rather fragmented segments that can partially recover the truth, thus, not solving the problem.

These deficiencies of CC are captured by our metrics of reconstruction. We provide

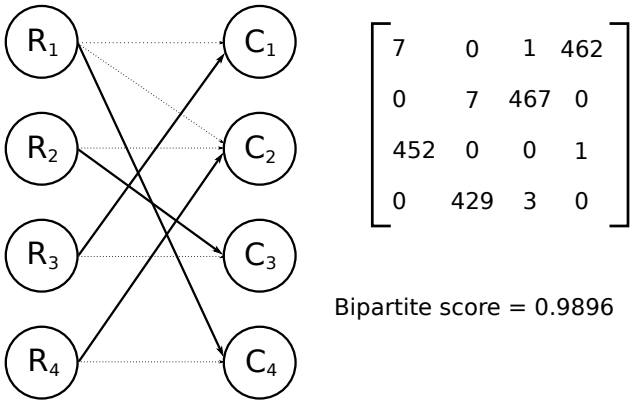


Figure 4.10: Matching Score between True and Recovered Segments

the true value of K to PPM, but also run our framework with incorrect segments to check the sensitivity. The results are best for correct estimates of K, but over-estimates do not degrade performance severely. Robustness is also applied to our basic framework to improve performance by assuming a fixed estimated fraction of outliers for high duplication regimes. Table 4.1 and Figure 4.11 clearly indicate the superiority of our algorithm over CC.

4.7 Reconstructing Virus Strains

Virus strains mutate at a high rate and pose severe challenges to antiviral drugs and vaccines. Whole genome sequencing of these strains from a mixture is thus needed to understand the diversity in the strains. More importantly, understanding the mutations in the coding regions of the genome provide crucial information about the environmental adaptations of the strains. The problem is similar to resolving segmental duplication inasmuch as one needs to perform polyploid phasing.

We utilized a publicly available data-set for this purpose [54]. Five viral strains from human immunodeficiency virus type 1 (HIV-1) are studied in this setup (Figure 4.13a). The strains (and their proportions) are HXB2 ($\sim$ 9.2%), 89.6 ($\sim$ 12.3%), JR-CSF ($\sim$ 27.9%), NL4-3 ($\sim$ 38.4%) and YU2 ($\sim$ 12.2%) respectively. The major challenge in this problem is that the viral strains have small pairwise edit distance (varying from 2.61% to 8.45%) in this

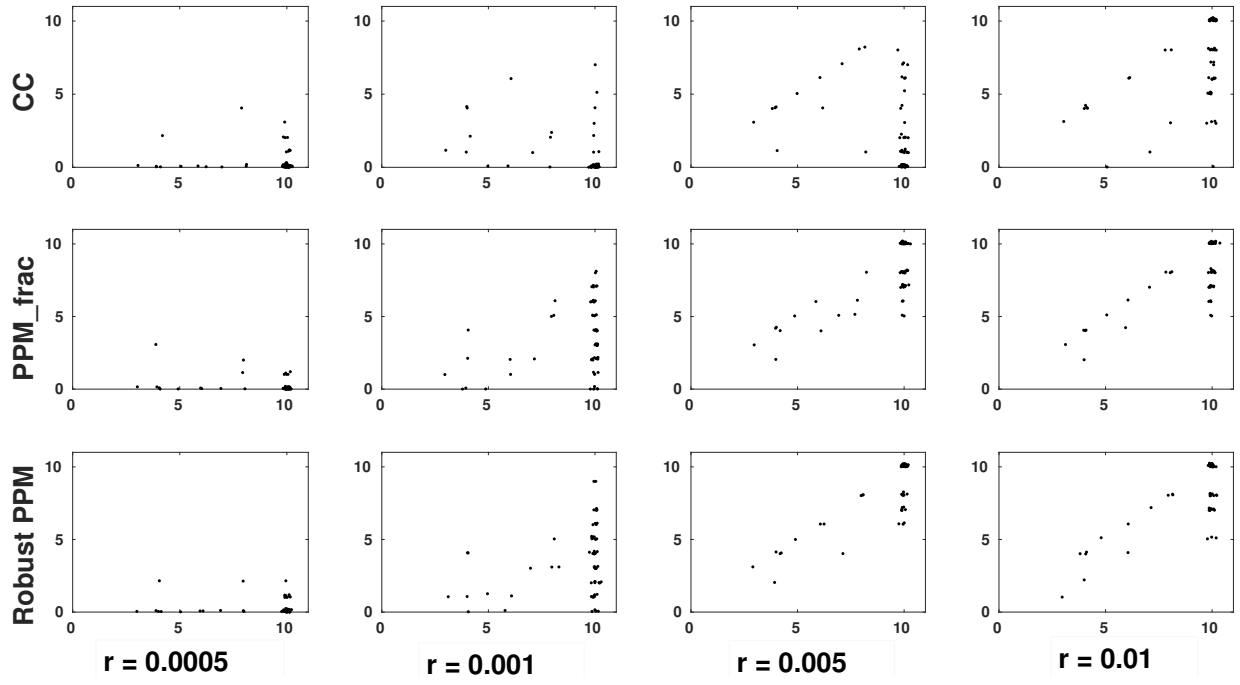
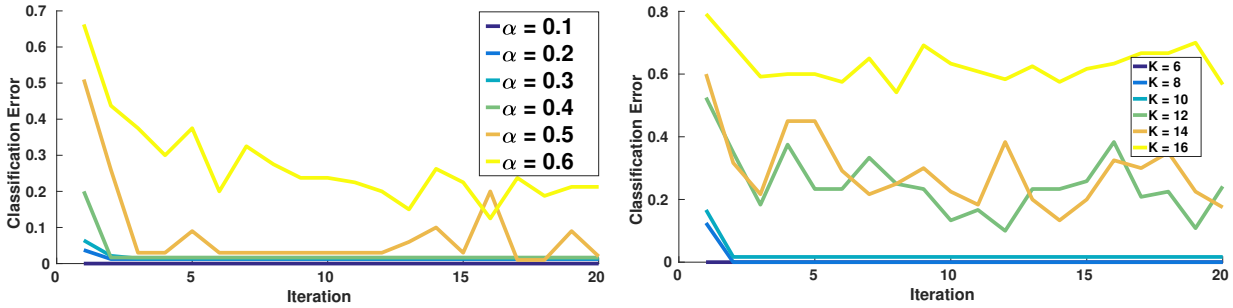


Figure 4.11: Count of Reconstructed Segments Vs True number of segments (Metric m-$10x$).

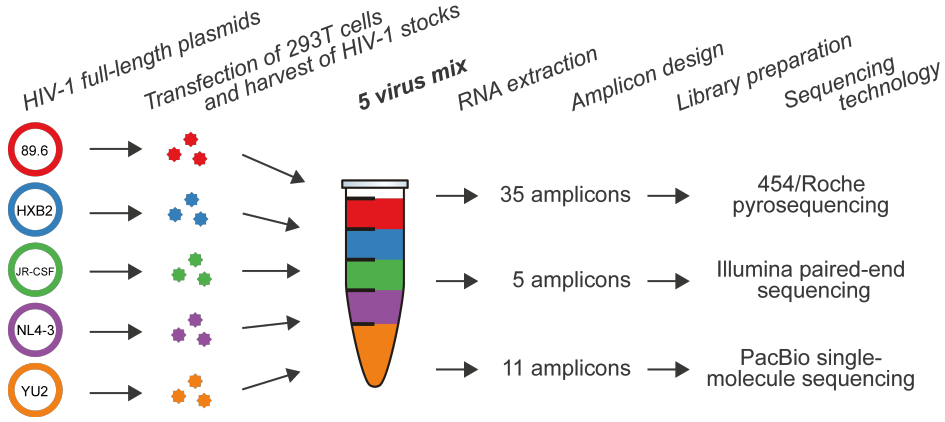


(a) Variation of α with $n = 200, K = 10$

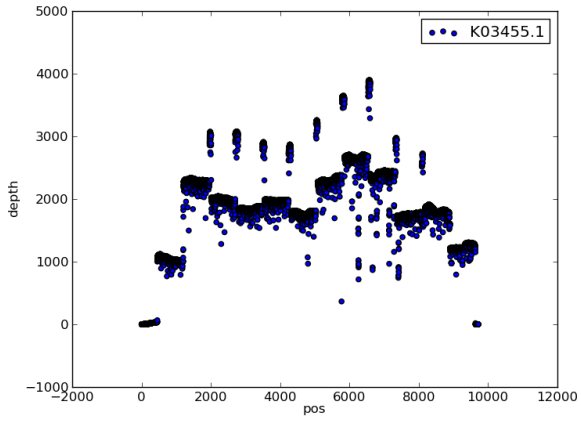
(b) Variation of K with $n = 200, \alpha = 0.4$

Figure 4.12: Robust algorithm reduces classification error

data-set. Yet, the algorithms need to differentiate the true variants from sequencing errors. We used PacBio sequencing reads in our pipeline. The reconstruction of entire genome of the strains is considered from base position 455 to 9635, from which sufficient number of



(a) Sequencing Virus Mixture (Courtesy [54])



(b) Depth profile of PacBio Reads

Figure 4.13: Data Generation Workflow for Virus Mixture

reads are available.

A total of 15932 PacBio reads were available and the average read length was approximately 1500 base pairs. The PacBio reads were first aligned to the reference genome using BWA Aligner. Variant calling was performed to detect SNV (single nucleotide variant) sites based on the counts of second-most frequent base. We chose sites where the minimum

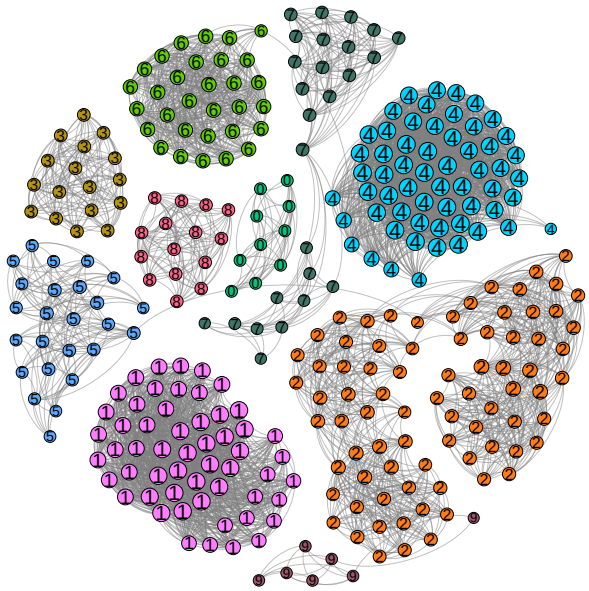


Figure 4.14: Communities Recovered by CC for the Virus Dataset. Some communities have very few nodes and could potentially be merged.

coverage was 200 and maximum was 700 for the second-most frequent base. Figure 4.13b shows the depth profile of most frequent base from BWA Aligner. After variant calling and pre-processing, the read-variant matrix had 12156 rows (corresponding to reads) and 670 columns (corresponding to variant sites). Note that variant calling to retain only SNV sites leads to a significant reduction in computation time for the algorithms, as inferring from entire genome would have led to 9181 columns.

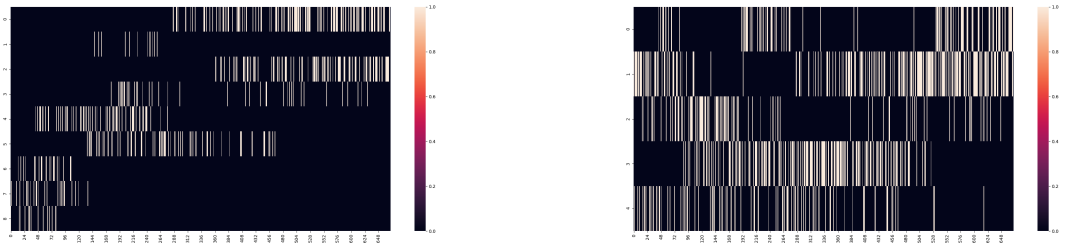
CC and PPM are then run on the data matrix to obtain output haplotypes. Since CC does not have any mechanism to input true number of haplotypes, we could not provide

Genomic Region	Phasing Algorithm	Virus Strains				
		1	**2**	**3**	**4**	**5**
gag pool	PredictHaplo	0.9995	1.0	1.0	1.0	1.0
	PPM	0.981	0.9879	0.9886	0.981	0.9984
	CC	0.362	0.9981	0.1953	0.9677	0.551
p17	PredictHaplo	1.0	1.0	1.0	1.0	1.0
	PPM	0.952	0.952	0.9899	1.0	0.9949
	CC	0.0	1.0	0.0	0.9621	0.0
p24	PredictHaplo	1.0	1.0	1.0	1.0	1.0
	PPM	0.9683	0.9683	1.0	0.9971	0.9971
	CC	0.0	1.0	0.0	0.9683	0.0
p7	PredictHaplo	0.9879	1.0	1.0	1.0	1.0
	PPM	0.9758	0.9879	1.0	1.0	1.0
	CC	0.0	1.0	0.0	0.9818	0.0
p6	PredictHaplo	1.0	1.0	1.0	1.0	1.0
	PPM	0.9686	1.0	1.0	0.9874	1.0
	CC	0.0	1.0	0.0	0.9748	0.0
gp41	PredictHaplo	0.999	0.9249	0.9306	0.9249	0.999
	PPM	0.9486	0.948	0.9894	0.9942	1.0
	CC	0.9904	0.0	0.9374	0.9942	1.0
gp120	PredictHaplo	1.0	0.9311	0.944	0.9349	1.0
	PPM	0.998	0.9873	0.9804	0.9846	1.0
	CC	0.985	0.2423	0.9204	0.9846	0.9869
nef	PredictHaplo	1.0	0.8841	0.9449	0.8867	0.9984
	PPM	0.9404	0.9308	0.9935	0.9754	1.0
	CC	0.9855	0.0	0.9417	0.9754	0.9565

