BAYESIAN LEARNING WITH HETEROGENEOUS DATA FOR LIFE SCIENCES

A Dissertation

by

MOHAMMAD EHSAN HAJIRAMEZANALI

Submitted to the Office of Graduate and Professional Studies of
Texas A&M University
in partial fulfillment of the requirements for the degree of

DOCTOR OF PHILOSOPHY

Chair of Committee,	Xiaoning Qian
Committee Members,	Paul de Figueiredo
	Krishna Narayanan
	Aniruddha Datta
Head of Department,	Miroslav M. Begovic

May 2021

Major Subject: Electrical Engineering

ABSTRACT

We propose a suite of Bayesian learning methods to address challenges arising from task and data heterogeneity in life science applications.

First, we develop a novel multi-domain negative binomial (NB) factorization model to analyze next-generation sequencing (NGS) count data, with the goal of enhancing cancer subtyping in the target domain with a limited number of NGS samples by leveraging surrogate data from other cancer types (source domains). In particular, such a Bayesian multi-domain learning (BMDL) method addresses data scarcity issues due to task heterogeneity by learning domain relevance through common latent factors based on given samples across domains. It automatically avoids "negative transfer", to which many existing transfer learning methods are amenable, and performs consistently better than single-domain learning regardless of the domain relevance level.

In addition to study task heterogeneity, investigating longitudinal heterogeneity of temporal NGS count data may help to better understand the underlying cellular mechanisms of living systems. We propose gamma Markov negative binomial (GMNB) as a fully Bayesian solution to study temporal RNA-seq data. A notable advantage is the capacity to capture a broad range of gene expression patterns over time by integrating a gamma Markov chain into the NB distribution model. We then adopt the Bayes Factor (BF) as a measure that exploits information collectively from all time points to detect the genes with significant variations in temporal expression patterns across phenotypes or treatment conditions.

Moving to more complicated experimental settings, we propose variational graph recurrent neural network (VGRNN) that combines additional structural heterogeneity to the longitudinal data. The use of high-level latent random variables in VGRNN can better capture potential variability observed in dynamic graphs as well as the uncertainty of node latent representations, with graphs capturing prior knowledge on dependency relationships. We further develop semi-implicit variational inference for this new VGRNN architecture (SI-VGRNN) to allow flexible non-Gaussian latent representations.

Finally, in the last chapter, we propose a novel Bayesian relation learning framework, BayReL, that infers interactions across different heterogeneous input datasets as different views from different types of bio-molecules, aiming at deriving meaningful biological knowledge for integrative multi-omics data analysis. BayReL can flexibly incorporate the available graph dependency structure of each view, exploits non-linear transformations, and provides probabilistic interpretation simultaneously.

DEDICATION

To Sahar – my love and life's partner

and

Toranj – whose dazzling light makes everything brighter

ACKNOWLEDGMENTS

First and foremost, I would like to thank my beloved, Sahar, for all of her sacrifices during my PhD. Her patience, love, and encouraging attitude have been always driving me to seek the best quality of work. I would like to sincerely thank my academic advisor, Professor Xiaoning Qian, for his help, advice, and unconditional encouragement and support. He was more than generous with his expertise and precious time. I also thank Professor Mingyuan Zhou for his constructive comments and good-natured support. It is my pleasure to also appreciate my family, whose constant support led me to this path and made this dissertation the beginning of my professional journey.

CONTRIBUTORS AND FUNDING SOURCES

Contributors

This work was supported by a dissertation committee consisting of Professor Xiaoning Qian [advisor] and Professors Krishna Narayanan and Aniruddha Datta of the Department of Electrical and Computer Engineering and Professor Paul de Figueiredo of the Department of Microbial Pathogenesis and Immunology.

All work for the dissertation was completed by the student, under the advisement of Professor Xiaoning Qian of the Department of Electrical and Computer Engineering.

Funding Sources

My work was supported by the National Science Foundation, through NSF awards CCF-1553281 and IIS-1812641.

NOMENCLATURE

NGS	Next-Generation Sequencing
RNA-seq	RNA Sequencing
GO	Gene Ontology
NB	Negative Binomial
CRT	Chinese Restaurant Process
GNN	Graph Neural Network
GMNB	Gamma Markov Negative Binomial
BMDL	Bayesian Multi-Domain Learning
DP	Dirichlet Process
HDP	Hierarchical Dirichlet Process
hGNBP	Hierarchical Gamma-Negative Binomial Process
NBFA	Negative Binomial Factor Analysis
IBP	Indian Buffet Process
MCMC	Markov Chain Monte Carlo
MTL	Multi-task Learning
TL	Transfer Learning
DA	Domain Adaptation
BF	Bayes Factor
PMF	Probability Mass Function
AR	Auto-regressive
ROC	Receiver Operating Characteristic
PR	Precision-Recall

AUC	Area Under the Curve
BayReL	Bayesian Relational Learning
GRNN	Graph Recurrent Neural Network
VGRNN	Variational Graph Recurrent Neural Network
VRNN	Variational Recurrent Neural Network
SI-VGRNN	Semi-implicit Variational Graph Recurrent Neural Network
GCN	Graph Convolutional Networks
VGAE	Variational Graph Autoencoder
Th17	T Helper 17
GCRN	Graph Convolutional Recurrent Neural Networks
LSTM	Long Short-term Memory
GRU	Gated Recurrent Units
SIVI	Semi-implicit Variational Inference
ELBO	Evidence Lower Bound
CCA	Canonical Correlation Analysis
PCCA	Probabilistic Canonical Correlation Analysis
BCCA	BayesianCanonical Correlation Analysis
SRCA	Spearman's Rank Correlation Analysis
FCNN	Fully Connected Neural Networks
CF	Cystic Fibrosis
rRNA	ribosomal RNA
TCGA	The Cancer Genome Atlas
BRCA	Breast Cancer
GRN	Gene Regulatory Networks
AML	Acute Myeloid Leukemia

TABLE OF CONTENTS

Page

LIST OF FIGURES

FIGURE Page

LIST OF TABLES

TABLE Page

1. INTRODUCTION

Demand for learning in biomedicine is higher than ever before. If successful, machine learning and artificial intelligence (ML/AI) can have significant impact in human society. Modern high-throughput biological technologies have produced rich high-dimensional biomedical data at different molecular levels, such as genome, epigenome, transcriptome, translatome, proteome, metabolome and interactome. Although such multi-view data provide views covering a diverse range of cellular activities, developing an understanding of how these data types quantitatively relate to each other and more critically the phenotypic characteristics of interest remains elusive. On the other hand, life and disease systems are highly complex, dynamic, stochastic, and heterogeneous that involve not only the within-level interaction but also the between-level regulation. The key question is how we may integrate multiple data types for deriving better insights into the underlying biological mechanisms. Due to the heterogeneity and high-dimensional nature of multi-omic data, it is necessary to develop effective and affordable learning methods for their integration and analysis.

Traditional machine leaning often assumes that training and testing data are homogeneous and/or come from identical distributions. However, in real-world applications, "heterogeneity" has been one of the major hurdles for machine learning to achieve generalizable predictions [2]. For example, the target application may consist of multiple heterogeneous data sets with different classes/tasks (task heterogeneity); each sample may be characterized with features from multiple sources of potentially mixed data types (e.g. categorical, binary, count, continuous, and etc.) (view heterogeneity); the data may be collected in different and irregular time-points (longitudinal heterogeneity) or under different experimental conditions (context heterogeneity); last but not least, data dependency may be manifested by heterogeneous graph relationships (structural heterogeneity).

Different machine learning methods have been proposed to address the above potential challenges. However, they still suffer from multiple shortcomings when being applied to life science applications: 1) Existing machine learning methods are not specifically designed to deal with biological profiling technologies, such as next-generation sequencing that produces special over-dispersed

count data; 2) The data-hungry nature of many of these methods makes them not applicable in life science as often the number of available training samples is much smaller than the number of features (molecular measurements), making incorporation of *a priori* known structured knowledge necessary in their learning; 3) Most of the available methods focus on addressing issues due to a single type of heterogeneity.

Through this dissertation, we propose a suite of Bayesian learning frameworks to address challenges arising from task and data heterogeneity in life science applications. We first try to propose Bayesian learning specifically designed for analysis of heterogeneous RNA sequencing (RNA-seq) count data. We further incorporate prior knowledge as graph structured data in our analysis. Last but not least, we develop a Bayesian model to combine different kinds of heterogeneous datasets in multi-omics data integrative analysis for studying interactions across different types of molecules to reveal, for example signal transduction mechanisms. In particular, the following issues have been addressed throughout the dissertation:

• **Bayesian multi-domain learning (BMDL):** First, in Chapter 2, we analysis count data from next-generation sequencing (NGS) experiments, with the goal of enhancing cancer subtyping in the target domain with a limited number of NGS samples by leveraging surrogate data from other domains, for example relevant data from other well-studied cancer types. We develop a novel multi-domain negative binomial (NB) factorization model for data with task heterogeneity as the different cancer types may consist of multiple heterogeneous data sets with different classes. The proposed method, directly applied to over-dispersed RNA-seq count data, obviates the need for multiple ad-hoc preprocessing steps as required in most of of gene expression analyses. Additionally, by introducing domain-dependent binary variables that assign latent factors to each domain, it explicitly learns the sample relevance across domains to guarantee the effectiveness of joint learning across multiple heterogeneous domains. Experimental results on both synthetic and real-world NGS datasets demonstrate the state-of-the-art performance of BMDL for effective multi-domain learning without "negative transfer" effects often seen in existing multi-task learning and transfer learning methods.

• **Gamma Markov Chain Differential Expression Analysis:** In Chapter 3, we propose a hierarchical model that integrates a gamma Markov chain into a negative binomial distribution, to model longitudinal heterogeneity in temporal RNA sequencing count data. We further derive an efficient Gibbs sampling with closed-form updating steps to infer the model parameters in GMNB. Using Bayes factors, GMNB enables more powerful temporal gene differential expression analysis across various phenotypes or treatment conditions. Additionally, the proposed method naturally handles the heterogeneity of sequencing depth in different samples. Extensive experiments on both simulated and real-world RNA-seq data demonstrate GMNB is flexible and powerful in detecting any changes (smooth or abrupt) over time and/or between time points. It helps to identify several known genes involved in Th17 differentiation as well as some new potential genetic markers when applied to the temporal NGS data of human T-cells.

• **Variational Graph Recurrent Neural Networks (VGRNN):** Many emerging high impact applications exhibit the coexistence of multiple types of heterogeneity [3]. For example, scientists can gather structured data in different and irregular time-points including number of COVID-19 cases and deaths in different regions, where relationships information between different regions can be model as a dynamic graph. Since all of these data sources interact within the same global system, it can be advantageous to analyze them together via data integration. In Chapter 4 of this dissertation, we propose a novel node embedding method for dynamic graphs that maps each node to a random vector in the latent space. To the best of our knowledge, this is the first method modeling uncertainty of node latent representations for dynamic graphs, capturing both topological evolution and dynamic attribute changes simultaneously. By imposing semi-implicit variational inference, we have further extended our original VGRNN model to increase the expressive power of the inferred posterior. Given the success of the experiments with multiple real-world dynamic graph datasets demonstrate that (SI)-VGRNN consistently outperform the existing baseline and state-of-the-art methods by a significant margin in dynamic link prediction.

• **Bayesian Relational Learning (BayReL):** Moving to more complicated experimental settings, in Chapter 5, we introduce a novel Bayesian representation learning method for studying

molecular interactions across different data types. In addition to the view heterogeneity, BayReL exploits structural information among features at each corresponding view that is available for biological data when analyzing multi-omics data. BayReL infers the relational interactions as a multi-partite graph across multi-omics data types through non-linear and deep transformations. Unlike co-embedding and matrix completion based methods, it infers relations between different molecular classes, without any pre-known interactions across views. Our experiments on several real-world datasets demonstrate enhanced performance of BayReL in inferring meaningful interactions compared to existing baselines.

2. BAYESIAN MULTI-DOMAIN LEARNING *

2.1 Overview

Precision medicine aims for personalized prognosis and therapeutics by utilizing recent genome-scale high-throughput profiling techniques, including next-generation sequencing (NGS). However, translating NGS data faces several challenges. First, NGS count data are often overdispersed, requiring appropriate modeling. Second, compared to the number of involved molecules and system complexity, the number of available samples for studying complex disease, such as cancer, is often limited, especially considering disease heterogeneity. The key question is whether we may integrate available data from all different sources or domains to achieve reproducible disease prognosis based on NGS count data. In this section, we develop a Bayesian Multi-Domain Learning (BMDL) model that derives domain-dependent latent representations of overdispersed count data based on hierarchical negative binomial factorization for accurate cancer subtyping even if the number of samples for a specific cancer type is small. Experimental results from both our simulated and NGS datasets from The Cancer Genome Atlas (TCGA) demonstrate the promising potential of BMDL for effective multi-domain learning without "negative transfer" effects often seen in existing multi-task learning and transfer learning methods.

2.2 Introduction

In this chapter, we study Bayesian Multi-Domain Learning (BMDL) for analyzing count data from next-generation sequencing (NGS) experiments, with the goal of enhancing cancer subtyping in the *target domain* with a limited number of NGS samples by leveraging surrogate data from other domains, for example relevant data from other well-studied cancer types. Due to both biological and technical limitations, it is often difficult and costly, if not prohibitive, to collect enough samples when studying complex diseases, especially considering the complexity of disease processes. When

*Reprinted with permission from "Bayesian multi-domain learning for cancer subtype discovery from next-generation sequencing count data" by E. Hajiramezanali, S. Z. Dadaneh, A. Karbalayghareh, M. Zhou, and X. Qian. Advances in Neural Information Processing Systems, pp. 9115-9124. 2018. Copyright 2018 by Curran Associates, Inc.

studying one cancer type, there are typically at most hundreds of samples available with tens of thousands of genes/molecules involved, including in the case of the arguably largest cancer consortium, The Cancer Genome Atlas (TCGA) [4]. Considering the heterogeneity in cancer and the potential cost of clinical studies and profiling, we usually have only less than one hundred samples, which often does not lead to generalizable results. Our goal here is to develop effective ways to derive predictive feature representations using available NGS data from different sources to help accurate and reproducible cancer subtyping.

The assumption of having only one domain is restrictive in many practical scenarios due to the nonstationarity of the underlying system and data heterogeneity. Multi-task learning (MTL), transfer learning (TL), and domain adaptation (DA) techniques have recently been utilized to leverage the relevant data and knowledge of different domains to improve the predictive power in all domains or one target domain [5, 6]. In MTL, there are D different labeled domains where data are related and the goal is to improve the predictive power of all domains altogether. In TL, there are $D-1$ source domains and one target domain such that we have plenty of labeled data in the source domains and a few labeled data in the target domain, and the goal is to take advantage of source data, for example by domain adaptation, to improve the predictive power in the target domain. Although many TL and MTL methods have been proposed, "negative transfer" may happen with degraded performance when the domains are not related but the methods force to "transfer" the data and model knowledge. There still lacks a rigorous theoretical understanding when data from different domains can help each other due to the discriminative nature of these methods.

In this chapter, instead of following most of TL/MTL methods relying on discriminative models $p(y|\theta, \vec{n})$ given high-dimensional count data $\vec{n}$, we propose a generative framework to learn more flexible latent representations of $\vec{n}$ from different domains. We first construct a Bayesian hierarchical model $p(\vec{n})$, which is essentially a factorization model for counts $\vec{n}$, to derive domain-dependent latent representations allowing both domain-specific and globally shared latent factors. Then the learned low-dimensional representations can be used together with any supervised or unsupervised predictive models for cancer subtyping. Due to its unsupervised nature when deriving latent

representations, we term our model as Bayesian Multi-Domain Learning (BMDL). This is desirable in cancer subtyping since we may not always have labeled data and thus the model flexibility of BMDL enables effective transfer learning across different domains, with or without labeled data.

By allowing the assignment of the inferred latent factors to each domain independently based on the amount of contribution of each latent factor to that domain, BMDL can automatically learn the sample relevance across domains based on the number of shared latent factors in a data-driven manner. On the other hand, the domain-specific latent factors help keep important information in each domain without severe information loss in the derived domain-dependent latent representations of the original count data. Therefore, BMDL automatically avoids “negative transfer" with which many TL/MTL methods are dealing. At the same time, the number of shared latent factors can serve as one possible measure of domain relevance that may lead to more rigorous theoretical study of TL/MTL methods.

Specifically, for BMDL, we propose a novel multi-domain negative binomial (NB) factorization model for over-dispersed NGS count data. Similar as [7] and [8], we employ NB distributions for count data to obviate the need for multiple *ad-hoc* pre-processing steps as required in most of gene expression analyses. More precisely, BMDL identifies domain-specific and globally shared latent factors in different sequencing experiments as domains, corresponding to gene modules significant for subtyping different cancer types for example, and then use them to improve subtyping performance in a target domain with a very small number of samples. We introduce latent binary “selector” variables which help assign the factors to different domains. Inspired by Indian Buffet Process (IBP) [9], we impose beta-Bernoulli priors over them, leading to sparse domain-dependent latent factor representations. By exploiting a novel data augmentation technique for the NB distribution [10], an efficient Gibbs sampling algorithm with closed-form updates is derived for BMDL. Our experiments on both synthetic and real-world RNA-seq datasets verify the benefits of our model in improving predictive power in domains with small training sets by borrowing information from domains with rich training data. In particular, we demonstrate a substantial increase in cancer subtyping accuracy by leveraging related RNA-seq datasets, and also show that

in scenarios with unrelated datasets, our method does not create adverse effects.

2.3 Related works

TL/MTL methods typically assume some notions of relevance across domains of the corresponding tasks: All tasks under study either possess a cluster structure [11, 12, 13], share feature representations in common low-dimensional subspaces [14, 15], or have parameters drawn from shared prior distributions [16]. Most of these methods force the corresponding assumptions for MTL to link the data across domains. However, when tasks are not related to the corresponding data from different underlying distributions, forcing MTL may lead to degraded performance. To solve this problem, [17] have proposed a Bayesian nonparametric MTL model by representing the task parameters as a mixture of latent factors. However, this model requires the number of both latent factors and mixtures to be less than the number of domains. This may lead to information loss and the model only has shown advantage when the number of domains is high. But in real-world applications, when analyzing cancer data for example, we may only have a small number of domains. [18] have assumed the task parameters within a group of related tasks lie in a low-dimensional subspace and allowed the tasks in different groups to overlap with each other in one or more bases. But this model requires a large number of training samples across domains.

The hierarchical Dirichlet process (HDP) [19] has been proposed to borrow statistical strengths across multiple groups by sharing mixture components. Although HDP is aimed for a general family of distributions, to make it more suitable for modeling count data, special efforts pertaining to the application need to be carried out. To directly model the counts assigned to mixture components as NB random variables, [10] have performed a joint count and mixture modeling via the NB process. Under the NB process and integrated to HDP [19], NB-HDP employed a Dirichlet process (DP) to model the rate measure of a Poisson process. However, NB-HDP is constructed by fixing the probability parameter of NB distribution. While fixing the probability parameter of NB is a natural choice in mixture modeling, where it appears irrelevant after normalization, it would make a restrictive assumption that each count vector has the same variance-to-mean ratio. This is not proper for NGS count modeling in this dissertation. Closely related to the multinomial mixed-membership

models, [20] have proposed the hierarchical gamma-negative binomial process (hGNBP) to support countably infinite factors for negative binomial factor analysis (NBFA), where each of the sample J is assigned with a sample-specific GNBP and a globally shared gamma process is mixed with all the J gamma-negative binomial Markov chains (GMNBs). Our BMDL also uses hGNBP to model the counts in each domain, but imposes a spike and slab model to ensure domain-specific latent factors can be identified.

In this chapter, we propose a hierarchical Bayesian model—BMDL—for multi-domain learning by deriving domain-dependent latent representations of observed data across domains. By jointly deriving latent representations with both domain-specific and shared latent factors, we take the best advantage of shared information across domains for effective multi-domain learning. In the context of cancer subtyping, we are interested in deriving such meaningful representations for accurate and reproducible subtyping in the target domain, where only a limited number of samples are available. We will show first in our experiments that when the source and target data share more latent factors, we can better help subtyping in the target domain with higher accuracy; more importantly, we will also show that even when the domains are distantly related, our method can selectively integrate the information from other domain(s) to improve subtyping in the target domain while prohibit using irrelevant knowledge to avoid performance degradation.

2.4 Method

We would like to model the observed counts $n_{vj}^{(d)}$ from next-generation sequencing (NGS) for gene $v \in \{1, ..., V\}$ in sample $j \in \{1, ..., J_d\}$ of domain $d \in \{1, ..., D\}$ to help cancer subtyping. The main modeling challenges here include: (1) NGS counts are often over-dispersed and requiring *ad-hoc* pre-processing that may lead to biased results; (2) there are a much smaller number of samples with respect to the number of genes ($V \gg J$), especially in the target domain of interest; and (3) it is often unknown how relevant/similar the samples across different domains are so that forcing the joint learning may lead to degraded performance.

We construct a Bayesian Multi-Domain Learning (BMDL) framework based on a domain-dependent latent negative binomial (NB) factor model for NGS counts so that (1) over-dispersion is

appropriately modeled and *ad-hoc* pre-processing is not needed; (2) low-dimensional representations of counts in different domains can help achieve more robust subtyping results; and most importantly, (3) the sample relevance across domains can be explicitly learned to guarantee the effectiveness of joint learning across multiple domains.

BMDL achieves flexible multi-domain learning by first constructing a NB factorization model of NGS counts, and then explicitly establishing the relevance of samples across different domains by introducing domain-dependent binary variables that assign latent factors to each domain. The graphical representation of BMDL is illustrated in Fig. 2.1.

We model NGS counts $n_{vj}^{(d)}$ based on the following representations

$$n_{vj}^{(d)} = \sum_{k=1}^{K} n_{vjk}^{(d)}, \quad n_{vjk}^{(d)} \sim \mathrm{NB}\left(\phi_{vk}\theta_{kj}^{(d)}, p_j^{(d)}\right), \tag{2.1}$$

where $n_{vj}^{(d)}$ is factorized by K sub-counts $n_{vjk}^{(d)}$, each of which is a latent factor distributed according to a NB distribution. The factor loading parameter ϕ_{vk} quantifies the association between gene v and latent factor k, while the score parameter $\theta_{kj}^{(d)}$ captures the popularity of factor k in sample j of domain d. It should be noted that the factor loadings are shared across all domains, and thus making their inference more robust when the number of samples is low, especially in the target domain. This does not put a restriction on the model flexibility in capturing the inter-domain variability as

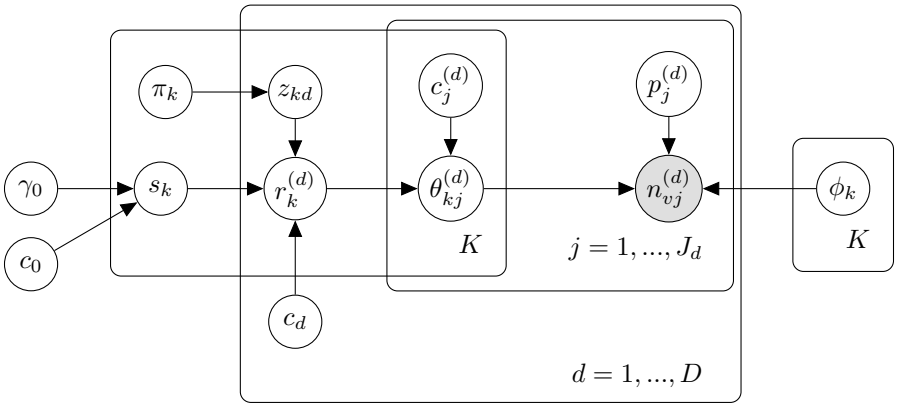


Figure 2.1: BMDL based on multi-domain negative binomial factorization model.

the score parameters determine the significance of corresponding latent factors across domains. The score parameter $\theta_{kj}^{(d)}$ is assumed to follow a gamma distribution:

$$\theta_{kj}^{(d)} \sim \text{Gamma}\left(r_k^{(d)}, 1/c_j^{(d)}\right), \tag{2.2}$$

with the scale parameter $c_j^{(d)}$ modeling the variability of sample j of domain d and the shape parameter $r_k^{(d)}$ capturing the popularity of factor k in domain d. To further enable domain-dependent latent representations, we introduce another hierarchical layer on the shape parameter:

$$r_k^{(d)} \sim \text{Gamma}\left(s_k z_{kd}, 1/c_d\right), \tag{2.3}$$

where the set of binary latent variables z_{kd} are considered as domain-dependent selector variables to allow different latent representations with the corresponding $r_k^{(d)}$ being present or absent across domains: When $z_{kd} = 1$, the latent factor k is present for factorization of counts in domain d; and it is absent otherwise. In our multi-domain learning framework, as the sample relevance across domains can vary significantly, this layer provides the additional model flexibility to model the sample relevance in the given data across domains. In (2.3), s_k is the global popularity of factor k in all domains. Inspired by the beta-Bernoulli process [21], whose marginal representation is also known as the Indian Buffet Process (IBP) [9], and its use in nonparametric Bayesian sparse factor analysis [22], we impose a beta-Bernoulli prior to the assignment variables:

$$z_{kd} \sim \text{Bernoulli}(\pi_k), \qquad \pi_k \sim \text{Beta}(c/K, c(1-1/K)), \tag{2.4}$$

which can be seen as an infinite spike-and-slab model as $K \to \infty$, where the spikes are provided by the beta-Bernoulli process and the slab is provided by the top-level gamma process. As a result, the proposed model assigns positive probability to only a subset of latent factors, selected independently of their masses.

We further complete the hierarchical Bayesian model for multi-domain learning by placing

appropriate priors on the model parameters in (2.1), (2.2), (2.3) and (2.4):

$$(\phi_{1k}, \ldots, \phi_{Vk}) \sim \mathrm{Dir}(\eta, \ldots, \eta), \quad \eta \sim \mathrm{Gamma}(s_0, w_0), \quad p_j^{(d)} \sim \mathrm{Beta}(a_0, b_0),$$
$$c_j^{(d)} \sim \mathrm{Gamma}(e_0, 1/f_0), \quad c_d \sim \mathrm{Gamma}(h_0, 1/u_0), \quad s_k \sim \mathrm{Gamma}(\gamma_0/K, 1/c_0),$$
$$\gamma_0 \sim \mathrm{Gamma}(a_0, 1/b_0), \quad c_0 \sim \mathrm{Gamma}(s_0, 1/t_0). \tag{2.5}$$

From a biological perspective, K factors may correspond to the underlying biological processes, cellular components, or molecular functions causing cancer subtypes, or more generally different phenotypes or treatment responses in biomedicine. The corresponding sub-counts $n_{vjk}^{(d)}$ can be viewed as the result of the contribution of underlying biological process k to the expression of gene v in sample j of domain d. The probability parameter $p_j^{(d)}$, which depends on the sample index, accounts for the potential effect of varying sequencing depth of sample j in domain d. More precisely, the expected expression of gene v in sample j and domain d is $\sum_{k=1}^{K} \phi_{vk}\theta_{kj}^{(d)} \frac{p_j^{(d))}}{1-p_j^{(d)}}$, and hence the term $(\sum_{k=1}^{K} \phi_{vk}\theta_{kj}^{(d)})$ can be viewed as the true abundance of gene v in domain d, after adjusting for the sequencing depth variation across samples. Specifically, it comprises of contributions from both domain-dependent and globally shared latent factors, where the amount of contribution of each latent factor can automatically be learned for the sample relevance across domains.

Given the BMDL model in Fig. 2.1, we derive an efficient Gibbs sampling algorithm with closed-form updating steps for inferring the model parameters by exploiting the data augmentation technique in [10]. The detailed Gibbs sampling procedure is provided in the *Appendix A*.

For real-world NGS datasets that are deeply sequenced and thus possess large counts, the steps in Gibbs sampling involving the Chinese Restaurant Table (CRT) distribution in [10] are the source of main computational burden. To speed up sampling from CRT, we propose the following scheme: to draw $\ell \sim \mathrm{CRT}(n, r)$, when n is large, we first draw $\ell_1 \sim \mathrm{CRT}(m, r)$, where $m \ll n$. Then, we draw $\ell_2 \sim \mathrm{Pois}\,(r[\psi(n+r) - \psi(m+r)])$, where $\psi(\cdot)$ is the digamma function. Finally, we have $\ell \approx \ell_1 + \ell_2$. This approximation is inspired by Le Cam's theory [23], and reduces the number

of Bernoulli draws required for CRT from n to m, and hence speeding up the Gibbs sampling substantially in our experiments with TCGA NGS data, where it is not uncommon for $n > 10^5$.

2.5 Experimental results

To verify the advantages of our BMDL model with the flexibility to capture the varying sample relevance across domains with both domain-specific and globally shared latent factors, we have designed experiments based on both simulated data and RNA-seq count data from TCGA [4]. We have implemented BMDL to extract domain-dependent low-dimensional latent representations and then examined how well using these extracted representations in an unsupervised manner can subtype new testing samples.

We also have compared the performance of BMDL to other Bayesian latent models for multi-domain learning, including

• **NB-HDP** [24], for which all domains are assumed to share a set of latent factors. This is done by involving a simple Bayesian hierarchy where the base measure for the child DPs is itself distributed according to a DP. It assumes the probability parameter of NB is fixed at $p_j^{(d)} = 0.5$.

• **HDP-NBFA**: To have fair comparison and make sure that the superior performance of BMDL is not only due to the modeling of the sequencing depth variation across samples, we apply HDP to model latent scores in NB factorization as well. More specifically we model count data as $n_{jk}^{(d)} \sim \mathrm{NB}(\phi_k \theta_{kj}^{(d)}, p_j^{(d)})$, where $\theta_{kj}^{(d)}$ is hierarchical DP instead of hierarchical gamma process in our model. Fixing $c_j^{(d)} = 1$ in (2.2) is considered here to construct an HDP, whose group-level DPs are normalized from gamma processes with the scale parameters as $1/c_j^{(d)} = 1$.

• **hGNBP** [20]: To evaluate the advantages of the beta-Bernoulli modeling in BMDL, we compare the results with hGNBP, which models count data as $n_{jk}^{(d)} \sim \mathrm{NB}(\phi_k \theta_{kj}^{(d)}, p_j^{(d)})$. Here, $\theta_{kj}^{(d)}$ is a hierarchical gamma process and the parameter z_{kd} in (2.4) is set to 1.

We illustrate that BMDL leads to more effective target domain learning compared to both HDP and hGNBP based models by assigning domain-specific latent factors to domains (using the beta-Bernoulli process) given observed samples, while learning the latent representations globally in a similar fashion as HDP and hGNBP. In addition, we also have compared with **hGNBP-NBFA**

[20], which can be considered as the baseline model as it extracts latent representations only using the samples from the target domain. Comparing to this baseline, we expect to show that BMDL effectively borrows the signal strength across domains to improve classification accuracy in a target domain with very small samples.

For all the experiments, we fix the truncation level $K = 100$ and consider $3{,}000$ Gibbs sampling iterations, and retain the weights $\{r_k^{(d)}\}_{1,K}$ and the posterior means of $\{\phi_k\}_{1,K}$ as factors, and use the last Markov chain Monte Carlo (MCMC) sample for the test procedure. With these K inferred factors and weights, we further apply $1{,}000$ blocked Gibbs sampling iterations and collect the last 500 MCMC samples to estimate the posterior mean of the latent factor score $\theta_j^{(d_t)}$, for every sample of target domain d_t in both the training and testing sets. We then train a linear support vector machine (SVM) classifier [25] on all $\bar{\theta}_j^{(d_t)}$ in the training set and use it to classify each $\bar{\theta}_j^{(d_t)}$ in the test set, where $\bar{\theta}_j^{(d_t)} \in \mathbb{R}^K$ is the estimated feature vector for sample j in the target domain. For each binary classification task, we report the classification accuracy based on ten independent runs. Note that although we fix K with a large enough value, we expect only a small subset of the K latent factors to be used and all the others to be shrunken towards zero. More precisely, inspired by the inherent shrinkage property of the gamma process, we have imposed $\mathrm{Gamma}(\gamma_0/K, 1/c_0)$ as the prior on each factor strength parameter s_k, leading to a truncated approximation of the gamma process using K atoms.

2.5.1 Synthetic data experiments

For synthetic experiments, we compare BMDL and the baseline hGNBP-NBFA using only target samples to illustrate multi-domain learning can help better prediction in the target domain.

For the first set of synthetic data experiments, we generate the varying sample relevance across domains. The degree of relevance is controlled by varying the number of latent factors shared by the domains. In this setup, we set two domains, $1{,}000$ features, 50 latent factors per domain, 200 samples in the source domain, and 20 samples in the target domain while the number of samples for both classes is 10. The number of shared latent factors between two domains changes from 50 to 0 to cover different degree of domain relevance. The factor loading matrix of the first

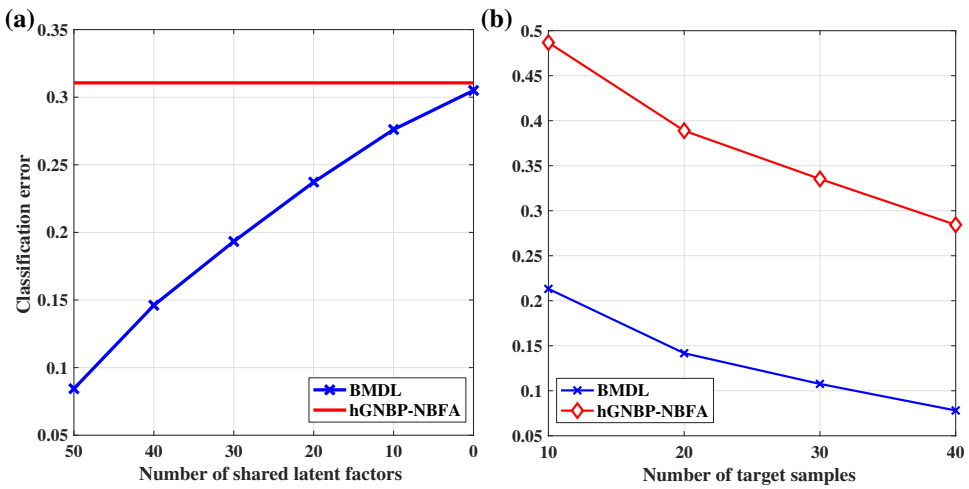


Figure 2.2: The classification error of BMDL and hGNBP-NBFA as a function of (a) domain relevance, and (b) the number of target samples.

domain is generated based on a Dirichlet distribution. To simulate the loading matrix for the second domain, we first select N_{K_c} shared latent factors from the first domain, and then randomly generate $50 - N_{K_c}$ latent factors as unique ones for the second domain, where $N_{K_c} \in \{0, 10, 20, \ldots, 50\}$. The dispersion parameters of both domains are generated from a gamma process: $\mathrm{Gamma}(s_k, 1/c_d)$, where s_k is generated by $\mathrm{Gamma}(\gamma_0/K, 1/c_0)$. The hyperparameters γ_0, and c_0 are drawn from $\mathrm{Gamma}(0.01, 0.01)$. To distinguish two classes of generated samples in the target domain, we generate their factor scores by different scale parameters $c_j^{(d)} \sim \mathrm{Gamma}(a, 0.01)$, where a is set to be 100 and 150 in the first and second class, respectively. From Figure 2(a), the first interesting observation is that BMDL automatically avoids "negative transfer": the classification errors of BMDL by jointly learning the latent representations are consistently lower than the classification errors using only the target domain data no matter how many shared latent factors exist across simulated domains. Furthermore, the classification error in the target domain decreases monotonically with the number of shared latent factors, which agrees with our intuition that BMDL can achieve higher predictive power when data across domains are more relevant. This demonstrates that the number of shared latent factors across domains may serve as a new measure of the domain relevance.

In the second simulation study, we investigate how the number of target samples affects the classification performance. In this simulation setup, we simulate two related domains with 40 shared latent factors out of 50 total ones. The number of samples in the target domain is changing from 10 to 40, keeping the other setups the same as in the first experiment. Figure 2(b) shows that increasing the number of target samples will improve the performance of both the baseline hGNBP-NBFA using only target data and BMDL integrating data across domains, which is again expected. More interestingly, the improvement of BMDL over hGNBP-NBFA decreases with the number of target samples, which agrees with the general trend shown in the TL/MTL literature [26, 27] that the prediction performance eventually converges to the optimal Bayes error when there are enough samples in the target domain.

2.5.2 Case study: Lung cancer

We consider two setups of analyzing RNA-seq count data from the studies on two subtypes of lung cancer, i.e. Lung Adenocarcinoma (LUAD) and Lung Squamous Cell Carcinoma (LUSC) from TCGA [4]. First, we take two types of NGS data, RNA-seqV2 and RNA-seq of the same lung cancer study, as two **highly-related** domains since the source and target domain difference is simply due to profiling techniques. Second, we use RNA-seq data from a Head and Neck Squamous Cell Carcinoma (HNSC) cancer study as the source domain and the above RNA-seq lung cancer data as the target domain. These are considered as **low-related** domains as these two cancer types have quite different disease mechanisms. In this set of experiments, we take 10 samples for each subtype of lung cancer in the target domain to test cancer subtyping performance. We also investigate the effect of the number of source samples, N_s, on cancer subtyping in the target domain by setting $N_s = 25$ and 100.

For all the TCGA NGS datasets, we first have selected the genes appeared in all the datasets and then filtered out the genes whose total read counts across samples are less than 50, resulting in roughly $14{,}000$ genes in each dataset. We first have divided the lung cancer datasets into training and test sets, and then the differential gene expression analysis has been performed on the training set using DeSeq2 [28], by which $1{,}000$ out of the top $5{,}000$ genes with higher log2 fold change

between LUAD and LUSC have been selected for consequent analyses. We first check the subtyping accuracy by directly applying linear SVM to the raw counts in the target domain, which gives an average accuracy of 59.28% with a sample standard deviation (STD) of 5.54% from ten independent runs. We also transform the count data to standard normal data after removing the sequencing depth effect using DESeq2 [28] and then apply regularized logistic regression provided by the LIBLINEAR (`https://www.csie.ntu.edu.tw/~cjlin/liblinear/`) package [29]. The classification accuracy becomes $74.10\% \pm 4.41\%$.

Table 2.1 provides cancer subtyping performance comparison between BMDL, NB-HDP, HDP-NBFA, hGNBP, as well as the baseline hGNBP-NBFA using only the samples form the target domain. In fact, when analyzing data across highly-related domains of lung cancer, from the identified 100 latent factors in the target domain by BMDL, there are 98 shared ones between two RNA-seq techniques. While for low-related domains of lung cancer and HMSC, only 25 of 62 extracted latent factors in lung cancer by BMDL are shared with HNSC. This is consistent with our biological knowledge regarding the sample relevance in two setups. From the table, BMDL consistently achieves better cancer subtyping in both highly- and low-related setups. On the contrary, as the results show, not only HDP based methods cannot improve the results in the low-related setup, but also the performance will be degraded with more severe "negative transfer" adversarial effects when using more source samples. The reason for this is that HDP assumes a latent factor with higher weight in the shared DP will occur more frequently within each sample [30]. This might be an undesirable assumption, especially when the domains are distantly related. For example, a latent factor might not be present throughout the HNSC samples but dominant within the samples of lung cancer. HDP based methods are not able to discover these latent factors given observed samples due to the limited number of lung cancer samples. In addition to this undesirable assumption, NB-HDP does not account for the sequencing-depth heterogeneity of different samples, which may lead to biased results deteriorating subtyping performance as shown in Table 2.1.

HDP-NBFA explores the advantages of modeling the NB dispersion and improves over the NB-HDP due to the flexibility of learning $p_j^{(d)}$, especially in the highly-related setup. This demonstrates

Table 2.1: Lung cancer subtyping results (average accuracy (%) and STD)

	highly-related (N_s)		low-related (N_s)	
Method	25	100	25	100
NB-HDP	55.22 ± 3.69	56.52 ± 4.61	54.57 ± 7.73	53.83 ± 7.79
HDP-NBFA	63.48 ± 1.23	65.65 ± 4.22	54.89 ± 7.38	51.83 ± 8.32
hGNBP	74.13 ± 7.07	77.61 ± 3.54	72.94 ± 1.70	74.55 ± 8.84
BMDL	$\mathbf{78.46} \pm 5.97$	$\mathbf{81.49} \pm 5.12$	$\mathbf{78.85} \pm 4.55$	$\mathbf{78.10} \pm 5.65$
hGNBP-NBFA	73.38 ± 7.29			

the benefits of inferring the sequencing depth in RNA-seq count applications. Although in highly-related setup the HDP-NBFA performance has been improved with the increasing number of source samples, we still observe the same "negative transfer" effect in the low-related setup. Again, integrating more source samples is beneficial when the samples across domains are highly relevant but it can be detrimental when the relevance assumption does not hold as both NB-HDP and HDP-NBFA force a similar structure of latent factors across domains.

The better performance of the gamma process based models compared to HDP based models, in both scenarios with low and high domain relevance, may be explained by the negative correlation structure that the Dirichlet process imposes on the weights of latent factors, while the gamma process models these weights independently, and hence allowing more flexibility for adjustment of latent representations across domains. On the other hand, when comparing the performance of BMDL and hGNBP, domain-specific latent factor assignment using the beta-Bernoulli process can be considered as the main reason for the superior performance of BMDL, especially in the low-related setup.