Table 4.2: Comparison of Phasing Algorithms for Virus Strain Reconstruction

Genomic Region	Phasing Algorithm	Virus Strains				
		1	2	3	4	5
vpu	PredictHaplo	1.0	1.0	1.0	1.0	1.0
	PPM	0.9939	1.0	0.9789	0.9872	1.0
	CC	0.9759	0.9873	0.9114	0.9872	1.0
vpr	PredictHaplo	1.0	1.0	0.9966	1.0	1.0
	PPM	0.9726	1.0	0.9897	1.0	0.9725
	CC	0.9726	1.0	0.9519	1.0	0.9966
vif	PredictHaplo	1.0	1.0	1.0	1.0	1.0
	PPM	0.9499	1.0	1.0	1.0	0.9447
	CC	0.9499	1.0	0.9655	1.0	0.981
int	PredictHaplo	1.0	1.0	1.0	1.0	1.0
	PPM	0.9746	0.9977	0.9896	0.9781	0.9988
	CC	0.9746	0.9977	0.97	0.9769	0.9769
RNAse	PredictHaplo	1.0	1.0	1.0	1.0	1.0
	PPM	1.0	1.0	0.9528	0.9583	0.9944
	CC	0.9556	1.0	0.0	0.9556	0.9944
RT	PredictHaplo	1.0	1.0	1.0	1.0	1.0
	PPM	0.997	0.997	0.9856	0.9705	1.0
	CC	0.2803	0.997	0.0	0.9674	0.8848
pro	PredictHaplo	1.0	1.0	1.0	1.0	1.0
	PPM	0.9865	0.9933	1.0	0.9832	1.0
	CC	0.0	0.9933	0.0	0.9529	0.0
Entire Genome	PredictHaplo	0.9997	0.9714	0.9783	0.9727	0.9996
	PPM	0.9756	0.98	0.9883	0.9854	0.9944
	CC	0.6552	0.6778	0.5528	0.9778	0.7403

Table 4.3: Comparison of Phasing Algorithms for Virus Strain Reconstruction



(a) CC (b) PPM

Figure 4.15: Variant Sites for the Recovered Haplotypes : The rows of the Heatmap denote the distinct segments and the columns denote each variant site. CC splits us the recovery into multiple segments, some of which only account for a small number of variants. PPM on the other hand has variants spread out throughout the segment and solves the problem effectively.

that information to the algorithm. For PPM, we ran it with $S = 5$, the correct number of virus strains. The acute issue of obtaining fragmented segments was observed for this real problem for CC. Similar to simulated cases, we found that CC's output haplotypes only partially recovered the virus strains and outputs corresponded to contigs rather than the strain as a whole. Figure 4.15 shows the how the variant sites of PPM are spread throughout each inferred haplotype. But for CC, some of the segments are only responsible for short contigs. We also ran PredictHaplo, a baseline algorithm designed specifically for this problem to ensure that our pipeline was correct. PredictHaplo recovered 6 segments, but it did not fragment virus strains.

Since the ground truth virus strains are known, we compared the reconstruction rate for each strain for the various algorithms. CC and PPM does not provide the haplotypes with bi-allelic or tri-allelic variations. This can be obtained accurately from the original PacBio reads. From the recovered haplotypes, we first took the Hamming distance of each read with each haplotype at the corresponding aligned coordinates. The read was assigned to the

haplotype index with minimum Hamming distance. Then we took a consensus (majority vote across all base positions) of all reads assigned to a haplotype index to obtain the recovered haplotype. A maximal weight bipartite matching was performed for all algorithms to pair up a recovered haplotype with a true virus strain.

Tables 4.2 and 4.3 show the reconstructed fraction for the various algorithms on gene segments of the 5 strains as well as the entire genome of the strains. Recovering variants in gene/coding regions is crucial since different proteins produced by the virus strained could be accounted for by this diversity information. We observe that since some recovered haplotypes of CC do not span all genome coordinates, it has 0 reconstruction rate for some genes. We found PPM and PredictHaplo performance to be comparable with PPM having slightly higher average recovery rate across entire genome.

4.8 Reconstructing Segmental Duplication in Human Chromosome

We applied our algorithms to Real data obtained from human chromosome. Specifically, duplicated regions for collapsed gene families were studied. The authors of [153] improved the CC algorithm further by merging small clusters using added heuristics. This addressed the issue of CC of splitting segments and recovering partial contigs. As a result, the comparison in this Section is with this improved version of CC (as opposed to the previous Sections).

We found the PacBio reads used in this setup to have lower error rates and they were also longer. In our PPM run, we reduced the error rate hyper-parameter to 0.04 from 0.08. The real data from human chromosomes had added challenges. The number of reads obtained from the various duplication were highly imbalanced in some genomic regions. PPM calculations rely on a model-based approach, where communities had been assumed to have approximately equal size. More, knowing the number of segments is crucial to the estimation steps in PPM. Fortunately, the estimation of number of duplication was done from coverage statistics and was correct in most data-sets. Despite these challenges, PPM was able to

[2]Improved version obtained from [153]

Genomic Region	# SNV Sites	# Duplication	Algorithms		
			CC^2	PPM	Robust PPM ($\alpha = 0.05$)
FCGR	703	2	0.8113	0.8137	0.8137
FRMPD2	243	2	0.9934	0.9907	0.9921
GTF2H2	96	3	0.7645	0.7679	0.7747
HYDIN2	1220	3	0.8150	0.8182	0.7025
NCF1	271	3	0.9917	0.9917	0.9917
NOTCH2	479	5	0.8095	0.7464	0.7269
NPY4	153	2	0.9606	0.8592	0.8938
SGRAP2	1600	5	0.7436	0.7601	0.7484

Table 4.4: Comparison of bipartite matching score w.r.t Read separation - Human Segmental Duplication

obtain comparable performance to improved CC.

4.9 Trade-offs between the various algorithms

To conclude, we summarize the trade-offs between the three algorithms we studied in this Chapter for resolving segmental duplication.

Algorithm	**Pros**	**Cons**
Discrete Matrix Completion for Phasing (DMP)	• General formulation. • Can accommodate multiple variant sites	• Overfits to read error when true variants are SNV • Sequential hence slow.
Correlation Clustering (CC)	• Does not need to know # segments	• Fragmented assembly in low coverage/high noise. • Heuristic.
Projected Power Method (PPM)	• Theoretical guarantees	• Needs to know # segments

Table 4.5: Trade-offs between the various algorithms

Chapter 5

UNSUPERVISED LEARNING IN THE REAL-WORLD : APPLICATIONS IN A NATURAL LANGUAGE TASK

In this Chapter[1], we study how unsupervised approaches blend with supervised learning to solve real-world problems. Having studied both model-agnostic and model-guided approaches, we realize that for more complicated problems when the generative process is not known, we need to adopt model-agnostic strategy. Contextual text generation is one such problem where the intricacies of natural language need to be *learnt* by our machine learning system to be able to accomplish human-level performance.

In recent times, smart assistants have evolved to help users efficiently perform day-to-day activities. For some people with special abilities, this is a need. For others, it is a means of increasing productivity and getting done with mundane chores. We focus now on the application domain of emails, a means of communication we often indulge in. Intelligent features in email service applications aim to increase productivity by helping people organize their folders, compose their emails and respond to pending tasks. In this work, we explore an interesting application, Smart To-Do, that allow users to diligently organize their tasks and schedules. The main objectives of this chapter are as follows:

1. To introduce a new task and data-set for automatically generating To-Do items from emails where the sender has promised to perform an action.

2. To design a two-stage algorithm leveraging recent advances in neural text generation and sequence-to-sequence learning. The first stage is an unsupervised matching based

[1]This Chapter is based on work primarily done during my internship at Microsoft. The work is done jointly with Subhabrata Mukherjee, Marcello Hasegawa, Ahmed Hassan Awadallah and Ryen White [107].

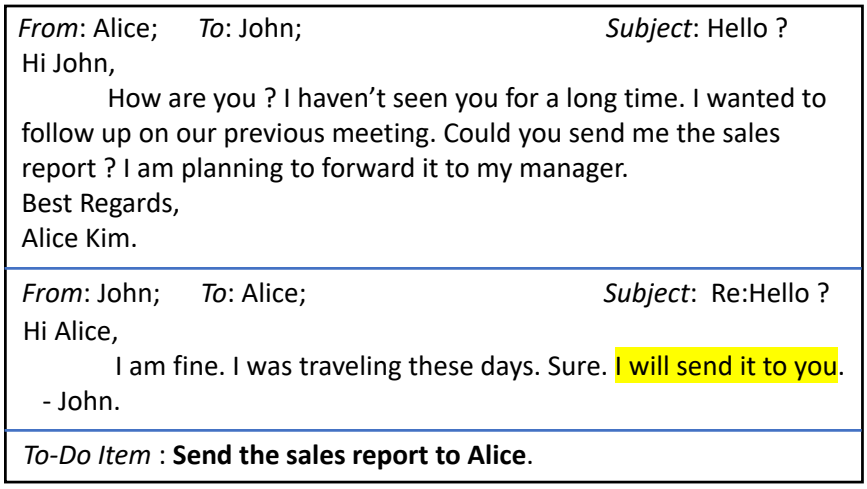


Figure 5.1: An illustration showing the email and a commitment sentence (in yellow) and the target To-Do item, along with other email meta-data.

extractive phase; the second stage utilizes existing neural frameworks for abstractive summarization.

To the best of our knowledge, this is the first work to address the problem of composing To-Do items from emails.

5.1 Introduction

Email is one of the most used forms of communication especially in enterprise and work settings [124]. With the growing number of users in email platforms, service providers are constantly seeking to improve user experience for a myriad of applications such as online retail, instant messaging and event management [46]. Smart Reply [72] and Smart Compose [26] are two recent features that provide contextual assistance to users aiming to reduce typing efforts. Another line of work in this direction is for automated task management and scheduling. For example, the recent Nudge feature in Gmail and Insights in Outlook are designed to remind users to follow-up on an email or pay attention to pending tasks.

Smart To-Do takes a step further in task assistance and seeks to boost user productivity by automatically generating To-Do items from their email context. Text generation from emails, like creating To-Do items, is replete with complexities due to the diversity of conversations in email threads, heterogeneous structure of emails and various meta-data involved. As opposed to prior work in text generation like news headlines, email subject lines and email conversation summarization, To-Do items are *action-focused*, requiring the identification of a specific task to be performed.

In this Chapter, we introduce the task of automatically generating To-Do items from email context and meta-data to assist users with following up on their promised actions (also referred to as commitments in this Chapter). Refer to Figure 5.1 for an illustration. Given an email, its temporal context (i.e. thread), and associated meta-data like the name of the sender and recipient, we want to generate a short and succinct To-Do item for the task mentioned in the email.

This requires identifying the task sentence (also referred to as a *query*), relevant sentences in the email that provide contextual information about the query along with the entities (e.g., people) associated with the task. We utilize existing work to identify the task sentence via a commitment classifier that detects action intents in the emails. Thereafter we use an unsupervised technique to extract key sentences in the email that are *helpful* in providing contextual information about the query. These pieces of information are further combined to generate the To-Do item using a sequence-to-sequence architecture with deep neural networks. Figure 5.2 shows a schematic diagram of the process. Since there is no existing work or dataset on this problem, our first step is to collect annotated data for this task.

Overall, our contributions can be summarized as follows:

- We create a new dataset for To-Do item generation from emails containing action items based on the publicly available email corpus *Avocado* [114].[2]

[2]Email examples provided in this Chapter are similar to those in our dataset but are not reproducing text from the Avocado dataset due to data sensitivity.

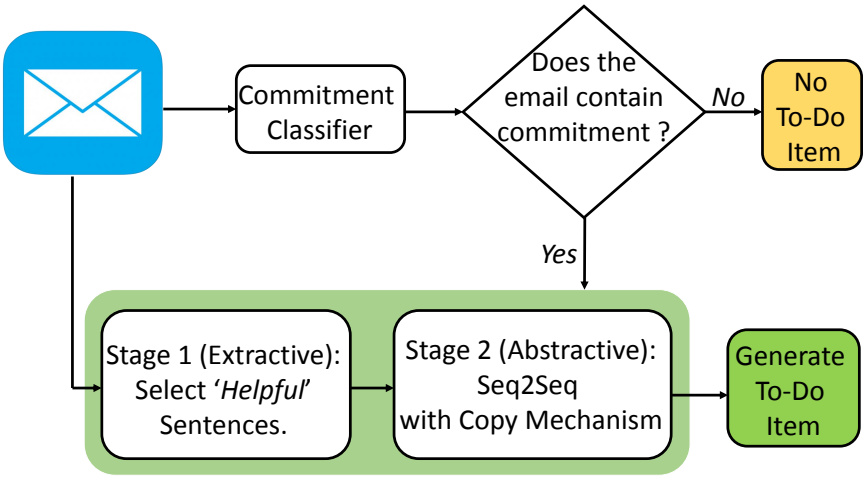


Figure 5.2: Smart To-Do flowchart: The email content is first scanned to detect any possible commitment sentence. If present, a To-Do item is generated using a two-stage Smart To-Do framework.

- We develop a two-stage algorithm, based on unsupervised task-focused content selection and subsequent text generation combining contextual information and email meta-data.

- We conduct experiments on this new dataset and show that our model performs at par with human judgments on multiple performance metrics.

5.2 Related Works

Summarization of email threads has been the focus of multiple research works in the past [126, 20, 40]. There has also been considerable research on identifying speech acts or tasks in emails [22, 81, 131] and how it can be robustly adapted across diverse email corpora [4]. Recently, novel neural architectures have been explored for modeling action items in emails [91] and identifying intents in email conversations [154]. However, there has been less focus on task-specific email summarization [32]. The closest to our work is that of email

subject line generation [161]. But it focuses on a common email theme and uses a supervised approach for sentence selection, whereas our method relies on identifying the task-related context.

5.3 Dataset Preparation

We build upon the Avocado dataset [114][3] containing an anonymized version of the Outlook mailbox for 279 employees with various meta-data and 938,035 emails overall.

5.3.1 Identifying Action Items in Emails

Emails contain various user intents including planning and scheduling meetings, requests for information, exchange of information, casual conversations, etc. [154]. For the purpose of this work, we first need to extract emails containing at least one sentence where the sender has promised to perform an action. It could be performing a task, providing some information, keeping others informed about a topic and so on. We use the term *commitment* to refer to such intent in an email and the term commitment sentence to refer to each sentence with that intent.

Commitment classifier: A commitment classifier $\mathcal{C} : \mathcal{S} \mapsto [0, 1]$ takes as input an email sentence $\mathcal{S}$ and returns a probability of whether the sentence is a commitment or not. The classifier is built using labels from an annotation task with 3 judges. The Cohen's kappa value is 0.694, depicting substantial agreement. The final label is obtained from the majority vote, generating a total of 9076 instances (with 2586 positive/commitment labels and 6490 negative labels). The dataset was split as 60% for training, 20% for validation and 20% for testing The classifier is an RNN-based model with word embeddings and self-attention geared for binary classification with the input being the entire email context [154]. The classifier has a precision of 86% and recall of 84% on the test set.

[3]Avocado is a more appropriate test bed than the Enron collection [76] since it contains additional meta-data and it entered the public domain via the cooperation and consent of the legal owner of the corpus.

5.3.2 To-Do Item Annotation

Candidate emails: We extracted $500k$ raw sentences from Avocado emails and passed them through the commitment classifier. We threshold the commitment classifier confidence to 0.9 and obtained $29k$ potential candidates for To-Do items. Of these, a random subset of $12k$ instances were selected for annotation.

Annotation guideline: For each candidate email e_c and the previous email in the thread e_p (if present), we obtained meta-data like '*From*', '*Sent-To*', '*Subject*' and '*Body*'. The commitment sentence in e_c was highlighted and annotators were asked to write a To-Do item using all of the information in e_c and e_p. The annotators could decide if for the given data instance a To-Do item can be written or not. Sometimes, there was missing data or lack of sufficient information to write a To-Do item. Sometimes, the commitment classifier picked sentences which were not commitments.

We prepared a comprehensive guideline to help human annotators write To-Do Items containing the definition and structure of To-Do Items and commitment sentences, along with illustrative examples. Annotators were instructed to use words and phrases from the email context as closely as possible and introduce new vocabulary only when required. Each instance was annotated by 2 judges.

Analysis of human annotations: Out of $12k$ instances, we obtained a total of 9349 email instances where a To-Do item was written by both annotators. In the remaining cases, either one or both judges felt that there was no To-Do item to write. The written To-Do items have a median token length of 9 and a mean length of 9.71. Given the email meta-data, the annotators were also asked whether the *subject* information was helpful in writing the To-Do Item. For 60.42% of the candidate emails, both annotators agreed that the subject line was helpful in writing the To-Do Item.

To analyze the annotation quality, we randomly sampled 100 annotated To-Do items and asked a judge to rate them on (a) *fluency* (grammatical and spelling correctness), and (b) *completeness* (capturing all the action items in the email) on a 4 point scale (1: Poor, 2:

One possible To-Do item	Update our quarterly sales in the head-office financial database.
Annotation	Update our quarterly sales in the database.
Fluency	4 (Grammatically correct, follows structure of To-Do item.)
Completeness	1 (Which *database* ? Does not include additional details available from email context.)
One possible To-Do item	Test the server for load fault on Friday morning PST and let Bob know the result.
Annotation	Testing on server load fault on Friday morning PST and let Bob know the result.
Fluency	2 (Grammatically incorrect; starts with 'ing' verb and deviates from To-Do structure.)
Completeness	4 (Explains the context and contains all keywords)

Table 5.1: Snapshot of qualitative analysis of human annotations for fluency and completeness.

Fair, 3: Good, 4: Excellent). Overall, we obtained a mean rating of 3.1 and 2.9 respectively for fluency and completeness. Table 5.1 shows a snapshot of the analysis.

5.4 Smart To-Do : Two Stage Generation

In this section, we describe our two-stage approach to generate To-Do items. In the first stage, we select sentences that are *helpful* in writing the To-Do item. Emails contain generic sentences such as salutations, thanks and casual conversations not relevant to the commitment task. The objective of the first stage is to select sentences containing informative concepts necessary to write the To-Do.

5.4.1 Identifying Helpful Sentences for Commitment Task

In the absence of reliable labels to extract helpful sentences in a supervised fashion, we resort to an unsupervised matching-based approach. Let the commitment sentence in the email be denoted as $\mathcal{H}$, and the rest of the sentences from the current email e_c and previous email e_p be denoted as $\{s_1, s_2, \ldots s_d\}$. The unsupervised approach seeks to obtain a relevance score $\Omega(s_i)$ for each sentence. The top K sentences with the highest scores will be selected as the extractive summary for the commitment sentence (also referred to as the query).

Enriched query context: We first extract top τ maximum frequency tokens from all the sentences in the given email, the commitment and the subject (i.e., $\{s_1, s_2, \ldots s_d\} \cup \mathcal{H} \cup$ Subject). Tokens are lemmatized and stop-words are removed. We set $\tau = 10$ in our experiments. An enriched context for the query $\mathcal{E}$ is formed by concatenating the commitment sentence $\mathcal{H}$, subject and top τ tokens.

Relevance score computation: Task-specific relevance score Ω for a sentence s_i is obtained by inner product in the embedding space with the enriched context. Let $h(\cdot)$ be the function denoting the embedding of a sentence with $\Omega(s_i) = h(s_i)^T h(\mathcal{E})$.

Our objective is to find helpful sentences for the commitment given by semantic similarity between concepts in the enriched context and a target sentence. In case of a short or less informative query, the subject and topic of the email provide useful information via the enriched context. We experiment with three different embedding functions.

(1) Term-frequency (Tf) – The binarized term frequency vector is used to represent the sentence.

(2) FastText Word Embeddings – We trained FastText embeddings [14] of dimension 300 on all sentences in the Avocado corpus. The embedding function $h(s_j)$ is given by taking the max (or mean) across the word-embedding dimension of all tokens in the sentence s_j. Although the training does not contain information combining email sentences together, it pertains to email structure and co-occurrence of words at sentence level. The embedding function $h(s_j)$ is then obtained by taking the max(or mean) across word-embedding dimen-

sions of all tokens in the sentence s_j.

(3) Contextualized Word Embeddings – We utilize recent advances in contextualized representations from pre-trained language models like BERT [37]. We use the second last layer of pre-trained BERT for sentence embeddings.

We also fine-tuned BERT on the labeled dataset for commitment classifier. The dataset is first made balanced (2586 positive and 2586 negative instances). Uncased BERT is trained for 5 epochs for commitment classification, with the input being word-piece tokenized email sentences. This model is denoted as BERT (Fine-tuned) in Table 5.2.

Evaluation of unsupervised approaches: Retrieving at-least one helpful sentence is crucial to obtain contextual information for the To-Do item. Therefore, we evaluate our approaches based on the proportion of emails where at-least one helpful sentence is present in the top K retrieved sentences.

We manually annotated 100 email instances and labeled every sentence as *helpful* or not based on (a) whether the sentence contains concepts appearing in the target To-Do item, and (b) whether the sentence helps to understand the task context. Inter-annotator agreement between 2 judgments for this task has a Cohen Kappa score of 0.69. This annotation task also demonstrates the importance of the previous email in a thread. Out of 100 annotated instances, 44 have a replied-to email of which 31 contains a *helpful* sentence in the replied-to email body (70.4%). Table 5.2 shows the performance of the various unsupervised extractive algorithms. FastText with max-pooling of embeddings performed the best and used in the subsequent generation stage.

5.4.2 To-Do Item Generation

The generation phase of our approach can be formulated as sequence-to-sequence (Seq2Seq) learning with attention [148, 5]. It consists of two neural networks, an encoder and a decoder. The input to the encoder consists of concatenated tokens from different meta-data fields of the email like 'sent-to', 'subject', commitment sentence $\mathcal{H}$ and extracted sentences $\mathcal{I}$ separated by special markers. For instance, the input to the encoder for the example in Figure 5.1 is

Algorithm	At-least One Helpful @ K=2	At-least One Helpful @ K=3
Tf	0.80	0.85
FastText (Mean)	0.76	0.90
FastText (Max)	**0.85**	**0.92**
BERT (Pre-trained)	0.76	0.89
BERT (Fine-tuned)	0.80	0.89

Table 5.2: Performance of unsupervised approaches in identifying helpful sentences for a given query.

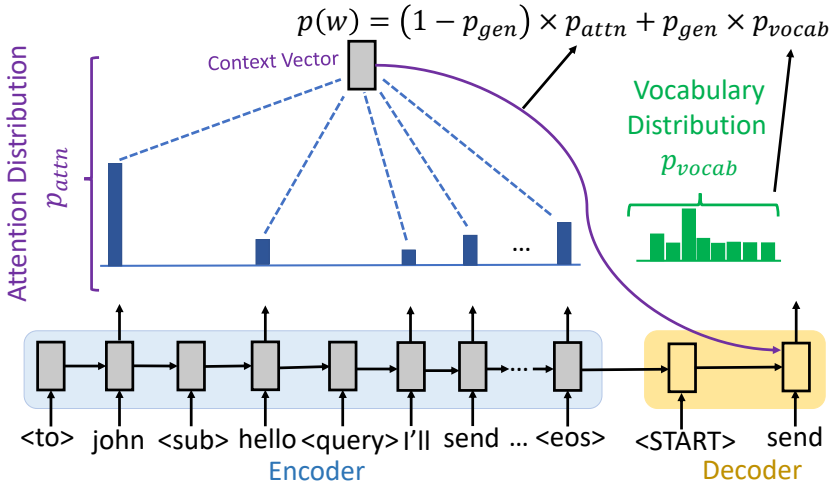


Figure 5.3: Seq2Seq with copy mechanism. Tokens involving named entities and task-specific keywords from the email are learned to copy in the To-Do item.

given as:

<to> alice <sub> hello ? <query> i will send it to you <sent>
could you send me the sales report ? <eos>

We experiment with multiple versions of the generation model as follows:

Vanilla Seq2Seq: Input tokens $\{x_1, x_2, \ldots x_T\}$ are passed through a word-embedding layer and a single layer LSTM to obtain encoded representations $h_t = f(x_t, h_{t-1}) \forall t$ for the input. The decoder is another LSTM that makes use of the encoder state h_t and prior decoder state s_{t-1} to generate the target words at every timestep t. We consider Seq2Seq with attention mechanism where the decoder LSTM uses attention distribution a_t over timesteps t to focus on important hidden states to generate the context vector h_t. This is the first baseline in our work.

$$e_{t,t'} = v^T tanh(W_h \cdot h_t + W_s \cdot s_{t'} + b) \tag{5.1}$$

$$a_{t,t'} = softmax(e_{t,t'}) \tag{5.2}$$

$$h_t = \sum_{t'} a_{t,t'} \cdot h_{t'} \tag{5.3}$$

Seq2Seq with copy mechanism: As the second model, we consider Seq2Seq with copy mechanism [133] to copy tokens from important email fields. Copying is pivotal for To-Do item generation since every task involves named entities in terms of the persons involved, specific times and dates when the task has to be accomplished and other task-specific details present in the email context. To understand the copy mechanism, consider the decoder input at each decoding step as y_t and the context vector as h_t. The decoder at each timestep t has the choice of generating the output word from the vocabulary $\mathcal{V}$ with probability $p_{\text{gen}} = \phi(h_t, s_t, y_t)$, or with probability $1 - p_{\text{gen}}$ it can copy the word from the input context. To allow that, the vocabulary is extended as $\mathcal{V}' = \mathcal{V} \cup \{x_1, x_2, \ldots x_T\}$. The model is trained end-to-end to maximize the log-likelihood of target words (To-Do items) given the email context.