Compared to the baseline hGNBP-NBFA, BMDL can clearly improve cancer subtyping performance. Even using a small portion of the related source domain samples, the subtyping accuracy can be improved up around 5%. With more highly-related source samples, the improvement can be up to 8%. Compared to the HDP based methods, BMDL can achieve up to 16% improvement in the highly-related setup due to the benefits brought by the gamma process modeling of count data

instead of using DP in HDP models, which forces negative correlation and restricts the distribution over latent factor abundance [30]. Compared to hGNBP, BMDL can achieve up to 4% and 6% accuracy improvement, respectively, in highly- and low-related setups due to domain-specific latent factor assignment using the beta-Bernoulli process. Since the selector variables z_{kd} in BMDL help to assign only a finite number of latent factors for each domain, it is sufficient merely to ensure that the sum of any finite subset of top-level atoms is finite. This eliminates the restrictions on factor score parameters imposed by DP, and improves subtyping accuracy since the latent factor abundance is independent.

BMDL also does not have any restriction on the number of domains and can be applied to more than two domains. To show this, we also have done another case study with three domains using both the highly- and low-related TCGA datasets. The accuracy of BMDL is $79.71\% \pm 5.32\%$ and $81.96\% \pm 4.96\%$ when using $N_s^{(d_{s1})} = N_s^{(d_{s2})} = 25$ and 100 samples for two source domains as described earlier, respectively. Compared to one source and one target domain with 25 source samples, the accuracy of using three domains has improved by 1%. Having two source domains with more samples ($N_s^{(d_{s1})} + N_s^{(d_{s2})} = 50$) leads to more robust estimation of ϕ_{vk} and improves the subtyping accuracy. When there are enough number of samples ($N_s^{(d_{s1})} = 100$) in highly-related domain, adding another low-related domain does not improve the subtyping results. But it is notable that the accuracy has increased around 4% when adding the highly-related domain with 100 samples to 100 low-related samples. The results show that 1) using more domains with more samples does help subtyping in the target domain; 2) BMDL avoids negative transfer even when adding samples from low-related domains.

We would like to emphasize again that, unlike existing methods, BMDL infers the domain relevance given in the data and derive domain-adaptive latent factors to improve predictive power in the target domain, regardless of the degree of domain relevance. This is important in real-world setups when the samples across domains are distantly related or the sample relevance is uncertain. As the results have demonstrated, BMDL achieves the similar performance improvement in the low-related setup as in the highly-related setup without "negative transfer" symptom, often witnessed

in existing TL/MTL methods. It shows the great potential for effective data integration and joint learning even in the low-related setup: the performance is better than competing methods as well as the baseline hGNBP-NBFA using only target samples and increasing the number of source samples does not hurt the performance.

3. GAMMA MARKOV CHAIN DIFFERENTIAL EXPRESSION ANALYSIS

3.1 Overview

To better understand the underlying cellular mechanisms by profiling temporal changes in living systems using next-generation sequencing (NGS), new statistical tools are required. The majority of existing NGS data analysis methods for static cross-sectional sequencing experiments lack the capability of exploiting the richer information embedded in temporal data and uncovering transitional responses. Several recent tools have been developed to analyze temporal NGS data but they typically impose strict model assumptions, such as smoothness on gene expression dynamic changes. To capture a broader range of gene expression dynamic patterns, we develop the gamma Markov negative binomial (GMNB) model that integrates a gamma Markov chain into a negative binomial distribution model, allowing flexible modeling of temporal variation in NGS count data. Using Bayes factors, GMNB enables more powerful temporal gene differential expression analysis across various phenotypes or treatment conditions. In addition, it naturally handles the heterogeneity of sequencing depth in different samples, removing the need for ad-hoc normalization. Efficient Gibbs sampling inference of the GMNB model parameters is achieved by exploiting novel data augmentation techniques. Extensive experiments on both simulated and real-world RNA-seq data demonstrate the benefits of GMNB. GMNB helps to identify several known genes involved in Th17 differentiation as well as some new potential genetic markers when applied to the temporal NGS data of human T-cells.

3.2 Introduction

Transcriptome analyses, including gene expression profiling and transcript quantification through RNA-sequencing (RNA-seq), can help better understand biological processes of interest when studying complex, dynamic, stochastic, and heterogeneous living and disease systems. It is challenging to model RNA-seq count data, not only because the data dimension is typically much higher than the sample size, but also because the sequencing counts are nonnegative, skewed, and highly

over-dispersed with large dynamical ranges [31]. With increasingly more temporal transcriptomic profiling data, an important task is to identify the genes that are differentially expressed over time across different conditions, which adds another level of difficulty compared to static RNA-seq data analysis tasks due to potential temporal dependencies and inconsistent temporal sample collections [32]. For example, in cell biology or drug discovery research, at the transcriptional and post-transcriptional regulatory levels, monitoring molecular expression changes in response to specific stimuli can help reveal underlying cellular mechanisms.

A large number of statistical tools have been developed for differential gene expression analysis of RNA-seq data [28]. Most of them have adopted the negative binomial (NB) distribution to account for over-dispersion as well as high uncertainty inherent in RNA-seq data due to the small number of replicate samples in typical differential expression experiments [28]. While these static methods can address some of the aforementioned issues of RNA-seq count data, they can neither handle inconsistencies in sample collection, such as potential missing values and misalignment, nor exploit the information embedded in temporal data. Such a temporal dependency not only can be used to analyze the timing of cellular programs, but also is critical to uncover transitional responses.

Recently, several dynamic differential RNA-seq analysis methods have been developed to better capture temporal dependency and deal with missing data. In general, these methods are divided into two categories. In the first category, the goal is to identify genes whose expression trajectories vary significantly over time from samples of one condition. The methods belonging to the second category focus on detecting genes with differential expression trajectories over time across different treatment conditions [33]. In this chapter, we concentrate on the latter scenario. For example, EBSeq-HMM [34] takes an empirical Bayesian mixture modeling approach; NB-AR(1) [35] is an autoregressive time-lagged model to detect the expression changes across consecutive time points under one treatment condition. Across different conditions, it is desirable to identify genes that have different dynamic patterns. SplineTimeR [36] treats time as a continuous variable and fits natural cubic spline functions across time, which requires measurements at densely sampled time points and can be prone to noise. In real-world scenarios when analyzing temporal RNA-seq

data, we often have noisy expression profiles at sparse time points, where SplineTimeR may not achieve the desired performance [37]. DyNB [1] has been proposed to model the temporal RNA-seq counts by NB distributions with non-parametric Gaussian Processes (GP) as their temporal expected values. DyNB can detect genes with differential dynamic patterns that static differential expression analysis, which considers individual time points, fails to discover. In addition to high computational complexity due to Markov chain Monte Carlo (MCMC) inference [38, 39], DyNB may fail to model potential abrupt expression changes due to its inherent smoothness assumptions [40]. Very recently, rmRNAseq [41] has been proposed for analysis of repeated-measured RNA-seq data by extending voom [42] with a parametric bootstrap method. Similar as voom, rmRNAseq models normalized log-transformed counts and associated precision weights in a GLM framework with a continuous autoregressive structure for temporal correlation [41]. Depending on the selected structure, it also imposes smoothness assumption across regularly sampled time points. ImpulseDE2 [33] has been proposed to solve the smoothness problem (i.e., to capture these abrupt expression changes). ImpulseDE2 models temporal RNA-seq counts by NB distributions using an impulse function for their temporal expected values [43]. However, the proposed impulse function can only capture the temporal profiles that have at most two significant changes in expression. Hence, it may not be appropriate for modeling complicated trajectories with multiple change points. In addition, as the model imposes the abrupt changes into the expression pattern, its performance will be degraded for the genes with smooth expression patterns. Consequently, its performance will drop when the number of time points is either too small or too large [37].

We present a new dynamic differential expression analysis method for temporal RNA-seq data, GMNB (gamma Markov negative binomial), a hierarchical model that employs a gamma Markov chain [44, 45] to model the potential dynamic transitions of the model parameters in NB distributions. With this new model for temporal RNA-seq data and an efficient inference algorithm, GMNB is expected to provide the following advantages over existing methods: 1) GMNB is flexible and powerful in detecting any changes (smooth or abrupt) over time and/or between time points; 2) Gibbs sampling with closed-form updating steps can be derived to infer the model parameters in

GMNB, which is computationally more efficient than existing methods; 3) For dynamic differential expression, the genes are ranked based on the Bayes factor (BF), which is very general especially when considering differential expression under multiple factors; 4) Last but not least, GMNB avoids the normalization preprocessing step due to the explicit modeling of the sequencing depth in NB distributions, as described in [46, 47], and we expect similar superior performance of GMNB compared to existing methods requiring such heuristic preprocessing steps.

3.3 Methods

Notation. Throughout this chapter, we use the NB distribution to model RNA-seq read counts. We parameterize a NB random variable as $Y \sim \text{NB}(r, p)$, where r is the nonnegative dispersion and p is the probability parameter. The probability mass function (PMF) of Y is expressed as $f_Y(y) = \frac{\Gamma(y+r)}{y!\Gamma(r)} p^y (1-p)^r$, where $\Gamma(\cdot)$ is the gamma function. The NB distribution $Y \sim \text{NB}(r, p)$ can be generated from a compound Poisson distribution:

$$Y = \sum_{t=1}^{L} U_t, \quad U_t \sim \text{Log}(p), \quad L \sim \text{Pois}(-r \ln(1-p)),$$

where $U \sim \text{Log}(p)$ corresponds to the logarithmic random variable [48], with PMF $f_U(u) = -\frac{p^u}{u \ln(1-p)}$, $u = 1, 2, \ldots$. As shown in [10], given y and r, the random count L follows a Chinese Restaurant Table (CRT) distribution, $(L \,|\, y, r) \sim \text{CRT}(y, r)$, which can be generated as $L = \sum_{t=1}^{y} B_t,\ B_t \sim \text{Bernoulli}(\frac{r}{r+t-1})$.

3.3.1 GMNB model

We model the dynamic gene expression changes in a temporal RNA-seq dataset by constructing a Markov chain where the expression of a gene at time t only depends on that of time $t-1$. Specifically, for the RNA-seq reads mapped to gene k in a given sample j under different conditions, the read count at time t follows:

$$y_{kj}^{(t)} \sim \text{NB}(r_k^{(t)}, p_j^{(t)}). \tag{3.1}$$

To impose the dependency between consecutive time points, we model the dispersion parameters dynamically by introducing a gamma Markov chain, in which $r_k^{(t)}$ is distributed according to a gamma distribution as:

$$r_k^{(t)} \sim \text{Gamma}(r_k^{(t-1)}, \frac{1}{c_k}). \tag{3.2}$$

Note in our notation, $x \sim \text{Gamma}(a, 1/c)$ has mean a/c and variance a/c^2.

Note that the scale parameter $1/c_k$ of the Gamma distribution in (3.2) is shared between different time points, thereby making statistical inference more robust by sharing statistical strength between samples across time. To complete the model we sample the dispersion parameter at the first time point as $r_k^{(0)} \sim \text{Gamma}(e_0, \frac{1}{f_0})$, and use conjugate priors as $c_k \sim \text{Gamma}(c_0, \frac{1}{d_0})$ and $p_j^{(t)} \sim \text{Beta}(a_0, b_0)$.

Let $s_j^{(t)} = \frac{p_j^{(t)}}{1-p_j^{(t)}}$. The probability parameters $p_j^{(t)}$, depending on the sample and time indices, can be considered as a parameter reflecting the heterogeneity of read counts due to the variation of the sequencing depths across different samples. This can be justified by the gene count expectation $\mathbb{E}[y_{kj}^{(t)}] = r_k^{(t)} s_j^{(t)}$, which is directly proportional to $s_j^{(t)}$. The term $s_j^{(t)}$ can be interpreted as the effect of the sequencing-depth heterogeneity of sample j at time t on the corresponding gene expression in this sample. This approach removes the need for an ad-hoc normalization step, as the model accounts for the sequencing-depth heterogeneity of different samples automatically, similar to the mechanisms employed in [47]. Due to the one-to-one mapping between $s_j^{(t)}$ and $p_j^{(t)}$, the variation of $p_j^{(t)}$ can be used to account for those of sequencing depths and other potential confounding factors, in the context of RNA-seq data. The remaining term in the expectation, $r_k^{(t)}$, can represent the true abundance of gene k at time t.

In addition to the flexibility of modeling temporal RNA-seq data, this GMNB model enables an efficient inference procedure by taking advantage of unique data augmentation and marginalization techniques for the NB distribution [10], as described in detail below.

3.3.2 Gibbs sampling inference

By exploiting the data augmentation techniques in [10], we implement an efficient Gibbs sampling algorithm with closed-form updating steps. More specifically, we infer the dispersion parameter of the NB distribution by first drawing latent random counts from the CRT distribution, and then update the dispersion by employing the gamma-Poisson conjugacy. Furthermore, due to the Markovian construction of the model, it is necessary to consider both backward and forward flow of information for the inference of $r_k^{(t)}$. First, in the backward stage, starting from the last time point $t = T$, we draw two sets of auxiliary random variables as

$$\begin{aligned} l_{kj}^{(t)} &\sim \text{CRT}(y_{kj}^{(t)}, r_k^{(t)}), \qquad l_{k.}^{(t)} = \sum_j l_{kj}^{(t)} \\ u_k^{(t-1)(t)} &\sim \text{CRT}(u_k^{(t)(t+1)} + l_{k.}^{(t)}, r_k^{(t-1)}), \end{aligned} \tag{3.3}$$

for $t = T, T-1, \ldots, 1$. For the last time point, we assume $u_k^{(T)(T+1)} = 0$. Next, in the forward stage of Gibbs sampling, we sample $r_k^{(t)}$ starting from $t = 0$ to $t = T$ as

$$(r_k^{(t)} \,|\, -) \sim \text{Gamma}\big(r_k^{(t-1)} + u_k^{(t)(t+1)} + l_{k.}^{(t)},\ \frac{1}{\theta_k^{(t)}}\big), \tag{3.4}$$

where $r_k^{(0)} = e_0$ and $\theta_k^{(t)} = c_k - \sum_j \ln(1 - p_j^{(t)}) - \ln(1 - q_k^{(t)})$. For $t = 0, ..., T-1$, $q_k^{(t)}$ is defined as

$$q_k^{(t)} = \frac{-\sum_j \ln(1 - p_j^{(t+1)}) - \ln(1 - q_k^{(t+1)})}{c_k - \sum_j \ln(1 - p_j^{(t+1)}) - \ln(1 - q_k^{(t+1)})}, \tag{3.5}$$

and $q_k^{(T)} = 0$. Finally, by taking advantage of conjugate priors, in each iteration of Gibbs sampling, c_k and $p_j^{(t)}$ can be drawn as

$$\begin{aligned} (c_k \,|\, -) &\sim \text{Gamma}(c_0 + \sum_{t=0}^{T-1} r_k^{(t)}, 1/(d_0 + \sum_{t=1}^{T} r_k^{(t)})), \\ (p_j^{(t)} \,|\, -) &\sim \text{Beta}(a_0 + \sum_k y_{kj}^{(t)}, b_0 + \sum_k r_k^{(t)}). \end{aligned} \tag{3.6}$$

The efficient augmentation technique employed in our Gibbs sampling inference removes the need for specifying a suitable proposal distribution, as required in the Metropolis-Hastings inference of both DyNB of [1] and NB-AR(1) of [35]. Our experiments in the next section demonstrate that the Gibbs sampling algorithm of GMNB converges fast and mixes well.

3.3.3 Dynamic differential expression using Bayes factor

In the GMNB model, since in the prior we have

$$\begin{aligned}
\mathbb{E}[y_{kj}^{(t)} \,|\, r_k^{(t)}, p_j^{(t)}] &= r_k^{(t)} p_j^{(t)}/(1-p_j^{(t)}),\\
\text{var}[y_{kj}^{(t)} \,|\, r_k^{(t)}, p_j^{(t)}] &= r_k^{(t)} p_j^{(t)}/(1-p_j^{(t)})^2,\\
&= \mathbb{E}[y_{kj}^{(t)} \,|\, r_k^{(t)}, p_j^{(t)}] + \frac{1}{r_k^{(t)}} \mathbb{E}^2[y_{kj}^{(t)} \,|\, r_k^{(t)}, p_j^{(t)}],
\end{aligned}$$

and in the conditional posterior, if $b_0 + \sum_k r_k^{(t)} > 1$, we have

$$\begin{aligned}
\mathbb{E}[r_k^{(t)} \,|\, -] &= \frac{r_k^{(t-1)} + u_k^{(t)(t+1)} + l_{k.}^{(t)}}{c_k - \sum_j \ln(1-p_j^{(t)}) - \ln(1-q_k^{(t)})},\\
\mathbb{E}[p_j^{(t)}/(1-p_j^{(t)}) \,|\, -] &= \big(a_0 + \sum_k y_{kj}^{(t)}\big)/\big(b_0 + \sum_k r_k^{(t)} - 1\big).
\end{aligned} \tag{3.7}$$

Thus, one may interpret $p_j^{(t)}/(1-p_j^{(t)})$ as a term that accounts for the sequencing depth of sample j at time point t, and may compare the posterior distributions of the NB shape parameters $r_k^{(t)}$ of the same gene across different conditions to assess differential expression of that gene. The conditional posterior of the scaling factor $p_j^{(t)}/(1-p_j^{(t)})$ is determined by not only $\sum_k y_{kj}^{(t)}$, the total counts of genes in sample j at time point t, but also $\sum_k r_k^{(t)}$, the total sum of all countably infinite gene-specific NB shape parameters; and the conditional expectation of $r_k^{(t)}$ is related to both $l_{k.}^{(t)}$ and $\sum_j \ln(1-p_j^{(t)})$, which aggregate their corresponding sample-specific values across all the J samples. Therefore, our GMNB model borrows statistical strengths across both the genes and samples to infer the conditional posterior of $r_k^{(t)}$. For an unexpressed gene, whose total count across all samples in a condition is 0, the posterior values of its corresponding $r_k^{(t)}$ would be fixed at 0.

The main goal of differential expression analysis is to identify genes whose expressions demon-

strate significant variations across conditions. In the classic static RNA-seq data analysis, this goal is usually obtained via the comparison of expression averages across conditions. In dynamic RNA-seq measurement settings, however, this task becomes more challenging as any change of temporal expression patterns between conditions may reflect interesting biological mechanisms. Hence, as in [1], we adopt the Bayes Factor (BF) as a measure that exploits information collectively from all time points to detect the genes with significant variations in temporal expression patterns across conditions.

To compute the BF, we first consider the null hypothesis H_0 that the genes are not differentially expressed across conditions, and thus the same set of parameters govern temporal gene expressions in all samples. In this case, we aggregate the counts $\mathcal{D}$ of both experimental conditions to fit the GMNB model M_0. Under the alternative hypothesis H_1, the differentially expressed genes possess different model parameters in each condition. Hence, GMNB models M_1 and M_2 independently to fit to the counts in conditions 1 ($\mathcal{D}_1$) and 2 ($\mathcal{D}_2$), respectively. Then, the BF can be calculated as

$$\mathrm{BF} = \frac{P(\mathcal{D} \,|\, \mathrm{H}_1)}{P(\mathcal{D} \,|\, \mathrm{H}_0)} = \frac{P(\mathcal{D}_1 \,|\, \mathrm{M}_1) P(\mathcal{D}_2 \,|\, \mathrm{M}_2)}{P(\mathcal{D} \,|\, \mathrm{M}_0)},$$

where we have assumed equal prior probabilities for both hypotheses. The BF computation requires marginalizing out the model parameters, which we conduct through Monte Carlo integration using posterior samples collected with the proposed Gibbs sampler. More specifically, we can write the likelihood of the data for each gene k in the model M_i as follows

$$P(y_k \,|\, r_{k,M_i}, p_{M_i}) = \sum_{j,t} P(y_{kj}^{(t)} \,|\, r_{k,M_i}^{(t)}, p_{j,M_i}^{(t)}),$$

where $r_{k,M_i}^{(t)}$ and $p_{j,M_i}^{(t)}$ are the samples corresponding to the posterior distribution of model M_i in equations (4) and (6), respectively. We further estimate the marginal likelihood $P(\mathcal{D}_i \,|\, \mathrm{M}_i)$, using the mean of the likelihoods of the samples from the posterior distribution.

In Appendix B, we discuss the potential generalization of BF for multi-level factor analysis and use it to identify the corresponding time intervals that the genes are differentially expressed.

3.4 Experimental results

To evaluate our proposed GMNB differential expression analysis method, we first examine its performance on synthetic data, generated from different generative models. Then, to show the practical utility of our GMNB, we analyze RNA-seq time-series datasets with application to human Th17 cell differentiation. In particular, we compare its performance with those of four state-of-the-art dynamic differential expression methods—DyNB [1], ImpulseDE2 [33], SplineTimeR [36], and rmRNAseq [41]. We also consider DESeq2 [28], which is a popular tool for differential expression analysis, however, not specifically designed for temporal RNA-seq data. We first consider synthetic RNA-seq data, and show that GMNB provides outstanding performance in terms of the area under the curves (AUCs) of the receiver operating characteristic (ROC) and precision-recall (PR) curves. Furthermore, we present two case studies on human Th17 cell differentiation [49, 50, 1], and explain the biomedical implications based on differential expression analysis over time by GMNB. We show that the proposed GMNB method identifies several known and novel genes involved in Th17 differentiation, revealing potential autoimmune mechanisms, which may lead to more effective and affordable treatments.

Throughout the experiments, in each run of Gibbs sampling inference for GMNB, 1000 MCMC samples of parameters are collected after 1000 burn-in iterations. The example MCMC sample trace plots in Figure C.26 suggest that the Markov chains for GMNB converge fast and mix well, supporting the practice of performing downstream analysis with 2,000 MCMC iterations. We use the collected MCMC samples to calculate the BF for each gene as explained in Section 3.3.3, and rank the genes according to these BFs. For DyNB, we follow the settings provided in [1] and rank the genes using the computed BFs.

We also follow the case-control setting of the available R package for ImpulseDE2 and rank the genes based on the p-values. We consider three different setups for differential expression analysis of temporal RNA-seq data using DESeq2. The first setup is denoted by DESeq2-GLM in the experiments. In this setup, time information is incorporated as a covariate of the generalized linear model in DESeq2 to determine temporal data in one model. In the second and third setups,

we apply DESeq2 to the data at different time points independently, and use the average and minimum of the computed p-values from the respective differential expression analyses as an overall measure of differential expression across conditions, denoted by DESeq2-avg and DESeq2-min in the experiments, respectively. For all the experiments, we use the default p-value adjustment method, Benjamini-Hochberg, for multiple test correction.

The running time also has been benchmarked with 20 parallel running processes on a single cluster node with Intel Xeon 2.5GHz E5-2670 v2 processor. For the first RNA-seq dataset with around 20K genes at 5 time points, it took 2 hours and 20 minutes for our GMNB method with 2,000 MCMC iterations, compared to 58 hours for DyNB. For the second time-course RNA-seq data with around 15K genes and 10 time points, it took 4 and 80 hours for GMNB and DyNB methods, respectively. For the non-Bayesian based methods, DESeq2, rmRNAseq, and SplineTimeR are very quick (within several minutes); however, ImpulseDE2 takes around 3 hours to run without any parallelization on both datasets.

3.4.1 Synthetic data

For comprehensive performance evaluation, we have generated synthetic data under different temporal RNA-seq models. More precisely, we simulate data under the following three different setups: the generative model based on our proposed GMNB, the DyNB-based generative model, and an auto-regressive (AR) based procedure (detailed in Appendix C.1). For each setting, to make the synthetic data closely resemble real-world RNA-seq data, we first infer the parameters of the corresponding model based on the human Th17 cell differentiation datasets, and then generate synthetic sequencing counts using these inferred model parameters. Following the instruction in the papers of DESeq2 [28], we generate count data for 10,000 genes across two conditions, each of which has three replicated samples. We randomly select 10% of the genes to be differentially expressed across two conditions, with the procedure described in details in the next subsection for the GMNB generative model. For each specific generative model, we change the corresponding model parameters to ensure that the expected expression changes of truly differentially expressed genes are different across two conditions.

3.4.1.1 Comparison based on GMNB-based generative model

In the first simulation study, we generate the synthetic RNA-seq count data for 10,000 genes under two conditions according to the GMNB-based generative model. In the GMNB setting, if gene k is up-regulated, we generate its counts by sampling the NB counts (3.1) with dispersion parameters Gamma$(r_k^{(t-1)}, \frac{1}{c_k})$ and Gamma$(r_k^{(t-1)}, \frac{b}{c_k})$ for the samples in the first and second conditions, respectively; if gene k is down-regulated, we generate its counts with dispersion parameters Gamma$(r_k^{(t-1)}, \frac{1}{c_k})$ and Gamma$(r_k^{(t-1)}, \frac{1}{bc_k})$ for the third samples in the first and second conditions, respectively. The gene-wise scale parameters c_k are drawn from the uniform distribution in the interval $[0.7, 2]$. The dispersion parameter at the initial time point, $r_k^{(0)}$, is generated for both conditions according to Gamma$(e_0, 1)$, where $e_0 \sim$ Uniform$(10, 100)$. To simulate the effect of potential varying sequencing depths, the probability parameters $p_j^{(t)}$ are drawn uniformly at random from the interval $[0.7, 0.98]$. Similar to the human Th17 cell dataset [1], the number of generated time points are 5. (All these values are chosen to mimic the situations in the human Th17 cell datasets.)

To assess how sensitive different methods under study are to different levels of differential expression, we set the fold change b^T at 1.4, 1.6, 1.8, or 2, where $T = 5$ in simulating synthetic data. For each fold change, we report the results of each method based on ten independent random trials.

For the synthetic data from the GMNB-based generative model, as shown in Figure 3.1(a), measured by both AUC-ROC and AUC-PR, DyNB has the worst overall performance, followed by splineTimeR, DESeq2-GLM and ImpulseDE2, and then by DESeq2-min, DESeq2-mean, and rmRNAseq. GMNB clearly outperforms all the other differential expression analysis algorithms with significant margins. To further compare the operating characteristics of different algorithms, we show in Figure C.25 the full ROC and PR curves for the fold change of $b = 2$. The proposed GMNB model outperforms all the other methods in both the ROC and PR curves.

More carefully examining Figures 3.1(a) and C.25, it is interesting to notice that for the synthetic data generated with the GMNB-based model, ImpulseDE2, DESeq2-GLM, splineTimeR, and DyNB, which incorporate temporal structures of data as either confounding factors or dynamic structures, in fact underperform the static methods based on DESeq2 especially when the fold

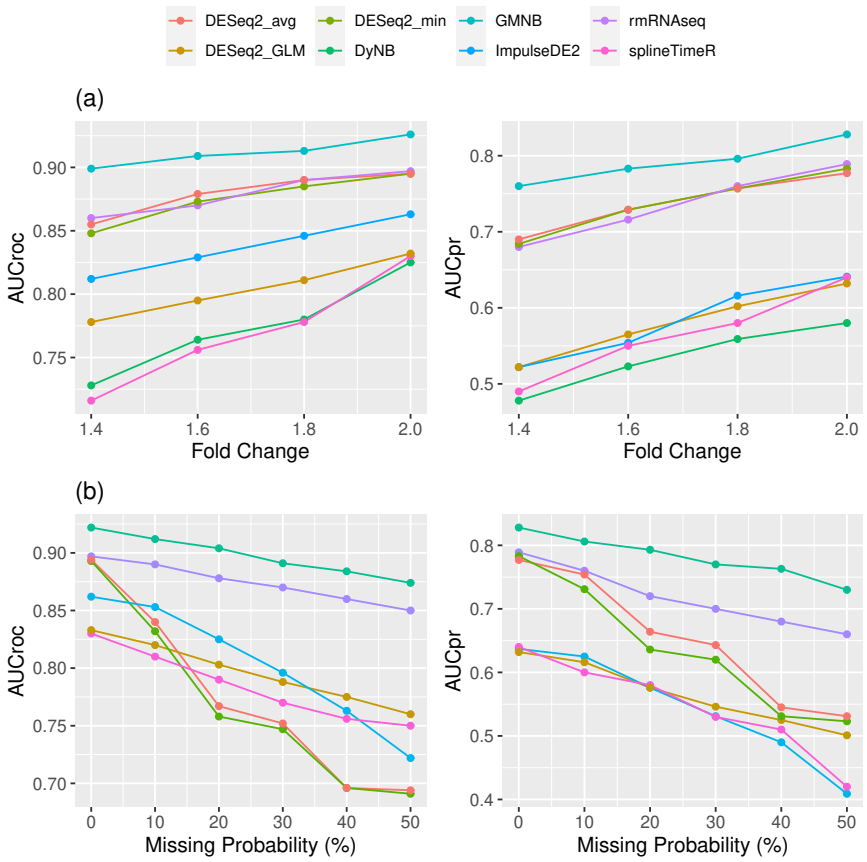


Figure 3.1: Left column: AUC-ROC values, Right column: AUC-PR values. Performance comparison of different methods in detecting differential gene expression over time under various (a) fold changes and (b) missing probability.

change is small, suggesting that explicitly modeling the temporal dependency is insufficient in detecting any changes over time and/or between time points. This raises the concern on the capability of ImpulseDE2, splineTimeR, and DyNB to analyze complex temporal RNA-seq data under this generative model. DyNB performs worse than DESeq2 based methods and ImpulseDE2, indicating the smooth assumption of DyNB may not always hold for the data generated by this gamma-Markov-chain based generative model.

To further compare the operating characteristics of different models in dealing with missing data, we show in Figure 3.1(b) the AUC-ROC and AUC-PR curves for the fold change of $b = 2$. To take account of missing data, at each time point there is a probability of not observing the expression of a

gene that we set this probability as 10%, 20%, 30%, 40%, or 50% in simulating synthetic data. It is interesting to notice that the performance of the static methods, i.e., DESeq2-min and DESeq2-avg, quickly drops even with 10% missing data, showing the importance of a need for a temporal model. In fact GMNB clearly outperforms the other methods that can consider temporal dependency in various missing probabilities. ImpulseDE2 outperforms DESeq2-min and DESeq2-avg in terms of AUC-ROC when the missing probability is higher than 10%. When there are more than 10% missing values, performance of DESeq2-GLM and splineTimeR are better than the static methods and increasing the missing probability will increase their difference. This shows that incorporating all data is necessary when the data are not complete, often witnessed in real-world datasets. Note that, as the DyNB implementation cannot deal with different numbers of samples in different time points, we are unable to compare its performance for missing data.

3.4.2 Human Th17 cell induction

Having validated our method on simulated data, we present a case study consisting of 57 human samples during the priming of T helper 17 (Th17) cell differentiation [51] to further illustrate how GMNB may help identify differentially expressed genes from temporal RNA-seq data for biologically significant results. The main goal of designing this case study is to gain insights into the differentiation process by unraveling dependency between different genetic factors in various pathways, which may serve as potential biomarkers of immunological diseases for therapeutic intervention design. In this dataset [49], at $0, 0.5, 1, 2, 4, 6, 12, 24, 48$, and 72 hours of Th17 polarized cells and control Th0 cells, three biological replicates were collected for transcript profiling by RNA-seq. The accession number of this dataset is GSE52260 [49, 50].

When checking the ten most differentially expressed genes based on their BFs by GMNB, we find that all of them have been reported to be differentially expressed in other studies investigating Th17 cell differentiation. Among them, the top differentially expressed gene is thrombospondin-1 (*TSP1*), whose encoded protein participates in the differentiation of Th17 cells by activating transforming growth factor beta (*TGF-*β) and enhancing the inflammatory response in experimental autoimmune encephalomyelitis (*EAE*) [52]. The second gene in the list is Lymphotoxin α (*LTA*), a

member of the tumor necrosis factor (*TNF*) superfamily that is both secreted and expressed on the cell surface of activated Th17 cells [53]. The third gene, *COL6A3*, contributes to adipose tissue inflammation [54] and responds quickly to Th17 cell polarizing stimulation [55]. The rest of the genes in the list, *CTSL1*, *FURIN*, *LMNA*, *FLNA*, *SBNO2*, *ACTB*, and *NOTCH1*, are reported as either the immune response regulators or T cell activation genes, as detailed in Appendix C.2.

We then investigate how the results of DyNB differ from those of GMNB. The majority of the above genes are indeed ranked relatively high by DyNB as differentially expressed, except two genes: *FLNA* and *ACTB*. For these two genes, their expression levels change abruptly after 12 hours of T17 differentiation. These two genes show that the DyNB method may fail to detect temporal differential expression when the temporal gene expression trends are not smooth. As an instance, Figure 3.2 shows that DyNB is not able to capture the temporal expression changes of gene FLNA accurately. More precisely, Figure 3.2(a) shows the posterior means of expected gene expression μ_k based on DyNB and their corresponding confidence intervals, where circles and diamonds represent the normalized counts from Th0 and Th17 lineages, respectively. In this chapter, the normalized expression profiles based on read counts on the y-axis for DyNB are obtained by using the normalization method of DESeq.

To further assess the power of the models in reproducing the observed gene counts, for each model, we generate 1000 gene counts per sample and time points based on the inferred parameters, and then calculate the 99% confidence interval using these synthetically generated counts. Figure 3.2(b) shows the means and confidence intervals of the counts generated via this procedure for DyNB.

As similarly shown in the plots in Figures 3.2(a) and 3.2(b), we perform the same examination on the expression pattern of *FLNA* by the GMNB model. To demonstrate the expression levels of the kth gene between two conditions, DyNB compares the posterior NB mean parameters μ_k, whereas GMNB compares the posterior NB shape parameters r_k. One may consider that the expression level of gene k is assumed to roughly follow a function of the shape parameter r_k in GMNB, but the observed counts should be demonstrated in the same scale as the shape parameter. The difference

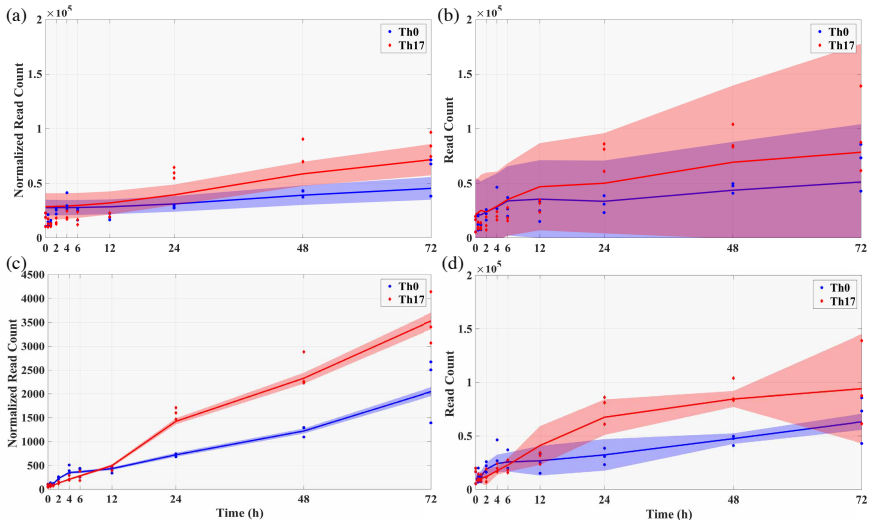


Figure 3.2: Inferred expression profiles of *FLNA*, detected by GMNB but not by DyNB. Left column: The normalized expression profiles: The solid blue and red curves are the posterior means of (a) μ_k by DyNB and (c) r_k by GMNB under Th0 and Th17 lineages, respectively, with 99% CIs (shaded areas). Right column: The inferred read counts over time: The solid blue and red curves with shaded areas correspond to the inferred parameters by (b) DyNB and (d) GMNB similarly.

between the posterior shape parameters r_k explains the differences between the means, since if $y_{kj}^{(t)} \sim \text{NB}(r_k^{(t)}, p_j^{(t)})$, then $\mathbb{E}[y_{kj}^{(t)}] = r_k^{(t)} p_j^{(t)}/(1-p_j^{(t)})$. Therefore, Figure 3.2(c) shows the posterior means of r_k based on GMNB and their corresponding confidence intervals, where the circles and diamonds are obtained by dividing the observed counts by the parameter $p_j^{(t)}/(1-p_j^{(t)})$ representing the sequencing depth in the proposed model. Additionally, Figure 3.2(d) demonstrates the means and confidence intervals for synthetically generated gene counts based on the inferred parameters of GMNB, where the read counts on the y-axis are observed read counts. Not only does GMNB improve the model fitting over 24h to 72h, but also it has more robust estimation of expression patterns for the starting time points with lower counts (Figures 3.2(c) and 3.2(d)). The calculated BFs for the gene *FLNA* are 2.3461 and 1.60×10^{308} by DyNB and GMNB, respectively.

GMNB also identifies *ACTB* as a gene with significant differential temporal expression (BF $>$ 10) but DyNB again fails to capture the abrupt expression changes and thereby associates it with low BF (Appendix C). The corresponding temporal expression plots are depicted in Figure C.4.

Compared to the results from other methods for this dataset, ImpulseDE2 identifies *FLNA* (p-value$= 1.7 \times 10^{-12}$) as differentially expressed, but not *ACTB* (p-values$= 1$). Figure C.5 shows the expression patterns of these two genes estimated by ImpulseDE2. Similar as ImpulseDE2, rmRNAseq identifies *FLNA* (p-value$= 7.1 \times 10^{-3}$) as differentially expressed, but not *ACTB* (p-values$= 0.99$). This might be due to the inherent smoothness assumption of rmRNAseq, as the expression pattern of *ACTB* at last time point does not follow its expression trend at the previous time points. The method splineTimeR identifies both genes as differentially expressed where the p-values are 2.2×10^{-5} and 8.5×10^{-3} for *FLNA* and *ACTB*, respectively. Figure C.6 shows the expression patterns of these two genes estimated by splineTimeR. The better performance of splineTimeR compared to ImpulseDE2 is expected for temporal gene expression with high signal-to-noise ratio and more time points, as pointed out by [37].

On the other hand, *LGALS1*, *SEPT5*, *COL1A2*, and *ENO2* are four genes out of 90 differentially expressed genes detected by DyNB with BFs 2.59×10^7, 472.34, 404.34, and 398.43, respectively, whereas they are associated with BFs lower than 10 by GMNB. Figure 3.3 illustrates the expression profile of the gene *LGALS1* inferred by DyNB and GMNB, indicating that DyNB is not able to filter out those low count genes, for which the replicated Th0 and Th17 lineages are seemingly similar, leading to this potential false positive. On the contrary, GMNB considers this gene not significantly differentially expressed with similar inferred temporal expression profiles across conditions, as demonstrated in Figures 3.3(c) and 3.3(d). This may be explained by the fact that GMNB employs a fully generative model of gene expressions, including the sequencing depth, while DyNB uses a deterministic ad-hoc procedure to normalize gene counts, and thus neglecting the uncertainty over the sequencing depth when computing the BF, leading to potential false positives. We provide more detailed discussions for the other genes in Appendix C.2.

To further demonstrate the advantages of GMNB, we investigate the distribution of each top 100 differentially expressed gene set over the other five methods. More specifically, the overlap of six approaches (GMNB, ImpulseDE2, DyNB, rmRNAseq, splineTimeR, and DESeq2-min), for 100 most differentially expressed genes identified by GMNB is depicted as the UpSet diagram [56] in

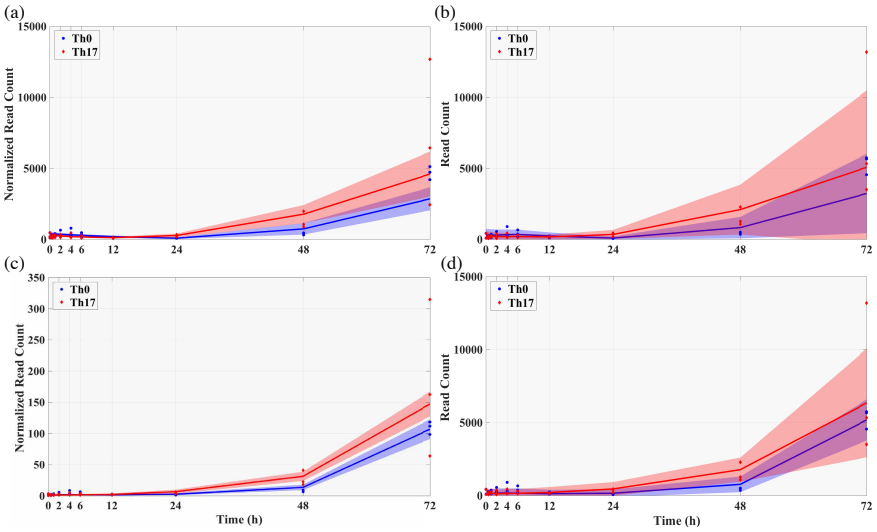


Figure 3.3: Inferred expression profiles of *LGALS1*, detected by DyNB but not by GMNB. They are analogous plots to those in Figure 3.2, with different genes.

Figures 3.4. Throughout the chapter, we consider a gene to be differentially expressed if BF > 10 or adjusted $p-$value < 0.05 depending on the methods.

Out of the top 100 differentially expressed genes identified by GMNB, seven genes are not identified by the other five methods, demonstrating the potential capability of GMNB in identifying a broader range of gene expression patterns. Among these genes, *EGR1* is a transcription factor known to inhibit the expression of *GFI1*, a negative regulator of Th17 differentiation, by directly binding to its promoter and its expression is detected only in the early phase of Th17 differentiation [57]. The gene *MYC* has been reported as one of the key transcript factors for Th17 differentiation [58]. The critical roles of the other genes have also been reported in other studies investigating Th17 cell differentiation. We provide more detailed discussions together with their temporal expression plots indicating differential dynamic patterns identified by GMNB, in Section C.2 of the Appendix C.

This again illustrates the advantage of GMNB on better modeling temporal dynamic changes to detect biologically meaningful genes, which show significant differences in temporal changes but do not show significant differential expression when studying them at individual time points.

All the top 100 genes identified by either DyNB or DESeq2-min are considered differentially

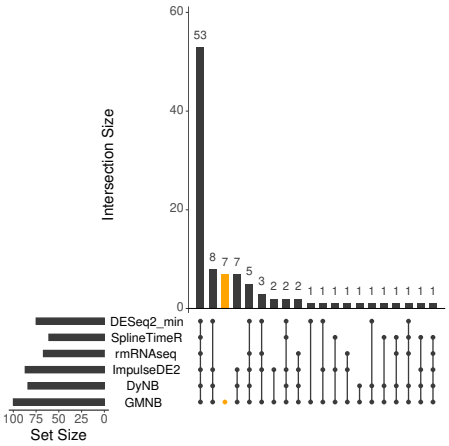


Figure 3.4: The UpSet diagram representing the distribution of the top 100 differentially expressed genes detected by GMNB over other methods.

expressed by GMNB, i.e. their BFs are greater than 10 (Figure C.13). It is important to notice that seven genes in the top expressed gene set of DyNB is only identified by GMNB but not by other methods, indicating GMNB can also better capture the genes with smooth expression dynamic patterns. Unlike ImpulseDE2 whose performance is limited to the trajectories with abrupt changes, GMNB is a powerful approach to model both smooth expression trajectories and the patterns with abrupt changes.

Other methods, i.e. ImpulseDE2, rmRNAseq, and SplineTimeR, have a couple of false positives. We provide the same UpSet diagrams with more detailed discussions in the Appendix C

3.4.2.1 Gene Set Enrichment Analysis

To further demonstrate the biological relevance of the detected genes by GMNB, gene ontology (GO) analysis of top 100 differentially expressed genes ($\log(\text{BF}) > 100$) has been performed using Fisher's exact test. Enriched GO terms (Table C.2) by these genes agree with the current biological understanding of the Th17 differentiation process. The most significantly enriched GO terms are related to the organ development (p-value $< 2 \times 10^{-23}$), immune system process (p-value $< 6 \times 10^{-21}$), immune response (p-value $< 1 \times 10^{-19}$), response to stimulus (p-value $< 3^{-19}$), cell differentiation (p-value $< 3 \times 10^{-18}$), and defense response (p-value $< 2 \times 10^{-16}$). In particular,

Table 3.1: Comparison of the ranking for the validated genes by DyNB.

Genes	*RORC*	*IL17F*	*IL17A*
GMNB	**3** (2.98×10^{48})	**14** (2.53×10^{9})	**31** (3.90×10^{4})
DyNB	33 (2.26×10^{93})	351 (1.74×10^{15})	754 (6.96×10^{8})
ImpulseDE2	12 (1.01×10^{-47})	361 (2.08×10^{-4})	2067 (1)
rmRNAseq	56 (3.24×10^{-5})	87 (1.30×10^{-4})	46 (1.63×10^{-5})
splineTimeR	24 (4.67×10^{-8})	690 (0.09)	644 (0.08)
DESeq2-GLM	6 (1.35×10^{-17})	187 (4.52×10^{-9})	320 (0.01)
DESeq2-mean	4 (0.20)	214 (0.45)	282 (0.56)
DESeq2-min	5 (1.06×10^{-64})	229 (1.02×10^{-4})	78 (7.44×10^{-11})

38% and 74% of these 100 genes are annotated to immune response and response to stimulus, respectively, supported by the central role of Th17 cells in the pathogenesis of autoimmune and inflammatory diseases [59].

To investigate how *a priori* known sets of Th17 genes are being captured as statistically significant by different models, we also compare the results of all methods in terms of the enrichment of the archived Th17 genes in MSigDB [60]. We rank the results of GMNB and DyNB based on the BF while all the other methods based on their p-values. The enrichment scores for the gene set GSE27241, i.e. Genes up-regulated in polarizing CD4 Th17 cells, are 0.89, 0.40, 0.43, 0.32, 0.40, 0.25, 0.28, and 0.13 for GMNB, DyNB, ImpulseDE2, rmRNAseq, splineTimeR, DESeq2-GLM, DESeq2-min, and DESeq2-mean, respectively. Table C.4 provides the enrichment analysis of five different sets of Th17 genes. We also investigate how many Th17 related genes are among top 1000 genes identified by different methods as detailed in the Appendix C. Again, GMNB identifies more Th17 related genes compared to other methods, demonstrating its biological significance.

3.4.3 RNA-seq data in [1]: Human-activated T- and Th17 cells

We further illustrate the practical utility of GMNB to identify differentially expressed genes from another temporal RNA-seq data studying Th17 cell lineage as detailed in Section C.3 of the Appendix C [1].In [1], DyNB was implemented for dynamic differential expression analysis. Out of its identified 698 genes, three genes were investigated and discussed with the qRT-PCR validation: *IL17A*, *IL17F*, and *RORC*.

We apply GMNB to analyze the same Th17 cell lineage dataset to identify differentially expressed genes. Table 3.1 provides the ranking of these three validated genes [1], as well as the computed BFs or p-values by GMNB, DyNB, ImpulseDE2, rmRNAseq, splineTimeR, DESeq2-GLM, DESeq2-mean, and DESeq2-min. After differential expression analysis, we first remove the genes which their fold changes are less than 2 and then ranked the genes based on their p-values or BFs. In the case of GMNB, DyNB, DESeq2-mean, and DESeq2-min methods, we keep the genes whose fold changes are lager than 2 at least in one time point. While *RORC* is highly ranked by all methods, the other two genes are particularly ranked higher by GMNB compared to the other methods. This is mostly due to the higher expression of *RORC* but relatively lower expression of two other genes, showing the superior performance of GMNB in low expression scenarios, which can be more difficult for DE analysis in practice. While the cytokine *IL17A* is known to be highly expressed in Th17 cells [61, 62], ImpulseDE2 and splineTimeR can not identify this gene as differentially expressed. The expression of *IL17A* is commonly used to assess the Th17 polarization efficiency [63]. This again shows that ImpulseDE2 performs poorly when the expression change is small but statistically significant. Note that splineTimeR can identify neither *IL17F* nor *IL17A* as differentially expressed. This is consistent with the finding in [37] that the overall performance of splinTimeR can go down when either the expression value change or the number of available time points is small. Th17 cells also have been shown to be important in autoimmunity and clearance of mucosal infection by producing proinflammatory cytokines *IL17F* [64]. Figure C.24 shows the estimated trajectories of these genes based on GMNB, DyNB, ImpulseDE2, and splineTimeR.

3.5 Discussion and conclusions

We propose gamma Markov negative binomial (GMNB) as a fully Bayesian solution to study temporal RNA-seq data. A notable advantage is the capacity to capture a broad range of gene expression patterns over time by the integration of a gamma Markov chain into a negative binomial distribution model. This allows GMNB to offer consistent performance over different generative models and makes it be robust for studies with different numbers of replicates by borrowing the statistical strength across both genes and samples. Another critical characteristic is the efficient

closed-form Gibbs sampling inference of the model parameters, which improves the computational complexity compared to the state-of-the-art methods. This is achieved by using a statistically well-grounded data augmentation solution. In addition, GMNB explicitly models the potential sequencing depth heterogeneity so that no heuristic preprocessing step is required. Multi-level factor analysis based on the proposed Bayes factor might lead to inefficient analysis if more complex experiment design is needed. We plan to extend GMNB for our future study to incorporate other confounding covariates to achieve more accurate genetic marker identification, for example, as similarly done in our recent Bayesian negative binomial regression [65].

4. VARIATIONAL GRAPH RECURRENT NEURAL NETWORKS*

4.1 Overview

Representation learning over graph structured data has been mostly studied in static graph settings while efforts for modeling dynamic graphs are still scant. In this chapter, we develop a novel hierarchical variational model that introduces additional latent random variables to jointly model the hidden states of a graph recurrent neural network (GRNN) to capture both topology and node attribute changes in dynamic graphs. We argue that the use of high-level latent random variables in this variational GRNN (VGRNN) can better capture potential variability observed in dynamic graphs as well as the uncertainty of node latent representation. With semi-implicit variational inference developed for this new VGRNN architecture (SI-VGRNN), we show that flexible non-Gaussian latent representations can further help dynamic graph analytic tasks. Our experiments with multiple real-world dynamic graph datasets demonstrate that SI-VGRNN and VGRNN consistently outperform the existing baseline and state-of-the-art methods by a significant margin in dynamic link prediction.

4.2 Introduction

Node embedding maps each node in a graph to a vector in a low-dimensional latent space, in which classical feature vector-based machine learning formulations can be adopted [66]. Most of the existing node embedding techniques assume that the graph is static and that learning tasks are performed on fixed sets of nodes and edges [67, 68, 69, 70, 71, 72, 73]. However, many real-world problems are modeled by *dynamic* graphs, where graphs are constantly evolving over time. Such graphs have been typically observed in social networks, citation networks, and financial transaction networks. A naive solution to node embedding for dynamic graphs is simply applying static methods to each snapshot of dynamic graphs. Among many potential problems of such a naive solution, it is

*Reprinted with permission from "Variational graph recurrent neural networks" by E. Hajiramezanali, A. Hasanzadeh, K. Narayanan, N. Duffield, M. Zhou, and X. Qian. Advances in Neural Information Processing Systems, pp. 10701-10711. 2019. Copyright 2019 by Curran Associates, Inc.

clear that it ignores the temporal dependencies between snapshots.

Several node embedding methods have been proposed to capture the temporal graph evolution for both networks without attributes [74, 75] and attributed networks [76, 77]. However, all of the existing dynamic graph embedding approaches represent each node by a deterministic vector in a low-dimensional space [78]. Such deterministic representations lack the capability of modeling uncertainty of node embedding, which is a natural consideration when having multiple information sources, i.e. node attributes and graph structure.

In this chapter, we propose a novel node embedding method for dynamic graphs that maps each node to a random vector in the latent space. More specifically, we first introduce a dynamic graph autoencoder model, namely graph recurrent neural network (GRNN), by extending the use of graph convolutional recurrent neural networks (GCRN) [79] to dynamic graphs. Then, we argue that GRNN lacks the expressive power for fully capturing the complex dependencies between topological evolution and time-varying node attributes because the output probability in standard RNNs is limited to either a simple unimodal distribution or a mixture of unimodal distributions [80, 81, 82, 83, 84, 85, 86, 87, 88]. Next, to increase the expressive power of GRNN in addition to modeling the uncertainty of node latent representations, we propose variational graph recurrent neural network (VGRNN) by adopting high-level latent random variables in GRNN. Our proposed VGRNN is capable of learning interpretable latent representation as well as better modeling of very sparse dynamic graphs.

To further boost the expressive power and interpretability of our new VGRNN method, we integrate semi-implicit variational inference [89] with VGRNN. We show that semi-implicit variational graph recurrent neural network (SI-VGRNN) is capable of inferring more flexible and complex posteriors. Our experiments demonstrate the superior performance of VGRNN and SI-VGRNN in dynamic link prediction tasks in several real-world dynamic graph datasets compared to baseline and state-of-the-art methods.

4.3 Background

Graph convolutional recurrent networks (GCRN). GCRN was introduced by [79] to model time series data defined over nodes of a static graph. Series of frames in videos and spatio-temporal measurements on a network of sensors are two examples of such datasets. GCRN combines graph convolutional networks (GCN) [90] with recurrent neural networks (RNN) to capture spatial and temporal patterns in data. More precisely, given a graph G with N nodes, whose topology is determined by the adjacency matrix $\mathbf{A} \in \mathbb{R}^{N \times N}$, and a sequence of node attributes $\mathcal{X} = \{\mathbf{X}^{(1)}, \mathbf{X}^{(2)}, \ldots, \mathbf{X}^{(T)}\}$, GCRN reads M-dimensional node attributes $\mathbf{X}^{(t)} \in \mathbb{R}^{N \times M}$ and updates its hidden state $\mathbf{h}_t \in \mathbb{R}^p$ at each time step t:

$$\mathbf{h}_t = f\left(\mathbf{A}, \mathbf{X}^{(t)}, \mathbf{h}_{t-1}\right). \tag{4.1}$$

Here f is a non-probabilistic deep neural network. It can be any recursive network including gated activation functions such as long short-term memory (LSTM) or gated recurrent units (GRU), where the deep layers inside them are replaced by graph convolutional layers. GCRN models node attribute sequences by parameterizing a factorization of the joint probability distribution as a product of conditional probabilities such that

$$p\left(\mathbf{X}^{(1)}, \mathbf{X}^{(2)}, \ldots, \mathbf{X}^{(T)} \,|\, \mathbf{A}\right) = \prod_{t=1}^{T} p\left(\mathbf{X}^{(t)} \mid \mathbf{X}^{(<t)}, \mathbf{A}\right); \quad p\left(\mathbf{X}^{(t)} \mid \mathbf{X}^{(<t)}, \mathbf{A}\right) = g(\mathbf{A}, \mathbf{h}_{t-1}).$$

Due to the deterministic nature of the transition function f, the choice of the mapping function g here effectively defines the only source of variability in the joint probability distributions $p(\mathbf{X}^{(1)}, \mathbf{X}^{(2)}, \ldots, \mathbf{X}^{(T)} \,|\, \mathbf{A})$ that can be expressed by the standard GCRN. This can be problematic for sequences that are highly variable. More specifically, when the variability of X is high, the model tries to map this variability in hidden states h, leading to potentially high variations in h and thereafter overfitting of training data. Therefore, GCRN is not fully capable of modeling sequences with high variations. This fundamental problem of autoregressive models has been addressed for

non-graph-structured datasets by introducing stochastic hidden states to the model [91, 80, 92].

In this chapter, we integrate GCN and RNN into a graph RNN (GRNN) framework, which is a dynamic graph autoencoder model. While GCRN aims to model dynamic node attributes defined over a static graph, GRNN can get different adjacency matrices at different time snapshots and reconstruct the graph at time t by adopting an inner-product decoder on the hidden state $\mathbf{h}_t$. More specifically, $\mathbf{h}_t$ can be viewed as node embedding of the dynamic graph at time t. To further improve the expressive power of GRNN, we introduce stochastic latent variables by combining GRNN with variational graph autoencoder (VGAE) [72]. This way, not only we can capture time dependencies between graphs without making smoothness assumption, but also each node is represented with a distribution in the latent space. Moreover, the prior construction devised in VGRNN allows it to predict links in the future time steps.

Semi-implicit variational inference (SIVI). SIVI has been shown effective to learn posterior distributions with skewness, kurtosis, multimodality, and other characteristics, which were not captured by the existing variational inference methods [89]. To characterize the latent posterior $q(\mathbf{z}|\mathbf{x})$, SIVI introduces a mixing distribution on the parameters of the original posterior distribution to expand the variational family with a hierarchical construction: $\mathbf{z} \sim q(\mathbf{z}|\boldsymbol{\psi})$ with $\boldsymbol{\psi} \sim q_{\boldsymbol{\phi}}(\boldsymbol{\psi})$. $\boldsymbol{\phi}$ denotes the distribution parameter to be inferred. While the original posterior $q(\mathbf{z}|\boldsymbol{\psi})$ is required to have an analytic form, its mixing distribution is not subject to such a constraint, and so the marginal posterior distribution is often implicit and more expressive that has no analytic density function. It is also common that the marginal of the hierarchy is implicit, even if both the posterior and its mixing distribution are explicit. We will integrate SIVI in our new model to infer more flexible and interpretable node embedding for dynamic graphs.

4.4 Variational graph recurrent neural network (VGRNN)

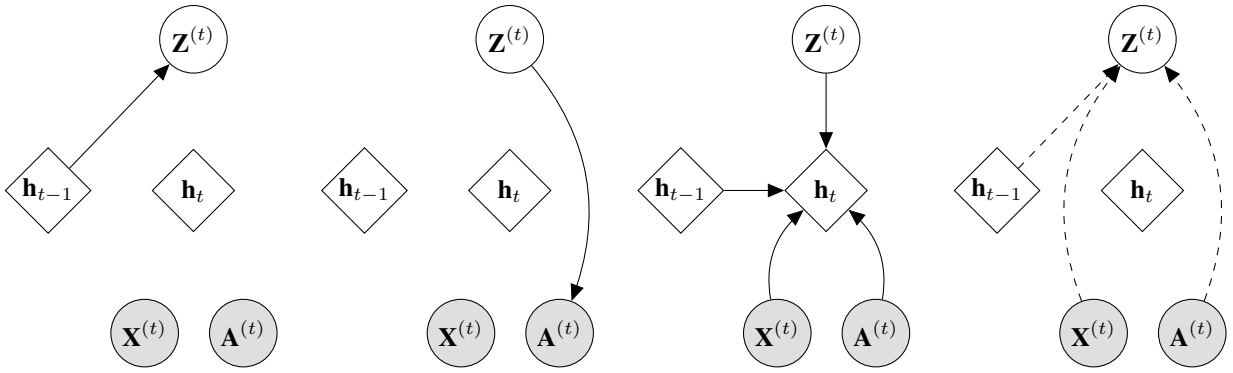
4.4.1 Overview

We consider a dynamic graph $\mathcal{G} = \{G^{(1)}, G^{(2)}, \ldots, G^{(T)}\}$ where $G^{(t)} = (\mathcal{V}^{(t)}, \mathcal{E}^{(t)})$ is the graph at time step t with $\mathcal{V}^{(t)}$ and $\mathcal{E}^{(t)}$ being the corresponding node and edge sets, respectively. In

this chapter, we aim to develop a model that is universally compatible with potential changes in both node and edge sets. In particular, the cardinality of both $\mathcal{V}^{(t)}$ and $\mathcal{E}^{(t)}$ can change across time. There are no constraints on the relationships between $(\mathcal{V}^{(t)}, \mathcal{E}^{(t)})$ and $(\mathcal{V}^{(t+1)}, \mathcal{E}^{(t+1)})$, namely new nodes can join the dynamic graph and create edges to the existing nodes or previous nodes can disappear from the graph. On the other hand, new edges can form between snapshots while existing edges can disappear. Let N_t denotes the number of nodes , i.e., the cardinality of $\mathcal{V}^{(t)}$, at time step t. Therefore, VGRNN can take as input a variable-length adjacency matrix sequence $\mathcal{A} = \{\mathbf{A}^{(1)}, \mathbf{A}^{(2)}, \ldots, \mathbf{A}^{(T)}\}$. In addition, when considering node attributes, different attributes can be observed at different snapshots with a variable-length node attribute sequence $\mathcal{X} = \{\mathbf{X}^{(1)}, \mathbf{X}^{(2)}, \ldots, \mathbf{X}^{(T)}\}$. Note that $\mathbf{A}^{(t)}$ and $\mathbf{X}^{(t)}$ are $N_t \times N_t$ and $N_t \times M$ matrices, respectively, where M is the dimension of the node attributes that is constant across time. Inspired by variational recurrent neural networks (VRNN) [80], we construct VGRNN by integrating GRNN and VGAE so that complex dependencies between topological and node attribute dynamics are modeled sufficiently and simultaneously. Moreover, each node at each time is represented with a distribution, hence uncertainty of latent representations of nodes are also modelled in VGRNN.

4.4.2 VGRNN model

Generation. The VGRNN model adopts a VGAE to model each graph snapshot. The VGAEs across time are conditioned on the state variable $\mathbf{h}_{t-1}$, modeled by a GRNN. Such an architecture design will help each VGAE to take into account the temporal structure of the dynamic graph. More critically, unlike a standard VGAE, our VGAE in VGRNN takes a new prior on the latent random variables by allowing distribution parameters to be modelled by either explicit or implicit complex functions of information of the previous time step. More specifically, instead of imposing a standard multivariate Gaussian distribution with deterministic parameters, VGAE in our VGRNN learns the prior distribution parameters based on the hidden states in previous time steps. Hence, our VGRNN allows more flexible latent representations with greater expressive power that captures dependencies between and within topological and node attribute evolution processes. In particular, we can write



(a) Prior (b) Generation (c) Recurrence (d) Inference

Figure 4.1: Graphical illustrations of each operation of VGRNN; (a) computing the conditional prior by (4.2); (b) decoder function (4.3); (c) updating the GRNN hidden states using (4.4); and (d) inference of the posterior distribution by (4.4.2).

the construction of the prior distribution adopted in our experiments as follows,

$$p\left(\mathbf{Z}^{(t)}\right)=\prod_{i=1}^{N_t} p\left(\mathbf{Z}_i^{(t)}\right);\;\; \mathbf{Z}_i^{(t)}\sim\mathcal{N}\left(\boldsymbol{\mu}_{i,\text{prior}}^{(t)},\text{diag}((\boldsymbol{\sigma}_{i,\text{prior}}^{(t)})^2)\right),\;\; \left\{\boldsymbol{\mu}_{\text{prior}}^{(t)},\boldsymbol{\sigma}_{\text{prior}}^{(t)}\right\}=\varphi^{\text{prior}}(\mathbf{h}_{t-1}), \tag{4.2}$$

where $\boldsymbol{\mu}_{\text{prior}}^{(t)}\in\mathbb{R}^{N_t\times l}$ and $\boldsymbol{\sigma}_{\text{prior}}^{(t)}\in\mathbb{R}^{N_t\times l}$ denote the parameters of the conditional prior distribution, and $\boldsymbol{\mu}_{i,\text{prior}}^{(t)}$ and $\boldsymbol{\sigma}_{i,\text{prior}}^{(t)}$ are the i-th row of $\boldsymbol{\mu}_{\text{prior}}^{(t)}$ and $\boldsymbol{\sigma}_{\text{prior}}^{(t)}$, respectively. Moreover, the generating distribution will be conditioned on $\mathbf{Z}^{(t)}$ as:

$$\mathbf{A}^{(t)}\,|\,\mathbf{Z}^{(t)}\sim\text{Bernoulli}\left(\pi^{(t)}\right),\quad \pi^{(t)}=\varphi^{\text{dec}}\left(\mathbf{Z}^{(t)}\right), \tag{4.3}$$

where $\pi^{(t)}$ denotes the parameter of the generating distribution; φ^{prior} and φ^{dec} can be any highly flexible functions such as neural networks.

On the other hand, the backbone GRNN enables flexible modeling of complex dependency involving both graph topological dynamics and node attribute dynamics. The GRNN updates its hidden states using the recurrence equation:

$$\mathbf{h}_t=f\left(\mathbf{A}^{(t)},\varphi^{\mathbf{x}}\left(\mathbf{X}^{(t)}\right),\varphi^{\mathbf{z}}\left(\mathbf{Z}^{(t)}\right),\mathbf{h}_{t-1}\right), \tag{4.4}$$

where f is originally the transition function from equation (4.1). Unlike the GRNN defined in [79], graph topology can change in different time steps as it does in real-world dynamic graphs, and the adjacency matrix $\mathbf{A}^{(t)}$ is time dependent in VGRNN. To further enhance the expressive power, $\varphi^{\mathbf{x}}$ and $\varphi^{\mathbf{z}}$ are deep neural networks which operate on each node independently and extract features from $\mathbf{X}^{(t)}$ and $\mathbf{Z}^{(t)}$, respectively. These feature extractors are crucial for learning complex graph dynamics. Based on (4.4), $\mathbf{h}_t$ is a function of $\mathbf{A}^{\leq(t)}$, $\mathbf{X}^{\leq(t)}$, and $\mathbf{Z}^{\leq(t)}$. Therefore, the prior and generating distributions in equations (4.2) and (4.3) define the distributions $p(\mathbf{Z}^{(t)} \,|\, \mathbf{A}^{(<t)}, \mathbf{X}^{(<t)}, \mathbf{Z}^{(<t)})$ and $p(\mathbf{A}^{(t)} \,|\, \mathbf{Z}^{(t)})$, respectively. The generative model can be factorized as

$$p\left(\mathbf{A}^{(\leq T)}, \mathbf{Z}^{(\leq T)} \,|\, \mathbf{X}^{(<T)}\right) = \prod_{t=1}^{T} p\left(\mathbf{Z}^{(t)} \,|\, \mathbf{A}^{(<t)}, \mathbf{X}^{(<t)}, \mathbf{Z}^{(<t)}\right) p\left(\mathbf{A}^{(t)} \,|\, \mathbf{Z}^{(t)}\right), \tag{4.5}$$

where the prior of the first snapshot is considered to be a standard multivariate Gaussian distribution, i.e. $p(\mathbf{Z}_i^{(0)} \,|\, -) \sim \mathcal{N}(0, \mathbf{I})$ for $i \in \{1, \ldots, N_0\}$ and $\mathbf{h}_0 = \mathbf{0}$. Also, if a previously unobserved node is added to the graph at snapshot t, we consider the hidden state of that node at snapshot $t-1$ is zero and hence the prior for that node at time t is $\mathcal{N}(0, \mathbf{I})$. If node deletion occurs, we assume that the identity of nodes can be maintained thus removing a node, which is equivalent to removing all the edges connected to it, will not affect the prior construction for the next step. More specifically, the sizes of $\mathbf{A}$ and $\mathbf{X}$ can change in time while their latent space maintains across time.

Inference. With the VGRNN framework, the node embedding for dynamic graphs can be derived by inferring the posterior distribution of $\mathbf{Z}^{(t)}$ which is also a function of $\mathbf{h}_{t-1}$. More specifically,

$$\begin{aligned} q\left(\mathbf{Z}^{(t)} \,|\, \mathbf{A}^{(t)}, \mathbf{X}^{(t)}, \mathbf{h}_{t-1}\right) &= \prod_{i=1}^{N_t} q\left(\mathbf{Z}_i^{(t)} \,|\, \mathbf{A}^{(t)}, \mathbf{X}^{(t)}, \mathbf{h}_{t-1}\right) = \prod_{i=1}^{N_t} \mathcal{N}\left(\boldsymbol{\mu}_{i,\text{enc}}^{(t)}, \text{diag}((\boldsymbol{\sigma}_{i,\text{enc}}^{(t)})^2)\right), \\ \boldsymbol{\mu}_{\text{enc}}^{(t)} &= \text{GNN}_{\mu}\left(\mathbf{A}^{(t)}, \text{CONCAT}\left(\varphi^{\mathbf{x}}\left(\mathbf{X}^{(t)}\right), \mathbf{h}_{t-1}\right)\right), \\ \boldsymbol{\sigma}_{\text{enc}}^{(t)} &= \text{GNN}_{\sigma}\left(\mathbf{A}^{(t)}, \text{CONCAT}\left(\varphi^{\mathbf{x}}\left(\mathbf{X}^{(t)}\right), \mathbf{h}_{t-1}\right)\right), \end{aligned} \tag{4.6}$$

where $\boldsymbol{\mu}_{\text{enc}}^{(t)}$ and $\boldsymbol{\sigma}_{\text{enc}}^{(t)}$ denote the parameters of the approximated posterior, and $\boldsymbol{\mu}_{i,\text{enc}}^{(t)}$ and $\boldsymbol{\sigma}_{i,\text{enc}}^{(t)}$ are the i-th row of $\boldsymbol{\mu}_{\text{enc}}^{(t)}$ and $\boldsymbol{\sigma}_{\text{enc}}^{(t)}$, respectively. GNN_{μ} and GNN_{σ} are the encoder functions and can be

any of the various types of graph neural networks, such as GCN [93], GCN with Chebyshev filters [90] and GraphSAGE [94].

Learning. The objective function of VGRNN is derived from the variational lower bound at each snapshot. More precisely, using equation (4.5) , the evidence lower bound of VGRNN can be written as follows,

$$\begin{aligned}\mathcal{L} = \sum_{t=1}^{T} \Bigg\{ & \mathbb{E}_{\mathbf{Z}^{(t)} \sim q\left(\mathbf{Z}^{(t)} \,|\, \mathbf{A}^{(\leq t)}, \mathbf{X}^{(\leq t)}, \mathbf{Z}^{(<t)}\right)} \log p\left(\mathbf{A}^{(t)} \,|\, \mathbf{Z}^{(t)}\right) \\ & - \mathbf{KL}\Bigg(q\left(\mathbf{Z}^{(t)} \,|\, \mathbf{A}^{(\leq t)}, \mathbf{X}^{(\leq t)}, \mathbf{Z}^{(<t)}\right) \,||\, p\left(\mathbf{Z}^{(t)} \,|\, \mathbf{A}^{(<t)}, \mathbf{X}^{(<t)}, \mathbf{Z}^{(<t)}\right)\Bigg)\Bigg\}.\end{aligned} \tag{4.7}$$

We learn the parameters of the generative and inference models jointly by optimizing the variational lower bound with respect to the variational parameters. The graphical representation of VGRNN is illustrated in Fig. 4.1, operations (a)–(d) correspond to equations (4.2) – (4.4), and (4.4.2), respectively. We note that if we don't use hidden state variables $\mathbf{h}_{t-1}$ in the derivation of the prior distribution, then the prior in (4.2) becomes independent across snapshots and reduces to the prior of vanilla VGAE.

The inner-product decoder is adopted in VGRNN for the experiments in this chapter– φ^{dec} in (4.3)–to clearly demonstrate the advantages of the stochastic recurrent models for the encoder. Potential extensions with other decoders can be integrated with VGRNN if necessary. More specifically,

$$p\left(\mathbf{A}^{(t)} \,|\, \mathbf{Z}^{(t)}\right) = \prod_{i=1}^{N_t} \prod_{j=1}^{N_t} p\left((A_{i,j}^{(t)} \,|\, \mathbf{z}_i^{(t)}, \mathbf{z}_j^{(t)}\right); \; p\left(A_{i,j}^{(t)} = 1 \,|\, \mathbf{z}_i^{(t)}, \mathbf{z}_j^{(t)}\right) = \text{sigmoid}\left(\mathbf{z}_i^{(t)} (\mathbf{z}_j^{(t)})^T\right), \tag{4.8}$$

where $\mathbf{z}_i^{(t)}$ corresponds to the embedding representation of node $v_i^{(t)} \in \mathcal{V}^{(t)}$ at time step t. Note the generating distribution can also be conditioned on $\mathbf{h}_{t-1}$ if we want to generate $\mathbf{X}^{(t)}$ in addition to the adjacency matrix for other applications. In such cases, φ^{dec} should be a highly flexible neural network instead of a simple inner-product function.

4.4.3 Semi-implicit VGRNN (SI-VGRNN)

To further increase the expressive power of the variational posterior of VGRNN, we introduce a SI-VGRNN dynamic node embedding model. We impose a mixing distributions on the variational distribution parameters in (4.8) to model the posterior of VGRNN with a semi-implicit hierarchical construction:

$$\mathbf{Z}^{(t)} \sim q(\mathbf{Z}^{(t)} \mid \boldsymbol{\psi}_t), \qquad \boldsymbol{\psi}_t \sim q_{\boldsymbol{\phi}}(\boldsymbol{\psi}_t \mid \mathbf{A}^{(\leq t)}, \mathbf{X}^{(\leq t)}, \mathbf{Z}^{(<t)}) = q_{\boldsymbol{\phi}}(\boldsymbol{\psi}_t | \mathbf{A}^{(t)}, \mathbf{X}^{(t)}, \mathbf{h}_{t-1}). \tag{4.9}$$

While the variational distribution $q(\mathbf{Z}^{(t)} \mid \boldsymbol{\psi}_t)$ is required to be explicit, the mixing distribution, $q_{\boldsymbol{\phi}}$, is not subject to such a constraint, leading to considerably flexible $\mathbb{E}_{\boldsymbol{\psi}_t \sim q_{\boldsymbol{\phi}}(\boldsymbol{\psi}_t | \mathbf{A}^{(t)}, \mathbf{X}^{(t)}, \mathbf{h}_{t-1})}(q(\mathbf{z}_t | \boldsymbol{\psi}_t))$. More specifically, SI-VGRNN draws samples from $q_{\boldsymbol{\phi}}$ by transforming random noise $\boldsymbol{\epsilon}_t$ via a graph neural network, which generally leads to an implicit distribution for $q_{\boldsymbol{\phi}}$.

Inference. Under the SI-VGRNN construction, the generation, prior and recurrence models are the same as VGRNN (equations (4.2) to (4.5)). We indeed have updated the encoder functions as follows:

$$\boldsymbol{\ell}_j^{(t)} = \text{GNN}_j(\mathbf{A}^{(t)}, \text{CONCAT}(\mathbf{h}_{t-1}, \boldsymbol{\epsilon}_j^{(t)}, \boldsymbol{\ell}_{j-1}^{(t)})); \ \boldsymbol{\epsilon}_j^{(t)} \sim q_j(\boldsymbol{\epsilon}) \text{ for } j = 1, \ldots, L, \ \boldsymbol{\ell}_0^{(t)} = \varphi_\tau^{\mathbf{x}}\left(\mathbf{X}^{(t)}\right)$$

$$\boldsymbol{\mu}_{\text{enc}}^{(t)}(\mathbf{A}^{(t)}, \mathbf{X}^{(t)}, \mathbf{h}_{t-1}) = \text{GNN}_\mu(\mathbf{A}^{(t)}, \boldsymbol{\ell}_L^{(t)}), \quad \boldsymbol{\Sigma}_{\text{enc}}^{(t)}(\mathbf{A}^{(t)}, \mathbf{X}^{(t)}, \mathbf{h}_{t-1}) = \text{GNN}_\Sigma(\mathbf{A}^{(t)}, \boldsymbol{\ell}_L^{(t)}),$$

$$q(\mathbf{Z}_i^{(t)} \mid \mathbf{A}^{(t)}, \mathbf{X}^{(t)}, \mathbf{h}_{t-1}, \boldsymbol{\mu}_{i,\text{enc}}^{(t)}, \boldsymbol{\Sigma}_{i,\text{enc}}^{(t)}) = \mathcal{N}(\boldsymbol{\mu}_{i,\text{enc}}^{(t)}(\mathbf{A}^{(t)}, \mathbf{X}^{(t)}, \mathbf{h}_{t-1}), \boldsymbol{\Sigma}_{i,\text{enc}}^{(t)}(\mathbf{A}^{(t)}, \mathbf{X}^{(t)}, \mathbf{h}_{t-1})),$$

where L is the number of stochastic layers and $\boldsymbol{\epsilon}_j^{(t)}$ is N_t-dimensional random noise drawn from a distribution q_j with N_t denoting number of nodes at time t. Note that given $\{\mathbf{A}^{(t)}, \mathbf{X}^{(t)}, \mathbf{h}_{t-1}\}$, $\boldsymbol{\mu}_{i,\text{enc}}^{(t)}$ and $\boldsymbol{\Sigma}_{i,\text{enc}}^{(t)}$ are now random variables rather than analytic and thus the posterior is not Gaussian after marginalizing.

Learning. In this construction, because the parameters of the posterior are random variables, the

Table 4.1: Dataset statistics for VGRNN.

Metrics	Enron	COLAB	Facebook	HEP-TH	Cora	Social Evolution
Number of Snapshots	11	10	9	40	11	27
Number of Nodes	184	315	663	1199-7623	708-2708	84
Number of Edges	115-266	165-308	844-1068	769-34941	406-5278	303-1172
Average Density	0.01284	0.00514	0.00591	0.00117	0.00154	0.21740
Number of Node Attributes	-	-	-	-	1433	168

ELBO goes beyond the simple VGRNN in (7) and can be written as

$$\begin{aligned}\mathcal{L} = \sum_{t=1}^{T} \Bigg\{ & \mathbb{E}_{\boldsymbol{\psi}_t \sim q_{\boldsymbol{\phi}}(\boldsymbol{\psi}_t | \mathbf{A}^{(t)}, \mathbf{X}^{(t)}, \mathbf{h}_{t-1})} \mathbb{E}_{\mathbf{Z}^{(t)} \sim q(\mathbf{Z}^{(t)} \mid \boldsymbol{\psi}_t)} \log \Big(p(\mathbf{A}^{(t)} \mid \mathbf{Z}^{(t)}, \mathbf{h}_{t-1}) \Big) \\ & - \mathbf{KL} \Bigg(\mathbb{E}_{\boldsymbol{\psi}_t \sim q_{\boldsymbol{\phi}}(\boldsymbol{\psi}_t | \mathbf{A}^{(t)}, \mathbf{X}^{(t)}, \mathbf{h}_{t-1})} q \left(\mathbf{Z}^{(t)} \mid \boldsymbol{\psi}_t \right) \, || \, p(\mathbf{Z}^{(t)} \mid \mathbf{h}_{t-1}) \Bigg) \Bigg\}. \end{aligned} \tag{4.10}$$

Direct optimization of the ELBO in SIVI is not tractable [89], hence to infer variational parameters of SI-VGRNN, we derive a lower bound for the ELBO as follows (see the Appendix E for more details.).

$$\underline{\mathcal{L}} = \sum_{t=1}^{T} \mathbb{E}_{\boldsymbol{\psi}_t \sim q_{\boldsymbol{\phi}}(\boldsymbol{\psi}_t | \mathbf{A}^{(t)}, \mathbf{X}^{(t)}, \mathbf{h}_{t-1})} \mathbb{E}_{\mathbf{Z}^{(t)} \sim q(\mathbf{Z}^{(t)} \mid \boldsymbol{\psi}_t)} \log \left(\frac{p(\mathbf{A}^{(t)} \mid \mathbf{Z}^{(t)}, \mathbf{h}_{t-1}) \, p(\mathbf{Z}^{(t)} \mid \mathbf{h}_{t-1})}{q(\mathbf{Z}^{(t)} \mid \boldsymbol{\psi}_t)} \right). \tag{4.11}$$

4.5 Experiments

Datasets. We evaluate our proposed methods, VGRNN and SI-VGRNN, and baselines on six real-world dynamic graphs as described in Table 4.1. More detailed descriptions of the datasets can be found in Appendix F.1.

Competing methods. We compare the performance of our proposed methods against four competing node embedding methods, three of which have the capability to model evolving graphs with changing node and edge sets. Among these four, two (**DynRNN** and **DynAERNN** [95]) are based on RNN models. By comparing our models to these methods, we will be able to see how much improvement we may obtain by improving the backbone RNN with our new prior construction compared to these RNNs with deterministic hidden states. We also compare our methods against a

deep autoencoder with fully connected layers (**DynAE** [95]) to show the advantages of RNN based sequential learning methods. Last but not least, our methods are compared with **VGAE** [72], which is implemented to analyze each snapshot separately, to demonstrate how temporal dependencies captured through hidden states in the backbone GRNN can improve the performance. More detailed descriptions of these selected competing methods are described in Appendix F.2.

Evaluation tasks. In the dynamic graph embedding literature, the term *link prediction* has been used with different definitions. While some of the previous works focused on link prediction in a transductive setting and others proposed inductive models, our models are capable of working in both settings. We evaluate our proposed models on three different *link prediction* tasks that have been widely used in the dynamic graph representation learning studies. More specifically, given partially observed snapshots of a dynamic graph $\mathcal{G} = \{G^{(1)}, \ldots, G^{(T)}\}$ with node attributes $\mathcal{X} = \{\mathbf{X}^{(1)}, \ldots, \mathbf{X}^{(T)}\}$, dynamic link prediction problems are defined as follows: 1) **dynamic link detection**, i.e. detect unobserved edges in $G^{(T)}$; 2) **dynamic link prediction**, i.e. predict edges in $G^{(T+1)}$; 3) **dynamic new link prediction**, i.e. predict edges in $G^{(T+1)}$ that are not in $G^{(T)}$.

Experimental setups. For performance comparison, we evaluate different methods based on their ability to correctly classify true and false edges. For dynamic link detection problem, we randomly remove 5% and 10% of all edges at each time for validation and test sets, respectively. We also randomly select the equal number of non-links as validation and test sets to compute average precision (AP) and area under the ROC curve (AUC) scores. For dynamic (new) link prediction, all (new) edges are set to be true edges and the same number of non-links are randomly selected to compute AP and AUC scores. In all of our experiments, we test the model on the last three snapshots of dynamic graphs while learning the parameters of the models based on the rest of the snapshots except for HEP-TH where we test the model on the last 10 snapshots. For the datasets without node attributes, we consider the N_t-dimensional identity matrix as node attributes at time t. Numbers show mean results and standard error for 10 runs on random datasets splits with random initializations.

For all datasets, we set up our VGRNN model to have a single recurrent hidden layer with 32

GRU units. All φ's in equations (3), (4), and (6) are modeled by a 32-dimensional fully-connected layer. We use two 32-dimensional fully-connected layers for φ^{prior} in (4.2) and 2-layer GCN with sizes equal to [32, 16] to model $\boldsymbol{\mu}_{\text{enc}}^{(t)}$ and $\boldsymbol{\sigma}_{\text{enc}}^{(t)}$ in (6). For SI-VGRNN, a stochastic GCN layer with size 32 and an additional GCN layer of size 16 are used to model the $\boldsymbol{\mu}$. The dimension of injected standard Gaussian noise ϵ is 16. The covariance matrix $\boldsymbol{\Sigma}$ is deterministic and is inferred through two layers of GCNs with sizes equal to [32, 16]. For fair comparison, the number of parameters are the same for the competing methods. In all experiments, we train the models for 1500 epochs with the learning rate 0.01. We use the validation set for the early stopping. The Appendix F contains additional implementation details with hyperparmaeter selection. We implemented (SI-)VGRNN in PyTorch [96] and the implementation of our proposed models is accessible at `https://github.com/VGraphRNN/VGRNN`.

4.5.1 Results and discussion

Dynamic link detection. Table 4.2 summarizes the results for inductive link detection in different datasets. Our proposed methods, VGRNN and SI-VGRNN, outperform competing methods across all datasets by large margins. Improvement made by (SI-)VGRNN compared to GRNN and DynAERNN supports our claim that latent random variables carry more information than deterministic hidden states specially for dynamic graphs with complex temporal changes. Comparing the (SI-)VGRNN with VGAE, which is a static graph embedding method, shows that the improvement of the proposed methods is not only because of introducing stochastic latent variables, but also successful modelling of temporal dependencies. We note that methods that take node attributes as input, i.e VGAE, GRNN and (SI-)VGRNN, outperform other competing methods by a larger margin in Cora dataset which includes node attributes.