Algorithm	BLEU-4	Rouge-1	Rouge-2	Rouge-L
Concatenate	0.13	0.52	0.28	0.50
Oracle	0.15	0.47	0.27	0.50
Seq2Seq (vanilla)	0.14	0.53	0.31	0.56
Seq2Seq (copy)	**0.23**	**0.60**	**0.41**	**0.63**
Seq2Seq (BiFocal)	0.18	0.56	0.34	0.58
Human Agreement	0.21	0.60	0.37	0.60

Table 5.3: Comparison of various models for To-Do generation with BLEU and ROUGE (higher is better).

Seq2Seq BiFocal: As a third model, we experimented with query-focused attention having two encoders – one containing only tokens of the query and the other containing rest of the input context. We use a bifocal copy mechanism that can copy tokens from either of the encoders. We refer the reader to the Appendix for more details about training and hyper-parameters used in our models.

5.5 Experimental Results

We trained the above neural networks for To-Do item generation on our annotated dataset. Of the 9349 email instances with To-Do items, we used 7349 for training and 1000 each for validation and testing. For each instance, we chose the annotation with fewer tokens as ground-truth reference.

It is important to note here that the commitment sentence is not selected automatically by the Seq2Seq framework. It was already made available by the commitment classifier throughout our two stage algorithm. After the commitment classifier identifies the commitment sentence, our pipeline aims to identify the *helpful* sentences from the email thread and then combine all the meta-data together to obtain the To-Do summary.

The median token length of the encoder input is 43 (including the helpful sentence). Table 5.3 shows the performance comparison of various models. We report BLEU-4 [116] and the F1-scores for Rouge-1, Rouge-2 and Rouge-L [90]. We also report the human performance for this task in terms of the above metrics computed between annotations from the two judges. A trivial baseline – which concatenates tokens from the 'sent-to' and 'subject' fields and the commitment sentence – is included for comparison. The Oracle baseline consists of just the commitment sentence which was selected by the commitment classifier. This metric provides a basis for gauging the improvement obtained by our Smart To-Do pipeline.

The best performance is obtained with Seq2Seq using copying mechanism. We observe our model to perform at par with human performance for writing To-Do items. Table 5.4 shows some examples of To-Do item generation from our best model. We found the query-focused bifocal copy mechanism to be slightly worse than using a single copy-mechanism resulting from the ambiguity on whether to copy from the query or the context at a given time step.

From: John Carter *To:* Helena Watson; Daniel Craig; Rupert Grint *Subject:* Thanks Thank you for helping me prepare the paper draft for ACL conference. Attached is the TeX file. Please feel free to make any changes to the revised version. I sent to my other collaborators already and am waiting for their suggestions. I'll keep you posted. Thanks, John.
GOLD: Keep Helena posted about paper draft for ACL conference.
PRED: Keep Helana posted about ACL conference.
From: Raymond Jiang *To:*support@company.com *Subject:* Bug 62 Hi, there is a periodic bug 62 appearing in my cellphone browser, whenever I choose to open the request. It might be a JavaScript issue on our side, but it would be nice if you take a look. Thanks, Ray.
From: Criag Johnson *To:* Raymond Jiang *Subject:* Bug 62 Good Morning Ray, I shall take a look at it and get back to you.
GOLD: Take a look at Bug 62 and get back to Raymond.
PRED: Take a look at periodic and get back to Raymond.

Table 5.4: Generation example (GOLD: manual annotation, PRED: machine-generated) with email context.

Chapter 6

CONCLUSION AND FUTURE DIRECTIONS

In this thesis, we explored unsupervised learning algorithms using the notion of model-agnostic and model-guided strategies. Our journey led us to the development of ClusterGAN, a novel general purpose clustering algorithm using GANs. Recent advances in using mutual information in clustering urged us to design a stable estimator for mutual information in high dimensions. As a side result, we obtained an estimator for conditional mutual information and used it for conditional independence testing. We also explored application areas for unsupervised representation learning where domain-knowledge can be incorporated in the learning algorithm via an explicit generative model. Finally, a real-world natural language application showed how unsupervised methods may be included in a pipeline and blend gracefully with supervised end-to-end training. There are multiple research directions that arise from this thesis and we discuss them in the subsequent sections.

6.1 Deep Clustering and Interpolation

ClusterGAN applied to datasets with high intra-cluster variability led to clusters that did not align with semantic labels. While this problem has been addressed in recent works [63] [69] by incorporating mutual information, there are still open directions for clustering large scale image datasets. Perhaps the first work in this direction is DeepCluster [21], which showed significant improvement over clustering baselines on ImageNet. The alternating algorithm used by DeepCluster in reminiscent of the auto-encoder based clustering algorithm DEC [156].

Clustering in ImageNet is challenging since many of the images could have multiple co-occurring semantic features. However, the seminal work SN-GAN [101] could sample from

all the 1000 modes in ImageNet. Moreover, StyleGAN [73] has been able to generate high-resolution images with stylized content. The promising advancement in GAN training leads us to the next question:

Research Question 1: *Can we use GANs for clustering in large scale datasets such as ImageNet ?*

Using a Generator-Encoder-Discriminator triplet with SN-GAN could be a viable starting point. But to address the high intra-cluster variability, some information maximization between semantic and pixel patches need to be included as well.

One of the added motivations of using ClusterGAN was its interpolation property in the latent space. So, exploring this unique interpolation property for various applications leads to the next research question.

Research Question 2: *What are the implications of interpolation in latent space for applications such as synthetic molecule discovery and lineage estimate in single cells?*

We demonstrated how moving from one cluster to another generates realistic images, even though samples were not drawn from these portions of the latent space during training. Interesting applications can arise from this property, such as generation of synthetic molecules between clusters with distinct chemical properties. Lineage estimation in the latent space for single cell RNA sequencing is another possibility.

Research Question 3: *Can ClusterGAN achieve state-of-the-art performance for semi-supervised classification?*

The focus of our entire thesis has been unsupervised learning where no labeled information is available for data points. However, in some applications, there may be availability of few labeled samples for each class. Even in the extreme case of 1 labeled sample per class, there is crucial discriminative information that can be leveraged by ClusterGAN to separate out the high-level semantic features.

6.2 Information Estimation

The estimation of mutual information in high dimensions opens up the possibility of using it as a regularizer in representation learning. Recent research [63] has shown promising representation learning in images. But it is yet to be explored for other data modalities.

Research Question 4: *How can mutual information enable better representation learning in natural language tasks ?*

Pointwise mutual information has been used in topic models for capturing collocations and associations between tokens. It remains to be seen whether mutual information maximization between word embedding of tokens and semantic embedding of the sentence provides benefits in representation learning.

Research Question 5: *How to obtain an upper bound of mutual information ?*

The estimator proposed in Chapter 3 was based on lower bounds of mutual information. When we subtract two lower bounds to obtain CMI estimate, we cannot provide any lower or upper bound for CMI. This can be resolved once an upper bound for MI is available. It can have an immense impact on calibrating classifier accuracy, since for a given supervised dataset, the maximum mutual information between data points $\mathcal{X}$ and labels $\mathcal{Y}$ would be known. This will further enable us to characterize what can be the best classification accuracy on the given dataset.

Research Question 6: *Can the mutual information estimator be extended for directed information estimation in time series ? Does it lead to improved causal discovery ?*

An equivalent information theoretic quantity in time series is the directed information. For a finite order Markov Chain, the directed information is indeed a sum of CMI terms. More interesting is the quantity directed information rate, which has a close correspondence with Granger causality. When we condition a time series based on the past values, the dimension of the conditioning variable increases dramatically with the order of the Markov chain. This provides another direction where accurate high dimensional CMI estimation can lead to improvements in estimation of directed information rate.

6.3 Task-focused Summarization

The unique feature in task-focused summarization is that it needs to provide context for the given *query* or task sentence. This is different from summarization of the entire content where a general theme may be prevalent. In task-focused summarization, however, some sentences may be irrelevant to the task. Even though we demonstrated a complete pipeline to achieving the task-focused To-Do item generation, there is plenty of scope for improvement. The unsupervised extractive phase works through semantic matching; but it is not jointly trained with the sequence-to-sequence.

Research Question 6: *Can we combine extractive and abstractive stages for task-focused summarization ?*

The architecture needs to accommodate a variable number of sentences as input. The supervised tokens of the To-Do item would provide end-to-end signal for deciding which sentence is relevant to the task. Such hierarchical classification has been explored in [154], but it is not well explored for sequence-to-sequence tasks. Another general direction is to combine attention weights from two sources at the time of decoding. In a generic attention based architecture, the distribution over all the encoder tokens is considered. But in a query focused situation, it may be necessary to copy (or attend to) different tokens at different times from multiple sources. We are not aware of extensive work on such bi-focal attention mechanism.

BIBLIOGRAPHY

[1] Derek Aguiar and Sorin Istrail. Hapcompass: a fast cycle basis algorithm for accurate haplotype assembly of sequence data. *Journal of computational biology : a journal of computational molecular cell biology*, 19(6):577–90, Jun 2012.

[2] Derek Aguiar and Sorin Istrail. Haplotype assembly in polyploid genomes and identical by descent shared tracts. *Bioinformatics*, 29(13):i352–i360, 2013.

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

[4] Hosein Azarbonyad, Robert Sim, and Ryen W White. Domain adaptation for commitment detection in email. In *Proceedings of the twelfth ACM international conference on web search and data mining*, pages 672–680, 2019.

[5] Dzmitry Bahdanau, Kyunghyun Cho, and Yoshua Bengio. Neural machine translation by jointly learning to align and translate. *ICLR*, 2014.

[6] Nikhil Bansal, Avrim Blum, and Shuchi Chawla. Correlation clustering. *Machine Learning*, 56(1-3):89–113, 2004.

[7] Vikas Bansal and Vineet Bafna. Hapcut: an efficient and accurate algorithm for the haplotype assembly problem. *Bioinformatics*, 24(16):i153–i159, 2008.

[8] Jan Beirlant, Edward J Dudewicz, László Györfi, and Edward C Van der Meulen. Nonparametric entropy estimation: An overview. *International Journal of Mathematical and Statistical Sciences*, 6(1):17–39, 1997.

[9] Mohamed Ishmael Belghazi, Aristide Baratin, Sai Rajeshwar, Sherjil Ozair, Yoshua Bengio, Aaron Courville, and Devon Hjelm. Mutual information neural estimation. In *Proceedings of the 35th International Conference on Machine Learning*, 2018.

[10] 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.

[11] Emily Berger, Deniz Yorukoglu, Jian Peng, and Bonnie Berger. Haptree: A novel bayesian framework for single individual polyplotyping using ngs data. *PLoS Comput Biol*, 10(3):e1003502, 2014.

[12] Konstantin Berlin, Sergey Koren, Chen-Shan Chin, James P Drake, Jane M Landolin, and Adam M Phillippy. Assembling large genomes with single-molecule sequencing and locality-sensitive hashing. *Nature biotechnology*, 33(6):623–630, 2015.

[13] Kush Bhatia, Prateek Jain, and Purushottam Kar. Robust regression via hard thresholding. In *Advances in Neural Information Processing Systems*, pages 721–729, 2015.

[14] Piotr Bojanowski, Edouard Grave, Armand Joulin, and Tomas Mikolov. Enriching word vectors with subword information. *Transactions of the Association for Computational Linguistics*, 5:135–146, 2017.

[15] Paola Bonizzoni, Riccardo Dondi, Gunnar W Klau, Yuri Pirola, Nadia Pisanti, and Simone Zaccaria. On the minimum error correction problem for haplotype assembly in diploid and polyploid genomes. *Journal of Computational Biology*, 2016.

[16] Ashish Bora, Ajil Jalal, Eric Price, and Alexandros G. Dimakis. Compressed sensing using generative models. In *Proceedings of the 34th International Conference on Machine Learning, ICML 2017, Sydney, NSW, Australia, 6-11 August 2017*, pages 537–546, 2017.

[17] Changxiao Cai, Sujay Sanghavi, and Haris Vikalo. Structured low-rank matrix factorization for haplotype assembly. *J. Sel. Topics Signal Processing*, 10(4):647–657, 2016.

[18] Jian-Feng Cai, Emmanuel J Candès, and Zuowei Shen. A singular value thresholding algorithm for matrix completion. *SIAM Journal on Optimization*, 20(4):1956–1982, 2010.

[19] Emmanuel J. Candès and Benjamin Recht. Exact matrix completion via convex optimization. *Commun. ACM*, 55(6):111–119, 2012.

[20] Giuseppe Carenini, Raymond T Ng, and Xiaodong Zhou. Summarizing email conversations with clue words. In *Proceedings of the 16th international conference on World Wide Web*, pages 91–100. ACM, 2007.