Comparing SI-VGRNN with VGRNN shows that the Gaussian latent distribution may not always be the best choice for latent node representations. SI-VGRNN with flexible variational inference can learn more complex latent structures. The results for the Cora dataset, which also includes attributes, clearly magnify the benefits of flexible posterior as SI-VGRNN improves the accuracy by 2% compared to VGRNN. We also note that the improvement made by SI-VGRNN

Table 4.2: AUC and AP scores of inductive dynamic link detection on dynamic graphs.

Metrics	Methods	Enron	COLAB	Facebook	Social Evo.	HEP-TH	Cora
AUC	VGAE	88.26 ± 1.33	70.49 ± 6.46	80.37 ± 0.12	79.85 ± 0.85	79.31 ± 1.97	87.60 ± 0.54
	DynAE	84.06 ± 3.30	66.83 ± 2.62	60.71 ± 1.05	71.41 ± 0.66	63.94 ± 0.18	53.71 ± 0.48
	DynRNN	77.74 ± 5.31	68.01 ± 5.50	69.77 ± 2.01	74.13 ± 1.74	72.39 ± 0.63	76.09 ± 0.97
	DynAERNN	91.71 ± 0.94	77.38 ± 3.84	81.71 ± 1.51	78.67 ± 1.07	82.01 ± 0.49	74.35 ± 0.85
	GRNN	91.09 ± 0.67	86.40 ± 1.48	85.60 ± 0.59	78.27 ± 0.47	89.00 ± 0.46	91.35 ± 0.21
	VGRNN	**94.41 ± 0.73**	**88.67 ± 1.57**	**88.00 ± 0.57**	**82.69 ± 0.55**	**91.12 ± 0.71**	**92.08 ± 0.35**
	SI-VGRNN	**95.03 ± 1.07**	**89.15± 1.31**	**88.12 ± 0.83**	**83.36 ± 0.53**	**91.05 ± 0.92**	**94.07 ± 0.44**
AP	VGAE	89.95 ± 1.45	73.08 ± 5.70	79.80 ± 0.22	79.41 ± 1.12	81.05 ± 1.53	89.61 ± 0.87
	DynAE	86.30 ± 2.43	67.92 ± 2.43	60.83 ± 0.94	70.18 ± 1.98	63.87 ± 0.21	53.84 ± 0.51
	DynRNN	81.85 ± 4.44	73.12 ± 3.15	70.63 ± 1.75	72.15 ± 2.30	74.12 ± 0.75	76.54 ± 0.66
	DynAERNN	93.16 ± 0.88	83.02 ± 2.59	83.36 ± 1.83	77.41 ± 1.47	85.57 ± 0.93	79.34 ± 0.77
	GRNN	93.47 ± 0.35	88.21 ± 1.35	84.77 ± 0.62	76.93± 0.35	89.50 ± 0.42	91.37 ± 0.27
	VGRNN	**95.17 ± 0.41**	**89.74 ± 1.31**	**87.32 ± 0.60**	**81.41 ± 0.53**	**91.35 ± 0.77**	**92.92 ± 0.28**
	SI-VGRNN	**96.31 ± 0.72**	**89.90 ± 1.06**	**87.69 ± 0.92**	**83.20± 0.57**	**91.42 ± 0.86**	**94.44 ± 0.52**

compared to VGRNN is marginal in Facebook dataset. The reason could be that Gaussian latent variables already represent the graph well. Therefore, more flexible posteriors do not enhance the performance significantly.

Dynamic (new) link prediction. Tables 4.3 and 4.4 show the results for link prediction and new link prediction, respectively. Since GRNN is trained as an autoencoder, it cannot predict edges in the next snapshot. However, in (SI-)VGRNN, the prior construction based on previous time steps allows us to predict links in the future. Note that none of the methods can predict new nodes, therefore, HEP-TH and Cora datasets are not evaluated for these tasks. VGRNN and SI-VGRNN outperform the competing methods significantly in both tasks for all of the datasets which proves that our proposed models have better generalization, which is the result of including random latent variables in our model. We note that our proposed methods improve new link prediction more substantially which shows that they can capture temporal trends better than the competing methods. Comparing VGRNN with SI-VGRNN shows that the prediction results are almost the same for all datasets. The reason is that although the posterior is more flexible in SI-VGRNN, the prior on which our predictions are based, is still Gaussian, hence the improvement is marginal. A possible avenue for further improvements is constructing more flexible priors such as semi-implicit priors proposed by [97], which we leave for future studies.

To find out when VGRNN and SI-VGRNN show more improvements compared to the baselines,

Table 4.3: AUC and AP scores of dynamic link prediction on dynamic graphs.

Metrics	Methods	Enron	COLAB	Facebook	Social Evo.
AUC	DynAE	74.22 ± 0.74	63.14 ± 1.30	56.06 ± 0.29	65.50 ± 1.66
	DynRNN	86.41 ± 1.36	75.7 ± 1.09	73.18 ± 0.60	71.37 ± 0.72
	DynAERNN	87.43 ± 1.19	76.06 ± 1.08	76.02 ± 0.88	73.47 ± 0.49
	VGRNN	$\mathbf{93.10 \pm 0.57}$	$\mathbf{85.95 \pm 0.49}$	$\mathbf{89.47 \pm 0.37}$	$\mathbf{77.54 \pm 1.04}$
	SI-VGRNN	$\mathbf{93.93 \pm 1.03}$	$\mathbf{85.45 \pm 0.91}$	$\mathbf{90.94 \pm 0.37}$	$\mathbf{77.84 \pm 0.79}$
AP	DynAE	76.00 ± 0.77	64.02 ± 1.08	56.04 ± 0.37	63.66 ± 2.27
	DynRNN	85.61 ± 1.46	78.95 ± 1.55	75.88 ± 0.42	69.02 ± 1.71
	DynAERNN	89.37 ± 1.17	81.84 ± 0.89	78.55 ± 0.73	71.79 ± 0.81
	VGRNN	$\mathbf{93.29 \pm 0.69}$	$\mathbf{87.77 \pm 0.79}$	$\mathbf{89.04 \pm 0.33}$	$\mathbf{77.03 \pm 0.83}$
	SI-VGRNN	$\mathbf{94.44 \pm 0.85}$	$\mathbf{88.36 \pm 0.73}$	$\mathbf{90.19 \pm 0.27}$	$\mathbf{77.40 \pm 0.43}$

we take a closer look at three of the datasets. Figure 4.2 shows the temporal evolution of density and clustering coefficients of COLAB, Enron, and Facebook datasets. Enron shows the highest density and clustering coefficients, indicating that it contains dense clusters who are densely connected with each other. COLAB have low density and high clustering coefficients across time, which means that although it is very sparse but edges are mostly within the clusters. Facebook, which has both low density and clustering coefficients, is very sparse with almost no clusters. Looking back at (new) link prediction results, we see that the improvement margin of (SI-)VGRNN compared to competing methods is more substantial for Facebook. Moreover, the improvement margin diminishes when the graph has more clusters and is more dense. Predicting the evolution very sparse graphs with no clusters is indeed a very difficult task (arguably more difficult than dense graphs), in which our proposed (SI-)VGRNN is very successful. The stochastic latent variables in our models can capture the temporal trend while other methods tend to overfit very few observed links.

4.5.2 Interpretable latent representations

To show that VGRNN learns more interpretable latent representations, we simulated a dynamic graph with three communities in which a node (red colored node) transfers from one community into another in two time steps (Figure 4.3). We embedded the node into 2-d latent space using VGRNN (Figure 4.4) and DynAERNN (the best performed baseline; Figure F.1). While the advantages of modeling uncertainty for latent representations and its relation to node labels (classes) for static graphs have been discussed in [78], we argue that the uncertainty is also directly related to

Table 4.4: AUC and AP scores of dynamic new link prediction on dynamic graphs.

Metrics	Methods	Enron	COLAB	Facebook	Social Evo.
AUC	DynAE	66.10 ± 0.71	58.14 ± 1.16	54.62 ± 0.22	55.25 ± 1.34
	DynRNN	83.20 ± 1.01	71.71± 0.73	73.32 ± 0.60	65.69 ± 3.11
	DynAERNN	83.77 ± 1.65	71.99 ± 1.04	76.35 ± 0.50	66.61 ± 2.18
	VGRNN	**88.43 ± 0.75**	**77.09 ± 0.23**	**87.20 ± 0.43**	**75.00 ± 0.97**
	SI-VGRNN	**88.60 ± 0.95**	**77.95 ± 0.41**	**87.74 ± 0.53**	**76.45 ± 1.19**
AP	DynAE	66.50 ± 1.12	58.82 ± 1.06	54.57 ± 0.20	54.05 ± 1.63
	DynRNN	80.96 ± 1.37	75.34 ± 0.67	75.52 ± 0.50	63.47 ± 2.70
	DynAERNN	85.16 ± 1.04	77.68 ± 0.66	78.70 ± 0.44	65.03 ± 1.74
	VGRNN	**87.57 ± 0.57**	**79.63 ± 0.94**	**86.30 ± 0.29**	**73.48 ± 1.11**
	SI-VGRNN	**87.88 ± 0.84**	**81.26 ± 0.38**	**86.72 ± 0.54**	**73.85 ± 1.33**

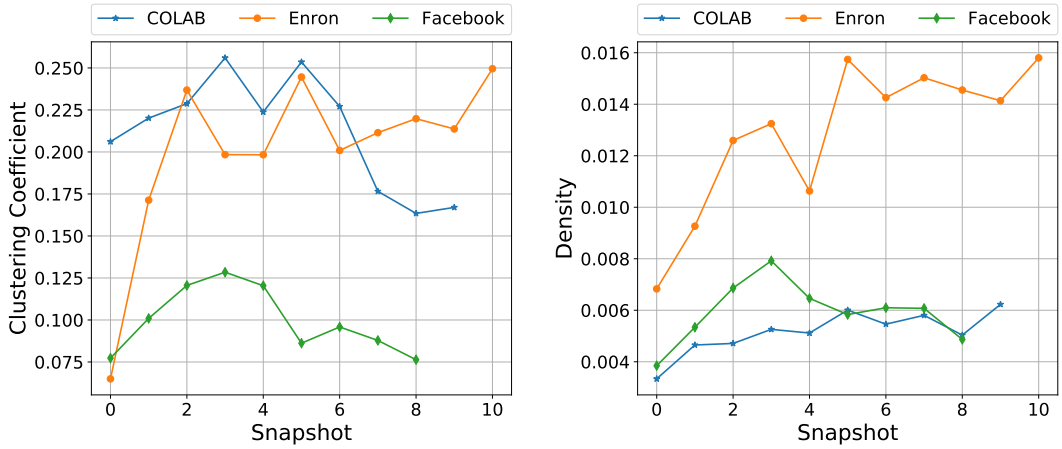


Figure 4.2: Evolution of graph statistics through time.

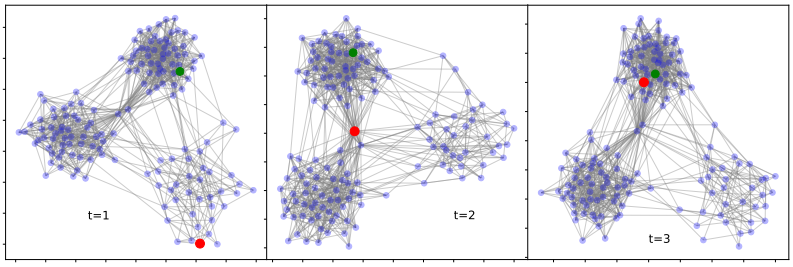


Figure 4.3: Evolution of simulated graph topology through time.

topological evolution in dynamic graphs.

More specifically, the variance of the latent variables for the node of interest increases in time (left to right) marked with the red contour. In time steps 2 and 3 (where the node is moving in

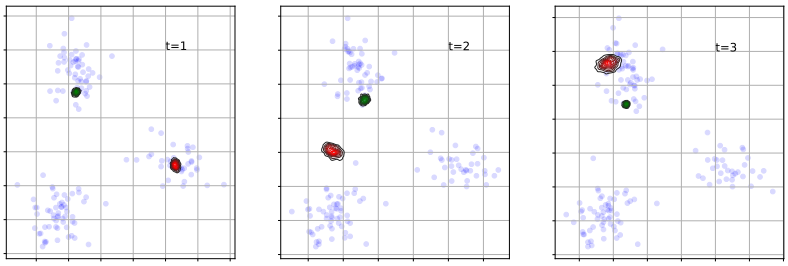


Figure 4.4: Latent representations of the simulated graph in different time steps in 2-d space using VGRNN.

the graph), the information from previous and current time steps contradicts each other; hence we expect the representation uncertainty to increase. We also plotted the variance of a node whose community doesn't change in time (marked with the green contour). As we expected, the variance of this node does not increase over time. We argue that the uncertainty helps to better encode non-smooth evolution, in particular abrupt changes, in dynamic graphs. Moreover, at time step 2, the moving node have multiple edges with nodes in two communities. Considering the inner-product decoder, which is based on the angle between the latent representations, the moving node can be connected to both of the communities which is consistent with the graph topology. We note that DynAERNN (Figure F.1) fails to produce such an interpretable latent representation. We can see that VGRNN can separate the communities in the latent space more distinctively than what DynAERNN does.

5. BAYESIAN RELATIONAL LEARNING *

5.1 Overview

High-throughput molecular profiling technologies have produced high-dimensional multi-omics data, enabling systematic understanding of living systems at the genome scale. Studying molecular interactions across different data types helps reveal signal transduction mechanisms across different classes of molecules. In this chapter, we develop a novel Bayesian representation learning method that infers the relational interactions across multi-omics data types. Our method, **Bay**esian **Re**lational **L**earning (BayReL) for multi-omics data integration, takes advantage of *a priori* known relationships among the same class of molecules, modeled as a graph at each corresponding view, to learn view-specific latent variables as well as a multi-partite graph that encodes the interactions across views. Our experiments on several real-world datasets demonstrate enhanced performance of BayReL in inferring meaningful interactions compared to existing baselines.

5.2 Introduction

Modern high-throughput molecular profiling technologies have produced rich high-dimensional data for different bio-molecules at the genome, constituting genome, transcriptome, translatome, proteome, metabolome, epigenome, and interactome scales [98, 46, 99]. Although such multi-view (multi-omics) data span a diverse range of cellular activities, developing an understanding of how these data types quantitatively relate to each other and to phenotypic characteristics remains elusive. Life and disease systems are highly non-linear, dynamic, and heterogeneous due to complex interactions not only within the same classes of molecules but also across different classes [100]. One of the most important bioinformatics tasks when analyzing such multi-omics data is how we may integrate multiple data types for deriving better insights into the underlying biological mechanisms. Due to the heterogeneity and high-dimensional nature of multi-omics data, it is

*Reprinted with permission from "BayReL: Bayesian Relational Learning for Multi-omics Data Integration" by E. Hajiramezanali, A. Hasanzadeh, N. Duffield, K. Narayanan, and X. Qian. Advances in Neural Information Processing Systems, 2020. Copyright 2020 by Curran Associates, Inc.

necessary to develop effective and affordable learning methods for their integration and analysis [98].

Modeling data across two views with the goal of extracting shared components has been typically performed by canonical correlation analysis (CCA). Given two random vectors, CCA aims to find the linear projections into a shared latent space for which the projected vectors are maximally correlated, which can help understand the overall dependency structure between these two random vectors [101]. However, it is well known that the classical CCA suffers from a lack of probabilistic interpretation when applied to high dimensional data [102] and it also cannot handle non-linearity [103]. To address these issues, probabilistic CCA (PCCA) has been proposed and extended to non-linear settings using kernel methods and neural networks [104]. Due to explicit uncertainty modeling, PCCA is particularly attractive for biomedical data of small sample sizes but high-dimensional features [105].

Despite the success of the existing CCA methods, their main limitation is that they do not exploit structural information among features that is available for biological data such as gene-gene and protein-protein interactions when analyzing multi-omics data. Using available structural information, one can gain better understanding and obtain more biologically meaningful results. Besides that, traditional CCA methods focus on aggregated association across data but are often difficult to interpret and are not very effective for inferring interactions between individual features of different datasets.

The presented work contains three major contributions: 1) We propose a novel Bayesian relation learning framework, BayReL, that can flexibly incorporate the available graph dependency structure of each view. 2) It can exploit non-linear transformations and provide probabilistic interpretation simultaneously. 3) It can infer interactions across different heterogeneous features of input datasets, which is critical to derive meaningful biological knowledge for integrative multi-omics data analysis.

5.3 Method

We propose a new graph-structured data integration method, **Bay**esian **Re**lational **L**earning (BayReL), for integrative analysis of multi-omics data. Consider data for different molecular classes

as corresponding data views. For each view, we are given a graph $G_v = (\mathcal{V}_v, \mathcal{E}_v)$ with $N_v = |\mathcal{V}_v|$ nodes, adjacency matrix $\mathbf{A}^v$, and node features in a $N_v \times D$ matrix $\mathbf{X}_v$. We note that G_v is completely defined by $\mathbf{A}^v$, hence we use them interchangeably where it does not cause confusion. We define sets $\mathcal{G} = \{G_1, \dots G_V\}$ and $\mathcal{X} = \{\mathbf{X}_1, \dots \mathbf{X}_V\}$ as the input graphs and attributes of all V views. The goal of our model is to find *inter-relations* between nodes of the graphs in different views. We model these relations as edges of a multi-partite graph $\mathfrak{G}$. The nodes in the multi-partite graph $\mathfrak{G}$ are the union of the nodes in all views, i.e. $\mathcal{V}_{\mathfrak{G}} = \bigcup_{v=1}^{V} \mathcal{V}_v$; and the edges, that will be inferred in our model, are captured in a multi-adjacency tensor $\mathcal{A} = \{\mathbf{A}^{vv'}\}_{v,v'=1,v\neq v'}^{V}$ where $\mathbf{A}^{vv'}$ is the $N_v \times N_{v'}$ bi-adjacency matrix between $\mathcal{V}_v$ and $\mathcal{V}_{v'}$. We emphasize that unlike matrix completion models, none of the edges in $\mathfrak{G}$ are assumed to be observed in our model. We infer our proposed probabilistic model using variational inference. We now introduce each of the involved latent variables in our model as well as their corresponding prior and posterior distributions. The graphical model of BayReL is illustrated in Figure 5.1.

Embedding nodes to the latent space. The first step is to embed the nodes in each view into a D_u dimensional latent space. We use view-specific latent representations, denoted by a set of $N_v \times D_u$ matrices $\mathcal{U} = \{\mathbf{U}_v\}_{v=1}^{V}$, to reconstruct the graphs as well as inferring the inter-relations. In particular, we parametrize the distribution over the adjacency matrix of each view $\mathbf{A}^v$ independently:

$$\int p_\theta(\mathcal{G}, \mathcal{U})\, \mathrm{d}\mathcal{U} = \prod_{v=1}^{V} \int p_\theta(\mathbf{A}^v, \mathbf{U}_v)\, \mathrm{d}\mathbf{U}_v = \prod_{v=1}^{V} \int p_\theta(\mathbf{A}^v \,|\, \mathbf{U}_v)\, p(\mathbf{U}_v)\, \mathrm{d}\mathbf{U}_v, \tag{5.1}$$

where we employ standard diagonal Gaussian as the prior distribution for $\mathbf{U}_v$'s. Given the input data $\{\mathbf{X}_v, G_v\}_{v=1}^{V}$, we approximate the distribution of $\mathcal{U}$ with a factorized posterior distribution:

$$q(\mathcal{U} \,|\, \mathcal{X}, \mathcal{G}) = \prod_{v=1}^{V} q(\mathbf{U}_v \,|\, \mathbf{X}_v, G_v) = \prod_{v=1}^{V} \prod_{i=1}^{N_v} q(\mathbf{u}_{i,v} \,|\, \mathbf{X}_v, G_v), \tag{5.2}$$

where $q(\mathbf{u}_{i,v} \,|\, \mathbf{X}_v, G_v)$ can be any parametric or non-parametric distribution that is derived from the input data. For simplicity, we use diagonal Gaussian whose parameters are a function of the input. More specifically, we use two functions denoted by $\varphi_v^{\mathrm{emb},\mu}(\mathbf{X}_v, G_v)$ and $\varphi_v^{\mathrm{emb},\sigma}(\mathbf{X}_v, G_v)$ to infer the

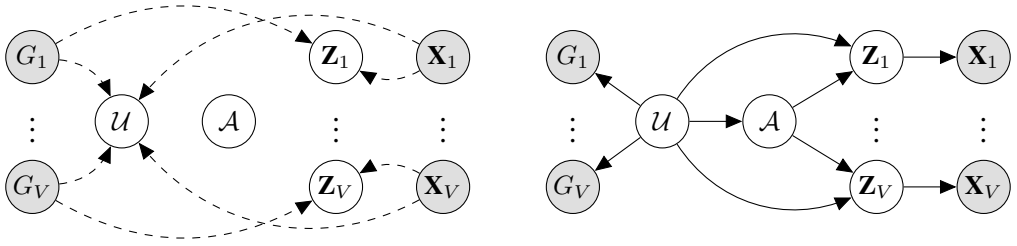


Figure 5.1: Graphical model for our proposed BayReL. Left: Inference; Right: Generative model.

mean and variance of the posterior at each view from input data. These functions could be highly flexible functions that can capture graph structure such as many variants of graph neural networks including GCN [90, 93], GraphSAGE [94], and GIN [106]. We reconstruct the graphs at each view by deploying inner-product decoder on view specific latent representations. More specifically,

$$p(\mathcal{G} \,|\, \mathcal{U}) = \prod_{v=1}^{V} \prod_{i,j=1}^{N_v} p(\mathbf{A}_{ij}^{v} \,|\, \mathbf{u}_{i,v}, \mathbf{u}_{j,v}); \qquad p(\mathbf{A}_{ij}^{v} \,|\, \mathbf{u}_{i,v}, \mathbf{u}_{j,v}) = \text{Bernoulli}\left(\sigma(\mathbf{u}_{i,v}\, \mathbf{u}_{j,v}^{T})\right), \quad (5.3)$$

where $\sigma(\cdot)$ is the sigmoid function. The above formulation for node embedding ensures that similar nodes at each view are close to each other in the latent space.

Constructing relational multi-partite graph. The next step is to construct a dependency graph among the nodes across different views. Given the latent embedding $\mathcal{U}$ that we obtain as described previously, we construct a set of bipartite graphs with multi-adjacency tensor $\mathcal{A} = \{\mathbf{A}^{vv'}\}_{v,v'=1,v\neq v'}^{V}$, where $\mathbf{A}^{vv'}$ is the bi-adjacency matrix between $\mathcal{V}_v$ and $\mathcal{V}_{v'}$. $\mathbf{A}_{ij}^{vv'} = 1$ if the node i in view v is connected to the node j in view v'. We model the elements of these bi-adjacency matrices as Bernoulli random variables. More specifically, the distribution of bi-adjacency matrices are defined as follows

$$p(\mathbf{A}^{vv'} \,|\, \mathbf{U}_v, \mathbf{U}_{v'}) = \prod_{i=1}^{N_v} \prod_{j=1}^{N_{v'}} \text{Bernoulli}\left(\mathbf{A}_{ij}^{vv'} \,|\, \varphi^{\text{sim}}(\mathbf{u}_{i,v}, \mathbf{u}_{j,v'})\right), \quad (5.4)$$

where $\varphi^{\text{sim}}(\cdot,\cdot)$ is a score function measuring the similarity between the latent representations of nodes. The inner-product link decoder $\varphi_{\text{ip}}^{\text{sim}}(\mathbf{u}_{i,v}, \mathbf{u}_{j,v'}) = \sigma(\mathbf{u}_{i,v}\, \mathbf{u}_{j,v'}^{T})$ and Bernoulli-Poisson link decoder $\varphi_{\text{bp}}^{\text{sim}}(\mathbf{u}_{i,v}, \mathbf{u}_{j,v'}) = 1 - \exp(-\sum_{k=1}^{D_u} \tau_k\, u_{ik,v}\, u_{jk,v'})$ are two examples of potential score

functions [107, 108]. In practice, we use the concrete relaxation [109, 110] during training while we sample from Bernoulli distributions in the testing phase.

We should point out that in many cases, we have a hierarchical structure between views. For example, in systems biology, proteins are products of genes. In these scenarios, we can construct the set of directed bipartite graphs, where the direction of edges embeds the hierarchy between nodes in different views. We may use an asymmetric score function or prior knowledge to encode the direction of edges. We leave this for future study.

Inferring view-specific latent variables. Having obtained the node representations $\mathcal{U}$ and the dependency multi-adjacency tensor $\mathcal{A}$, we can construct view-specific latent variables, denoted by set of $N_v \times D_z$ matrices $\mathcal{Z} = \{\mathbf{Z}_v\}_{v=1}^V$, which can be used to reconstruct the input node attributes. We parametrize the distributions for node attributes at each view independently as follows

$$\int p_\theta(\mathcal{X}, \mathcal{Z} \,|\, \mathcal{G}, \mathcal{A}, \mathcal{U}) \,\mathrm{d}\mathcal{Z} \;=\; \prod_{v=1}^{V} \prod_{i=1}^{N_v} \int p_\theta\left(\mathbf{z}_{i,v} \,|\, \mathcal{G}, \mathcal{A}, \mathcal{U}\right)\, p_\theta(\mathbf{x}_{i,v} \,|\, \mathbf{z}_{i,v}) \,\mathrm{d}\mathbf{z}_{i,v}. \tag{5.5}$$

In our formulation, the distribution of $\mathcal{X}$ is dependent on the graph structure at each view as well as inter-relations across views. This allows the local latent variable $\mathbf{z}_{i,v}$ to summarize the information from the neighboring nodes. We set the prior distribution over $\mathbf{z}_{i,v}$ as a diagonal Gaussian whose parameters are a function of $\mathbf{A}$ and $\mathcal{U}$. More specifically, first we construct the overall graph consisting of all the nodes and edges in all multi-partite graphs. We can view $\mathcal{U}$ as node attributes on this overall graph. We apply a graph neural network over this overall graph and its attributes to construct the prior. More formally, the following prior is adopted:

$$p_\theta(\mathcal{Z} \,|\, \mathcal{G}, \mathcal{A}, \mathcal{U}) = \prod_{v=1}^{V} \prod_{i=1}^{N_v} p_\theta(\mathbf{z}_{i,v} \,|\, \mathcal{G}, \mathcal{A}, \mathcal{U}), \qquad p_\theta(\mathbf{z}_{i,v} \,|\, \mathcal{G}, \mathcal{A}, \mathcal{U}) = \mathcal{N}(\boldsymbol{\mu}_{i,v}^{\text{prior}}, \boldsymbol{\sigma}_{i,v}^{\text{prior}}), \tag{5.6}$$

where $\boldsymbol{\mu}^{\text{prior}} = [\boldsymbol{\mu}_{i,v}^{\text{prior}}]_{i,v} = \varphi^{\text{prior},\mu}(\mathcal{A}, \mathcal{U})$, $\boldsymbol{\sigma}^{\text{prior}} = [\boldsymbol{\sigma}_{i,v}^{\text{prior}}]_{i,v} = \varphi^{\text{prior},\sigma}(\mathcal{A}, \mathcal{U})$, and $\varphi^{\text{prior},\mu}$ and $\varphi^{\text{prior},\sigma}$ are graph neural networks. Given input $\{\mathbf{X}_v, G_v\}_{v=1}^V$, we approximate the posterior of latent

variables with the following variational distribution:

$$q(\mathcal{Z} \mid \mathcal{X}, \mathcal{G}) = \prod_{v=1}^{V} \prod_{i=1}^{N_v} q(\mathbf{z}_{i,v} \mid \mathbf{X}_v, G_v), \qquad q(\mathbf{z}_{i,v} \mid \mathbf{X}_v, G_v) = \mathcal{N}(\boldsymbol{\mu}_{i,v}^{\text{post}}, \boldsymbol{\sigma}_{i,v}^{\text{post}}) \tag{5.7}$$

where $\boldsymbol{\mu}^{\text{post}} = [\boldsymbol{\mu}_{i,v}^{\text{post}}]_{i,v} = \{\varphi_v^{\text{post},\mu}(\mathbf{X}_v, G_v)\}_{v=1}^{V}$, $\boldsymbol{\sigma}^{\text{post}} = [\boldsymbol{\sigma}_{i,v}^{\text{post}}]_{i,v} = \{\varphi_v^{\text{post},\sigma}(\mathbf{X}_v, G_v)\}_{v=1}^{V}$, and $\varphi_v^{\text{post},\mu}$ and $\varphi_v^{\text{post},\sigma}$ are graph neural networks. The distribution over node attributes $p_\theta(\mathbf{x}_{i,v} \mid \mathbf{z}_{i,v})$ can vary based on the given data type. For instance, if $\mathcal{X}$ is count data it can be modeled by a Poisson distribution; if it is continuous, Gaussian may be an appropriate choice. In our experiments, we model the node attributes as normally distributed with a fixed variance, and we reconstruct the mean of the node attributes at each view by employing a fully connected neural network φ_v^{dec} that operates on $\mathbf{z}_{i,v}$'s independently.

Overall likelihood and learning. Putting everything together, the marginal likelihood is

$$p_\theta(\mathcal{X},\, \mathcal{G}) = \int \prod_{v=1}^{V} p_\theta(\mathbf{X}_v \mid \mathbf{Z}_v)\, p_\theta(\mathbf{Z}_v \mid \mathcal{G}, \mathcal{A}, \mathcal{U})\, p(\mathcal{A} \mid \mathcal{U})\, p(\mathcal{G} \mid \mathcal{U})\, p(\mathcal{U})\, \mathrm{d}\mathbf{Z}_1 \,\ldots\, \mathrm{d}\mathbf{Z}_V\, \mathrm{d}\mathcal{A}\, \mathrm{d}\mathcal{U}.$$

We deploy variational inference to optimize the model parameters θ and variational parameters ϕ by minimizing the following derived Evidence Lower Bound (ELBO) for BayReL:

$$\begin{aligned} \mathcal{L} = \sum_{v=1}^{V} \Big[& \mathbb{E}_{q_\phi(\mathbf{Z}_v \mid \mathcal{G}, \mathcal{X})} \log p_\theta(\mathbf{X}_v \mid \mathbf{Z}_v) + \mathbb{E}_{q_\phi(\mathbf{Z}_v, \mathcal{U} \mid \mathcal{G}, \mathcal{X})} \log p_\theta(\mathbf{Z}_v \mid \mathcal{G}, \mathcal{A}, \mathcal{U}) \\ & - \mathbb{E}_{q_\phi(\mathbf{Z}_v \mid \mathcal{G}, \mathcal{X})} q_\phi(\mathbf{Z}_v \mid \mathcal{G}, \mathcal{X}) \Big] - \mathrm{KL}\left(q_\phi(\mathcal{U} \mid \mathcal{G}, \mathcal{X}) \,||\, p(\mathcal{U})\right), \end{aligned} \tag{5.8}$$

where KL denotes the Kullback–Leibler divergence.

5.4 Related works

Graph-regularized CCA (gCCA). There are several recent CCA extensions that learn shared low-dimensional representations of multiple sources using the graph-induced knowledge of common sources [111, 112]. They directly impose the dependency graph between *samples* into a regularizer term, but are not capable of considering the dependency graph between *features*. These methods are

closely related to classic graph-aware regularizers for dimension reduction [113], data reconstruction, clustering [114], and classification. Similar to classical CCA methods, they cannot cope with high-dimensional data of small sample sizes while multi-omics data is typically that way when studying complex disease. In addition, these methods focus on latent representation learning but do not explicitly model relational dependency between features across views. Hence, they often require ad-hoc post-processing steps, such as taking correlation and thresholding, to infer inter-relations.

Bayesian CCA. Beyond classical linear algebraic solution based CCA methods, there is a rich literature on generative modelling interpretation of CCA [104, 115, 102]. These methods are attractive for their hierarchical construction, improving their interpretability and expressive power, as well as dealing with high dimensional data of small sample size. Some of them, such as [104, 102], are generic factor analysis models that decompose the data into shared and view-specific components and include an additional constraint to extract the statistical dependencies between views. Most of the generative methods retain the linear nature of CCA, but provide inference methods that are more robust than the classical solution. There are also a number of recent variational autoencoder based models that incorporate non-linearity in addition to having the probabilistic interpretability of CCA [115, 116]. Our BayReL is similar as these methods in allowing non-linear transformations. However, these models attempt to learn low-dimensional latent variables for multiple views while the focus of BayReL is to take advantage of *a priori* known relationships among features of the same type, modeled as a graph at each corresponding view, to infer a multi-partite graph that encodes the interactions across views.

Link prediction. In recent years, several graph neural network architectures have been shown to be effective for link prediction by low-dimensional embedding [94, 72, 107]. The majority of these methods do not incorporate heterogeneous graphs, with multiple types of nodes and edges, or graphs with heterogeneous node attributes [117]. In this chapter, we have to deal with multiple types of nodes, edges, and attributes in multi-omics data integration. The node embedding of our model is closely related to the Variational Graph AutoEncoder (VGAE) introduced by [72]. However, the original VGAE is designed for node embedding in a single homogeneous graph setting while in our

model we learn node embedding for all views. Furthermore, our model can be used for prediction of missing edges in each specific view. BayReL can also be adopted for graph transfer learning between two heterogeneous views to improve the link prediction in each view instead of learning them separately. We leave this for future study.

Geometric matrix completion. There have been attempts to incorporate graph structure in matrix completion for recommender systems [118, 119, 120, 121]. These methods take advantage of the known item-item and user-user relationships and their attributes to complete the user-item rating matrix. These methods either add a graph-based regularizer [120, 121], or use graph neural networks [119] in their analyses. Our method is closely related to the latter one. However, all of these methods assume that the matrix (i.e. inter-relations) is partially observed while we do not require such an assumption in BayReL, which is inherent advantage of formulating the problem as a generative model. In most of existing integrative multi-omics data analyses, there are no *a priori* known inter-relations.

5.5 Experiments

We test the performance of BayReL on capturing meaningful inter-relations across views on three real-world datasets. We compare our model with two baselines, Bayesian CCA (BCCA) [102] and Spearman's Rank Correlation Analysis (SRCA) of raw datasets. We emphasize that deep Bayesian CCA models as well as deep latent variable models are not capable of inferring inter-relations across views (even with post-processing). Specifically, these models derive low-dimensional non-linear embedding of the input samples. However, in the applications of our interest, we focus on identifying interactions between nodes across views. From this perspective, only matrix factorization based methods can achieve the similar utility for which the factor loading parameters can be used for downstream interaction analysis across views. Hence, we consider only BCCA, a well-known matrix factorization method, but not other deep latent models for benchmarking with multi-omics data.

We implement our model in TensorFlow [122]. For all datasets, we used the same architecture for BayReL as follows: Two-layer GCNs are used with a shared 16-dimensional first layer and

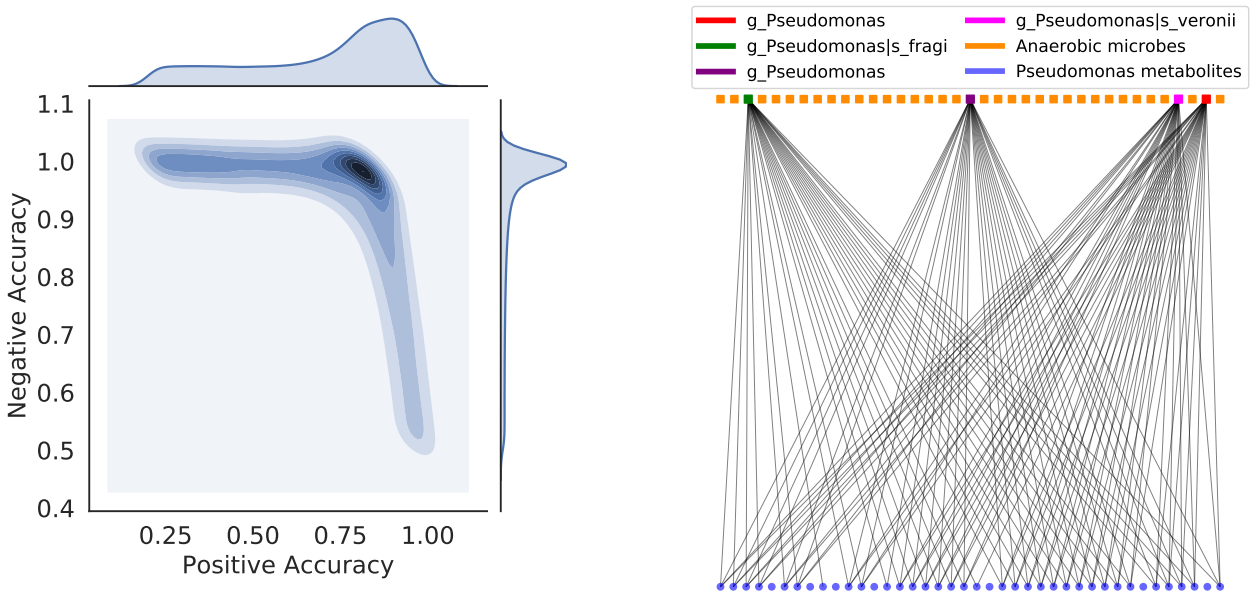


Figure 5.2: Left: Distribution of positive and negative accuracy in different training epochs for BayReL on CF dataset. Right: A sub-network of dependency graph consisting of *P. aeruginosa* micorbes, their validated targets, and anaerobic microbes, inferred using BayReL.

separate 8-dimensional output layers as $\varphi_v^{\text{emb},\mu}$, and $\varphi_v^{\text{emb},\sigma}$. We use the same embedding function for all views. Inner-product decoder is used for φ_v^{sim}. Also, we employ a one-layer 8-dimensional GCN as φ^{prior} to learn the mean of the prior. We set the variance of the prior to be one. We deploy view-specific two-layer fully connected neural networks (FCNNs) with 16 and 8 dimensional layers, followed by a two-layer GCN (16 and 8 dimensional layers) shared across views as $\varphi_v^{\text{post},\mu}$, and $\varphi_v^{\text{post},\sigma}$. Finally, we use a view-specific three-layer FCNN (8, input_dim, and input_dim dimensional layers) as φ_v^{dec}. ReLU activation functions are used. The model is trained with Adam optimizer. Also in our experiments, we multiply the term $\log p_\theta(\mathbf{Z}_v \mid \mathcal{G}, \mathcal{A}, \mathcal{U})$ in the objective function by a scalar $\alpha = 30$ during training in order to infer more accurate inter-relations. To have a fair comparison we choose the same latent dimension for BCCA as BayReL, i.e. 8. All of our results are averaged over four runs with different random seeds. More implementation details are included in the Appendix G.

5.5.1 Microbiome-metabolome interactions in cystic fibrosis

To validate whether BayReL can detect known microbe-metabolite interactions, we consider a study on the lung mucus microbiome of patients with Cystic Fibrosis (CF).

Data description. CF microbiome community within human lungs has been shown to be effected by altering the chemical environment [123]. Anaerobes and pathogens, two major groups of microbes, dominate CF. While anaerobes dominate in low oxygen and low pH environments, pathogens, in particular *P. aeruginosa*, dominate in the opposite conditions [124]. The dataset includes 16S ribosomal RNA (rRNA) sequencing and metabolomics for 172 patients with CF. Following [124], we filter out microbes that appear in less than ten samples, due to the overwhelming sparsity of microbiome data, resulting in 138 unique microbial taxa and 462 metabolite features. We use the reported target molecules of *P. aeruginosa* in studies [123] and [124] as a validation set for the microbiome-metabolome interactions.

Experimental details and evaluation metrics. We first construct the microbiome and metabolomic networks based on their taxonomies and compound names, respectively. For the microbiome network, we perform a taxonomic enrichment analysis using Fisher's test and calculate p-values for each pairs of microbes. The Benjamini-Hochberg procedure [125] is adopted for multiple test correction and an edge is added between two microbes if the adjusted p-value is lower than 0.01, resulting in 984 edges in total. The graph density of the microbiome network is 0.102. For the metabolomics network, there are 1185 edges in total, with each edge representing a connection between metabolites via a same chemical construction [124]. The graph density of the metabolite network is 0.011.

We evaluate BayReL and baselines in two metrics – 1) accuracy to identify the validated molecules interacting with *P. aeruginosa* which will be referred as *positive accuracy*, 2) accuracy of *not* detecting common targets between anaerobic microbes and notable pathogen which we refer to this measure as *negative accuracy*. More specifically, we do not expect any common metabolite targets between known anaerobic microbes (*Veillonella*, *Fusobacterium*, *Prevotella*, and *Streptococcus*) and notable pathogen *P. aeruginosa*. If a metabolite molecule x is associated with

a anaerobic microbe y, then x is more likely not to be associated with pathogen *P. aeruginosa* and vice versa. More formally, given two disjoint sets of metabolites $\mathfrak{s}_1$ and $\mathfrak{s}_2$ and the set of all microbes $\mathfrak{T}$ negative accuracy is defined as $1 - \frac{\sum_{i\in\mathfrak{s}_1}\sum_{j\in\mathfrak{s}_2}\sum_{k\in\mathfrak{T}}\mathbb{1}(i \textbf{ and } j \text{ are connected to } k)}{|\mathfrak{s}_1|\times|\mathfrak{s}_2|\times|\mathfrak{T}|}$, where $\mathbb{1}(\cdot)$ is an indicator function. Higher negative accuracy is better as there are fewer common targets between two sets of microbiomes.

Numerical results. Considering higher than 97% negative accuracy, the best positive accuracy of BayReL, BCCA, and SRCA are $82.7\% \pm 4.7$, $28.30\% \pm 3.21$, and 26.41%, respectively. Clearly, BayReL substantially outperforms the baselines with up to 54% margin. This shows that BayReL not only infers meaningful interactions with high accuracy, but also identify meicrobiome-metabolite pairs that should not interact.

We also plot the distribution of positive and negative accuracy in different training epochs for BayReL (Figure 5.2). We can see that the mass is concentrated on the top right corner, indicating that BayReL consistently generates accurate interactions in the inferred bipartite graph. Figure 5.2 also shows a sub-network of the inferred bipartite graph consisting *P. aeruginosa*, *anaerobic* microbes, and validated target nodes of *P. aeruginosa* and all of the inferred interactions by BayReL between them. While 78% of the validated edges of *P. aeruginosa* are identified by BayReL, it did not identify any connection between validated targets of *P. aeruginosa* and anaerobic microbes, i.e. negative accuracy of 100%. However, BCCA at negative accuracy of 100% could identify only one of these validated interactions. This clearly shows the effectiveness and interpretability of BayReL to identify inter-interactions.

When checking the top ten microbiome-metabolite interactions based on the predicted interaction probabilities, we find that four of them have been reported in other studies investigating CF. Among them, microbiome *Bifidobacterium*, marker of a healthy gut microbiota, has been qPCR validated to be less abundant in CF patients [126]. *Actinobacillus* and *capnocytophaga*, are commonly detected by molecular methods in CF respiratory secretions [127]. Significant decreases in the proportions of *Dialister* has been reported in CF patients receiving PPI therapy [128].

Table 5.1: Comparison of prediction sensitivity (in %) in TCGA dataset for different graph densities.

Avg. deg.	**SRCA**	**BCCA**	**BayReL**
0.2	17.58	21.08 ± 0.0	$\mathbf{34.06} \pm 2.5$
0.3	28.26	31.18 ± 0.7	$\mathbf{47.46} \pm 2.6$
0.4	37.55	41.12 ± 0.2	$\mathbf{59.50} \pm 3.0$

5.5.2 miRNA-mRNA interactions in breast cancer

We further validate whether BayReL can identify potential microRNA (miRNA)-mRNA interactions contributing to pathogenesis of breast cancer, by integrating miRNA expression with RNA sequencing (RNA-Seq) data from The Cancer Genome Atlas (TCGA) dataset [129].

Data description. It has been shown that miRNAs play critical roles in regulating genes in cell proliferation [130, 131]. To identify miRNA-mRNA interactions that have a combined effect on a cancer pathogenesis, we conduct an integrative analysis of miRNA expressions with the consequential alteration of expression profiles in target mRNAs. The TCGA data contains both miRNA and gene expression data for 1156 breast cancer (BRCA) tumor patients. For RNA-Seq data, we filter out the genes with low expression, requiring each gene to have at least 10 count per million in at least 25% of the samples, resulting in 11872 genes for our analysis. We further remove the sequencing depth effect using edgeR [132]. For miRNA data, we have the expression data of 432 miRNAs in total.

Experimental details and evaluation metrics. To take into account mRNA-mRNA and miRNA-miRNA interactions due to their involved roles in tumorigenesis, we construct a gene regulatory network (GRN) based on publicly available BRCA expression data from Clinical Proteomic Tumor Analysis Consortium (CPTAC) using the R package GENIE3 [133]. For the miRNA-miRNA interaction networks, we construct a weighted network based on the functional similarity between pairs of miRNAs using MISIM v2.0 [134]. We used miRNA-mRNA interactions reported by miRNet [135] as validation set. We calculate prediction sensitivity of interactions among validated ones while tracking the average density of the overall constructed graphs. We note that predicting

meaningful interactions while inferring sparse graphs is more desirable as the interactions are generally sparse.

Numerical results. The results for prediction sensitivity (i.e. true positive rate) of BayReL and baselines with different average node degrees based on the interaction probabilities in the inferred bipartite graph are reported in Table 5.1. As we can see, BayReL outperforms baselines by a large margin in all settings. With the increasing average node degree (i.e. more dense bipartite graph), the improvement in sensitivity is more substantial for BayReL.

We also investigate the robustness of BayReL and BCCA to the number of training samples. Table 5.2 shows the prediction sensitivity of both models while using different percentage of samples to train the models. Using 50% of all the samples, while the average prediction sensitivity of BayReL reduces less than 2% in the worst case scenario (i.e. average node density 0.20), BCCA's performance degraded around 6%. This clearly shows the robustness of BayReL to the number of training samples. In addition, we compare BayReL and BCCA in terms of consistency of identifying significant miRNA-mRNA interactions as well. We leave out 75% and 50% of all samples to infer the bipartite graphs, and then compare them with the identified miRNA-mRNA interactions using all of the samples. The Jensen–Shannon (JS) divergence values between the Bernoulli distribution of two inferred bipartite graphs for BayReL are 0.35 and 0.31 when using 25% and 50% of samples, respectively. The JS divergence values for BCCA are 0.64 and 0.62, using 25% and 50% of samples, respectively. The results prove that BayReL performs better than BCCA with fewer number of observed samples.

To further show the interpretability of BayReL, we inspect the top inferred interactions. Within them, multiple miRNAs appeared repeatedly. One of them is mir-155 which has been shown to regulate cell survival, growth, and chemosensitivity by targeting FOXO3 in breast cancer [136]. Another identified miRNA is mir-148b which has been reported as the biomarker for breast cancer prognosis [137].

5.5.3 Precision medicine in acute myeloid leukemia

We apply BayReL to identify molecular markers for targeted treatment of acute myeloid leukemia (AML) by integrating gene expression profiles and *in vitro* sensitivity of tumor samples to chemotherapy drugs with multi-omics prior information incorporated.

Classical multi-omics data integration to identify all gene markers of each drug faces several challenges. First, compared to the number of involved molecules and system complexity, the number of available samples for studying complex disease, such as cancer, is often limited, especially considering disease heterogeneity. Second, due to the many biological and experimental confounders, drug response could be associated with gene expressions that do not reflect the underlying drug's biological mechanism (i.e., false positive associations) [138]. We show even with a small number of samples, BayReL improves the performance of the classical methods by incorporating prior knowledge.

Data description. This *in vitro* drug sensitivity study has both gene expression and drug sensitivity data to a panel of 160 chemotherapy drugs and targeted inhibitors across 30 AML patients [139]. While 62 drugs are approved by the U.S. Food and Drug Administration (FDA) and encompassed a broad range of drug action mechanisms, the others are investigational drugs for cancer patients. Following [139], we study 53 out of 160 drugs that have less than 50% cell viability in at least half of the patient samples. Similar at the Cancer Cell Line Encyclopedia (CCLE) [138] and MERGE [139] studies, we use the area under the curve (AUC) to indicate drug sensitivity across a range of drug concentrations. For gene expression, we pre-processed RNA-Seq data for 9073 genes [139].

Table 5.2: Prediction sensitivity (in %) in TCGA dataset for different percentage of training samples.

	BCCA			**BayReL**		
Avg. degree	# of training samples			# of training samples		
	289 (25%)	578 (50%)	1156 (100%)	289 (25%)	578 (50%)	1156 (100%)
0.20	17.4 ± 0.8	17.6 ± 1.0	21.0 ± 0.0	31.9 ± 3.0	32.1 ± 1.0	34.0 ± 2.5
0.30	26.0 ± 0.8	26.4 ± 1.0	31.1 ± 0.7	45.8 ± 3.1	45.9 ± 1.5	47.4 ± 2.6
0.40	35.4 ± 0.8	35.5 ± 0.7	41.1 ± 0.2	57.6 ± 4.4	58.7 ± 1.3	59.5 ± 3.0

Experimental details and evaluation metrics. To apply BayReL, we first construct the GRN based on the publicly available expression data of the 14 AML cell lines from CCLE using R package GENIE3. We also construct drug-drug interaction networks based on their action mechanisms. Specifically, the selected 53 drugs are categorized into 20 broad pharmacodynamics classes [139]; 14 classes contain more than one drugs. Only 16 out of the 53 drugs are shared across two classes. We consider that two drugs interact if they belong to the same class.

We evaluate BayReL on this dataset in two ways: 1) The prediction sensitivity of identifying reported drug-gene interactions based on 797 interactions archived in The Drug–Gene Interaction Database (DGIdb) [140]. Note that DGIdb contains only the interactions for 43 of the 53 drugs included in our study. 2) Consistency of significant gene-drug interactions in two different AML datasets with 30 patients and 14 cell lines. We compare BayReL with BCCA in consistency of significant gene-drug interactions, where all 30 patient samples are used for discovery and the discovered interactions are validated using 14 cell lines.

Numerical results. Table 5.3 shows BayReL outperforms both SRCA and BCCA at different average node degrees in terms of identifying validated gene-drug interactions. If we compare the results by BayReL and BCCA, their performance difference increases with the increasing density of the bipartite graph. While BayReL outperforms BCCA by 8% at the average degree 0.10, the improved margin increases to 10.7% at the average degree 0.50. This confirms that BayReL can identify potential gene-drug interactions more robustly.

We also compare the gene-drug interactions when we learn the graph using all 30 patient samples and 14 cell lines. The JL divergence between two inferred bipartite graphs are 0.38 and 0.67 for BayReL and BCCA, respectively. This could potentially account for the lower consistency rate of BCCA compared to BayReL. The capability of flexibly incorporating prior knowledge as view-specific graphs is an important factor for BayReL achieving more consistent results.

Table 5.3: Comparison of prediction sensitivity (in %) in AML dataset for different graph densities.

Avg. degree	0.10	0.15	0.20	0.25	0.30	0.40	0.50
SRCA	8.03	12.00	17.15	20.70	26.85	34.93	45.79
BCCA	9.65 ± 0.75	14.34 ± 0.06	18.96 ± 0.42	23.29 ± 0.52	28.22 ± 0.66	38.02 ± 2.15	46.88 ± 1.88
BayReL	$\mathbf{15.56} \pm 0.75$	$\mathbf{21.70} \pm 0.65$	$\mathbf{27.20} \pm 0.17$	$\mathbf{32.43} \pm 1.02$	$\mathbf{37.76} \pm 0.85$	$\mathbf{47.90} \pm 0.43$	$\mathbf{56.76} \pm 0.50$

6. CONCLUSIONS

In this dissertation, several Bayesian learning models and corresponding inference methods for heterogeneous data in life sciences have been developed.

To address issues due to task heterogeneity in NGS count data as well as limited sample size when focusing on the task of interest, we have developed a multi-domain NB latent factorization model for Bayesian multi-domain learning (BMDL). By introducing this hierarchical Bayesian model with selector variables to flexibly assign both domain-specific and globally shared latent factors to different domains, the derived latent representations of NGS data preserves predictive information in corresponding domains so that accurate cancer subtyping is possible even with a limited number of samples. As BMDL learns domain relevance based on given samples across domains and enables the flexibly of sharing useful information through common latent factors (if any), BMDL performs consistently better than single-domain learning regardless of the domain relevance level. Our experiments have shown the promising potential of BMDL in accurate and reproducible cancer subtyping with "small" data through effective multi-domain learning by taking advantage of available data from different sources.

A fully Bayesian solution based on a gamma Markov chain model, GMNB, is developed to study temporal RNA-seq data with longitudinal heterogeneity. The most notable advantage is the capacity to capture a broad range of gene expression patterns over time by the integration of a gamma Markov chain into a negative binomial distribution model. This allows GMNB to offer consistent performance over different generative models and makes it be robust for studies with different numbers of replicates by borrowing the statistical strength across both genes and samples. Similar as BMDL, one of the critical advantage of GMNB is its efficient closed-form Gibbs sampling inference of the model parameters, which improves the computational complexity compared to the state-of-the-art methods. This is achieved by using a statistically well-grounded data augmentation solution. In addition, GMNB explicitly models the potential sequencing depth heterogeneity so that no heuristic preprocessing step is required. Experimental results on both synthetic and real-world

RNA-seq data demonstrate the state-of-the-art performance of the GMNB method for temporal differential expression analysis of RNA-seq data.

In Chapter 4, we have introduced VGRNN and SI-VGRNN, the first node embedding methods for dynamic graphs that capture both longitudinal and structural heterogeneity. VGRNN embeds each node of dynamic graphs to a random vector in the latent space. We argue that adding high level latent variables to graph recurrent neural networks not only increases its expressiveness to better model the complex dynamics of graphs, but also generates interpretable random latent representation for nodes. SI-VGRNN is also developed by combining VGRNN and semi-implicit variational inference for flexible variational inference. We have tested our proposed methods on dynamic link prediction tasks and they outperform competing methods substantially, specially for very sparse graphs.

Finally, we have developed BayReL, a novel Bayesian relational representation learning method that infers interactions across multi-omics data types. BayReL takes advantage of *a priori* known relationships among the same class of molecules, modeled as a graph at each corresponding view. By learning view-specific latent variables as well as a multi-partite graph, more accurate and robust interaction identification across views can be achieved. We have tested BayReL on three different real-world omics datasets, which demonstrates that not only BayReL captures meaningful inter-relations across views, but also it substantially outperforms competing methods in terms of prediction sensitivity, robustness, and consistency.

As a summary, throughout this dissertation, we have tackled Bayesian learning with data having different kinds of heterogeneity and their combinations, especially those that are common in the life science applications. We have proposed novel Bayesian learning models for task and longitudinal heterogeneity in over-dispersed RNA-seq count data. Structural heterogeneity when considering interdependency relationships, for example from prior knowledge, has been considered when analyzing dynamic graph structured data by VGRNN. Finally, we have proposed BayReL, as a unified Bayesian representation learning method, to deal with view and structural heterogeneity. There are multiple remaining problems that can be further explored in future

based on this dissertation. Multi-level factor analysis based on the proposed Bayes factor analysis might lead to inefficient analysis if more complex experiment design is needed. We can further extend GMNB to incorporate other confounding covariates to achieve more accurate genetic marker identification. Context and longitudinal heterogeneity can also be combined with view heterogeneity. The remaining question is that how BayReL should be modified to handle these problems in different biomedical scenario.

REFERENCES

[1] T. Äijö, V. Butty, Z. Chen, V. Salo, S. Tripathi, C. B. Burge, R. Lahesmaa, and H. Lähdesmäki, "Methods for time series analysis of RNA-Seq data with application to human Th17 cell differentiation," *Bioinformatics*, vol. 30, no. 12, pp. i113–i120, 2014.

[2] E. Chu and P. Liu, "Meansum: a neural model for unsupervised multi-document abstractive summarization," in *International Conference on Machine Learning*, pp. 1223–1232, 2019.

[3] Z. Xu, Z. Liu, C. Sun, K. Murphy, W. T. Freeman, J. B. Tenenbaum, and J. Wu, "Unsupervised discovery of parts, structure, and dynamics," *arXiv preprint arXiv:1903.05136*, 2019.

[4] The Cancer Genome Atlas Research Network *et al.*, "Comprehensive genomic characterization defines human glioblastoma genes and core pathways," *Nature*, vol. 455, no. 7216, p. 1061, 2008.

[5] S. J. Pan and Q. Yang, "A survey on transfer learning," *IEEE Transactions on knowledge and data engineering*, vol. 22, no. 10, pp. 1345–1359, 2010.

[6] V. M. Patel, R. Gopalan, R. Li, and R. Chellappa, "Visual domain adaptation: A survey of recent advances," *IEEE signal processing magazine*, vol. 32, no. 3, pp. 53–69, 2015.

[7] S. Z. Dadaneh, X. Qian, and M. Zhou, "BNP-Seq: Bayesian nonparametric differential expression analysis of sequencing count data," *Journal of the American Statistical Association*, vol. 113, no. 521, pp. 81–94, 2018.

[8] E. Hajiramezanali, S. Z. Dadaneh, P. de Figueiredo, S.-H. Sze, M. Zhou, and X. Qian, "Differential expression analysis of dynamical sequencing count data with a gamma markov chain," *arXiv preprint arXiv:1803.02527*, 2018.

[9] Z. Ghahramani and T. L. Griffiths, "Infinite latent feature models and the Indian buffet process," in *Advances in neural information processing systems*, pp. 475–482, 2006.

[10] M. Zhou and L. Carin, "Negative binomial process count and mixture modeling," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 37, no. 2, pp. 307–320, 2015.

[11] Y. Xue, X. Liao, L. Carin, and B. Krishnapuram, "Multi-task learning for classification with Dirichlet process priors," *Journal of Machine Learning Research*, vol. 8, no. Jan, pp. 35–63, 2007.

[12] L. Jacob, J.-P. Vert, and F. R. Bach, "Clustered multi-task learning: A convex formulation," in *Advances in neural information processing systems*, pp. 745–752, 2009.

[13] Z. Kang, K. Grauman, and F. Sha, "Learning with whom to share in multi-task feature learning.," in *ICML*, pp. 521–528, 2011.

[14] A. Argyriou, T. Evgeniou, and M. Pontil, "Multi-task feature learning," in *Advances in neural information processing systems*, pp. 41–48, 2007.

[15] P. Rai and H. Daume III, "Infinite predictor subspace models for multitask learning," in *Proceedings of the Thirteenth International Conference on Artificial Intelligence and Statistics*, pp. 613–620, 2010.

[16] C. Chelba and A. Acero, "Adaptation of maximum entropy capitalizer: Little data can help a lot," *Computer Speech & Language*, vol. 20, no. 4, pp. 382–399, 2006.

[17] A. Passos, P. Rai, J. Wainer, and H. Daume III, "Flexible modeling of latent task structures in multitask learning," *arXiv preprint arXiv:1206.6486*, 2012.

[18] A. Kumar and H. Daume III, "Learning task grouping and overlap in multi-task learning," *arXiv preprint arXiv:1206.6417*, 2012.

[19] Y. W. Teh, M. I. Jordan, M. J. Beal, and D. M. Blei, "Sharing clusters among related groups: Hierarchical Dirichlet processes," in *Advances in neural information processing systems*, pp. 1385–1392, 2005.

[20] M. Zhou, "Nonparametric Bayesian negative binomial factor analysis," *Bayesian Analysis*, pp. 1061–1089, 2018.

[21] R. Thibaux and M. I. Jordan, "Hierarchical beta processes and the Indian buffet process," in *Artificial Intelligence and Statistics*, pp. 564–571, 2007.

[22] M. Zhou, H. Chen, L. Ren, G. Sapiro, L. Carin, and J. W. Paisley, "Non-parametric Bayesian dictionary learning for sparse image representations," in *Advances in neural information processing systems*, pp. 2295–2303, 2009.

[23] L. Le Cam, "An approximation theorem for the Poisson binomial distribution," *Pacific Journal of Mathematics*, vol. 10, no. 4, pp. 1181–1197, 1960.

[24] M. Zhou and L. Carin, "Augment-and-conquer negative binomial processes," in *Advances in Neural Information Processing Systems*, pp. 2546–2554, 2012.

[25] B. Schölkopf and A. J. Smola, *Learning with kernels: Support vector machines, regularization, optimization, and beyond*. MIT press, 2002.

[26] D. Pardoe and P. Stone, "Boosting for regression transfer," in *Proceedings of the 27th International Conference on International Conference on Machine Learning*, pp. 863–870, Omnipress, 2010.

[27] A. Karbalayghareh, X. Qian, and E. R. Dougherty, "Optimal Bayesian Transfer Learning," *IEEE Transactions on Signal Processing*, 2018.

[28] M. I. Love, W. Huber, and S. Anders, "Moderated estimation of fold change and dispersion for RNA-Seq data with DESeq2," *Genome biology*, vol. 15, no. 12, p. 550, 2014.

[29] R.-E. Fan, K.-W. Chang, C.-J. Hsieh, X.-R. Wang, and C.-J. Lin, "Liblinear: A library for large linear classification," *Journal of machine learning research*, vol. 9, no. Aug, pp. 1871–1874, 2008.

[30] S. Williamson, C. Wang, K. A. Heller, and D. M. Blei, "The IBP compound Dirichlet process and its application to focused topic modeling," in *ICML*, 2010.

[31] S. Anders, P. T. Pyl, and W. Huber, "HTSeq–a Python framework to work with high-throughput sequencing data," *Bioinformatics*, vol. 31, no. 2, pp. 166–169, 2015.

[32] J. Lienau, K. Schmidt-Bleek, A. Peters, H. Weber, H. J. Bail, G. N. Duda, C. Perka, and H. Schell, "Insight into the molecular pathophysiology of delayed bone healing in a sheep model," *Tissue Engineering Part A*, vol. 16, no. 1, pp. 191–199, 2009.

[33] D. S. Fischer, F. J. Theis, and N. Yosef, "Impulse model-based differential expression analysis of time course sequencing data," *Nucleic acids research*, vol. 46, no. 20, pp. e119–e119, 2018.

[34] N. Leng, Y. Li, B. E. McIntosh, B. K. Nguyen, B. Duffin, S. Tian, J. A. Thomson, C. N. Dewey, R. Stewart, and C. Kendziorski, "Ebseq-hmm: a Bayesian approach for identifying gene-expression changes in ordered RNA-Seq experiments," *Bioinformatics*, vol. 31, no. 16, pp. 2614–2622, 2015.

[35] S. Oh, S. Song, G. Grabowski, H. Zhao, and J. P. Noonan, "Time series expression analyses using RNA-Seq: a statistical approach," *BioMed research international*, vol. 2013, 2013.

[36] A. Michna, H. Braselmann, M. Selmansberger, A. Dietz, J. Hess, M. Gomolka, S. Hornhardt, N. Bluethgen, H. Zitzelsberger, and K. Unger, "Natural cubic spline regression modeling followed by dynamic network reconstruction for the identification of radiation-sensitivity gene association networks from time-course transcriptome data," *PloS one*, vol. 11, no. 8, p. e0160791, 2016.

[37] D. Spies, P. F. Renz, T. A. Beyer, and C. Ciaudo, "Comparative analysis of differential gene expression tools for RNA sequencing time course data," *Briefings in bioinformatics*, 2017.

[38] D. Spies and C. Ciaudo, "Dynamics in transcriptomics: advancements in RNA-Seq time course and downstream analysis," *Computational and structural biotechnology journal*, vol. 13, pp. 469–477, 2015.

[39] X. Sun, D. Dalpiaz, D. Wu, J. S. Liu, W. Zhong, and P. Ma, "Statistical inference for time course RNA-Seq data using a negative binomial mixed-effect model," *BMC bioinformatics*, vol. 17, no. 1, p. 324, 2016.

[40] C. E. Rasmussen and C. K. Williams, *Gaussian processes for machine learning*, vol. 1. MIT press Cambridge, 2006.

[41] Y. Nguyen and D. Nettleton, "rmRNAseq: Differential expression analysis for repeated-measures RNA-seq data," *Bioinformatics*, 2020.

[42] C. W. Law, Y. Chen, W. Shi, and G. K. Smyth, "Voom: precision weights unlock linear model analysis tools for RNA-Seq read counts," *Genome biology*, vol. 15, no. 2, p. R29, 2014.

[43] G. Chechik and D. Koller, "Timing of gene expression responses to environmental changes," *Journal of Computational Biology*, vol. 16, no. 2, pp. 279–290, 2009.

[44] A. Acharya, J. Ghosh, and M. Zhou, "Nonparametric Bayesian Factor Analysis for Dynamic Count Matrices," in *Proceedings of the Eighteenth International Conference on Artificial Intelligence and Statistics*, vol. 38, (California, USA), pp. 1–9, 09–12 May 2015.

[45] A. Schein, H. Wallach, and M. Zhou, "Poisson-gamma dynamical systems," in *Advances in Neural Information Processing Systems*, pp. 5005–5013, 2016.

[46] E. Hajiramezanali, S. Z. Dadaneh, A. Karbalayghareh, M. Zhou, and X. Qian, "Bayesian multi-domain learning for cancer subtype discovery from next-generation sequencing count data," in *Advances in Neural Information Processing Systems*, pp. 9115–9124, 2018.

[47] S. Z. Dadaneh, X. Qian, and M. Zhou, "BNP-seq: Bayesian nonparametric differential expression analysis of sequencing count data," *Journal of the American Statistical Association*, vol. 113, no. 521, pp. 81–94, 2018.

[48] N. L. Johnson, A. W. Kemp, and S. Kotz, *Univariate discrete distributions*, vol. 444. John Wiley & Sons, 2005.

[49] S. Tuomela, S. Rautio, H. Ahlfors, V. Öling, V. Salo, U. Ullah, Z. Chen, S. Hämälistö, S. K. Tripathi, T. Äijö, *et al.*, "Comparative analysis of human and mouse transcriptomes of Th17 cell priming," *Oncotarget*, vol. 7, no. 12, p. 13416, 2016.

[50] Y. H. Chan, J. Intosalmi, S. Rautio, and H. Lähdesmäki, "A subpopulation model to analyze heterogeneous cell differentiation dynamics," *Bioinformatics*, vol. 32, no. 21, pp. 3306–3313, 2016.

[51] S. Tuomela, V. Salo, S. K. Tripathi, Z. Chen, K. Laurila, B. Gupta, T. Äijö, L. Oikari, B. Stockinger, H. Lähdesmäki, *et al.*, "Identification of early gene expression changes during human Th17 cell differentiation," *Blood*, vol. 119, no. 23, pp. e151–e160, 2012.

[52] K. Yang, J. L. Vega, M. Hadzipasic, J. P. S. Peron, B. Zhu, Y. Carrier, S. Masli, L. V. Rizzo, and H. L. Weiner, "Deficiency of thrombospondin-1 reduces Th17 differentiation and attenuates experimental autoimmune encephalomyelitis," *Journal of autoimmunity*, vol. 32, no. 2, pp. 94–103, 2009.

[53] E. Y. Chiang, G. A. Kolumam, X. Yu, M. Francesco, S. Ivelja, I. Peng, P. Gribling, J. Shu, W. P. Lee, C. J. Refino, *et al.*, "Targeted depletion of lymphotoxin-α–expressing TH1 and Th17 cells inhibits autoimmune disease," *Nature medicine*, vol. 15, no. 7, pp. 766–773, 2009.

[54] M. Pasarica, B. Gowronska-Kozak, D. Burk, I. Remedios, D. Hymel, J. Gimble, E. Ravussin, G. A. Bray, and S. R. Smith, "Adipose tissue collagen VI in obesity," *The Journal of Clinical Endocrinology & Metabolism*, vol. 94, no. 12, pp. 5155–5162, 2009.

[55] S. K. Tripathi, Z. Chen, A. Larjo, K. Kanduri, K. Nousiainen, T. Äijo, I. Ricaño-Ponce, B. Hrdlickova, S. Tuomela, E. Laajala, *et al.*, "Genome-wide analysis of STAT3-mediated transcription during early human Th17 cell differentiation," *Cell Reports*, vol. 19, no. 9, pp. 1888–1901, 2017.

[56] J. R. Conway, A. Lex, and N. Gehlenborg, "Upsetr: an r package for the visualization of intersecting sets and their properties," *Bioinformatics*, vol. 33, no. 18, pp. 2938–2940, 2017.

[57] Y. Kurebayashi, S. Nagai, A. Ikejiri, M. Ohtani, K. Ichiyama, Y. Baba, T. Yamada, S. Egami, T. Hoshii, A. Hirao, *et al.*, "PI3K-Akt-mTORC1-S6K1/2 axis controls Th17 differentiation by regulating Gfi1 expression and nuclear translocation of RORγ," *Cell reports*, vol. 1, no. 4, pp. 360–373, 2012.

[58] J. Gnanaprakasam and R. Wang, "MYC in regulating immunity: metabolism and beyond," *Genes*, vol. 8, no. 3, p. 88, 2017.

[59] J. C. Waite and D. Skokos, "Th17 response and inflammatory autoimmune diseases," *International journal of inflammation*, vol. 2012, 2011.

[60] A. Subramanian, P. Tamayo, V. K. Mootha, S. Mukherjee, B. L. Ebert, M. A. Gillette, A. Paulovich, S. L. Pomeroy, T. R. Golub, E. S. Lander, *et al.*, "Gene set enrichment analysis: a knowledge-based approach for interpreting genome-wide expression profiles," *Proceedings of the National Academy of Sciences*, vol. 102, no. 43, pp. 15545–15550, 2005.

[61] N. Yosef, A. K. Shalek, J. T. Gaublomme, H. Jin, Y. Lee, A. Awasthi, C. Wu, K. Karwacz, S. Xiao, M. Jorgolli, *et al.*, "Dynamic regulatory network controlling Th17 cell differentiation," *Nature*, vol. 496, no. 7446, p. 461, 2013.

[62] T. Korn, E. Bettelli, M. Oukka, and V. K. Kuchroo, "IL-17 and Th17 cells," *Annual review of immunology*, vol. 27, pp. 485–517, 2009.

[63] V. Brucklacher-Waldert, K. Steinbach, M. Lioznov, M. Kolster, C. Hölscher, and E. Tolosa, "Phenotypical characterization of human Th17 cells unambiguously identified by surface IL-17A expression," *The Journal of Immunology*, vol. 183, no. 9, pp. 5494–5501, 2009.

[64] Y. Chung, S. H. Chang, G. J. Martinez, X. O. Yang, R. Nurieva, H. S. Kang, L. Ma, S. S. Watowich, A. M. Jetten, Q. Tian, *et al.*, "Critical regulation of early Th17 cell differentiation by interleukin-1 signaling," *Immunity*, vol. 30, no. 4, pp. 576–587, 2009.

[65] S. Z. Dadaneh, M. Zhou, and X. Qian, "Bayesian negative binomial regression for differential expression with confounding factors," *Bioinformatics*, vol. 34, no. 19, pp. 3349–3356, 2018.

[66] C. Donnat, M. Zitnik, D. Hallac, and J. Leskovec, "Learning structural node embeddings via diffusion wavelets," in *International ACM Conference on Knowledge Discovery and Data Mining (KDD)*, vol. 24, 2018.

[67] E. Hajiramezanali, A. Hasanzadeh, N. Duffield, K. Narayanan, and X. Qian, "BayReL: Bayesian relational learning for multi-omics data integration," *Advances in Neural Information Processing Systems*, vol. 33, 2020.

[68] B. Perozzi, R. Al-Rfou, and S. Skiena, "Deepwalk: Online learning of social representations," in *Proceedings of the 20th ACM SIGKDD international conference on Knowledge discovery and data mining*, pp. 701–710, ACM, 2014.

[69] J. Tang, M. Qu, M. Wang, M. Zhang, J. Yan, and Q. Mei, "Line: Large-scale information network embedding," in *Proceedings of the 24th International Conference on World Wide Web*, pp. 1067–1077, International World Wide Web Conferences Steering Committee, 2015.

[70] A. Grover and J. Leskovec, "node2vec: Scalable feature learning for networks," in *Proceedings of the 22nd ACM SIGKDD international conference on Knowledge discovery and data mining*, pp. 855–864, ACM, 2016.

[71] L. F. Ribeiro, P. H. Saverese, and D. R. Figueiredo, "struc2vec: Learning node representations from structural identity," in *Proceedings of the 23rd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pp. 385–394, ACM, 2017.

[72] T. N. Kipf and M. Welling, "Variational graph auto-encoders," *arXiv preprint arXiv:1611.07308*, 2016.

[73] M. Armandpour, P. Ding, J. Huang, and X. Hu, "Robust negative sampling for network embedding," in *Proceedings of the AAAI Conference on Artificial Intelligence*, vol. 33, pp. 3191–3198, AAAI, 2019.

[74] P. Goyal, N. Kamra, X. He, and Y. Liu, "Dyngem: Deep embedding method for dynamic graphs," *arXiv preprint arXiv:1805.11273*, 2018.

[75] L. Zhou, Y. Yang, X. Ren, F. Wu, and Y. Zhuang, "Dynamic network embedding by modeling triadic closure process," in *Thirty-Second AAAI Conference on Artificial Intelligence*, 2018.

[76] R. Trivedi, M. Farajtabar, P. Biswal, and H. Zha, "Dyrep: Learning representations over dynamic graphs," in *International Conference on Learning Representations*, 2019.

[77] J. Li, H. Dani, X. Hu, J. Tang, Y. Chang, and H. Liu, "Attributed network embedding for learning in a dynamic environment," in *Proceedings of the 2017 ACM on Conference on Information and Knowledge Management*, pp. 387–396, ACM, 2017.

[78] A. Bojchevski and S. Günnemann, "Deep gaussian embedding of graphs: Unsupervised inductive learning via ranking," in *International Conference on Learning Representations*, 2018.

[79] Y. Seo, M. Defferrard, P. Vandergheynst, and X. Bresson, "Structured sequence modeling with graph convolutional recurrent networks," in *International Conference on Neural Information Processing*, pp. 362–373, Springer, 2018.

[80] J. Chung, K. Kastner, L. Dinh, K. Goel, A. C. Courville, and Y. Bengio, "A recurrent latent variable model for sequential data," in *Advances in neural information processing systems*, pp. 2980–2988, 2015.

[81] S. Shabanian, D. Arpit, A. Trischler, and Y. Bengio, "Variational bi-lstms," *arXiv preprint arXiv:1711.05717*, 2017.

[82] M. Fraccaro, S. r. K. Sø nderby, U. Paquet, and O. Winther, "Sequential neural models with stochastic layers," in *Advances in Neural Information Processing Systems 29* (D. D. Lee, M. Sugiyama, U. V. Luxburg, I. Guyon, and R. Garnett, eds.), pp. 2199–2207, Curran Associates, Inc., 2016.

[83] A. P. Goyal, A. Goyal, A. Sordoni, M.-A. Côté, N. R. Ke, and Y. Bengio, "Z-forcing: Training stochastic recurrent networks," in *Advances in Neural Information Processing Systems 30* (I. Guyon, U. V. Luxburg, S. Bengio, H. Wallach, R. Fergus, S. Vishwanathan, and R. Garnett, eds.), pp. 6713–6723, Curran Associates, Inc., 2017.

[84] E. Hajiramezanali, A. Hasanzadeh, N. Duffield, K. Narayanan, M. Zhou, and X. Qian, "Semi-implicit stochastic recurrent neural networks," in *ICASSP 2020-2020 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pp. 3342–3346, IEEE, 2020.

[85] M. Hajiramezanali, S. H. Fouladi, J. A. Ritcey, and H. Amindavar, "Stochastic differential equations for modeling of high maneuvering target tracking," *ETRI Journal*, vol. 35, no. 5, pp. 849–858, 2013.

[86] S. H. Fouladi, M. Hajiramezanali, H. Amindavar, J. A. Ritcey, and P. Arabshahi, "Denoising based on multivariate stochastic volatility modeling of multiwavelet coefficients," *IEEE transactions on signal processing*, vol. 61, no. 22, pp. 5578–5589, 2013.

[87] M. Hajiramezanali and H. Amindavar, "Maneuvering target tracking based on sde driven by garch volatility," in *2012 IEEE Statistical Signal Processing Workshop (SSP)*, pp. 764–767, IEEE, 2012.

[88] M. Hajiramezanali and H. Amindavar, "Maneuvering target tracking based on combined stochastic differential equations and garch process," in *2012 11th International Conference on Information Science, Signal Processing and their Applications (ISSPA)*, pp. 1293–1297, IEEE, 2012.

[89] M. Yin and M. Zhou, "Semi-implicit variational inference," in *International Conference on Machine Learning*, pp. 5660–5669, 2018.

[90] M. Defferrard, X. Bresson, and P. Vandergheynst, "Convolutional neural networks on graphs with fast localized spectral filtering," in *Advances in Neural Information Processing Systems*, pp. 3844–3852, 2016.

[91] M. Fraccaro, S. K. Sønderby, U. Paquet, and O. Winther, "Sequential neural models with stochastic layers," in *Advances in neural information processing systems*, pp. 2199–2207, 2016.

[92] A. Goyal, A. Sordoni, M.-A. Côté, N. R. Ke, and Y. Bengio, "Z-forcing: Training stochastic recurrent networks," in *Advances in neural information processing systems*, pp. 6713–6723, 2017.

[93] T. N. Kipf and M. Welling, "Semi-supervised classification with graph convolutional networks," in *International Conference on Learning Representations*, 2017.

[94] W. Hamilton, Z. Ying, and J. Leskovec, "Inductive representation learning on large graphs," in *Advances in Neural Information Processing Systems*, pp. 1024–1034, 2017.

[95] P. Goyal, S. R. Chhetri, and A. Canedo, "dyngraph2vec: Capturing network dynamics using dynamic graph representation learning," *Knowledge-Based Systems*, 2019.

[96] A. Paszke, S. Gross, S. Chintala, G. Chanan, E. Yang, Z. DeVito, Z. Lin, A. Desmaison, L. Antiga, and A. Lerer, "Automatic differentiation in pytorch," in *NIPS-W*, 2017.

[97] D. Molchanov, V. Kharitonov, A. Sobolev, and D. Vetrov, "Doubly semi-implicit variational inference," *arXiv preprint arXiv:1810.02789*, 2018.

[98] S. Huang, K. Chaudhary, and L. X. Garmire, "More is better: recent progress in multi-omics data integration methods," *Frontiers in genetics*, vol. 8, p. 84, 2017.

[99] E. Hajiramezanali, M. Imani, U. Braga-Neto, X. Qian, and E. R. Dougherty, "Scalable optimal bayesian classification of single-cell trajectories under regulatory model uncertainty," *BMC genomics*, vol. 20, no. 6, p. 435, 2019.

[100] E. Andrés-León, I. Cases, S. Alonso, and A. M. Rojas, "Novel mirna-mrna interactions conserved in essential cancer pathways," *Scientific reports*, vol. 7, p. 46101, 2017.

[101] B. Thompson, *Canonical correlation analysis: Uses and interpretation*. Sage, 1984.

[102] A. Klami, S. Virtanen, and S. Kaski, "Bayesian canonical correlation analysis," *Journal of Machine Learning Research*, vol. 14, no. Apr, pp. 965–1003, 2013.

[103] G. Andrew, R. Arora, J. Bilmes, and K. Livescu, "Deep canonical correlation analysis," in *International conference on machine learning*, pp. 1247–1255, 2013.

[104] F. R. Bach and M. I. Jordan, "A probabilistic interpretation of canonical correlation analysis," 2005.

[105] Z. Ghahramani, "Probabilistic machine learning and artificial intelligence," *Nature*, vol. 521, no. 7553, pp. 452–459, 2015.

[106] K. Xu, W. Hu, J. Leskovec, and S. Jegelka, "How powerful are graph neural networks?," in *International Conference on Learning Representations*, 2019.

[107] A. Hasanzadeh, E. Hajiramezanali, N. Duffield, K. R. Narayanan, M. Zhou, and X. Qian, "Semi-implicit graph variational auto-encoders," in *Advances in Neural Information Processing Systems*, 2019.

[108] E. Hajiramezanali, A. Hasanzadeh, N. Duffield, K. R. Narayanan, M. Zhou, and X. Qian, "Variational graph recurrent neural networks," in *Advances in Neural Information Processing Systems*, 2019.

[109] Y. Gal, J. Hron, and A. Kendall, "Concrete dropout," in *Advances in neural information processing systems*, pp. 3581–3590, 2017.

[110] A. Hasanzadeh, E. Hajiramezanali, S. Boluki, M. Zhou, N. Duffield, K. Narayanan, and X. Qian, "Bayesian graph neural networks with adaptive connection sampling," *arXiv preprint arXiv:2006.04064*, 2020.

[111] J. Chen, G. Wang, and G. B. Giannakis, "Multiview canonical correlation analysis over graphs," in *ICASSP 2019-2019 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pp. 2947–2951, IEEE, 2019.

[112] J. Chen, G. Wang, Y. Shen, and G. B. Giannakis, "Canonical correlation analysis of datasets with a common source graph," *IEEE Transactions on Signal Processing*, vol. 66, no. 16, pp. 4398–4408, 2018.

[113] B. Jiang, C. Ding, B. Luo, and J. Tang, "Graph-laplacian pca: Closed-form solution and robustness," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 3492–3498, 2013.

[114] F. Shang, L. Jiao, and F. Wang, "Graph dual regularization non-negative matrix factorization for co-clustering," *Pattern Recognition*, vol. 45, no. 6, pp. 2237–2250, 2012.

[115] S. Virtanen, A. Klami, and S. Kaski, "Bayesian cca via group sparsity," in *Proceedings of the 28th International Conference on International Conference on Machine Learning*, pp. 457–464, 2011.

[116] G. Gundersen, B. Dumitrascu, J. T. Ash, and B. E. Engelhardt, "End-to-end training of deep probabilistic cca on paired biomedical observations," in *35th Conference on Uncertainty in Artificial Intelligence, UAI 2019*, 2019.

[117] C. Zhang, D. Song, C. Huang, A. Swami, and N. V. Chawla, "Heterogeneous graph neural network," in *Proceedings of the 25th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*, pp. 793–803, 2019.

[118] R. v. d. Berg, T. N. Kipf, and M. Welling, "Graph convolutional matrix completion," *arXiv preprint arXiv:1706.02263*, 2017.

[119] F. Monti, M. Bronstein, and X. Bresson, "Geometric matrix completion with recurrent multi-graph neural networks," in *Advances in Neural Information Processing Systems*, pp. 3697–3707, 2017.

[120] V. Kalofolias, X. Bresson, M. Bronstein, and P. Vandergheynst, "Matrix completion on graphs," *arXiv preprint arXiv:1408.1717*, 2014.

[121] H. Ma, D. Zhou, C. Liu, M. R. Lyu, and I. King, "Recommender systems with social regularization," in *Proceedings of the fourth ACM international conference on Web search and data mining*, pp. 287–296, 2011.