[21] Mathilde Caron, Piotr Bojanowski, Armand Joulin, and Matthijs Douze. Deep clustering for unsupervised learning of visual features. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pages 132–149, 2018.

[22] Vitor R Carvalho and William W Cohen. On the collective classification of email speech acts. In *Proceedings of the 28th annual international ACM SIGIR conference on Research and development in information retrieval*, pages 345–352. ACM, 2005.

[23] Mark J Chaisson. Blasr, 2016.

[24] Mark J Chaisson, Sudipto Mukherjee, Sreeram Kannan, and Evan E Eichler. Resolving multicopy duplications de novo using polyploid phasing. In *International Conference on Research in Computational Molecular Biology*, pages 117–133. Springer, 2017.

[25] Moses Charikar, Venkatesan Guruswami, and Anthony Wirth. Clustering with qualitative information. In *Foundations of Computer Science, 2003. Proceedings. 44th Annual IEEE Symposium on*, pages 524–533. IEEE, 2003.

[26] Mia Xu Chen, Benjamin N. Lee, Gagan Bansal, Yuan Cao, Shuyuan Zhang, Justin Lu, Jackie Tsay, Yinan Wang, Andrew M. Dai, Zhifeng Chen, Timothy Sohn, and Yonghui Wu. Gmail smart compose: Real-time assisted writing. In *Proceedings of the 25th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*, KDD '19. ACM, 2019.

[27] Xi Chen, Yan Duan, Rein Houthooft, John Schulman, Ilya Sutskever, and Pieter Abbeel. Infogan: Interpretable representation learning by information maximizing generative adversarial nets. In *Advances in Neural Information Processing Systems*, pages 2172–2180, 2016.

[28] Yuxin Chen and Emmanuel Candes. The projected power method: An efficient algorithm for joint alignment from pairwise differences. *arXiv preprint arXiv:1609.05820*, 2016.

[29] Yuxin Chen, Govinda Kamath, Changho Suh, and David Tse. Community recovery in graphs with locality. *arXiv preprint arXiv:1602.03828*, 2016.

[30] Yuxin Chen, Govinda Kamath, Changho Suh, and David Tse. Community recovery in graphs with locality. In *International Conference on Machine Learning*, pages 689–698, 2016.

[31] Adam Coates, Andrew Ng, and Honglak Lee. An analysis of single-layer networks in unsupervised feature learning. In *Proceedings of the fourteenth international conference on artificial intelligence and statistics*, pages 215–223, 2011.

[32] Simon Corston-Oliver, Eric Ringger, Michael Gamon, and Richard Campbell. Task-focused summarization of email. In *Text Summarization Branches Out*. Association for Computational Linguistics, 2004.

[33] Antonia Creswell and Anil A Bharath. Inverting the generator of a generative adversarial network (ii). *arXiv preprint arXiv:1802.05701*, 2018.

[34] Shreepriya Das and Haris Vikalo. Sdhap: Haplotype assembly for diploids and polyploids via semi-definite programming. *BMC Genomics*, 16(1), 4 2015.

[35] Erik D Demaine and Nicole Immorlica. Correlation clustering with partial information. In *Approximation, Randomization, and Combinatorial Optimization.. Algorithms and Techniques*, pages 1–13. Springer, 2003.

[36] Megan Y Dennis, Xander Nuttle, Peter H Sudmant, Francesca Antonacci, Tina A Graves, Mikhail Nefedov, Jill A Rosenfeld, Saba Sajjadian, Maika Malig, Holland Kotkiewicz, et al. Evolution of human-specific neural srgap2 genes by incomplete segmental duplication. *Cell*, 149(4):912–922, 2012.

[37] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. Bert: Pre-training of deep bidirectional transformers for language understanding. In *NAACL-HLT*, 2019.

[38] Jeff Donahue, Philipp Krähenbühl, and Trevor Darrell. Adversarial feature learning. *arXiv preprint arXiv:1605.09782*, 2016.

[39] G Doran, K Muandet, K Zhang, and B Schölkopf. A permutation-based kernel conditional independence test. In *30th Conference on Uncertainty in Artificial Intelligence (UAI 2014)*.

[40] Mark Dredze, Hanna M Wallach, Danny Puller, and Fernando Pereira. Generating summary keywords for emails using topics. In *Proceedings of the 13th international conference on Intelligent user interfaces*, pages 199–206. ACM, 2008.

[41] Vincent Dumoulin, Ishmael Belghazi, Ben Poole, Olivier Mastropietro, Alex Lamb, Martin Arjovsky, and Aaron Courville. Adversarially learned inference. *arXiv preprint arXiv:1606.00704*, 2016.

[42] Peter Edge, Vineet Bafna, and Vikas Bansal. Hapcut2: robust and accurate haplotype assembly for diverse sequencing technologies. *Genome research*, 27(5):801–812, 2017.

[43] Evan E Eichler. Recent duplication, domain accretion and the dynamic mutation of the human genome. *TRENDS in Genetics*, 17(11):661–669, 2001.

[44] Dotan Emanuel and Amos Fiat. Correlation clustering–minimizing disagreements on arbitrary weighted graphs. In *European Symposium on Algorithms*, pages 208–220. Springer, 2003.

[45] Doris Entner and Patrik O Hoyer. Estimating a causal order among groups of variables in linear models. In *International Conference on Artificial Neural Networks*, pages 84–91. Springer, 2012.

[46] Tanya Feddern-Bekcan. Google calendar. *Journal of the Medical Library Association: JMLA*, 96(4):394, 2008.

[47] François Fleuret. Fast binary feature selection with conditional mutual information. *Journal of Machine learning research*, 5(Nov):1531–1555, 2004.

[48] Santo Fortunato. Community detection in graphs. *Physics reports*, 486(3):75–174, 2010.

[49] Stefan Frenzel and Bernd Pompe. Partial mutual information for coupling analysis of multivariate time series. *Physical review letters*, 99(20):204101, 2007.

[50] Shuyang Gao, Greg Ver Steeg, and Aram Galstyan. Efficient estimation of mutual information for strongly dependent variables. In *Artificial Intelligence and Statistics*, pages 277–286, 2015.

[51] Weihao Gao, Sreeram Kannan, Sewoong Oh, and Pramod Viswanath. Estimating mutual information for discrete-continuous mixtures. In *Advances in Neural Information Processing Systems*, pages 5988–5999, 2017.

[52] Weihao Gao, Sewoong Oh, and Pramod Viswanath. Breaking the bandwidth barrier: Geometrical adaptive entropy estimation. In *Advances in Neural Information Processing Systems*, pages 2460–2468, 2016.

[53] Weihao Gao, Sewoong Oh, and Pramod Viswanath. Demystifying fixed k-nearest neighbor information estimators. *IEEE Transactions on Information Theory*, 64(8):5629–5661, 2018.

[54] Francesca Di Giallonardo, Armin Töpfer, Melanie Rey, Sandhya Prabhakaran, Yannick Duport, Christine Leemann, Stefan Schmutz, Nottania K Campbell, Beda Joos, Maria Rita Lecca, et al. Full-length haplotype reconstruction to infer the structure of heterogeneous virus populations. *Nucleic acids research*, 42(14):e115–e115, 2014.

[55] Federico M Giorgi, Gonzalo Lopez, Jung H Woo, Brygida Bisikirska, Andrea Califano, and Mukesh Bansal. Inferring protein modulation from gene expression data using conditional mutual information. *PloS one*, 9(10):e109569, 2014.

[56] Ian Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio. Generative adversarial nets. In *Advances in neural information processing systems*, pages 2672–2680, 2014.

[57] David Gordon, John Huddleston, Mark JP Chaisson, Christopher M Hill, Zev N Kronenberg, Katherine M Munson, Maika Malig, Archana Raja, Ian Fiddes, LaDeana W Hillier, et al. Long-read sequence assembly of the gorilla genome. *Science*, 352(6281):aae0344, 2016.

[58] Ishaan Gulrajani, Faruk Ahmed, Martin Arjovsky, Vincent Dumoulin, and Aaron C Courville. Improved training of wasserstein gans. In *Advances in Neural Information Processing Systems*, pages 5769–5779, 2017.

[59] Chuan Guo, Geoff Pleiss, Yu Sun, and Kilian Q Weinberger. On calibration of modern neural networks. In *Proceedings of the 34th International Conference on Machine Learning-Volume 70*, pages 1321–1330. JMLR. org, 2017.

[60] Swaminathan Gurumurthy, Ravi Kiran Sarvadevabhatla, and R Venkatesh Babu. Deligan: Generative adversarial networks for diverse and limited data. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 166–174, 2017.

[61] Maria Halkidi, Yannis Batistakis, and Michalis Vazirgiannis. On clustering validation techniques. *Journal of intelligent information systems*, 17(2-3):107–145, 2001.

[62] Martin Heusel, Hubert Ramsauer, Thomas Unterthiner, Bernhard Nessler, and Sepp Hochreiter. Gans trained by a two time-scale update rule converge to a local nash equilibrium. In *Advances in Neural Information Processing Systems*, pages 6626–6637, 2017.

[63] R Devon Hjelm, Alex Fedorov, Samuel Lavoie-Marchildon, Karan Grewal, Phil Bachman, Adam Trischler, and Yoshua Bengio. Learning deep representations by mutual information estimation and maximization. *arXiv preprint arXiv:1808.06670*, 2018.

[64] Jaroslav Hlinka, David Hartman, Martin Vejmelka, Jakob Runge, Norbert Marwan, Jürgen Kurths, and Milan Paluš. Reliability of inference of directed climate networks using conditional mutual information. *Entropy*, 15(6):2023–2045, 2013.

[65] Kurt Hornik, Maxwell Stinchcombe, and Halbert White. Multilayer feedforward networks are universal approximators. *Neural networks*, 2(5):359–366, 1989.

[66] Andrew Ilyas, Ajil Jalal, Eirini Asteri, Constantinos Daskalakis, and Alexandros G Dimakis. The robust manifold defense: Adversarial training using generative models. *arXiv preprint arXiv:1712.09196*, 2017.

[67] Prateek Jain, Praneeth Netrapalli, and Sujay Sanghavi. Low-rank matrix completion using alternating minimization. In *Proceedings of the Forty-fifth Annual ACM Symposium on Theory of Computing*, STOC '13, pages 665–674, New York, NY, USA, 2013. ACM.

[68] Pan Ji, Tong Zhang, Hongdong Li, Mathieu Salzmann, and Ian Reid. Deep subspace clustering networks. In *Advances in Neural Information Processing Systems*, pages 23–32, 2017.

[69] Xu Ji, João F Henriques, and Andrea Vedaldi. Invariant information clustering for unsupervised image classification and segmentation. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 9865–9874, 2019.

[70] Jiantao Jiao, Weihao Gao, and Yanjun Han. The nearest neighbor information estimator is adaptively near minimax rate-optimal. In *Advances in neural information processing systems*, 2018.

[71] Kirthevasan Kandasamy, Akshay Krishnamurthy, Barnabas Poczos, Larry Wasserman, et al. Nonparametric von mises estimators for entropies, divergences and mutual informations. In *Advances in Neural Information Processing Systems*, pages 397–405, 2015.

[72] Anjuli Kannan, Karol Kurach, Sujith Ravi, Tobias Kaufmann, Andrew Tomkins, Balint Miklos, Greg Corrado, Laszlo Lukacs, Marina Ganea, Peter Young, et al. Smart reply: Automated response suggestion for email. In *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 955–964. ACM, 2016.

[73] Tero Karras, Samuli Laine, and Timo Aila. A style-based generator architecture for generative adversarial networks. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 4401–4410, 2019.

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

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

[76] Bryan Klimt and Yiming Yang. The enron corpus: A new dataset for email classification research. In *European Conference on Machine Learning*, pages 217–226. Springer, 2004.

[77] Sergey Koren, Brian P Walenz, Konstantin Berlin, Jason R Miller, and Adam M Phillippy. Canu: scalable and accurate long-read assembly via adaptive k-mer weighting and repeat separation. *bioRxiv*, page 071282, 2016.

[78] LF Kozachenko and Nikolai N Leonenko. Sample estimate of the entropy of a random vector. *Problemy Peredachi Informatsii*, 23(2):9–16, 1987.

[79] Alexander Kraskov, Harald Stögbauer, and Peter Grassberger. Estimating mutual information. *Physical review E*, 69(6):066138, 2004.

[80] Balaji Lakshminarayanan, Alexander Pritzel, and Charles Blundell. Simple and scalable predictive uncertainty estimation using deep ensembles. In *Advances in Neural Information Processing Systems*, pages 6402–6413, 2017.

[81] Andrew Lampert, Robert Dale, and Cecile Paris. Detecting emails containing requests for action. In *Human Language Technologies: The 2010 Annual Conference of the North American Chapter of the Association for Computational Linguistics*, pages 984–992. Association for Computational Linguistics, 2010.

[82] Giuseppe Lancia, Vineet Bafna, Sorin Istrail, Ross Lippert, and Russell Schwartz. Snps problems, complexity, and algorithms. In *European symposium on algorithms*, pages 182–193. Springer, 2001.

[83] Daniel D Lee and H Sebastian Seung. Learning the parts of objects by non-negative matrix factorization. *Nature*, 401(6755):788, 1999.