[122] M. Abadi, A. Agarwal, P. Barham, E. Brevdo, Z. Chen, C. Citro, G. S. Corrado, A. Davis, J. Dean, M. Devin, S. Ghemawat, I. Goodfellow, A. Harp, G. Irving, M. Isard, Y. Jia, R. Jozefowicz, L. Kaiser, M. Kudlur, J. Levenberg, D. Mané, R. Monga, S. Moore, D. Murray, C. Olah, M. Schuster, J. Shlens, B. Steiner, I. Sutskever, K. Talwar, P. Tucker, V. Vanhoucke, V. Vasudevan, F. Viégas, O. Vinyals, P. Warden, M. Wattenberg, M. Wicke, Y. Yu, and

X. Zheng, "TensorFlow: Large-scale machine learning on heterogeneous systems," 2015. Software available from tensorflow.org.

[123] R. A. Quinn, K. Whiteson, Y.-W. Lim, P. Salamon, B. Bailey, S. Mienardi, S. E. Sanchez, D. Blake, D. Conrad, and F. Rohwer, "A winogradsky-based culture system shows an association between microbial fermentation and cystic fibrosis exacerbation," *The ISME journal*, vol. 9, no. 4, pp. 1024–1038, 2015.

[124] J. T. Morton, A. A. Aksenov, L. F. Nothias, J. R. Foulds, R. A. Quinn, M. H. Badri, T. L. Swenson, M. W. Van Goethem, T. R. Northen, Y. Vazquez-Baeza, *et al.*, "Learning representations of microbe–metabolite interactions," *Nature methods*, vol. 16, no. 12, pp. 1306–1314, 2019.

[125] Y. Benjamini and Y. Hochberg, "Controlling the false discovery rate: a practical and powerful approach to multiple testing," *Journal of the Royal statistical society: series B (Methodological)*, vol. 57, no. 1, pp. 289–300, 1995.

[126] F. Miragoli, S. Federici, S. Ferrari, A. Minuti, A. Rebecchi, E. Bruzzese, V. Buccigrossi, A. Guarino, and M. L. Callegari, "Impact of cystic fibrosis disease on archaea and bacteria composition of gut microbiota," *FEMS microbiology ecology*, vol. 93, no. 2, p. fiw230, 2017.

[127] A. Bevivino, G. Bacci, P. Drevinek, M. T. Nelson, L. Hoffman, and A. Mengoni, "Deciphering the ecology of cystic fibrosis bacterial communities: towards systems-level integration," *Trends in molecular medicine*, 2019.

[128] D. Burke, F. Fouhy, M. Harrison, M. C. Rea, P. D. Cotter, O. O'Sullivan, C. Stanton, C. Hill, F. Shanahan, B. J. Plant, *et al.*, "The altered gut microbiota in adults with cystic fibrosis," *BMC microbiology*, vol. 17, no. 1, p. 58, 2017.

[129] K. Tomczak, P. Czerwińska, and M. Wiznerowicz, "The cancer genome atlas (tcga): an immeasurable source of knowledge," *Contemporary oncology*, vol. 19, no. 1A, p. A68, 2015.

[130] D. J. Skok, N. Hauptman, E. Boštjančič, and N. Zidar, "The integrative knowledge base for mirna-mrna expression in colorectal cancer," *Scientific Reports*, vol. 9, no. 1, pp. 1–9, 2019.

[131] X. Meng, J. Wang, C. Yuan, X. Li, Y. Zhou, R. Hofestädt, and M. Chen, "Cancernet: a database for decoding multilevel molecular interactions across diverse cancer types," *Oncogenesis*, vol. 4, no. 12, pp. e177–e177, 2015.

[132] M. D. Robinson, D. J. McCarthy, and G. K. Smyth, "edgeR: a Bioconductor package for differential expression analysis of digital gene expression data," *Bioinformatics*, vol. 26, no. 1, pp. 139–140, 2010.

[133] A. I. Vân Anh Huynh-Thu, L. Wehenkel, and P. Geurts, "Inferring regulatory networks from expression data using tree-based methods," *PloS one*, vol. 5, no. 9, 2010.

[134] J. Li, S. Zhang, Y. Wan, Y. Zhao, J. Shi, Y. Zhou, and Q. Cui, "Misim v2. 0: a web server for inferring microrna functional similarity based on microrna-disease associations," *Nucleic acids research*, vol. 47, no. W1, pp. W536–W541, 2019.

[135] Y. Fan and J. Xia, "mirnet—functional analysis and visual exploration of mirna–target interactions in a network context," in *Computational cell biology*, pp. 215–233, Springer, 2018.

[136] W. Kong, L. He, M. Coppola, J. Guo, N. N. Esposito, D. Coppola, and J. Q. Cheng, "Microrna-155 regulates cell survival, growth, and chemosensitivity by targeting foxo3a in breast cancer," *Journal of Biological Chemistry*, vol. 285, no. 23, pp. 17869–17879, 2010.

[137] J. Shen, Q. Hu, M. Schrauder, L. Yan, D. Wang, L. Medico, Y. Guo, S. Yao, Q. Zhu, B. Liu, *et al.*, "Circulating mir-148b and mir-133a as biomarkers for breast cancer detection," *Oncotarget*, vol. 5, no. 14, p. 5284, 2014.

[138] J. Barretina, G. Caponigro, N. Stransky, K. Venkatesan, A. A. Margolin, S. Kim, C. J. Wilson, J. Lehár, G. V. Kryukov, D. Sonkin, *et al.*, "The cancer cell line encyclopedia enables predictive modelling of anticancer drug sensitivity," *Nature*, vol. 483, no. 7391, pp. 603–607, 2012.

[139] S.-I. Lee, S. Celik, B. A. Logsdon, S. M. Lundberg, T. J. Martins, V. G. Oehler, E. H. Estey, C. P. Miller, S. Chien, J. Dai, *et al.*, "A machine learning approach to integrate big data

for precision medicine in acute myeloid leukemia," *Nature communications*, vol. 9, no. 1, pp. 1–13, 2018.

[140] A. H. Wagner, A. C. Coffman, B. J. Ainscough, N. C. Spies, Z. L. Skidmore, K. M. Campbell, K. Krysiak, D. Pan, J. F. McMichael, J. M. Eldred, *et al.*, "Dgidb 2.0: mining clinically relevant drug–gene interactions," *Nucleic acids research*, vol. 44, no. D1, pp. D1036–D1044, 2016.

[141] T. J. Hardcastle and K. A. Kelly, "baySeq: empirical Bayesian methods for identifying differential expression in sequence count data," *BMC bioinformatics*, vol. 11, no. 1, p. 422, 2010.

[142] J. Reiser, B. Adair, and T. Reinheckel, "Specialized roles for cysteine cathepsins in health and disease," *The Journal of clinical investigation*, vol. 120, no. 10, p. 3421, 2010.

[143] M. Pesu, W. T. Watford, L. Wei, L. Xu, I. Fuss, W. Strober, J. Andersson, E. Shevach, M. Quezado, A. Roebroek, *et al.*, "T cell-expressed proprotein convertase furin is essential for maintenance of peripheral tolerance," *Nature*, vol. 455, no. 7210, p. 246, 2008.

[144] J. M. González-Granado, C. Silvestre-Roig, V. Rocha-Perugini, L. Trigueros-Motos, D. Cibrián, G. Morlino, M. Blanco-Berrocal, F. G. Osorio, J. M. P. Freije, C. López-Otín, *et al.*, "Nuclear envelope lamin-A couples actin dynamics with immunological synapse architecture and T cell activation," *Science signaling*, vol. 7, no. 322, p. ra37, 2014.

[145] K. Hayashi and A. Altman, "Filamin A is required for T cell activation mediated by protein kinase C-θ," *The Journal of Immunology*, vol. 177, no. 3, pp. 1721–1728, 2006.

[146] S. Zhao, W.-P. Fung-Leung, A. Bittner, K. Ngo, and X. Liu, "Comparison of RNA-Seq and microarray in transcriptome profiling of activated T cells," *PloS one*, vol. 9, no. 1, p. e78644, 2014.

[147] S. Keerthivasan, R. Suleiman, R. Lawlor, J. Roderick, T. Bates, L. Minter, J. Anguita, I. Juncadella, B. J. Nickoloff, I. C. Le Poole, *et al.*, "Notch signaling regulates mouse and human Th17 differentiation," *The Journal of Immunology*, vol. 187, no. 2, pp. 692–701, 2011.

[148] Y. Doi, S. Oki, T. Ozawa, H. Hohjoh, S. Miyake, and T. Yamamura, "Orphan nuclear receptor NR4A2 expressed in T cells from multiple sclerosis mediates production of inflammatory cytokines," *Proceedings of the National Academy of Sciences*, vol. 105, no. 24, pp. 8381–8386, 2008.

[149] M. S. Fassett, W. Jiang, A. M. D'Alise, D. Mathis, and C. Benoist, "Nuclear receptor Nr4a1 modulates both regulatory T-cell (Treg) differentiation and clonal deletion," *Proceedings of the National Academy of Sciences*, vol. 109, no. 10, pp. 3891–3896, 2012.

[150] S. E. Corcoran and L. A. O'Neill, "HIF1α and metabolic reprogramming in inflammation," *The Journal of clinical investigation*, vol. 126, no. 10, pp. 3699–3707, 2016.

[151] M. Zhang, Y. Wang, J.-S. Wang, J. Liu, M.-M. Liu, and H.-B. Yang, "The roles of Egr-2 in autoimmune diseases," *Inflammation*, vol. 38, no. 3, pp. 972–977, 2015.

[152] Y.-K. Shih and S. Parthasarathy, "Identifying functional modules in interaction networks through overlapping markov clustering," *Bioinformatics*, vol. 28, no. 18, pp. i473–i479, 2012.

[153] S. Anders and W. Huber, "Differential expression analysis for sequence count data," *Genome biology*, vol. 11, no. 10, p. R106, 2010.

[154] L. Zhu, D. Guo, J. Yin, G. Ver Steeg, and A. Galstyan, "Scalable temporal latent space inference for link prediction in dynamic social networks," *IEEE Transactions on Knowledge and Data Engineering*, vol. 28, no. 10, pp. 2765–2777, 2016.

[155] Z. Zhang, P. Cui, J. Pei, X. Wang, and W. Zhu, "Timers: Error-bounded svd restart on dynamic networks," in *Thirty-Second AAAI Conference on Artificial Intelligence*, 2018.

[156] W. Yu, W. Cheng, C. C. Aggarwal, K. Zhang, H. Chen, and W. Wang, "Netwalk: A flexible deep embedding approach for anomaly detection in dynamic networks," in *Proceedings of the 24th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*, pp. 2672–2681, ACM, 2018.

[157] G. H. Nguyen, J. B. Lee, R. A. Rossi, N. K. Ahmed, E. Koh, and S. Kim, "Continuous-time dynamic network embeddings," in *Companion Proceedings of the The Web Conference 2018*, pp. 969–976, International World Wide Web Conferences Steering Committee, 2018.

[158] A. Sankar, Y. Wu, L. Gou, W. Zhang, and H. Yang, "Dynamic graph representation learning via self-attention networks," 2019.

[159] L. Zhou, Y. Yang, X. Ren, F. Wu, and Y. Zhuang, "Dynamic network embedding by modeling triadic closure process," in *Thirty-Second AAAI Conference on Artificial Intelligence*, 2018.

[160] R. Trivedi, H. Dai, Y. Wang, and L. Song, "Know-evolve: Deep temporal reasoning for dynamic knowledge graphs," in *International Conference on Machine Learning*, pp. 3462–3471, 2017.

[161] Z. Yang, W. W. Cohen, and R. Salakhutdinov, "Revisiting semi-supervised learning with graph embeddings," *arXiv preprint arXiv:1603.08861*, 2016.

[162] P. Sarkar, S. M. Siddiqi, and G. J. Gordon, "A latent space approach to dynamic embedding of co-occurrence data," in *Artificial Intelligence and Statistics*, pp. 420–427, 2007.

[163] C. E. Priebe, J. M. Conroy, D. J. Marchette, and Y. Park, "Scan statistics on enron graphs," *Computational & Mathematical Organization Theory*, vol. 11, no. 3, pp. 229–247, 2005.

[164] K. S. Xu and A. O. Hero, "Dynamic stochastic blockmodels for time-evolving social networks," *IEEE Journal of Selected Topics in Signal Processing*, vol. 8, no. 4, pp. 552–562, 2014.

[165] M. Rahman and M. Al Hasan, "Link prediction in dynamic networks using graphlet," in *Joint European Conference on Machine Learning and Knowledge Discovery in Databases*, pp. 394–409, Springer, 2016.

[166] B. Viswanath, A. Mislove, M. Cha, and K. P. Gummadi, "On the evolution of user interaction in facebook," in *Proceedings of the 2nd ACM workshop on Online social networks*, pp. 37–42, ACM, 2009.

[167] J. Gehrke, P. Ginsparg, and J. Kleinberg, "Overview of the 2003 kdd cup," *ACM SIGKDD Explorations Newsletter*, vol. 5, no. 2, pp. 149–151, 2003.

[168] J. Leskovec and A. Krevl, "SNAP Datasets: Stanford large network dataset collection." `http://snap.stanford.edu/data`, June 2014.

[169] P. Sen, G. Namata, M. Bilgic, L. Getoor, B. Galligher, and T. Eliassi-Rad, "Collective classification in network data," *AI magazine*, vol. 29, no. 3, p. 93, 2008.

[170] X. Liu, P. Hsieh, N. Duffield, R. Chen, M. Xie, and X. Wen, "Streaming network embedding through local actions," *CoRR*, vol. abs/1811.05932, 2018.

[171] R. Liersch, J. Gerss, C. Schliemann, M. Bayer, C. Schwöppe, C. Biermann, I. Appelmann, T. Kessler, B. Löwenberg, T. Büchner, *et al.*, "Osteopontin is a prognostic factor for survival of acute myeloid leukemia patients," *Blood, The Journal of the American Society of Hematology*, vol. 119, no. 22, pp. 5215–5220, 2012.

[172] E. B. Bächli, D. J. Schaer, R. B. Walter, J. Fehr, and G. Schoedon, "Functional expression of the cd163 scavenger receptor on acute myeloid leukemia cells of monocytic lineage," *Journal of leukocyte biology*, vol. 79, no. 2, pp. 312–318, 2006.

[173] M. J. Christopher, A. A. Petti, M. P. Rettig, C. A. Miller, E. Chendamarai, E. J. Duncavage, J. M. Klco, N. M. Helton, M. O'Laughlin, C. C. Fronick, *et al.*, "Immune escape of relapsed aml cells after allogeneic transplantation," *New England Journal of Medicine*, vol. 379, no. 24, pp. 2330–2341, 2018.

[174] Y.-Y. Chen, W.-A. Chang, E.-S. Lin, Y.-J. Chen, and P.-L. Kuo, "Expressions of hla class ii genes in cutaneous melanoma were associated with clinical outcome: Bioinformatics approaches and systematic analysis of public microarray and rna-seq datasets," *Diagnostics*, vol. 9, no. 2, p. 59, 2019.

APPENDIX A

GIBBS SAMPLING INFERENCE FOR BMDL

We provide the detailed Gibbs sampling procedure by exploiting the augmentation techniques for negative binomial (NB) factor analysis in [10].

Sampling ϕ_{vk} and $\theta_{kj}^{(d)}$: The NB random variable $n \sim \text{NB}(r, p)$ can be generated from a compound Poisson distribution:

$$n = \sum_{t=1}^{\ell} u_t, \quad u_t \sim \text{Log}(p), \quad \ell \sim \text{Pois}(-r \ln(1-p)),$$

where $u \sim \text{Log}(p)$ corresponds to the logarithmic random variable [48], with the probability mass function (pmf) $f_U(u) = -\frac{p^u}{u \ln(1-p)}$, $u = 1, 2, ...$ As shown in [10], given n and r, the distribution of ℓ is a Chinese Restaurant Table (CRT) distribution: $(\ell | n, r) \sim \text{CRT}(n, r)$, a random variable from which can be generated as $\ell = \sum_{t=1}^{n} b_t$, with $b_t \sim \text{Bernoulli}(\frac{r}{r+t-1})$.

Utilizing the above data augmentation technique, for each observed count $n_{vj}^{(d)}$, a latent count is sampled as

$$(\ell_{vj}^{(d)} | -) \sim \text{CRT}\left(n_{vj}^{(d)}, \sum_{k=1}^{K} \phi_{vk} \theta_{kj}^{(d)}\right). \tag{A.1}$$

These counts can further split into latent sub-counts [20] using a multinomial distribution:

$$(\ell_{vj1}^{(d)}, \ldots, \ell_{vjK}^{(d)} | -) \sim \text{Mult}\left(\ell_{vj}^{(d)}; \frac{\phi_{v1}\theta_{1j}^{(d)}}{\sum_{k=1}^{K} \phi_{vk}\theta_{kj}^{(d)}}, \ldots, \frac{\phi_{vK}\theta_{Kj}^{(d)}}{\sum_{k=1}^{K} \phi_{vk}\theta_{kj}^{(d)}}\right). \tag{A.2}$$

These latent counts can be generated as $\ell_{vjk}^{(d)} \sim \text{Pois}(q_j^{(d)} \phi_{vk} \theta_{kj}^{(d)})$, where $q_j^{(d)} := \ln(1 - p_j^{(d)})$. Hence, using the gamma-Poisson conjugacy, and denoting $\ell_{v.k}^{(.)} = \sum_{d=1}^{D} \sum_{j=1}^{J} \ell_{vjk}^{(d)}$ and $\ell_{.jk}^{(d)} = \sum_{v=1}^{V} \ell_{vjk}^{(d)}$, ϕ_{vk} and $\theta_{kj}^{(d)}$ are updated as

$$(\phi_{1k}, \ldots, \phi_{Vk} | -) \sim \text{Dir}(\eta + \ell_{1.k}^{(.)}, \ldots, \eta + \ell_{V.k}^{(.)}); \ \theta_{kj}^{(d)} \sim \text{Gamma}\left(r_k^{(d)} + \ell_{.jk}^{(d)}, \frac{1}{c_j^{(d)} - q_j^{(d)}}\right). \tag{A.3}$$

Approximation: Rather than sampling $\ell_{vj}^{(d)}$ using (A.1), we can approximate it as follows to further speed up the inference procedure:

$$\begin{aligned}
\mathrm{CRT}(n,r) &= \sum_{i=1}^{n} \mathrm{Bernoulli}\left(\frac{r}{i-1+r}\right) \\
&= \sum_{i=1}^{m} \mathrm{Bernoulli}\left(\frac{r}{i-1+r}\right) + \sum_{i=m+1}^{n} \mathrm{Bernoulli}\left(\frac{r}{i-1+r}\right) \\
&= \mathrm{CRT}(m,r) + \mathrm{Pois}(\lambda), \\
\lambda &= \sum_{i=1}^{n} \frac{r}{i-1+r} = r[\psi(n+r) - \psi(m+r)].
\end{aligned} \tag{A.4}$$

This approximation reduces the computational complexity for sampling all $\ell_{vjk}^{(d)}$ from $O[\sum_d \sum_v \sum_j n_{vj}^{(d)} K]$ to $O[\sum_d \sum_v \sum_j \min(n_{vj}^{(d)}, m)K]$, which can lead to significant computation saving for a large number of genes where large counts $n_{vj}^{(d)}$ are abundant.

Sampling $r_k^{(d)}$, s_k, and γ_0: Let $\tilde{p_j}^{(d)} = -q_j^{(d)}/(c_j^{(d)} - q_j^{(d)})$. Starting with $\ell_{\cdot jk}^{(d)} \sim \mathrm{Pois}(-q_j^{(d)}\theta_{kj}^{(d)})$, marginalizing out $\theta_{kj}^{(d)}$ leads to

$$\ell_{\cdot jk}^{(d)} \sim \mathrm{NB}(r_k^{(d)}, \tilde{p_j}^{(d)}). \tag{A.5}$$

Based on the CRT augmentation technique:

$$(\tilde{\ell}_{jk}^{(d)}|-) \sim \mathrm{CRT}(\ell_{\cdot jk}^{(d)}, r_k^{(d)}), \tag{A.6}$$

the Gibbs sampling update for $r_k^{(d)}$ can be written as

$$(r_k^{(d)}|-) \sim \mathrm{Gamma}\left(z_{kd}s_k + \tilde{\ell}_{\cdot k}^{(d)}, \frac{1}{c_k - \sum_j \ln(1-\tilde{p_j}^{(d)})}\right). \tag{A.7}$$

Following a similar procedure for s_k, first we draw

$$(\tilde{\tilde{\ell}}_k^{(d)}|-) \sim \mathrm{CRT}(\tilde{\ell}_{\cdot k}^{(d)}, s_k), \tag{A.8}$$

and then we update the conditional posterior of s_k as

$$(s_k|-) \sim \text{Gamma}\left(\gamma_0/K + \sum_d \tilde{\tilde{\ell}}_k^{(d)}, \frac{1}{c_0 - \tilde{q}_k}\right), \tag{A.9}$$

where $\tilde{q}_k := \sum_d z_{kd} \ln\left(1 + \sum_j \ln(1-\tilde{p_j}^{(d)})/(c_k - \sum_j \ln(1-\tilde{p_j}^{(d)}))\right)$. Similarly, we can update posterior of γ_0 as

$$(\acute{\ell}_k|-) \sim \text{CRT}(\sum_d \tilde{\tilde{\ell}}_k^{(d)}, \gamma_0/K), \;\; (\gamma_0|-) \sim \text{Gamma}\left(a_0 + \acute{\ell}_{\cdot}, \frac{1}{b_0 - \sum_k \ln(1 + \frac{\tilde{q}_k}{1-\tilde{q}_k})}\right). \tag{A.10}$$

Sampling z_{kd}: Denote $\tilde{\tilde{q}}_k^{(d)} := 1 + \sum_j \ln(1-\tilde{p_j}^{(d)})/(c_k - \sum_j \ln(1-\tilde{p_j}^{(d)}))$. Starting with $\tilde{\ell}_{\cdot k}^{(d)} \sim \text{Pois}(-z_{kd} s_k \sum_j \ln(1-\tilde{p_j}^{(d)}))$, marginalizing out s_k leads to $\tilde{\ell}_{\cdot k}^{(d)} \sim \text{NB}(z_{kd} s_k, \tilde{\tilde{q}}_k^{(d)})$. We can write

$$\begin{aligned} Pr(z_{kd}|\tilde{\ell}_{\cdot k}^{(d)} = 0) &\propto Pr(\tilde{\ell}_{\cdot k}^{(d)} = 0|z_{kd}) Pr(z_{kd}) \propto (\tilde{\tilde{q}}_k^{(d)})^{z_{kd} s_k} \pi_k^{z_{kd}} (1-\pi_k)^{1-z_{kd}} \\ &\propto ((\tilde{\tilde{q}}_k^{(d)})^{s_k} \pi_k)^{z_{kd}} (1-\pi_k)^{1-z_{kd}}, \end{aligned} \tag{A.11}$$

and thus we have $Pr(z_{kd}|\tilde{\ell}_{\cdot k}^{(d)} = 0) \sim \text{Bernoulli}\left(\frac{(\tilde{\tilde{q}}_k^{(d)})^{s_k}\pi_k}{(\tilde{\tilde{q}}_k^{(d)})^{s_k}\pi_k + (1-\pi_k)}\right)$. Therefore, we can update z_{kd} as

$$(z_{kd}|-) \sim \delta(\tilde{\ell}_{\cdot k}^{(d)} = 0)\text{Bernoulli}\left(\frac{(\tilde{\tilde{q}}_k^{(d)})^{s_k}\pi_k}{(\tilde{\tilde{q}}_k^{(d)})^{s_k}\pi_k + (1-\pi_k)}\right) + \delta(\tilde{\ell}_{\cdot k}^{(d)} > 0). \tag{A.12}$$

Sampling η: . To derive the update steps for Dirichlet hyperparameters, the likelihood for ϕ_k is

$$\mathcal{L}(\phi_k) \propto \prod_k \text{Mult}(\ell_{1\cdot k}^{(\cdot)}, \ldots, \ell_{V\cdot k}^{(\cdot)}; \ell_{\cdot\cdot k}^{(\cdot)}, \phi_k). \tag{A.13}$$

Marginalizing out ϕ_k from (A.13), the likelihood for η can be expressed as

$$\mathcal{L}(\eta) \propto \prod_k \text{DirMult}(\ell_{1\cdot k}^{(\cdot)}, \ldots, \ell_{V\cdot k}^{(\cdot)}; \ell_{\cdot\cdot k}^{(\cdot)}, \eta, \ldots, \eta). \tag{A.14}$$

where DirMult denotes the Dirichlet-Multinomial distribution [20]. The product of $\mathcal{L}(\eta)$ and

$\prod_k \mathrm{Beta}(q_k; \ell_{\cdot\cdot k}^{(\cdot)}, \eta V)$ can be expressed as

$$\mathcal{L}(\eta)\mathrm{Beta}(q_k; \ell_{\cdot\cdot k}^{(\cdot)}, \eta V) \propto \prod_k \prod_v \mathrm{NB}(\ell_{v\cdot k}^{(\cdot)}; \eta, q_k), \tag{A.15}$$

we can further apply the data augmentation technique for the NB distribution of [10] to derive the closed-form updates for η as

$$\begin{aligned} (q_k|-) &\sim \mathrm{Beta}(\ell_{\cdot\cdot k}^{(\cdot)}, \eta V), \quad u_{vk} \sim \mathrm{CRT}(\ell_{v\cdot k}^{(\cdot)}, \eta), \\ (\eta|-) &\sim \mathrm{Gamma}\left(s_0 + \sum_{k,v} u_{kv}, \frac{1}{w_0 - V\sum_k \ln(1-q_k)}\right) \end{aligned} \tag{A.16}$$

Sampling $p_j^{(d)}$: Using appropriate conditional conjugacy, we can sample the remaining parameters:

$$(p_j^{(d)}|-) \sim \mathrm{Beta}(a_0 + \sum_v n_{vj}^{(d)}, b_0 + \sum_k \theta_{jk}^{(d)}). \tag{A.17}$$

APPENDIX B

NOTES ON BAYES FACTOR (BF)

Similar as baySeq [141], GMNB computes BF for differentially gene expression analysis with more complex experimental design. To generalize it for any number of conditions, we split the counts $\mathcal{D}$ for the ith hypothesis to the intended partitioning. As an example, assume the experimental design involving samples from three different conditions. In this case, the counts are either identically distributed across all samples, or they are differently distributed in three conditions; or they are identically distributed under two conditions but not under the third one. There are thus five models that we would need to consider. We should also note, however, that in many applications we can exclude particular models based on biological knowledge.

Throughout the dissertation, GMNB looks at the whole temporal trajectory for computing BF; but it can be generalized to identify time intervals for the genes that are significantly over/under expressed in a condition. For example, after fitting the proposed GMNB model in each hypothesis, the BF is calculated for each time interval, and then the time intervals are ranked based on the calculated BF. Not only GMNB is able to identify differentially expressed genes, but also helps identify the correspondence time intervals that the genes are differentially expressed.

APPENDIX C

ADDITIONAL EXPERIMENTAL RESULTS FOR GMNB

C.1 Additional experimental results for synthetic data

In addition to the benchmark experiments with the simulated data based on the GMNB generative model discussed in the main text, we here provide more comprehensive results for the simulated data based on GMNB. More importantly, we also examined the power of GMNB to detect true differential expression of the simulation data based on two other generative models.

For each setting, following the instruction in the papers of DESeq2 [28] and [42], we generate count data for 10,000 genes across two conditions, each of which has three replicated samples. We randomly select 10% of the genes to be differentially expressed across two conditions, with the procedure described in details in the main text for the GMNB generative model and in the next subsections for two other generative models. For each specific generative model, we change the corresponding model parameters to ensure that the expected expression changes of truly differentially expressed genes are different across two conditions.

The fold change of differentially expressed genes is chosen as an adjustable parameter. To produce both up- and down-regulated differentially expressed genes, each differentially expressed gene is randomly set to be either up- or down-regulated. For each specific generative model, we change the corresponding model parameters to ensure that the expected expression changes of truly differentially expressed genes are different across two conditions. Using different human Th17 cell differentiation [51, 1] datasets to infer model parameters and different models to generate synthetic datasets allows us to assess the robustness of various methods in different practical settings.

C.1.1 False discovery rate comparison based on GMNB generative model

As in [141], false discovery rate (FDR) curves are used to highlight the performance of the method for the top ranked genes. Since there are 1000 truly differentially expressed genes, the top 1000 are selected (x-axis) and the number of false discoveries is plotted (y-axis) at each point. We

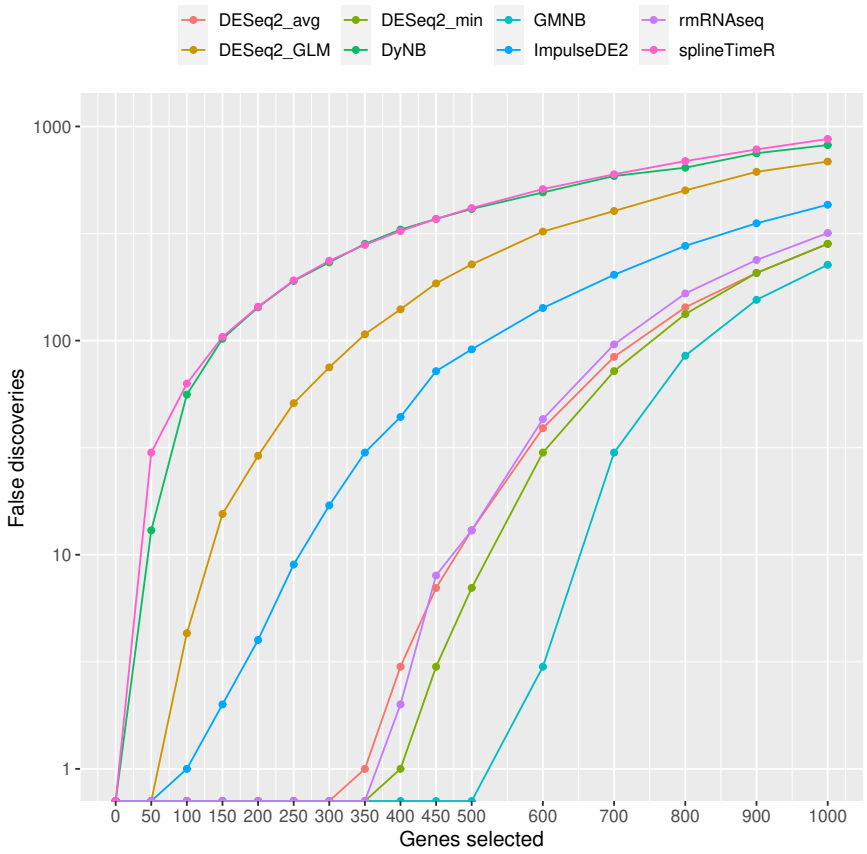


Figure C.1: False discovery plots for different methods on synthetic data generated based on the GMNB generative model. The x-axis shows the number of genes selected, in order of their detected differential expression levels, while the y-axis shows the number of selected genes that are false positives.

estimate the average false discovery rates for the top k genes over 10 simulations (Figure C.1). For the GMNB and DyNB methods, the genes are ordered by the Bayes factors; true and false positive rates are calculated on the basis of this ordering. For all the other methods, we order the genes on the basis of the computed p-values estimated by each method.

In this simulation, GMNB outperforms the existing methods with significant margins, particularly with lower numbers of selected genes. The performance of GMNB is essentially close to that of rmRNAseq, DESeq2-min, and DESeq2-mean when a higher number of genes are selected. DyNB, splineTimeR, and DESeq2-GLM always perform poorly compared with other methods. For the top

700 genes identified by GMNB, rmRNAseq, DESeq2-avg, DESeq2-min, ImpulseDE2, DESeq2-GLM, DyNB, and splineTimeR, we would expect in average 86 false positives for the GMNB, 96 from rmRNAseq, 133.4 from DESeq2-avg, 143.8 for DESeq2-min, 280 from ImpulseDE2, 509.9 for DESeq2-GLM, 650 from DyNB, and 598.7 for splineTimeR.

C.1.2 Comparison based on DyNB generative model

In the second simulation study, data is generated according to the DyNB model assumptions. To make the synthetic data closely resemble real-world RNA-seq data, we first infer the parameters of the corresponding model on the Human-activated T Cell dataset [1], and then generate synthetic sequencing counts using these inferred model parameters. More specifically, we draw the true mean values μ_k, for 1,000 genes from a Gaussian process with the mean m_k and the covariance matrix $\text{Cov}(t_i, t_j) = \theta_k \text{exp}(-\frac{1}{2\alpha_k}|t_i - t_j|)$, where m_k, θ_k, and α_k are uniformly distributed in the intervals $[1000, 2000]$, $[100, 10000]$, and $[0.5, 1]$, respectively. Similar to the real-world dataset [1], we consider five time points at 0, 12, 24, 48, and 72 hours. 10% of the genes are set to be truly differentially expressed across conditions by changing their mean values m_k and covariance function parameters $\{\theta_k, \alpha_k\}$ to $\{bm_k, c\theta_k, \alpha_k \pm d\}$, where $b = 1.5$, $c = 10$, and $d = 0.25$ determine the significance of expected expression changes across conditions. Similar to the simulation setup based on the GMNB generative model, four replicates are generated for each time point in the corresponding condition.

Figure C.2 demonstrates the performances of different methods applied to the data generated according to the above procedure. GMNB still clearly outperforms the other methods based on both the ROC and PR curves. The inferior performance of DyNB may be due to the small number of replicates for each time point, leading to poor estimates of both θ_k and μ_k, which are heuristically estimated in [1] by the data dependent value $10 \times \text{stdev}(\mathbf{Y})$ based on the observed replicates $\mathbf{Y} = \{\mathbf{y_1}, \ldots, \mathbf{y_J}\}$ and by $\frac{\min(\mathbf{Y})+\max(\mathbf{Y})}{2}$, respectively. On the contrary, the fully Bayesian nature of GMNB makes its performance robust to the number of replicates as well as potential noise at each time point. ImpulseDE2 performs worse than both GMNB and DyNB, indicating its limitation to analyze temporal data with smooth changes. In addition, DESeq2-GLM under-performs both

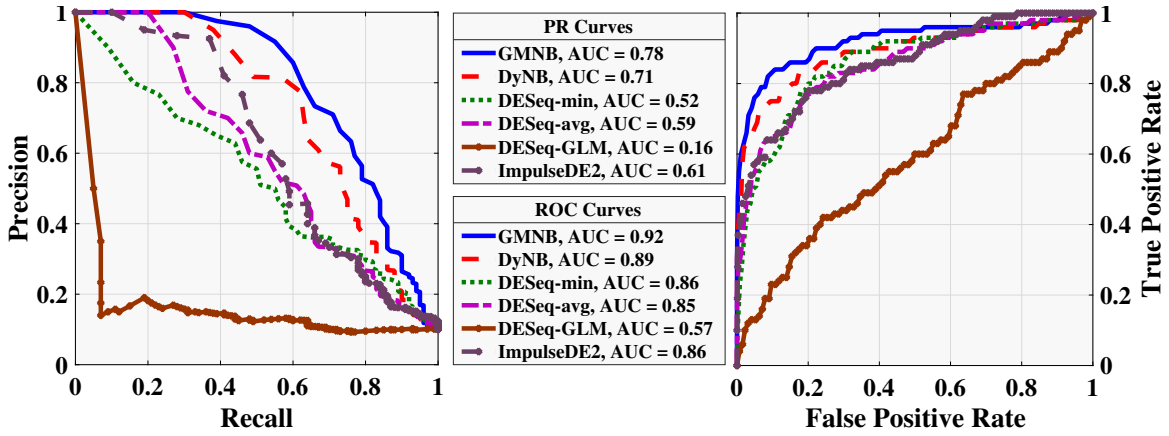


Figure C.2: Left column: PR Curves, Right column: ROC Curves. Performance comparison of different methods for differential gene expression over time based on the DyNB generative model. AUCs are given in the corresponding legends in the plots.

GMNB and DyNB substantially, as it neglects the inherent dynamics of RNA-seq experiments specifically.

C.1.3 Comparison based on NB-AR(1) generative model

In addition to synthetic data based on the GMNB and DyNB models, we evaluate these methods with the simulated data based on the NB-AR(1) model [35]. More precisely, the count for gene k at time t is distributed according to a NB distribution whose mean parameter satisfies $\log(\mu_k^{(t)}) = \omega_k^{(t)} + \beta_k$. Here we let β_k follow the uniform distribution in $[4.5, 5.5]$ to test the temporal differential expression performance with low read counts. The parameter $\omega_k^{(t)}$ is obtained through an auto-regressive process $\phi_k \omega_k^{(t-1)} + \epsilon^{(t)}$, where ϕ_k is randomly generated from the uniform distribution in $[0.1, 0.9]$, and $\epsilon^{(t)}$ denotes the Gaussian white noise. Similar to the previous simulation model, read counts are generated for 1,000 genes and 10% of them selected to be differentially expressed by changing the parameter ϕ_k to $b\phi_k$ for the second condition, where

$$b = \begin{cases} 3/2, & \text{if } \phi_k \leq 0.5, \\ 2/3, & \text{if } \phi_k > 0.5, \end{cases}$$

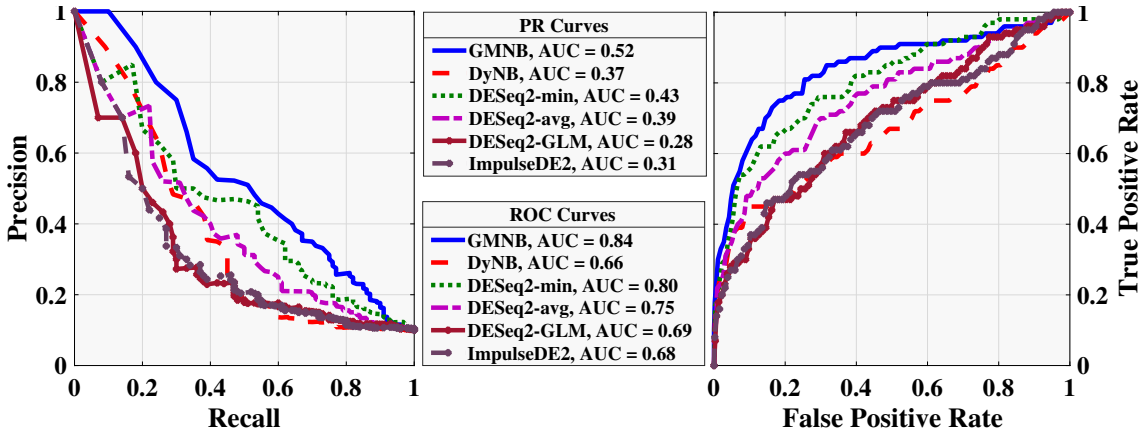


Figure C.3: Left column: PR Curves, Right column: ROC Curves. Performance comparison of different methods for differential gene expression over time based on the NB-AR(1) generative model. AUCs are given in the corresponding legends in the plots.

determines the significance of differential expression across conditions.

Figure C.3 demonstrates the performances of different methods applied to the NB-AR(1) data. GMNB again outperforms DyNB, ImpulseDE2, and DESeq2 with a remarkable margin in both the ROC and PR curves. This is due to the state-space nature of the NB-AR(1) simulation setup, in which differential expression is defined through the model parameter ϕ_k that controls the temporal dependence of gene expression. However, the temporal correlation assumption of the Gaussian process, different from this generative model, makes it less powerful to identify all differentially expressed genes. The results in Figure C.3 demonstrate the higher power of GMNB in detecting temporal differential expression patterns, especially with low expression levels (read counts are approximately 150 here).

As shown by the ROC and PR curves in both the GMNB and AR generative models, DESeq2-min outperforms both DyNB and ImpulseDE2. This indicates that the temporal correlation assumptions in DyNB may not fully capture the dynamic changes in these two state-space generative models, which can have abrupt non-smooth changes. In addition, the heuristic estimation of model parameters adopted in DyNB [1] when the number of replicates is low can be the other reason for the degraded performance. On the other hand, this shows that ImpulseDE2 cannot capture the temporal

Table C.1: Comparison of AUCs based on 20 independent runs for each method.

AUC	Model	GMNB	DyNB	ImpulseDE2	DESeq2-min
ROC	**GMNB**	**0.84 ± 0.02**	0.75 ± 0.05	0.67 ± 0.03	0.80 ± 0.07
	DyNB	**0.94 ± 0.01**	0.86 ± 0.08	0.89 ± 0.02	0.85 ± 0.03
	NB-AR(1)	**0.81 ± 0.03**	0.73 ± 0.07	0.67 ± 0.03	0.77 ± 0.07
PR	**GMNB**	**0.61 ± 0.04**	0.41 ± 0.08	0.23 ± 0.04	0.52 ± 0.06
	DyNB	**0.79 ± 0.03**	0.64 ± 0.20	0.63 ± 0.05	0.46 ± 0.06
	NB-AR(1)	**0.51 ± 0.06**	0.39 ± 0.07	0.28 ± 0.04	0.43 ± 0.10

profiles that have more than two significant changes in their expression which can be happen in these two generative models.

In summary, on synthetic RNA-seq count data from different generative models, comparison of both the ROC and PR curves shows that GMNB outperforms both the recently proposed temporal methods (DyNB and ImpulseDE2) and static differential analysis methods that aggregate differential statistics in heuristic ways (DESeq2 with different setups). Table C.1 summarizes the average AUCs and their standard deviation values of both the ROC and PR curves for 20 randomly generated synthetic datasets by the top four performing methods (GMNB, DyNB, ImpulseDE2, and DESeq2-min). GMNB improves the performances over DyNB, ImpulseDE2, and DESeq2-min, in terms of AUC-PR, at least by 12%, 16%, and 8%, respectively. In the best case scenario, GMNB improves the AUC-PR performances over DyNB, ImpulseDE2, and DESeq2-min up to 8%, 23%, and 33%, respectively. In terms of AUC-ROC, GMNB improves the best-case performances of DyNB, ImpulseDE2, and DESeq2-min by 8%, 5%, and 9%, respectively. Even with the data from the DyNB generative model, the fully Bayesian method GMNB outperforms DyNB, which estimates some of its model parameters in a heuristic manner [1]. In addition, GMNB achieves robust performance in both state-space (GMNB and NB-AR(1)) and functional (DyNB) generative models.