[84] Intae Lee. Sample-spacings-based density and entropy estimators for spherically invariant multidimensional data. *Neural Computation*, 22(8):2208–2227, 2010.

[85] Marek Leśniewicz. Expected entropy as a measure and criterion of randomness of binary sequences. *Przeglad Elektrotechniczny*, 90(1):42–46, 2014.

[86] Samuel Levy, Granger Sutton, Pauline C Ng, Lars Feuk, Aaron L Halpern, Brian P Walenz, Nelson Axelrod, Jiaqi Huang, Ewen F Kirkness, Gennady Denisov, et al. The diploid genome sequence of an individual human. *PLoS Biol*, 5(10):e254, 2007.

[87] Xiaopeng Li, Zhourong Chen, Leonard KM Poon, and Nevin L Zhang. Learning latent superstructures in variational autoencoders for deep multidimensional clustering. 2019.

[88] Zhaohui Li, Gaoxiang Ouyang, Duan Li, and Xiaoli Li. Characterization of the causality between spike trains with permutation conditional mutual information. *Physical Review E*, 84(2):021929, 2011.

[89] Kuo-Ching Liang and Xiaodong Wang. Gene regulatory network reconstruction using conditional mutual information. *EURASIP Journal on Bioinformatics and Systems Biology*, 2008(1):253894, 2008.

[90] Chin-Yew Lin. Rouge: A package for automatic evaluation of summaries. In *Text summarization branches out*, pages 74–81, 2004.

[91] Chu-Cheng Lin, Dongyeop Kang, Michael Gamon, and Patrick Pantel. Actionable email intent modeling with reparametrized rnns. In *Thirty-Second AAAI Conference on Artificial Intelligence*, 2018.

[92] Eugene Lin, Sudipto Mukherjee, and Sreeram Kannan. A deep adversarial variational autoencoder model for dimensionality reduction in single-cell RNA sequencing analysis. *BMC Bioinform.*, 2020.

[93] Zachary C Lipton and Subarna Tripathi. Precise recovery of latent vectors from generative adversarial networks. *arXiv preprint arXiv:1702.04782*, 2017.

[94] Dirk Loeckx, Pieter Slagmolen, Frederik Maes, Dirk Vandermeulen, and Paul Suetens. Nonrigid image registration using conditional mutual information. *IEEE transactions on medical imaging*, 29(1):19–29, 2010.

[95] David Lopez-Paz and Maxime Oquab. Revisiting classifier two-sample tests. *arXiv preprint arXiv:1610.06545*, 2016.

[96] Laurens van der Maaten and Geoffrey Hinton. Visualizing data using t-sne. *Journal of machine learning research*, 9(Nov):2579–2605, 2008.

[97] Alireza Makhzani, Jonathon Shlens, Navdeep Jaitly, Ian Goodfellow, and Brendan Frey. Adversarial autoencoders. *arXiv preprint arXiv:1511.05644*, 2015.

[98] Sebastian Mika, Bernhard Schölkopf, Alex J Smola, Klaus-Robert Müller, Matthias Scholz, and Gunnar Rätsch. Kernel pca and de-noising in feature spaces. In *Advances in neural information processing systems*, pages 536–542, 1999.

[99] Erik G Miller. A new class of entropy estimators for multi-dimensional densities. In *Acoustics, Speech, and Signal Processing, 2003. Proceedings.(ICASSP'03). 2003 IEEE International Conference on*, volume 3, pages III–297. IEEE, 2003.

[100] Mehdi Mirza and Simon Osindero. Conditional generative adversarial nets. *arXiv preprint arXiv:1411.1784*, 2014.

[101] Takeru Miyato, Toshiki Kataoka, Masanori Koyama, and Yuichi Yoshida. Spectral normalization for generative adversarial networks. *arXiv preprint arXiv:1802.05957*, 2018.

[102] Mehryar Mohri, Afshin Rostamizadeh, and Ameet Talwalkar. *Foundations of machine learning*. 2018.

[103] Arnab Kumar Mondal, Arnab Bhattacharya, Sudipto Mukherjee, Prathosh AP, Sreeram Kannan, and Himanshu Asnani. C-mi-gan : Estimation of conditional mutual information using minmax formulation. *Conference on Uncertainty in Artificial Intelligence (UAI)*, 2020.

[104] Abolfazl Motahari, Kannan Ramchandran, David Tse, and Nan Ma. Optimal dna shotgun sequencing: Noisy reads are as good as noiseless reads. *arXiv preprint arXiv:1304.2798*, 2013.

[105] Sudipto Mukherjee, Himanshu Asnani, and Sreeram Kannan. Ccmi: Classifier based conditional mutual information estimation. *Conference on Uncertainty in Artificial Intelligence (UAI)*, 2019.

[106] Sudipto Mukherjee, Himanshu Asnani, Eugene Lin, and Sreeram Kannan. Clustergan: Latent space clustering in generative adversarial networks. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 33, pages 4610–4617, 2019.

[107] Sudipto Mukherjee, Subhabrata Mukherjee, Marcello Hasegawa, Ahmed Hassan Awadallah, and Ryen White. Smart to-do: Automatic generation of to-do items from emails. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*. ACL, 2020.

[108] Eugene W Myers. Toward simplifying and accurately formulating fragment assembly. *Journal of Computational Biology*, 2(2):275–290, 1995.

[109] Gene Myers. Efficient local alignment discovery amongst noisy long reads. In *International Workshop on Algorithms in Bioinformatics*, pages 52–67. Springer, 2014.

[110] Ilya Nemenman, Fariel Shafee, and William Bialek. Entropy and inference, revisited. In *Advances in neural information processing systems*, pages 471–478, 2002.

[111] Andrew Y Ng, Michael I Jordan, and Yair Weiss. On spectral clustering: Analysis and an algorithm. In *Advances in neural information processing systems*, pages 849–856, 2002.

[112] XuanLong Nguyen, Martin J Wainwright, and Michael I Jordan. Estimating divergence functionals and the likelihood ratio by penalized convex risk minimization. In *Advances in neural information processing systems*, pages 1089–1096, 2008.

[113] Alexandru Niculescu-Mizil and Rich Caruana. Obtaining calibrated probabilities from boosting. In *UAI*, 2005.

[114] Douglas Oard, William Webber, David Kirsch, and Sergey Golitsynskiy. Avocado research email collection. In *LDC2015T03. DVD. Philadelphia: Linguistic Data Consortium*, 2015.

[115] Dávid Pál, Barnabás Póczos, and Csaba Szepesvári. Estimation of rényi entropy and mutual information based on generalized nearest-neighbor graphs. In *Advances in Neural Information Processing Systems*, pages 1849–1857, 2010.

[116] Kishore Papineni, Salim Roukos, Todd Ward, and Wei-Jing Zhu. Bleu: a method for automatic evaluation of machine translation. In *Proceedings of the 40th annual meeting on association for computational linguistics*, pages 311–318. Association for Computational Linguistics, 2002.

[117] Pekka Parviainen and Samuel Kaski. Bayesian networks for variable groups. In *Conference on Probabilistic Graphical Models*, pages 380–391, 2016.

[118] Murray Patterson, Tobias Marschall, Nadia Pisanti, Leo van Iersel, Leen Stougie, Gunnar W Klau, and Alexander Schönhuth. Whatshap: Haplotype assembly for future-generation sequencing reads. In *International Conference on Research in Computational Molecular Biology*, pages 237–249. Springer, 2014.

[119] Pavel A. Pevzner. Dna physical mapping and alternating eulerian cycles in colored graphs. *Algorithmica*, 13(1-2):77–105, 1995.

[120] Pavel A Pevzner, Haixu Tang, and Michael S Waterman. An eulerian path approach to dna fragment assembly. *Proceedings of the National Academy of Sciences*, 98(17):9748–9753, 2001.

[121] Ben Poole, Sherjil Ozair, Aäron van den Oord, Alexander A Alemi, and George Tucker. On variational lower bounds of mutual information.

[122] Zrinka Puljiz and Haris Vikalo. Decoding genetic variations: Communications-inspired haplotype assembly. *IEEE/ACM Transactions on Computational Biology and Bioinformatics*, 13(3):518–530, 2016.

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

[124] Sara Radicati and J Levenstein. Email statistics report, 2015-2019. *Radicati Group, Palo Alto, CA, USA, Tech. Rep*, 2015.

[125] Arman Rahimzamani, Himanshu Asnani, Pramod Viswanath, and Sreeram Kannan. Estimators for multivariate information measures in general probability spaces. In *Advances in Neural Information Processing Systems 31*. Curran Associates, Inc., 2018.

[126] Owen Rambow, Lokesh Shrestha, John Chen, and Chirsty Lauridsen. Summarizing email threads. In *Proceedings of HLT-NAACL 2004: Short Papers*, HLT-NAACL-Short '04. Association for Computational Linguistics, 2004.

[127] Benjamin Recht, Maryam Fazel, and Pablo A. Parrilo. Guaranteed minimum-rank solutions of linear matrix equations via nuclear norm minimization. *SIAM Rev.*, 52(3):471–501, August 2010.

[128] Jakob Runge. Conditional independence testing based on a nearest-neighbor estimator of conditional mutual information. In *Proceedings of the Twenty-First International Conference on Artificial Intelligence and Statistics*, 2018.

[129] Karen Sachs, Omar Perez, Dana Pe'er, Douglas A Lauffenburger, and Garry P Nolan. Causal protein-signaling networks derived from multiparameter single-cell data. *Science*, 308(5721):523–529, 2005.

[130] Shibani Santurkar, David Budden, and Nir Shavit. Generative compression. *arXiv preprint arXiv:1703.01467*, 2017.

[131] Simon Scerri, Gerhard Gossen, Brian Davis, and Siegfried Handschuh. Classifying action items for semantic email. In *LREC*, 2010.

[132] Russell Schwartz et al. Theory and algorithms for the haplotype assembly problem. *Communications in Information & Systems*, 10(1):23–38, 2010.

[133] Abigail See, Peter J Liu, and Christopher D Manning. Get to the point: Summarization with pointer-generator networks. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics*, 2017.

[134] Eran Segal, Dana Pe'er, Aviv Regev, Daphne Koller, and Nir Friedman. Learning module networks. *Journal of Machine Learning Research*, 6(Apr):557–588, 2005.

[135] Rajat Sen, Karthikeyan Shanmugam, Himanshu Asnani, Arman Rahimzamani, and Sreeram Kannan. Mimic and classify: A meta-algorithm for conditional independence testing. *arXiv preprint arXiv:1806.09708*, 2018.

[136] Rajat Sen, Ananda Theertha Suresh, Karthikeyan Shanmugam, Alexandros G Dimakis, and Sanjay Shakkottai. Model-powered conditional independence test. In *Advances in Neural Information Processing Systems*, pages 2951–2961, 2017.

[137] Omry Sendik, Dani Lischinski, and Daniel Cohen-Or. Unsupervised k-modal styled content generation. *arXiv preprint arXiv:2001.03640*, 2020.

[138] Jeong-Sun Seo, Arang Rhie, Junsoo Kim, Sangjin Lee, Min-Hwan Sohn, Chang-Uk Kim, Alex Hastie, Han Cao, Ji-Young Yun, Jihye Kim, et al. De novo assembly and phasing of a korean human genome. *Nature*, 2016.

[139] Andrew J Sharp, Devin P Locke, Sean D McGrath, Ze Cheng, Jeffrey A Bailey, Rhea U Vallente, Lisa M Pertz, Royden A Clark, Stuart Schwartz, Rick Segraves, et al. Segmental duplications and copy-number variation in the human genome. *The American Journal of Human Genetics*, 77(1):78–88, 2005.

[140] Hongbo Si, Haris Vikalo, and Sriram Vishwanath. Haplotype assembly: An information theoretic view. In *Information Theory Workshop (ITW), 2014 IEEE*, pages 182–186. IEEE, 2014.

[141] Harshinder Singh, Neeraj Misra, Vladimir Hnizdo, Adam Fedorowicz, and Eugene Demchuk. Nearest neighbor estimates of entropy. *American journal of mathematical and management sciences*, 23(3-4):301–321, 2003.

[142] Shashank Singh and Barnabás Póczos. Exponential concentration of a density functional estimator. In *Advances in Neural Information Processing Systems*, pages 3032–3040, 2014.

[143] Shashank Singh and Barnabás Póczos. Finite-sample analysis of fixed-k nearest neighbor density functional estimators. In *Advances in Neural Information Processing Systems*, pages 1217–1225, 2016.

[144] Kihyuk Sohn, Honglak Lee, and Xinchen Yan. Learning structured output representation using deep conditional generative models. In *Advances in Neural Information Processing Systems 28*. 2015.

[145] Kumar Sricharan, Raviv Raich, and Alfred O Hero. Estimation of nonlinear functionals of densities with confidence. *IEEE Transactions on Information Theory*, 58(7):4135–4159, 2012.

[146] Kumar Sricharan, Dennis Wei, and Alfred O Hero. Ensemble estimators for multivariate entropy estimation. *IEEE transactions on information theory*, 59(7):4374–4388, 2013.