On the other hand, the experimental results also indicate that ImpulseDE2 cannot capture the temporal profiles that have more than two abrupt changes in their expression, which can happen in both state-space generative models (GMNB and NB-AR(1)). Regarding ImpulseDE2, the insight gained from this study is in agreement with the previous study in [37]. The performance of

ImpulseDE2 degenerates when the expression pattern change is small, with significant latency, or when the trajectory is more complicated.

Sample collection in longitudinal studies can vary significantly across studies, for example, with different missing value patterns. While the performance difference between DESeq2-min and GMNB is 3% without missing values, their performance differences can reach around 10% and 15% with only 10% and 20% missing observations, respectively. Furthermore, the temporal methods show superior performance compared to static ones in terms of AUC-ROC with missing values, which is common in such applications. While the performances of GMNB, ImpulseDE2, and DESeq2-GLM are almost 1% worse by using data with 10% missing values, the performance of static methods drops by 6%.

C.2 Additional experimental results for human Th17 cell induction case study

We discussed the biological significance of three genes out of the ten most differentially expressed genes identified by GMNB in the main text. Here, we provide more discussion about the rest of the genes. These genes have been reported as either the immune response regulators or T cell activation genes. The gene Cathepsin L (CTSL1) is ranked at the fourth in the list and is linked to the regulation of immune responses at the level of MHC complex maturation and antigen (Ag) presentation influencing differentiation of CD4+ cells and autoimmune reactions [142]. The fifth gene, FURIN, has been reported as a T cell activation gene that regulates the T helper cell balance of the immune system [143]. The sixth gene lamin A (LMNA) has been identified as one of the immune response regulators [144]. The seventh gene, Filamin A (FLNA), is required for T cell activation [145]. The eighth gene, SBNO2, has been reported to influence Th17 cell differentiation [55]. [146] observed significant changes in the expression of the ninth gene ACTB in activated T cells. Finally, the tenth gene NOTCH1 is activated in both mouse and human in vitro-polarized Th17 cells and also in Th17 polarized cells as compared with Th0 control cells [147].

In addition to the results for the gene FLNA presented in the main text, we provide more detailed experimental results for the gene ACTB that is detected by GMNB and splineTimeR with differential temporal expression but missed by DyNB, ImpulseDE2, and rmRNAseq. Figures C.4(a) and C.4(b)

show that DyNB is not able to capture the change of expression evolution of gene ACTB accurately (BF = 1.37). On the contrary, GMNB identified this gene as differentially expressed over time with BF = 3.28×10^{300} (Figures C.4(c) and C.4(d)). More precisely, Figure C.4(a) shows the posterior means of μ_k based on DyNB and their corresponding confidence intervals, where the circles and diamonds represent the normalized counts from Th0 and Th17 lineages, respectively. Figure C.4(c) shows the posterior means of r_k based on GMNB and their corresponding confidence intervals, where the circles and diamonds are obtained by dividing the observed counts by the parameter $p_j^{(t)}/(1-p_j^{(t)})$. Figures C.4(b) and C.4(d) demonstrate the means and confidence intervals of the synthetically generated counts based on the inferred parameter of DyNB and GMNB, respectively, where the circles and diamonds represent the observed raw counts from Th0 and Th17 lineages, respectively. This indicates GMNB is more powerful to reproduce the observed gene counts. Figures C.5(a) and C.5(b) show ImpulseDE2 is able to capture the change of expression evaluation of gene FLNA (p-value = 1.7×10^{-12}), but not able to capture the change of expression evaluation of gene ACTB (p-value = 1), respectively. Figures C.6(a) and C.6(b) show splineTimeR is able to capture the change of expression evaluation of genes FLNA ($p-$value = 2.2×10^{-5}) and ACTB ($p-$value = 0.008), respectively.

We also provide more detailed experimental results for five genes EGR1, NR4A1, MYC, PKM2, and EGR2, that are detected by GMNB with differential temporal expression patterns but missed by other methods. We discussed about the critical rules of genes EGR1 and MYC in the main text. Here, we provide more detailed discussion about the rest of these genes. The gene NR4A1 plays critical roles in T cell apoptosis during the thymocyte development [148]. This gene is not only a proapoptotic transcription factor, but also reported as a survival factor and activator of metabolic pathways. Both facets show the NR4A1's role in T-cell differentiation as a balancing molecule in the fate determination [149]. PKM2 promotes the function of HIF1α that is critical to drive Th17 differentiation [150]. EGR2 has been identified as an important transcription factor in the development and function of Th17 cells [151].

As shown in Figures C.7(c) and C.7(d), GMNB infers the temporal evolution of EGR1 that

fits the RNA-seq counts better. The differences of temporal profiles across Th0 and Th17 lineages can be better detected by GMNB with BF = 2.31×10^{295}. Figures C.7(a) and C.7(b) illustrate the inferred temporal expression profile for EGR1 by DyNB. Due to the abrupt changes of read counts in the early phase, DyNB is not able to correctly infer the temporal evolution of gene expression, which leads to low BF values at 1.34×10^{-04}. Similar to the plots for EGR1, Figure C.8 illustrates the similar trends for NR4A1 based on the results by DyNB and GMNB. Again GMNB detects the abrupt expression changes in the early phase as shown in Figures C.8(c) and C.8(d), while DyNB is not able to successfully capture the trend in Figures C.8(a) and C.8(b). The corresponding BF values for NR4A1 are 7.52×10^{-5} and 8.21×10^{307} by DyNB and GMNB, respectively. While the GMNB detects MYC as differentially expressed over time with BF = 2.97×10^{70} (Figures C.9(c) and C.9(d)), DyNB fails to capture the expression pattern of this gene with BF = 2.71 due to the abrupt expression changes in its early phases (Figures C.9(a) and C.9(b)). Figures C.10(a) and C.10(b) illustrate that DyNB neither captures the expression evolution of gene PKM2 in the early phases nor in the final phases accurately. On the contrary, GMNB improves the model fitting over the whole temporal process with different count levels (Figures C.10(c) and C.10(d)). The calculated BFs for the gene PKM2 are 1.08 and 3.28×10^{65} by DyNB and GMNB, respectively. Similarly, Figures C.11(a) and C.11(b) illustrate that DyNB is not able to capture the expression evolution of gene EGR2 accurately (BF = 0.02). Figures C.11(c) and C.11(d) show GMNB can identify more robust estimated expression patterns under two conditions (BF = 3.01×10^{65}). These results illustrate the benefits of GMNB to identify different temporal patterns, including abrupt changes, where DyNB faces difficulty due to its inherent smooth assumption.

Similar as GMNB, the overlap sets of the results by six approaches (GMNB, ImpulseDE2, DyNB, rmRNAseq, splineTimeR, and DESeq2-min), for the top 100 most differentially expressed genes are depicted as the UpSet diagrams in Figures C.12 and C.13.

Figure C.12 shows eight genes among the top 100 genes of the ImpulseDE2 method are not considered being differentially expressed by GMNB. RP1-187N21.2, RP11-513I15.6, CDCA2, and CEP55 are four out of eight genes with small p-values, i.e., $\log(p\text{-value}) < -20$ by ImpulseDE2.

We investigate these genes and their estimated trends by ImpulseDE2. Figure C.14 shows the temporal expression profiles of the gene RP1-187N21.2 estimated by ImpulseDE2 and GMNB, for which the replicated Th0 and Th17 profiles are seemingly similar and thus likely false positives. More precisely, ImpulseDE2 could not truly estimate the trend of the combination of two conditions of gene RP1-187N21.2. Figure C.15(c) shows the similar trend for the gene RP11-513I15.6. This is mostly due to the limitation of ImpulseDE2 to capture only one peak. While the model overfits at the time points 2, 4, and 6 hours, it obviously could not model the latter time points. Figure C.15 also shows the transition times are not truly estimated for the genes CDCA2, and CEP55. More precisely, ImpulseDE2 could not estimate the monotonically increasing expression patterns without any abrupt jump. These cases show the limitations of ImpulseDE2 in capturing the monotonically increasing expression patterns without any abrupt changes.

Figure C.13(a) shows two genes, i.e. AC097713.4 and RP11-80H8.4, out of the top 100 genes identified by rmRNAseq, which are not considered as differentially expressed by GMNB. Also among all the top 100 genes detected as differentially expressed by SplineTimeR, only the gene AP000593.5 is not identified by GMNB (Figure C.13(b)). The average expression values of AC097713.4, RP11-80H8.4, and AP000593.5 are less than 14, 15, and 20, respectively, for which the data provide little information about differential expression [28]. Figure C.21 shows the expression patterns of the genes AC097713.4, RP11-80H8.4, and AP000593.5 based on splineTimeR. These examples raise the concerns about the performance of rmRNAseq and SplineTimeR in the low count genes with low signal-to-noise ratio as already reported by [37] for SplineTimeR.

Next we present the inferred temporal patterns for four genes: LGALS1, SEPT5, COL1A2, and ENO2, which are four out of 90 differentially expressed genes detected by DyNB with BF values 2.59×10^7, 472.34, 404.34, and 398.43, respectively. These four genes are not considered differentially expressed by GMNB with their corresponding BF values lower than 10, and ImpulseDE2, rmRNAseq, and splineTimeR with their corresponding $p-$value values lower than 0.05. Figures C.16(a), C.17(a), and C.18(a) demonstrate how the inferred temporal evolution patterns of the corresponding RNA-seq counts have high variances by DyNB, which may be due

to its sensitivity to noisy data especially with low counts. These genes are likely to be potential false positives by DyNB. On the contrary, Figures C.16(c), C.17(c), and C.18(c) show that the explicit modeling of the sequencing depth in GMNB can handle the noisy RNA-seq counts at low levels when their expression patterns are similar across Th0 and Th17 lineages. In addition, Figures C.16(b&d), C.17(b&d), and C.18(b&d) show the means and confidence intervals for 1000 generated counts data based on the inferred parameters of DyNB and GMNB models. The proposed model produces narrow confidence interval compare to DyNB. These results in this case study are consistent with the trends that we observe in our three simulation studies, where we show that GMNB achieves more stable results and outperforms DyNB according to ROC and PR curves for the genes with low expression. These genes are not identified by ImpulseDE2, rmRNAseq, and splineTimeR as differentially expressed. Figures C.19 and C.20 show the expression patterns of the genes LGALS1, SEPT5, COL1A2, and ENO2 based on ImpulseDE2 and splineTimeR, respectively. Please note that rmRNAseq does neither provide a plot function nor the intermediate expression estimation of the genes to plot.

To further demonstrate the biological relevance of the detected genes by GMNB, GO enrichment analysis of top 100 differentially expressed genes (log(BF) > 100) has been performed using Fisher test. Table S2 gives the list of enriched GO terms with p-value $< 1 \times 10^{-13}$. The top enriched GO terms agree with the current biological understanding of the Th17 differentiation process with the most of the enriched GO terms in immune systems and cell development. To have a fine-resolution GO enrichment analysis, we also evaluate the results based on high-level GO terms in all three categories (biological process (BP), molecular function (MF), and cellular component (CC)). In [152] the authors defined information content (IC) of a GO term g by IC(g) $= -\log(|\text{g}|/|\text{root}|)$, where "root" is the corresponding GO category of the GO term g. Any GO term with IC > 2, is considered as a high-level GO term [152]. Table S3 gives the list of enriched high-level GO terms with p-value $< 5 \times 10^{-13}$. This clearly demonstrates top high-level GO terms agree with the current biological understanding of the Th17 cells, which is its important role in autoimmune diseases and inflammation. We further compared the top 10 GO terms of different methods in Figures C.22 and

C.23. While Figure C.22 only includes the top 10 GO terms enriched in the corresponding set of identified genes by each method, Figure C.23 shows the p-values and gene ratios from different methods for the top 10 GO terms of all models. For the GO terms that agree with the current biological understanding of the Th17 cells, GMNB demonstrates lower p-values and higher gene ratios compared to other methods.

To have an unbiased evaluation of different methods in terms of identifying Th17 related genes, we also evaluate them based on the enrichment score (ES) of 77 well-established Th17 specific gene sets in MSigDB. Table S4 shows the top gene sets and the enrichment scores of different methods. This shows an unbiased comparison of how many Th17 related genes are being captured by different models. GMNB marginally outperforms all the methods in terms of ES.

We further investigate how many Th17 related genes are among top 1000 genes identified by different methods. Out of 128 Th17 related genes reported in GSE27241, 42 genes are among top 1000 genes of GMNB, 12 genes for rmRNAseq, 13 genes for splineTimeR, 28 genes for DyNB, 11 genes for ImpulseDE2, 9 genes for DESeq2-GLM, 8 genes for DESeq2-min, and 3 genes for DESeq2-mean. Please note that while 8 genes out of 128 Th17 related genes are among top 100 genes of GMNB, none of them are among the top 100 genes identified by rmRNAseq, splineTimeR, DESeq2-min, and DESeq2-mean. This number is 2, 1, and 4 for DESeq2-GLM, ImpulseDE2, and DyNB, respectively. Again, GMNB identifies more Th17 related genes compared to other methods, demonstrating its biological significance.

C.3 Additional experimental results for human Th17 cell differentiation case study

In this dataset, CD4+ T cells were activated and polarized as described in [51], and RNA-seq data were collected at 0, 12, 24, 48 and 72 hours of both the activation (Th0) and differentiation (Th17). At each time point, there are three biological replicates for both cell lineages. The original paper [1] performed DyNB to quantify Th17-specific gene expression dynamics.

[1] first normalized the RNA-seq counts by the DESeq pipeline [153], and then applied DyNB to the normalized expression values to identify differentially expressed genes between the Th0 and Th17 lineages. Genes were considered differentially expressed if both (i) BF > 10 and (ii)

fold change > 2 for at least one time point. Out of the 698 differentially expressed genes identified by DyNB, three genes were investigated and discussed in [1] with the qRT-PCR validation: IL17A, IL17F, and RORC.

Figure C.24 shows the estimated trajectories of these genes based on GMNB, DyNB, ImpulseDE2, and splineTimeR. While, the cytokine IL17A is known to be highly expressed in Th17 cells [62], ImpulseDE2 and splineTimeR could not identify this gene as differentially expressed. The expression of IL17A is commonly used to assess the Th17 polarization efficiency [63]. Th17 cells also have been shown to be important in autoimmunity and clearance of mucosal infection by producing proinflammatory cytokines IL17F [64]. However, splineTimeR is not able to identify it as differentially expressed. This might be due to small number of available time points or small changes of the expression value.

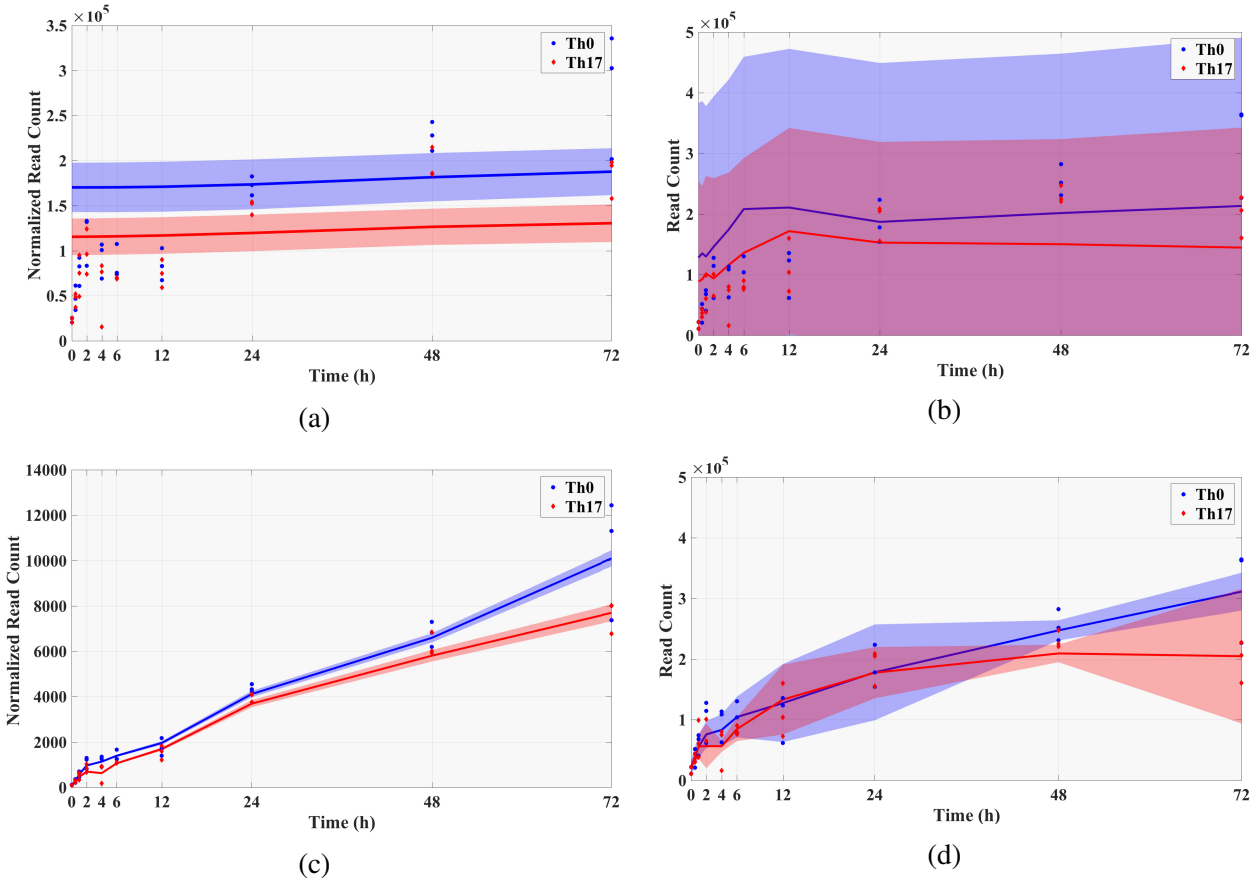
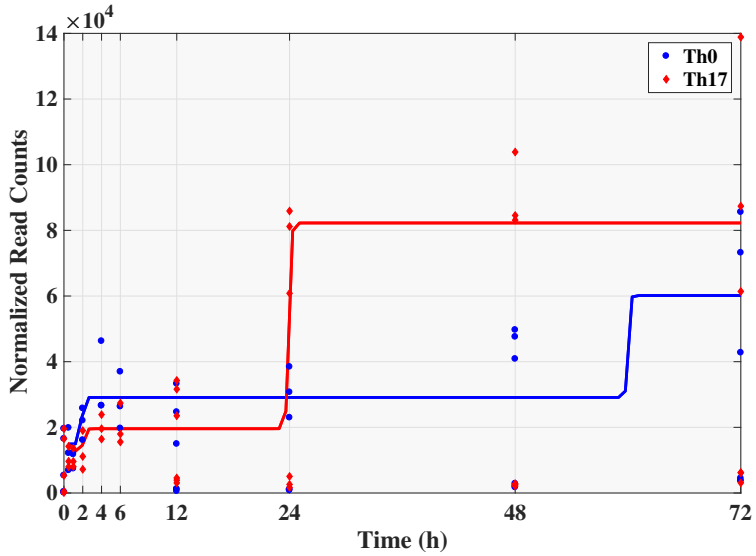
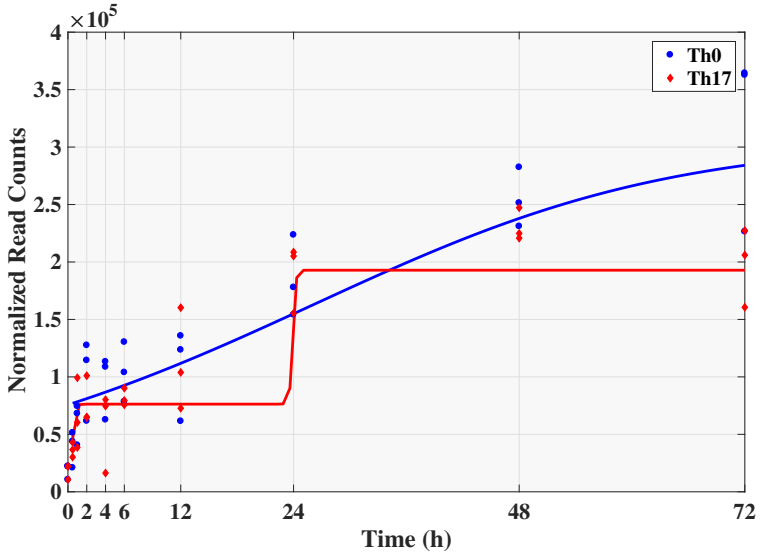


Figure C.4: Inferred expression profiles of *ACTB*, detected by GMNB but not by DyNB. Left column: The normalized expression profiles: The solid blue and red curves are the posterior means of (a) μ_k by DyNB and (c) r_k by GMNB under Th0 and Th17 lineages, respectively, with 99% CIs (shaded areas). Right column: The inferred read counts over time: The solid blue and red curves with shaded areas correspond to the inferred parameters by (b) DyNB and (d) GMNB similarly.

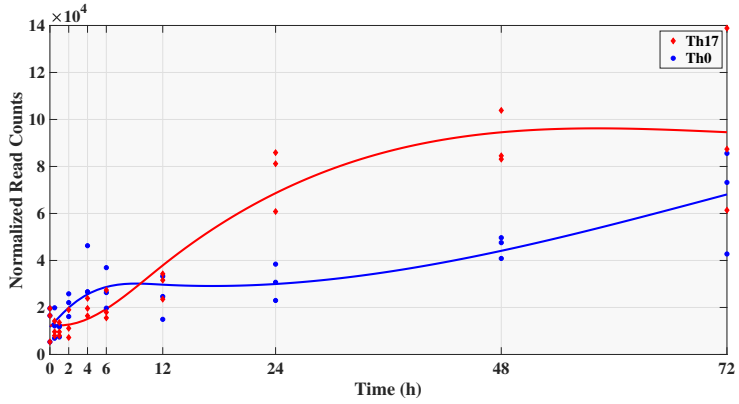


(a)

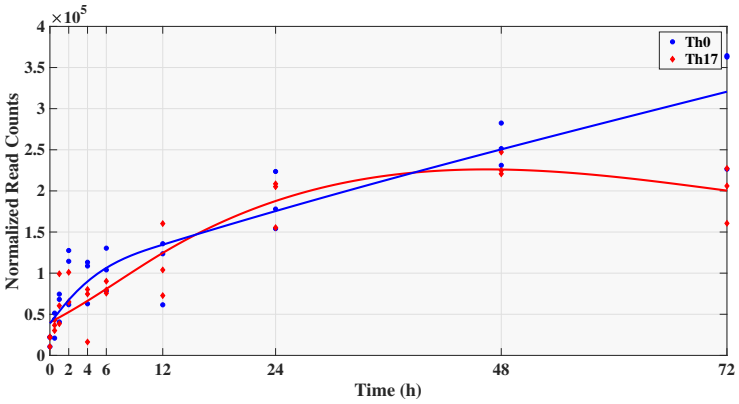


(b)

Figure C.5: Differentially expressed genes *FLNA* and *ACTB* detected by GMNB. (a) The normalized gene expression profile of *FLNA* over time estimated by ImpulseDE2 model. This gene is detected as differentially expressed by both GMNB and ImpulseDE2, but not by DyNB. (b) The normalized gene expression profile of *ACTB* over time estimated by ImpulseDE2 model. This gene is detected as differentially expressed by GMNB, but not by DyNB and ImpulseDE2.



(a)



(b)

Figure C.6: Differentially expressed genes *FLNA* and *ACTB* detected by GMNB and splineTimeR. (a) The normalized gene expression profile of *FLNA* over time estimated by splineTimeR model. (b) The normalized gene expression profile of *ACTB* over time estimated by splineTimeR model.

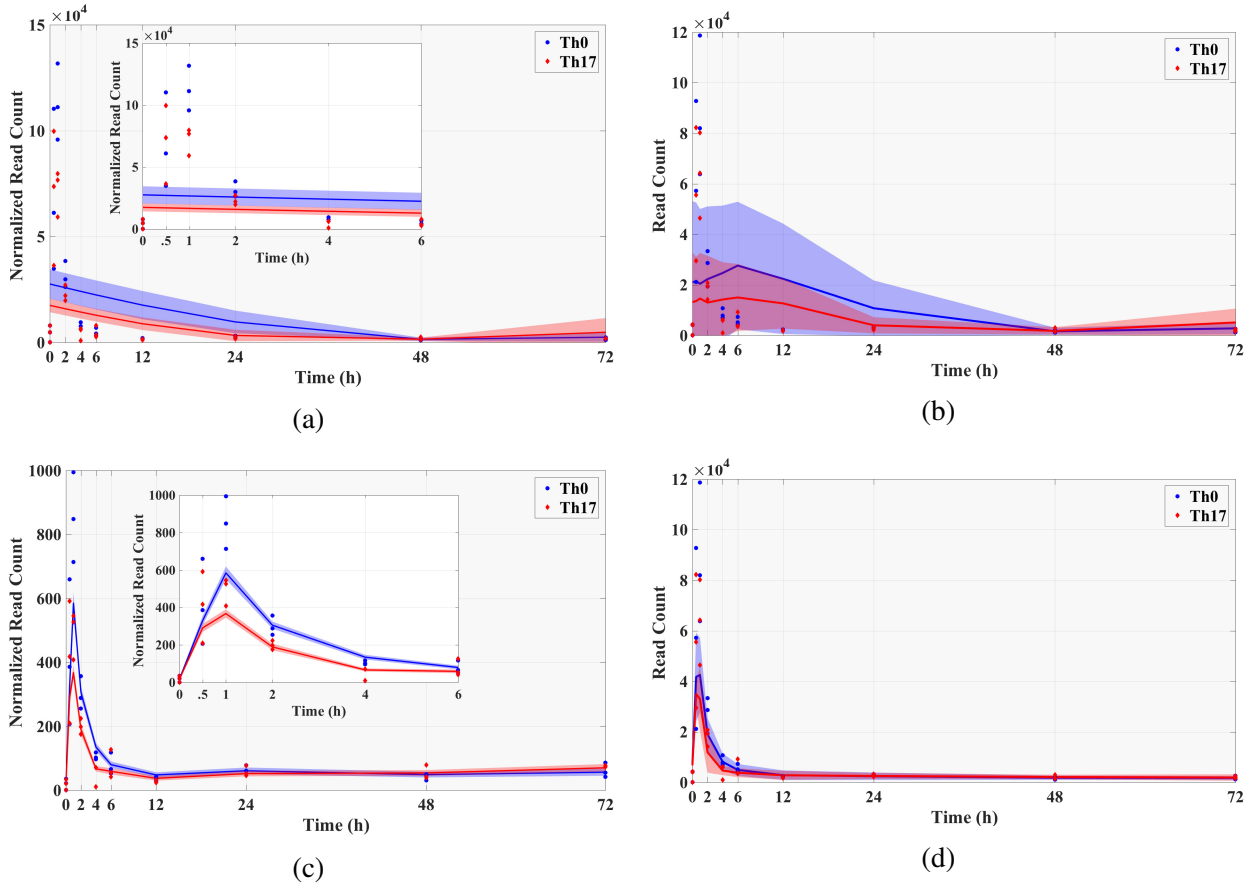


Figure C.7: Inferred expression profiles of *EGR1*, detected by GMNB (c,d) but not by DyNB (a,b) and DESeq2. Left column: The normalized expression profiles; Right column: The inferred read counts over time. They are analogous plots to those in Figure C.4, with different genes.

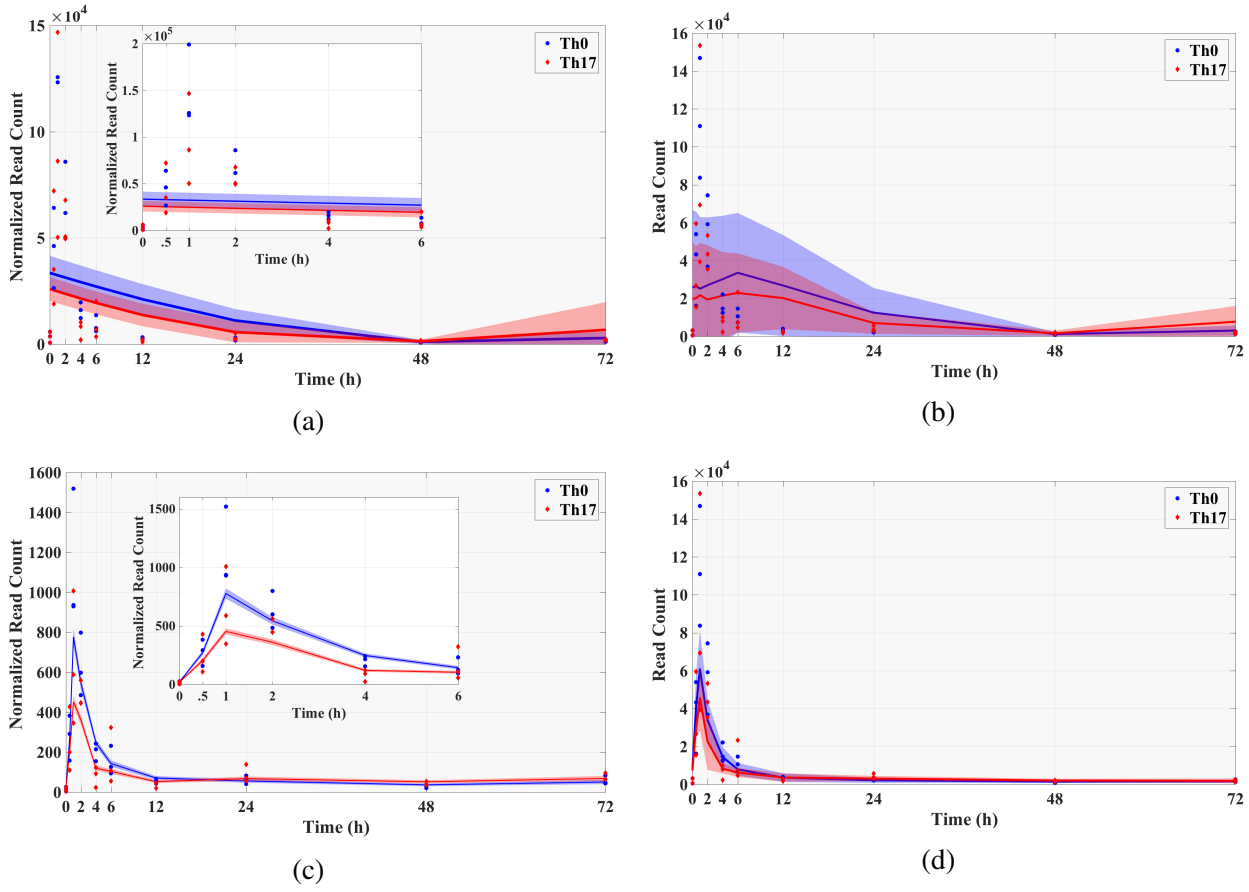


Figure C.8: Inferred expression profiles of *NR4A1*, detected by GMNB (c,d) but not by DyNB (a,b) and DESeq2. Left column: The normalized expression profiles; Right column: The inferred read counts over time. They are analogous plots to those in Figures C.4 and C.7, with different genes.

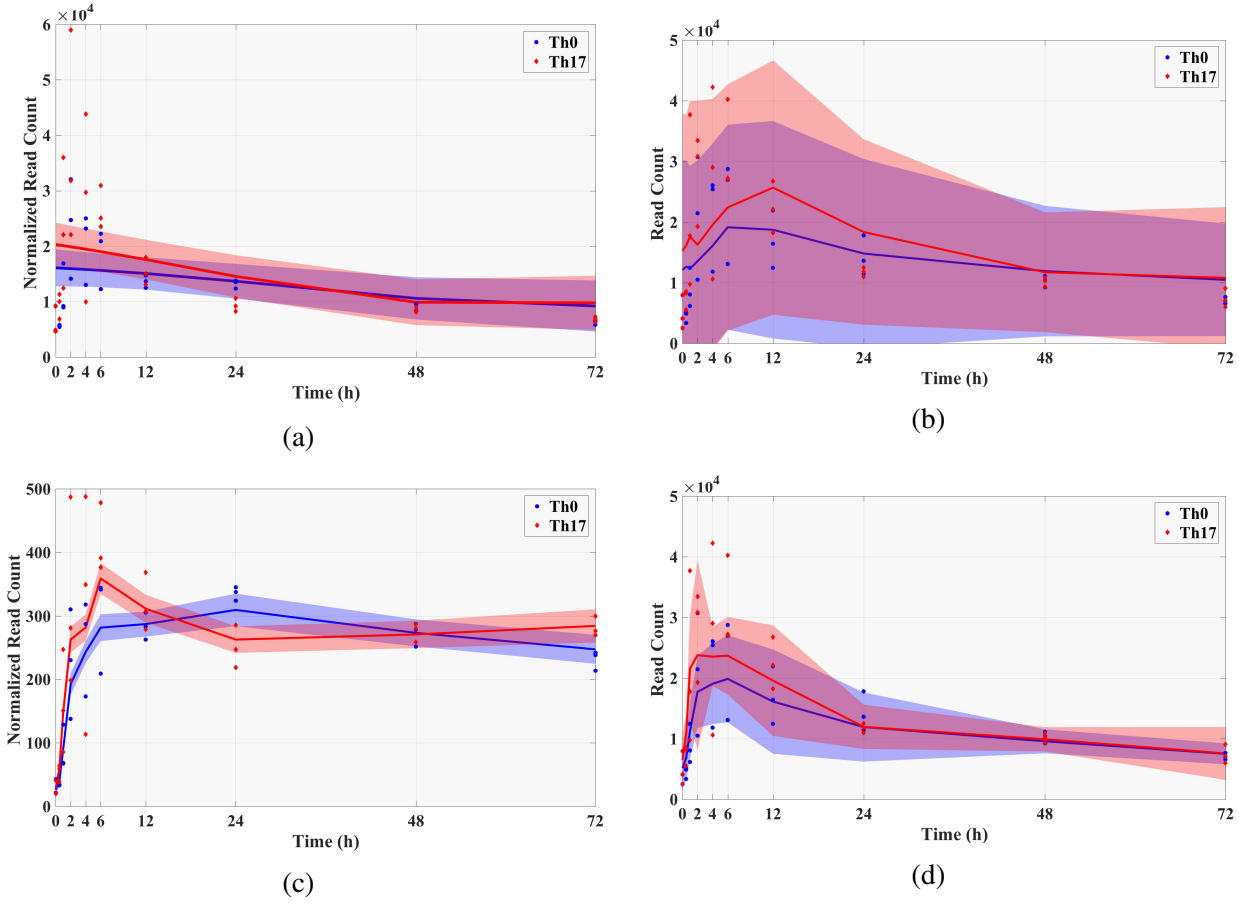


Figure C.9: Inferred expression profiles of *MYC*, detected by GMNB (c,d) but not by DyNB (a,b) and DESeq2. Left column: The normalized expression profiles; Right column: The inferred read counts over time. They are analogous plots to those in Figures C.4, C.7, and C.8, with different genes.

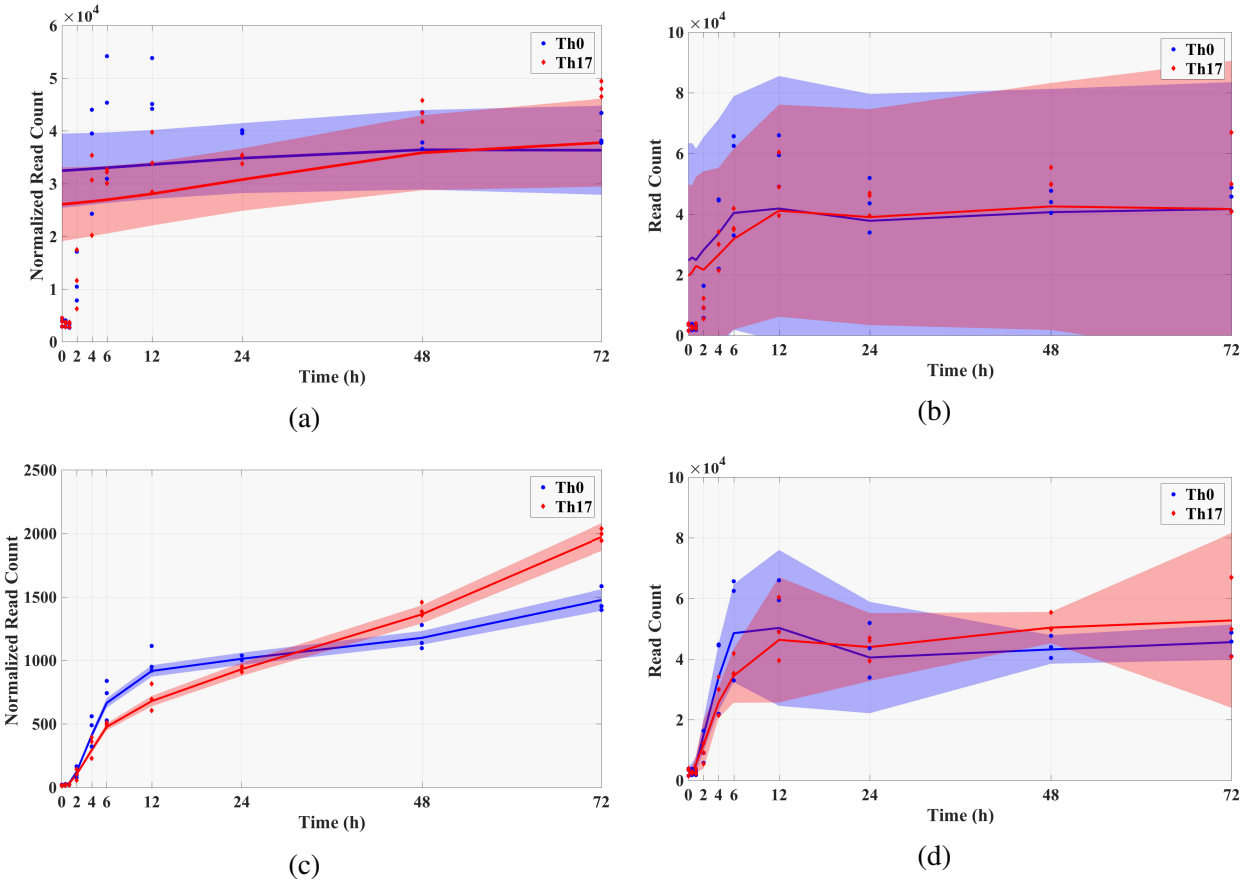


Figure C.10: Inferred expression profiles of *PKM2*, detected by GMNB (c,d) but not by DyNB (a,b) and DESeq2. Left column: The normalized expression profiles; Right column: The inferred read counts over time. They are analogous plots to those in Figures C.4 and C.7 – C.9, with different genes.

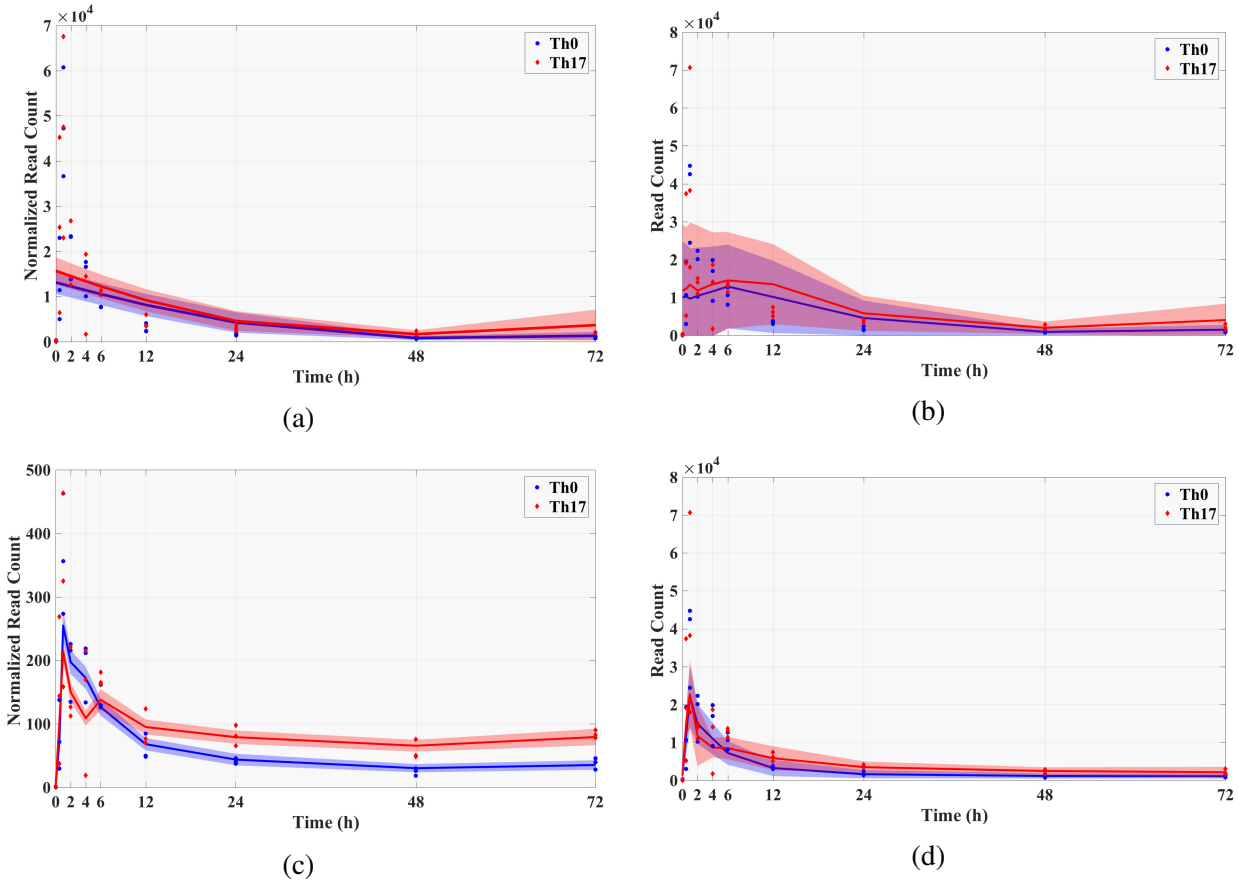


Figure C.11: Inferred expression profiles of *EGR2*, detected by GMNB (c,d) but not by DyNB (a,b) and DESeq2. Left column: The normalized expression profiles; Right column: The inferred read counts over time. They are analogous plots to those in Figures C.4 and C.7 − C.10, with different genes.