[147] Karyn Meltz Steinberg, Tina Graves-Lindsay, Valerie A Schneider, Mark JP Chaisson, Chad Tomlinson, John L Huddleston, Patrick Minx, Milinn Kremitzki, Derek Albrecht, Vincent Magrini, et al. High-quality assembly of an individual of yoruban descent. *bioRxiv*, page 067447, 2016.

[148] I Sutskever, O Vinyals, and QV Le. Sequence to sequence learning with neural networks. *Advances in NIPS*, 2014.

[149] Taiji Suzuki, Masashi Sugiyama, Jun Sese, and Takafumi Kanamori. Approximating mutual information by maximum likelihood density ratio estimation. In *New challenges for feature selection in data mining and knowledge discovery*, pages 5–20, 2008.

[150] Ilya Tolstikhin, Olivier Bousquet, Sylvain Gelly, and Bernhard Schoelkopf. Wasserstein auto-encoders. *arXiv preprint arXiv:1711.01558*, 2017.

[151] Martin Vejmelka and Milan Paluš. Inferring the directionality of coupling with conditional mutual information. *Physical Review E*, 77(2):026214, 2008.

[152] Pascal Vincent, Hugo Larochelle, Isabelle Lajoie, Yoshua Bengio, and Pierre-Antoine Manzagol. Stacked denoising autoencoders: Learning useful representations in a deep network with a local denoising criterion. *Journal of Machine Learning Research*, 11(Dec):3371–3408, 2010.

[153] Mitchell R Vollger, Philip C Dishuck, Melanie Sorensen, AnneMarie E Welch, Vy Dang, Max L Dougherty, Tina A Graves-Lindsay, Richard K Wilson, Mark JP Chaisson, and Evan E Eichler. Long-read sequence and assembly of segmental duplications. *Nature methods*, 16(1):88–94, 2019.

[154] Wei Wang, Saghar Hosseini, Ahmed Hassan Awadallah, Paul N. Bennett, and Chris Quirk. Context-aware intent identification in email conversations. In *Proceedings of the 42Nd International ACM SIGIR Conference on Research and Development in Information Retrieval*, SIGIR'19, 2019.

[155] Shiming Xiang, Feiping Nie, and Changshui Zhang. Learning a mahalanobis distance metric for data clustering and classification. *Pattern Recognition*, 41(12):3600–3612, 2008.

[156] Junyuan Xie, Ross Girshick, and Ali Farhadi. Unsupervised deep embedding for clustering analysis. In *International conference on machine learning*, pages 478–487, 2016.

[157] Huan Xu, Constantine Caramanis, and Shie Mannor. Robust regression and lasso. In *Advances in Neural Information Processing Systems*, pages 1801–1808, 2009.

[158] Bo Yang, Xiao Fu, Nicholas D. Sidiropoulos, and Mingyi Hong. Towards k-means-friendly spaces: Simultaneous deep learning and clustering. In *Proceedings of the 34th International Conference on Machine Learning, ICML 2017, Sydney, NSW, Australia, 6-11 August 2017*, pages 3861–3870, 2017.

[159] Yao-liang Yu, Özlem Aslan, and Dale Schuurmans. A polynomial-time form of robust regression. In *Advances in Neural Information Processing Systems*, pages 2483–2491, 2012.

[160] Martha A Zaidan, Ville Haapasilta, Rishi Relan, Pauli Paasonen, Veli-Matti Kerminen, Heikki Junninen, Markku Kulmala, and Adam S Foster. Exploring non-linear associations between atmospheric new-particle formation and ambient variables: a mutual information approach. *Atmospheric Chemistry and Physics*, 18(17):12699–12714, 2018.

[161] Rui Zhang and Joel R. Tetreault. This email could save your life: Introducing the task of email subject line generation. In *ACL*, 2019.

[162] Wei Zhang, Xiaogang Wang, Deli Zhao, and Xiaoou Tang. Graph degree linkage: Agglomerative clustering on a directed graph. In *European Conference on Computer Vision*, pages 428–441. Springer, 2012.

[163] Grace XY Zheng, Jessica M Terry, Phillip Belgrader, Paul Ryvkin, Zachary W Bent, Ryan Wilson, Solongo B Ziraldo, Tobias D Wheeler, Geoff P McDermott, Junjie Zhu, et al. Massively parallel digital transcriptional profiling of single cells. *Nature communications*, 8:14049, 2017.

Appendix A

SUPPLEMENTARY MATERIAL FOR CHAPTER 2

A.1 Hyperparameter and Architecture Details

The networks were trained with Adam Optimizer (learning rate $\eta = 1e$-04, $\beta_1 = 0.5$, $\beta_2 = 0.9$) for all datasets. The number of discriminator updates was 5 for each generator update in training. Gradient penalty coefficient for WGAN-GP was set to 10 for all experiments. The dimension of z_c is the same as the number of classes in the dataset. Most networks used Leaky ReLU activations and Batch Normalization (BN), details for each dataset are provided below. (In the architecture without encoder, Algorithm 1 used Adam optimizer to minimize the objective for 5000 iterations per point.)

Synthetic Data

We used batch size = 64, z_n of 6 dimensions. LReLU activation with leak = 0.2 was used. $\beta_n = 10$, $\beta_c = 10$.

Generator	Encoder	Discriminator
Input $z = (z_n, z_c) \in \mathbb{R}^{10}$	Input $X \in \mathbb{R}^{16}$	Input $X \in \mathbb{R}^{16}$
FC 256 LReLU BN	FC 256 LReLU BN	FC 256 LReLU BN
FC 256 LReLU BN	FC 256 LReLU BN	FC 256 LReLU BN
FC 16 Sigmoid	FC 10 linear for $\hat{z}$ Softmax on last 4 to obtain $\hat{z}_c$	FC 1 linear

MNIST and Fashion-MNIST

We used batch size = 64, z_n of 30 dimensions. LReLU activation with leak = 0.2 was used. $\beta_n = 10$ for MNIST and $\beta_n = 0$ for Fashion-MNIST, $\beta_c = 10$ for both .

Generator	Encoder	Discriminator
Input $z = (z_n, z_c) \in \mathbb{R}^{40}$	Input $X \in \mathbb{R}^{28 \times 28}$	Input $X \in \mathbb{R}^{28 \times 28}$
FC 1024 ReLU BN	4×4 conv, 64 stride 2 LReLU	4×4 conv, 64 stride 2 LReLU
FC $7 \times 7 \times 128$ ReLU BN	4×4 conv, 128 stride 2 LReLU	4×4 conv, 128 stride 2 LReLU
4×4 upconv, 64 stride 2 ReLU BN	FC 1024 LReLU	FC 1024 LReLU
4×4 upconv, 1 stride 2, Sigmoid	FC 40 linear for $\hat{z}$ Softmax on last 10 to obtain $\hat{z}_c$	FC 1 linear

For Fashion-MNIST, we used $z_n = 40$. Rest of the architecture remained identical.

10x_73k

We used batch size = 64, z_n of 30 dimensions. LReLU activation with leak = 0.2 was used. $\beta_n = 10$, $\beta_c = 10$.

Generator	Encoder	Discriminator
Input $z = (z_n, z_c) \in \mathbb{R}^{38}$	Input $X \in \mathbb{R}^{720}$	Input $X \in \mathbb{R}^{720}$
FC 256 LReLU	FC 256 LReLU	FC 256 LReLU
FC 256 LReLU	FC 256 LReLU	FC 256 LReLU
FC 720 Linear	FC 38 linear for $\hat{z}$ Softmax on last 8 to obtain $\hat{z}_c$	FC 1 linear

Pendigits

We used batch size = 64, z_n of 5 dimensions. LReLU activation with leak = 0.2 was used. $\beta_n = 10$, $\beta_c = 10$.

Generator	Encoder	Discriminator
Input $z = (z_n, z_c) \in \mathbb{R}^{15}$	Input $X \in \mathbb{R}^{16}$	Input $X \in \mathbb{R}^{16}$
FC 256 LReLU BN	FC 256 LReLU BN	FC 256 LReLU BN
FC 256 LReLU BN	FC 256 LReLU BN	FC 256 LReLU BN
FC 16 Sigmoid	FC 15 linear for $\hat{z}$ Softmax on last 10 to obtain $\hat{z}_c$	FC 1 linear

Coil-20, Coil-100 and CIFAR-10

For Coil-20, we used batch size = 64, z_n of 20 dimensions. For Coil-100, we used batch size = 512, z_n of 20 dimensions. For CIFAR-10, we used batch size = 64, z_n of 50 dimensions. $\beta_n = 10, \beta_c = 10$, LReLU activation with leak = 0.2 for all datasets.

Generator	Encoder	Discriminator
Input $z = (z_n, z_c) \in \mathbb{R}^{d_z}$	Input $X \in \mathbb{R}^{32\times32\times3}$	Input $X \in \mathbb{R}^{32\times32\times3}$
FC $2 \times 2 \times 448$ ReLU BN	4×4 conv, 64 stride 2 LReLU	4×4 conv, 64 stride 2 LReLU
4×4 upconv, 256 stride 2 ReLU BN	4×4 conv, 128 stride 2 LReLU BN	4×4 conv, 128 stride 2 LReLU BN
4×4 upconv, 128 stride 2 ReLU BN	4×4 conv, 256 stride 2 LReLU BN	4×4 conv, 256 stride 2 LReLU BN
4×4 upconv, 64 stride 2 ReLU BN	4×4 conv, 512 stride 2 LReLU BN	4×4 conv, 512 stride 2 LReLU BN
4×4 upconv, 3 stride 2, Sigmoid	FC d_z linear for $\hat{z}$ Softmax on last K to obtain $\hat{z}_c$	FC 1 linear

For InfoGAN, we used the implementation of the authors `https://github.com/openai/InfoGAN` for MNIST and Fashion-MNIST. For the other datasets, we used our hyperparame-

ters for Generator and Discriminator and added the Q network (FC 128-BN-LReLU-FC dim z_c). For "GAN with bp", we used the same Generator and Discriminator hyperparameters as ClusterGAN. Features for "GAN with Disc. ϕ" was obtained from the trained Discriminator of experiments "GAN with bp".

A.2 Reporting Clustering Performance

In [156], the authors ran all algorithms multiple times with a single hyperparamater change and reported the best accuracy. For fair comparison, we used 5 runs to determine the best model using validation. To be more precise, we first split our datasets into Train, Validation and Test portions. The GAN was trained only on the Train split of the data in an unsupervised manner, sometimes with a single hyper-parameter change such as β_n, z_n, leak or batch-norm . For each dataset, we saved the model with the best purity on the Validation split from the 5 runs. Table 4.1 reports the metrics on the Test split for the saved model. The metrics on the entire dataset for the saved model was either identical upto 2 decimals or slightly better than Test. But we report only for the Test split, since it has neither been used for training nor for validation runs.

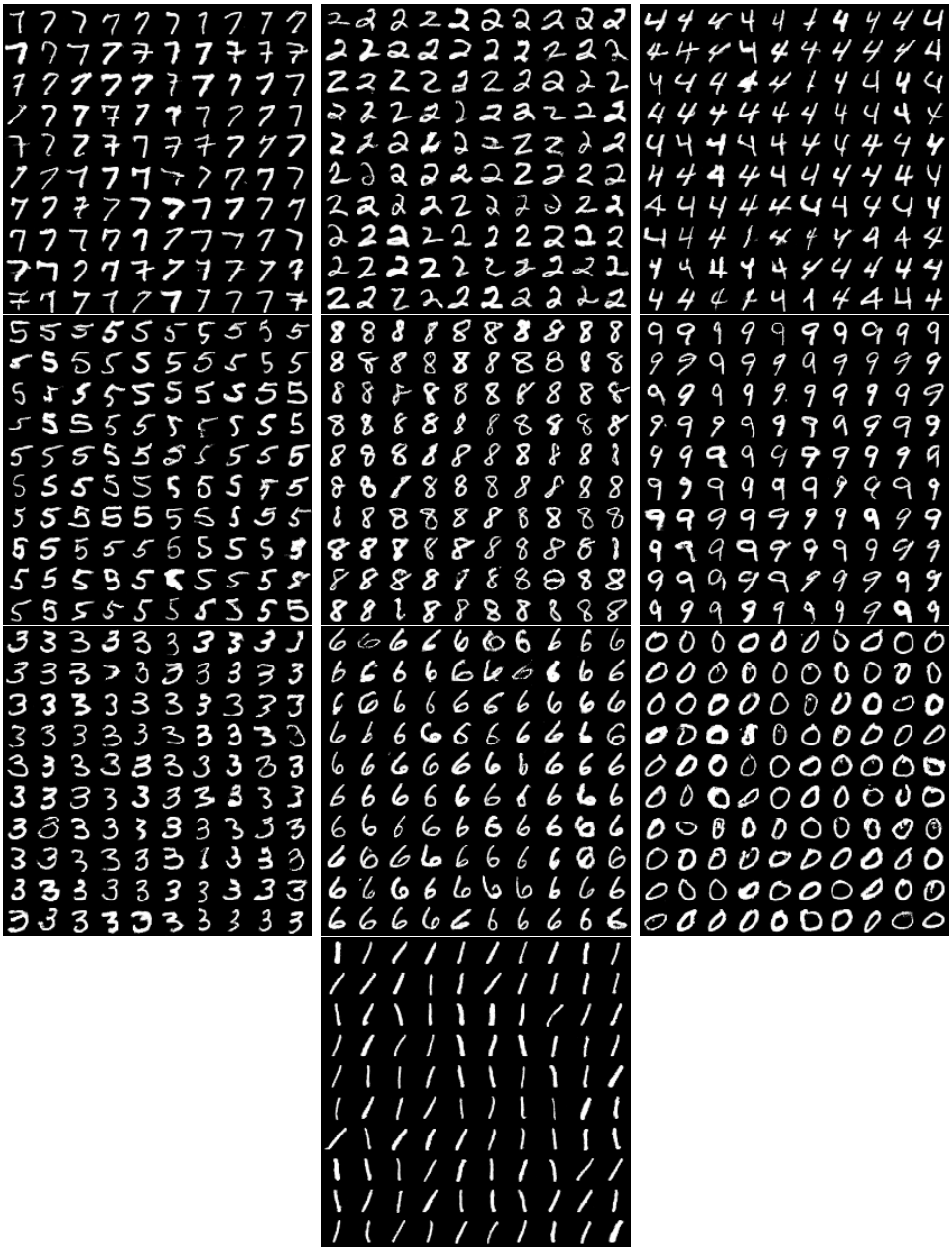


Figure A.1: Generated digits from distinct modes

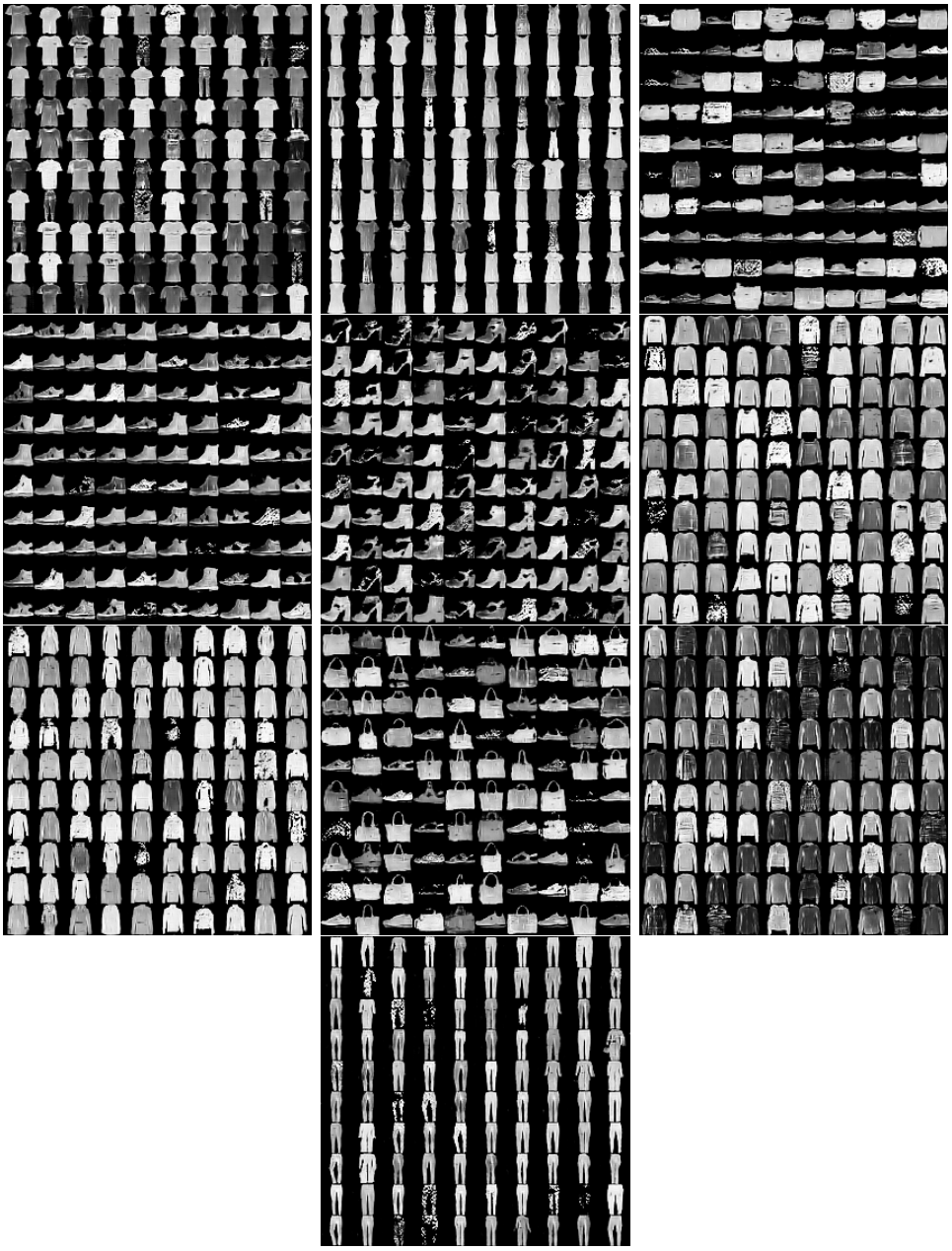


Figure A.2: Generated fashion items from distinct modes

Figure A.3: Generated categories from distinct modes of CIFAR-10

Appendix B

SUPPLEMENTARY MATERIAL FOR CHAPTER 5

B.1 Hyper-parameters

Hyper-parameter	**Value**
Rnn-type	LSTM
Rnn-size	256
# Layers	1
Word-embedding	100
Embedding init.	Glove
Batch size	64
Optimizer	Adagrad
Learning rate	0.15
Adagrad accumulator init.	0.1
Max. Gradient norm	2.0
Dropout	0.5
Attention dropout	0.5
Tokenizer	spacy
Vocabulary	Separate
Early Stopping (Patience)	5
Beam width	5

Table B.1: Seq2Seq with copy mechanism : Hyper-parameters for the best model.

We now provide the hyper-parameters and training details for ease of reproducibility of

our results in Chapter 5. The encoder-decode architecture consists of LSTM units. The word embedding look-up matrix is initialized using Glove embeddings and then trained jointly to adapt to the structure of the problem. We found this step crucial for improved performance. Using random initialization or static Glove embeddings degraded performance.

We also experimented with using either a shared or a separate vocabulary for the encoder and decoder. A token was included in the vocabulary if it occurred at least 2 times in the training input/target. Separate vocabulary for source and target had better performance. Typically, source vocabulary had higher number of tokens than target. A shared dictionary led to increased number of parameters in the decoder and to subsequent over-fitting. The validation data was used for early stopping. The patience was decreased whenever either the validation token accuracy or perplexity failed to improve. We used the OpenNMT framework in PyTorch for all our Seq2Seq experiments.

Table B.1 lists the hyper-parameters of the best performing model.

B.2 Illustrative Examples

In this Section, we provide further examples of the email threads along with the highlighted commitment sentence. Note that some of the emails have previous thread email present, and some do not have it. For each of these examples, we also provide the To-Do item written by the human judge (denoted as GOLD) and that predicted by our best model (denoted as PRED). As in the main Chapter, the sentences have been paraphrased and names changed due to the data sensitivity of Avocado.

From: Beverly Evans *To:* Carlos Simmons *Subject:* Amazon.com update Carlos, I came to know today from John Carter than we received a PHP script that is not decoding the correct database. Can you check with them why they sent us the eCommerce PHP code when the loss of functionality was not out fault? I have registered the error log in the eCommerce section because the staff scientist from Amazon mentioned it in his email. He also said they have not been able to resolve the issue and surprisingly did not mention who we should contact next. (This email exchange was about a week ago when I had handed them the cloud expenditures.) Also, we need to generate a PHP example to replicate the error. Could you update me if the team is working on it? Thanks, Beverly
From: Carlos Simmons *To:* Beverly Evans *Subject:* Amazon.com update The PHP they shared with us is an example. eCommerce is not what they want us to resolve. I feel we should wait until their engineers test all possibilities. Joseph informed us that they need to test the database more carefully and figure out which PHP code to send to us and whether they want our feedback on the database. I am not sure why they sent me a 'relevant PHP example' - I thought there was the only file they sent us yesterday. I will forward that to you and Renata.
GOLD: Forward PHP example to Beverly and Renata.
PRED: Forward eCommerce PHP to Beverly.

Table B.2: Illustrative Example 1

From: Kirstin Barnes *To:* Nannie Jacobs *Subject:* Ready for Product Launch Nannie, I am ready for the product launch. I need to include some of the enhancements in the presentation. I'll submit what is already completed and then do the remaining after the meeting.. Kirstin Barnes Product Engineer AvocadoIT, Inc.
GOLD: Submit presentation with product enhancements.
PRED: Submit the enhancements for product launch.

Table B.3: Illustrative Example 2

From: Rishabh Iyer *To:* R&D *Subject:* Software not ready yet for deployment Hello, Unlike our plan last month, the software is still not ready for deployment. The team put together some errors last week. We must plan to make it available latest by next week. I will keep you posted. Thanks, Rishabh Iyer. Software Engineer AvocadoIT Inc.
GOLD: Keep r&d posted about deployment of software.
PRED: Keep r&d posted about deployment.

Table B.4: Illustrative Example 3

From: Justine Sparrow *To:* Roma Patterson *Subject:* 24x7 Helpline Roma, I will bring this up in the Staff meeing today. I'll let you know the outcome. Could you confirm if this is for a license agreement or a shared solution ? Thanks, Justine.
GOLD: Let Roma know result.
PRED: Let Roma know about the license agreement.

Table B.5: Illustrative Example 4

From: Matthew White *To:* Frank; Paul; Dennis *Subject:* Draft Agenda for Software Training Dear All, As discussed before, we have finally come to a concrete plan. I have attached the draft for your review. Please go over it and let me know asap your suggestions so that I can send them to the organizers. Please check the agenda and the names of trainees. I'll put together the Training plan and the overall 5-day agenda as soon as I can. Matthew.
GOLD: Put together the training plan and the overall day agenda of software training.
PRED: Put together the draft agenda for software training.

Table B.6: Illustrative Example 5

From: Rebecca Anderson *To:* Julia Roberts *Subject:* Run a bash script while synchronize Julia, When synchronizing is done, we want to run a bash script to delete old records on the machine and remove all activity logs. How can I do this ? What is the way to perform this operation ? Also, in the bash script, is there a way to sort the dates so that we can identify older activities ? Thanks, Rebecca.
From: Julia Roberts *To:* Rebecca Anderson *Subject:* Run a bash script while synchronize Rebecca, We had exactly the same feature to delete activities which you mentioned in our previous release. But we no longer have that in the new version due to resource constraints. I will talk to John to review this again. Thanks, Julia.
GOLD: Talk to John to review bash script again.
PRED: Talk to John to review the activities.

Table B.7: Illustrative Example 6

From: Ramesh Paul *To:* Gopal Majumdar *Subject:* Updates List for 3/11 Here's the update for this week. 1. The R&D team is working on a presentation for the knowledge tranfer for v5. It should be ready within next two weeks. 2. I have received their email, but need to review the ppt. 3. Did you want to know more about the new cloud feature for automatic version management ? Or was it a different feature ? 4. I am constantly working on this. 5. Didn't we discuss this point in our last email ? 6. We are making similar tests in the desktop for v5 before migrating to the cloud. We first have to make sure things work well for the desktop. I will send you more details soon. Did you get a chance to update your blog with information about these new features ? Thanks, Ramesh.
GOLD: Send Gopal more details about tests in the desktop for v5.
PRED: Send Gopal more details on presentation for the knowledge transfer.

Table B.8: Illustrative Example 7

From: Lori Howard *To:* Karen James; Bruce Thomas; Steve Perry *Subject:* Room reservations Team, This needs to be done through a formal training session, but as of now let me point out some crucial points about room reservations. 1. In case you allocate a room for general meetings and administrative work, then make sure you book it for that month, but not for long periods of time. (Karen, can you check with Renata whether this is fulfilled for our meetings next week?) 2. In case of clients who do not need the entire month, make sure to reserve only for the particular month. If it exceeds that time, the system will authomatically resolve it and reserve it for next month. 3. For room reservation, either enter the number of hours required or the % of month, but not both. I would prefer precise hours. I will inform you when we can provide training, perhaps we can next week. Thanks, Lori.
GOLD: Let Karen know about the training provide for room reservations.
PRED: Let Karen know about room reservations.

Table B.9: Illustrative Example 8

From: Diana Wilson *To:* Alba Deacon *Subject:* DHL package from IBM Alba, I was able to track the package and as per the website it was in Sao Luis, Brazil at noon. I am not sure where it is, but it is Brazil so ... Send me an update if you receive it from them. I just tracked the package and as of 10:00am today it was in Toluca, Mexico. Where that is I have no idea but it is in Mexico so ... Let me know if you hear from them when they receive it. Thanks. Diana Wilson.
From: Alba Deacon *To:* Diana Wilson *Subject:* DHL package from IBM Thanks Diana. If I hear anything I'll let you know.. Alba.
GOLD: Let Diana know about DHL package from IBM.
PRED: Let Diana know about DHL package from IBM.

Table B.10: Illustrative Example 9