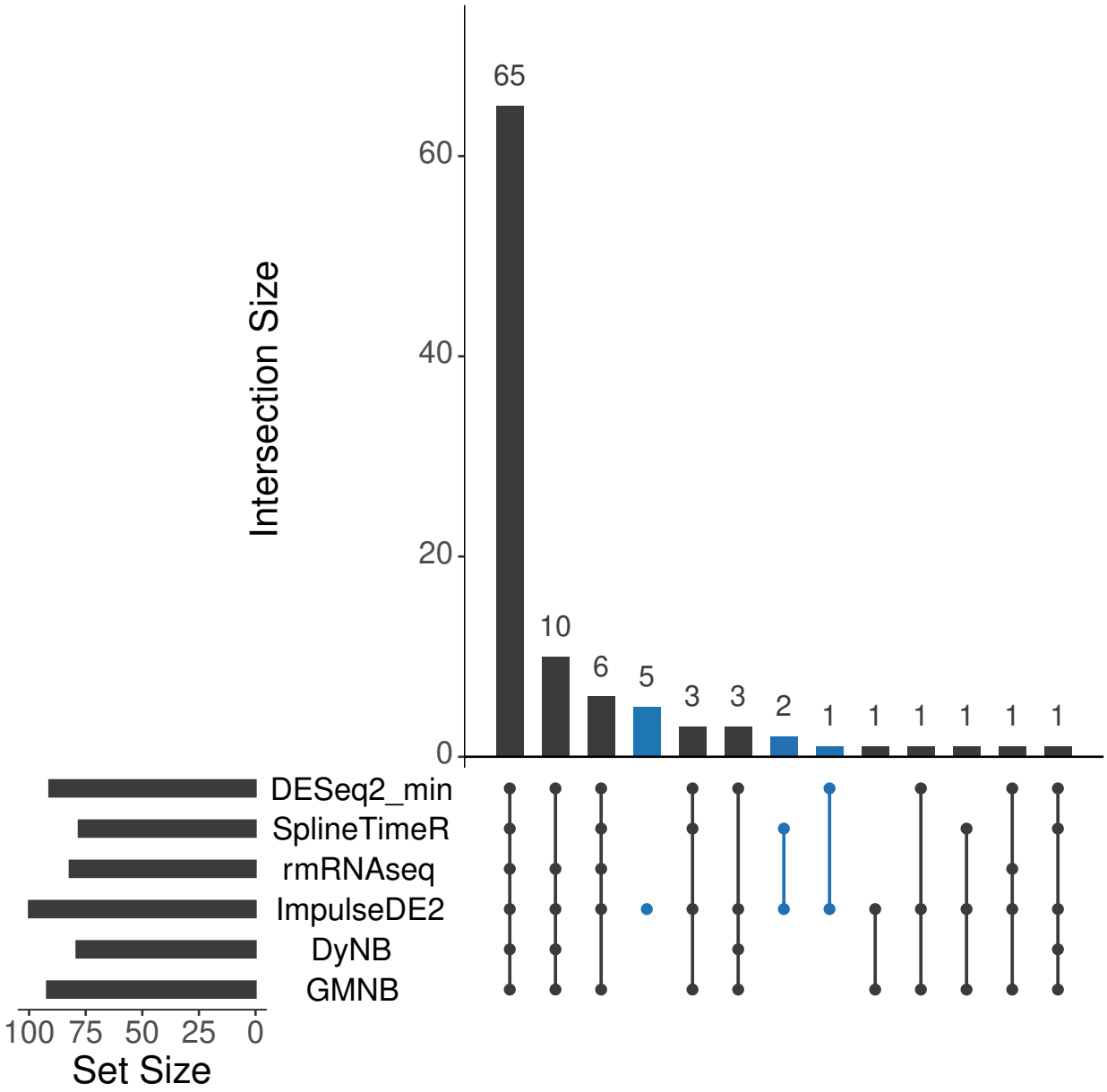
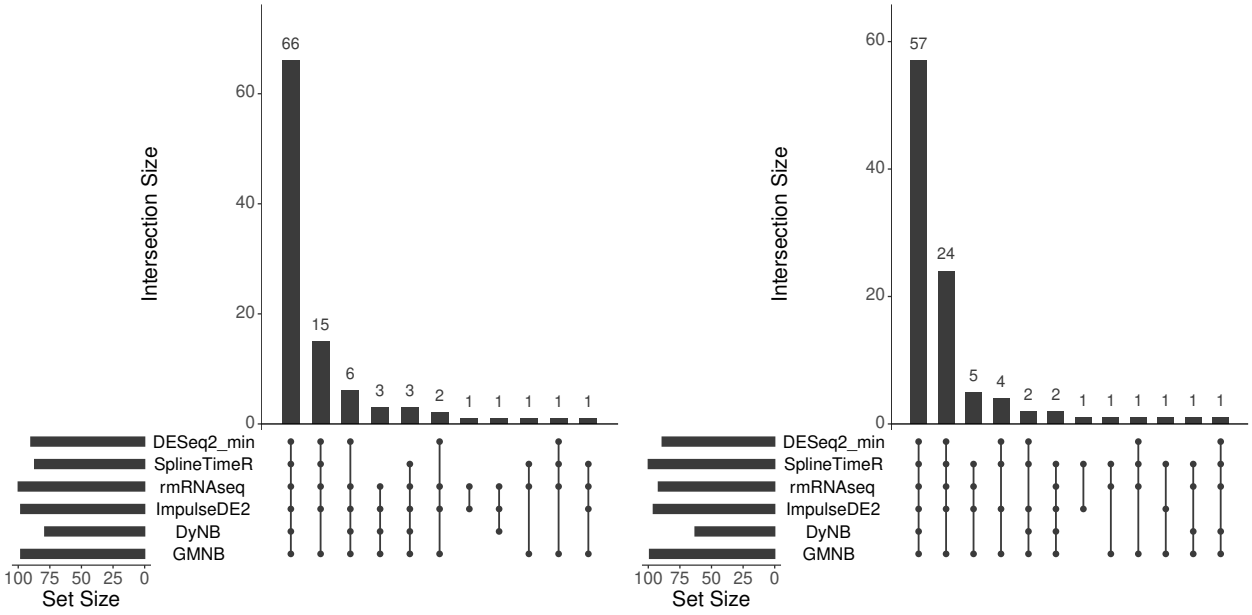
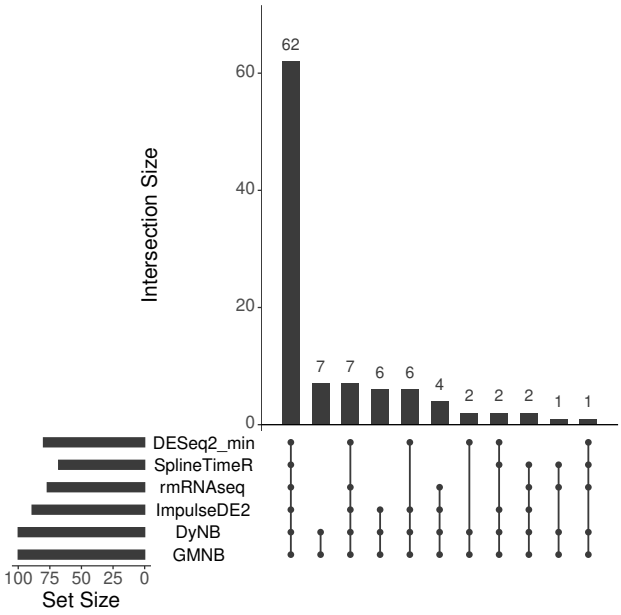


Figure C.12: The UpSet diagram representing the distribution of the top 100 differentially expressed genes detected by ImpulseDE2 over other methods. The blue bars shows the intersection of the sets that are not included GMNB.

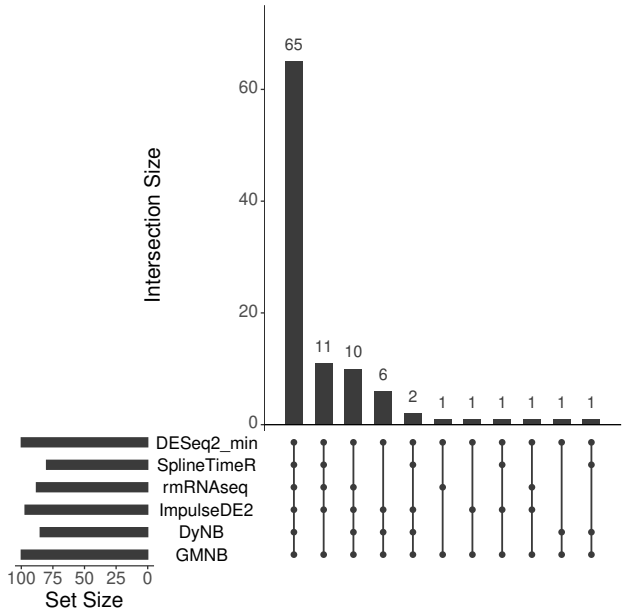


(a)

(b)



(c)



(d)

Figure C.13: The UpSet diagram representing the distribution of the top 100 differentially expressed genes detected by (a) rmRNAseq, (b) splineTimeR, (c) DyNB, and (d) DESeq2-min over other methods.

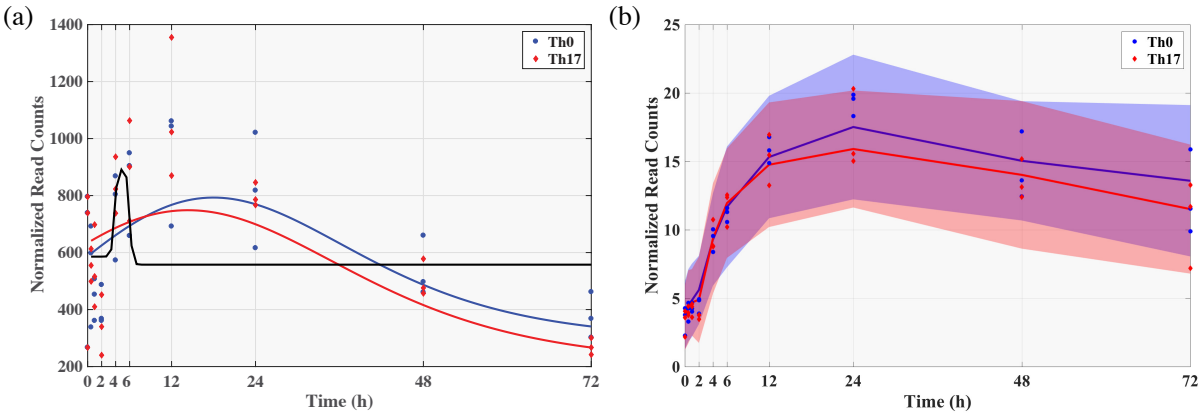


Figure C.14: Inferred expression profiles of *RP1-187N21.2* detected as differentially expressed by ImpulseDE2 but not by GMNB. (a) The blue and red curves are a separate Th0 and Th17 impulse fit (alternative hypothesis) and the black curves are a single impulse fit to all samples (null hypothesis). (b) This is analogous plot to those in Figures C.7 – C.11, with different genes.

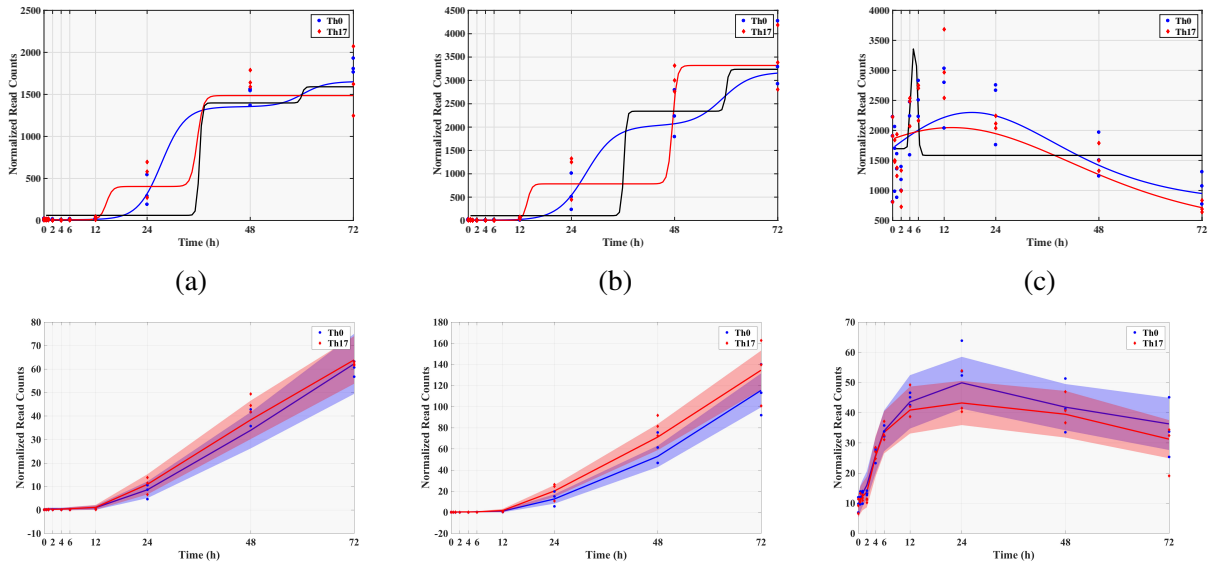


Figure C.15: Inferred expression profiles of *CDCA2* (left column), *CEP55* (middle column), and *RP11-513I15.6* (right column) detected as differentially expressed by ImpulseDE2 but not by GMNB. (a,b,c) The blue and red curves are a separate Th0 and Th17 impulse fit (alternative hypothesis) and the black curves are a single impulse fit to all samples (null hypothesis). (d,e,f) They are analogous plots to those in Figures C.4 and C.7, with different genes.

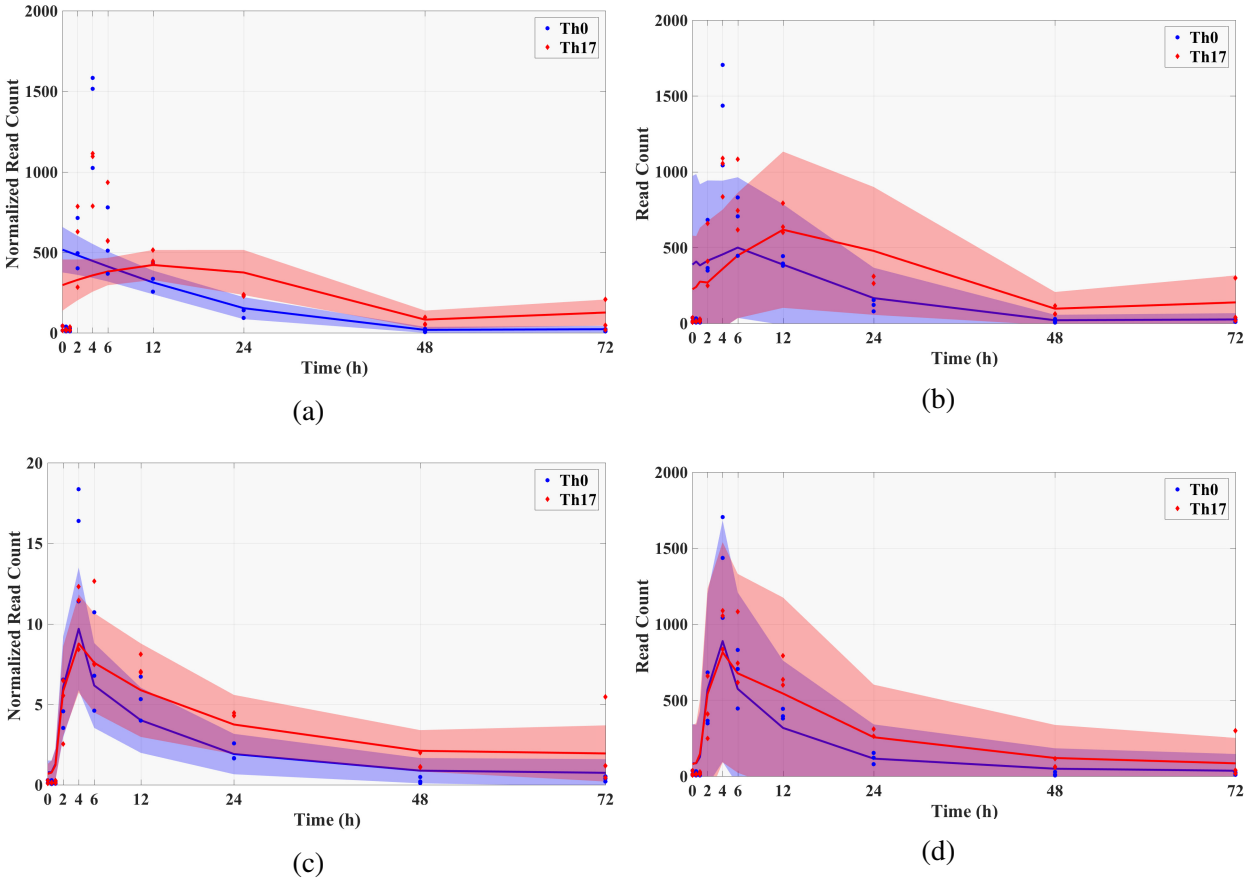


Figure C.16: Example of genes with low counts detected as differentially expressed by DyNB (a,b) but not by GMNB (c,d) and ImpulseDE2: *SEPT5*. Left column: The normalized expression profiles; Right column: The inferred read counts over time estimated.

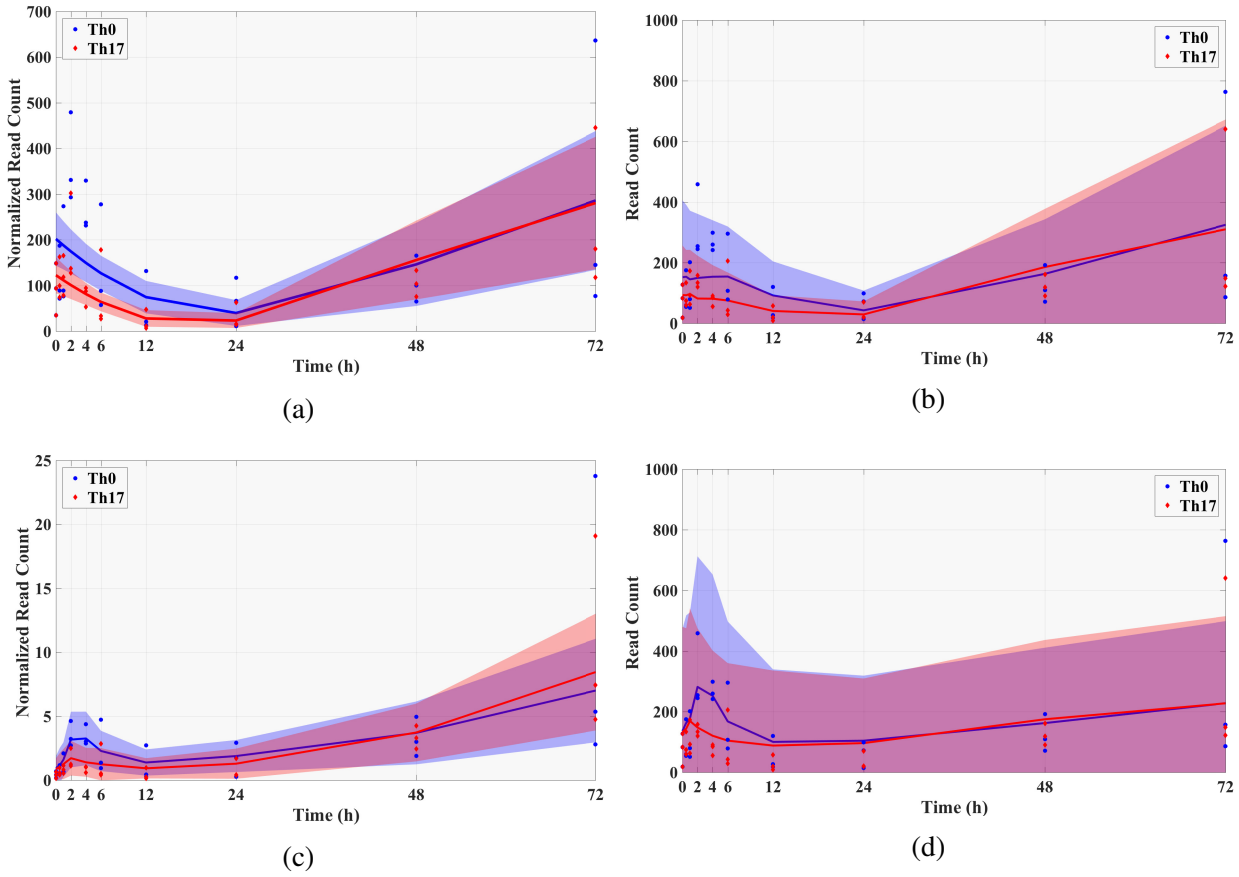


Figure C.17: Example of genes with low counts detected as differentially expressed by DyNB (a,b) but not by GMNB (c,d) and ImpulseDE2: *COL1A2*. Left column: The normalized expression profiles; Right column: The inferred read counts over time estimated.

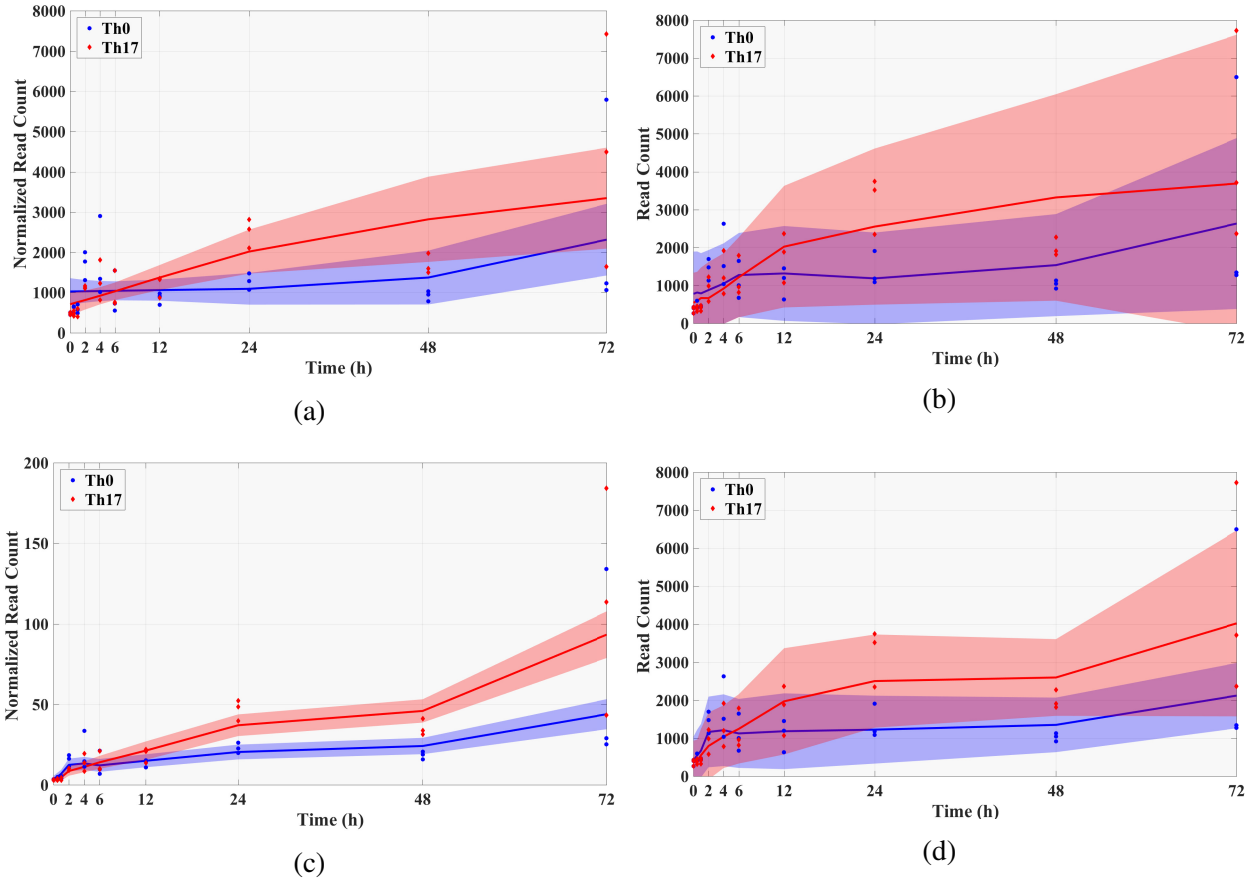


Figure C.18: Example of genes detected as differentially expressed by DyNB (a,b) but not by GMNB (c,d) and ImpulseDE2: *ENO2*. Left column: The normalized expression profiles; Right column: The inferred read counts over time estimated.

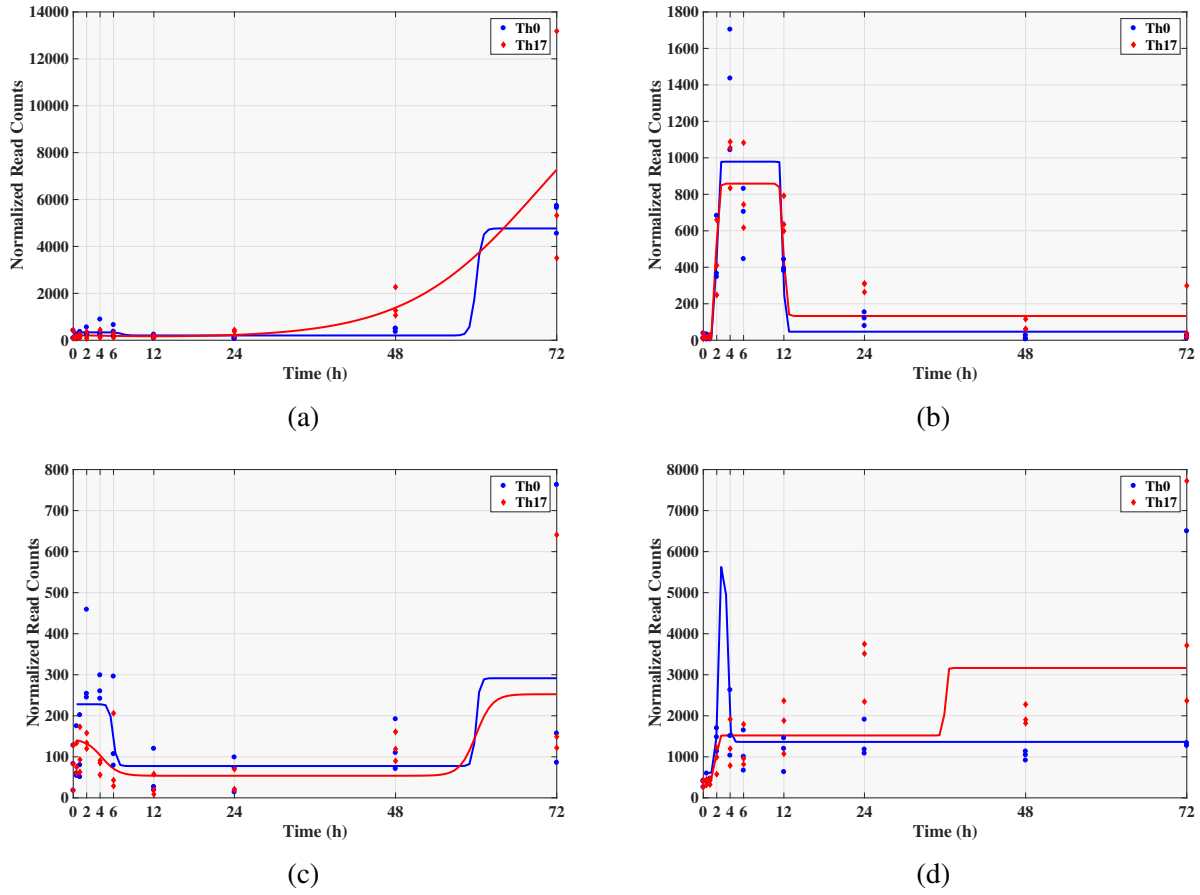


Figure C.19: Examples of genes with low counts detected as differentially expressed by DyNB but not by GMNB and ImpulseDE2. The normalized gene expression profiles of (a) *LGALS1*, (b) *SEPT5*, (c) *COL1A2*, and (d) *ENO2* over time estimated by ImpulseDE2 model.

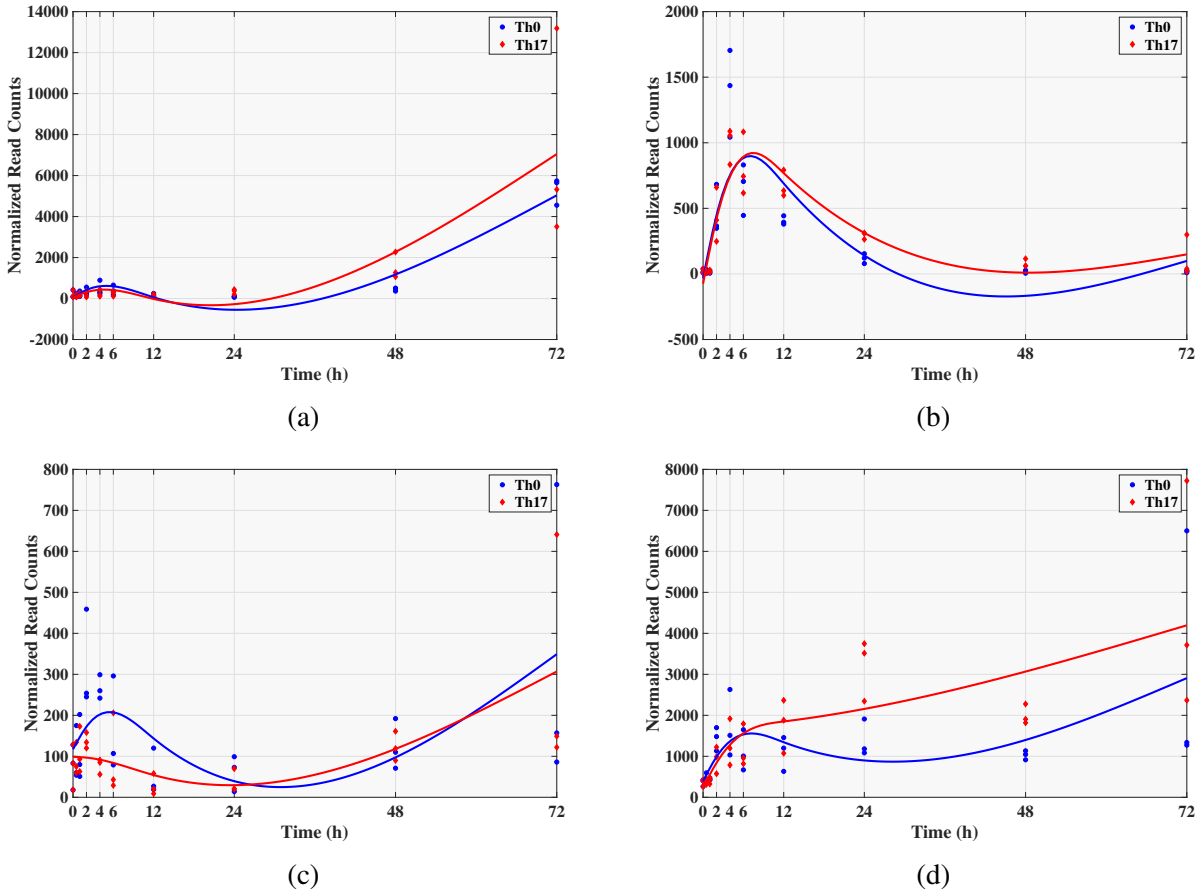
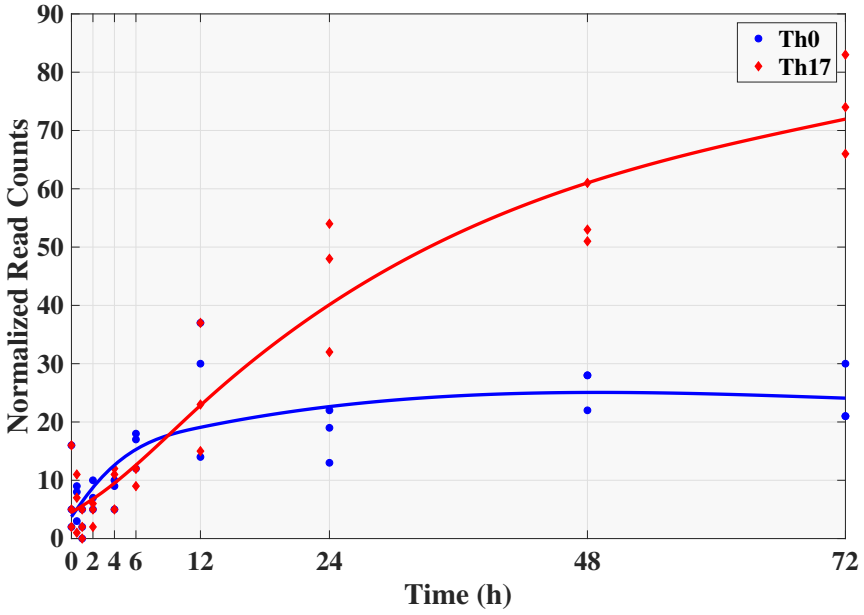
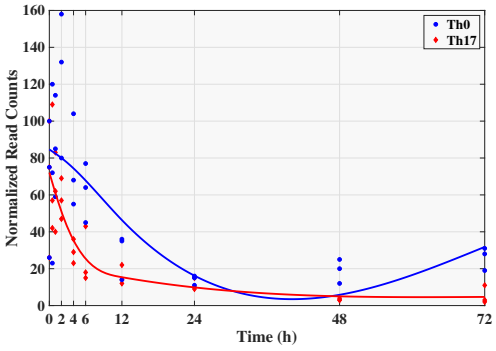


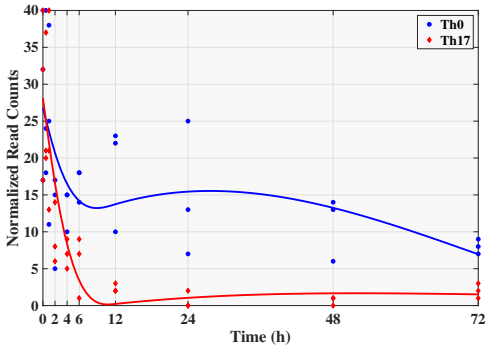
Figure C.20: Examples of genes with low counts detected as differentially expressed by DyNB but not by GMNB and splineTimeR. The normalized gene expression profiles of (a) *LGALS1*, (b) *SEPT5*, (c) *COL1A2*, and (d) *ENO2* over time estimated by splineTimeR model.



(a)



(b)



(c)

Figure C.21: Examples of genes with low counts detected as differentially expressed either by splineTimeR or rmRNAseq but not by GMNB. The normalized gene expression profiles of (a) *AP000593.5*, (b) *AC097713.4*, (c) and *RP11-80H8.4* over time estimated by splineTimeR model.

Table C.2: Enriched GO terms based on the top 100 differentially expressed genes identified by GMNB.

GO ID	Ontology	Description	p-value
GO:0048513	BP	organ development	2.00×10^{-23}
GO:0048731	BP	system development	3.08×10^{-21}
GO:0044707	BP	single-multicellular organism process	3.47×10^{-21}
GO:0002376	BP	immune system process	5.72×10^{-21}
GO:0032501	BP	multicellular organismal process	1.94×10^{-20}
GO:0007275	BP	multicellular organismal development	3.01×10^{-20}
GO:0048856	BP	anatomical structure development	6.80×10^{-20}
GO:0006955	BP	immune response	9.30×10^{-20}
GO:0032502	BP	developmental process	2.06×10^{-19}
GO:0050896	BP	response to stimulus	2.59×10^{-19}
GO:0048518	BP	positive regulation of biological process	3.59×10^{-19}
GO:0044767	BP	single-organism developmental process	6.08×10^{-19}
GO:0071310	BP	cellular response to organic substance	1.33×10^{-18}
GO:0030154	BP	cell differentiation	2.35×10^{-18}
GO:0007165	BP	signal transduction	2.96×10^{-18}
GO:0051716	BP	cellular response to stimulus	8.01×10^{-18}
GO:0044700	BP	single organism signaling	1.07×10^{-17}
GO:0023052	BP	signaling	1.14×10^{-17}
GO:0070887	BP	cellular response to chemical stimulus	1.35×10^{-17}
GO:0002682	BP	regulation of immune system process	1.62×10^{-17}
GO:0044763	BP	single-organism cellular process	1.83×10^{-17}
GO:0007154	BP	cell communication	1.95×10^{-17}
GO:0007166	BP	cell surface receptor signaling pathway	4.80×10^{-17}
GO:0009605	BP	response to external stimulus	5.06×10^{-17}
GO:0042221	BP	response to chemical	5.55×10^{-17}
GO:0048869	BP	cellular developmental process	8.14×10^{-17}
GO:0006952	BP	defense response	1.83×10^{-16}
GO:0010033	BP	response to organic substance	2.53×10^{-16}
GO:0044699	BP	single-organism process	2.54×10^{-16}
GO:0048583	BP	regulation of response to stimulus	1.28×10^{-15}
GO:0050794	BP	regulation of cellular process	1.31×10^{-15}
GO:0045944	BP	positive regulation of transcription from RNA polymerase II promoter	1.84×10^{-15}
GO:0048522	BP	positive regulation of cellular process	2.04×10^{-15}
GO:0048523	BP	negative regulation of cellular process	2.83×10^{-15}
GO:0065007	BP	biological regulation	4.92×10^{-15}
GO:0002520	BP	immune system development	7.31×10^{-15}
GO:0048519	BP	negative regulation of biological process	1.47×10^{-14}
GO:0010604	BP	positive regulation of macromolecule metabolic process	2.77×10^{-14}
GO:0050789	BP	regulation of biological process	4.98×10^{-14}
GO:0071495	BP	cellular response to endogenous stimulus	6.85×10^{-14}

Table C.3: Enriched high-level GO terms based on the top 100 differentially expressed genes identified by GMNB.

GO ID	Ontology	Description	p-value	IC
GO:0006955	BP	immune response	9.30×10^{-20}	2.32
GO:0071310	BP	cellular response to organic substance	1.33×10^{-18}	2.02
GO:0002682	BP	regulation of immune system process	1.62×10^{-17}	2.40
GO:0006952	BP	defense response	1.83×10^{-16}	2.23
GO:0045944	BP	positive regulation of transcription from RNA polymerase II promoter	1.84×10^{-15}	2.79
GO:0002520	BP	immune system development	7.31×10^{-15}	3.02
GO:0071495	BP	cellular response to endogenous stimulus	6.85×10^{-14}	2.59
GO:0002521	BP	leukocyte differentiation	1.29×10^{-13}	3.60
GO:0048534	BP	hematopoietic or lymphoid organ development	1.47×10^{-13}	3.08
GO:0009888	BP	tissue development	1.82×10^{-13}	2.24
GO:0031328	BP	positive regulation of cellular biosynthetic process	2.38×10^{-13}	2.25
GO:0051254	BP	positive regulation of RNA metabolic process	3.26×10^{-13}	2.41
GO:0009891	BP	positive regulation of biosynthetic process	3.37×10^{-13}	2.23
GO:0050793	BP	regulation of developmental process	3.82×10^{-13}	2.08
GO:0009611	BP	response to wounding	4.21×10^{-13}	2.79
GO:0051173	BP	positive regulation of nitrogen compound metabolic process	4.32×10^{-13}	2.24
GO:0010557	BP	positive regulation of macromolecule biosynthetic process	4.57×10^{-13}	2.32

Table C.4: Comparison different methods in terms of enrichment score for five different Th17 specific gene sets.

Gene Set	GSE27241	GSE43955	GSE26030	GSE11924	GSE14308
GMNB	0.89	0.87	0.85	0.85	0.83
DyNB	0.40	0.42	0.46	0.23	0.22
ImpulseDE2	0.43	0.47	0.44	0.44	0.43
rmRNAseq	0.32	0.40	0.35	0.35	0.39
splineTimeR	0.40	0.39	0.32	0.31	0.38
DESeq2-GLM	0.25	0.33	0.32	0.29	0.31
DESeq2-Min	0.28	0.38	0.32	0.34	0.37
DESeq2-Mean	0.13	0.20	0.24	0.13	0.12

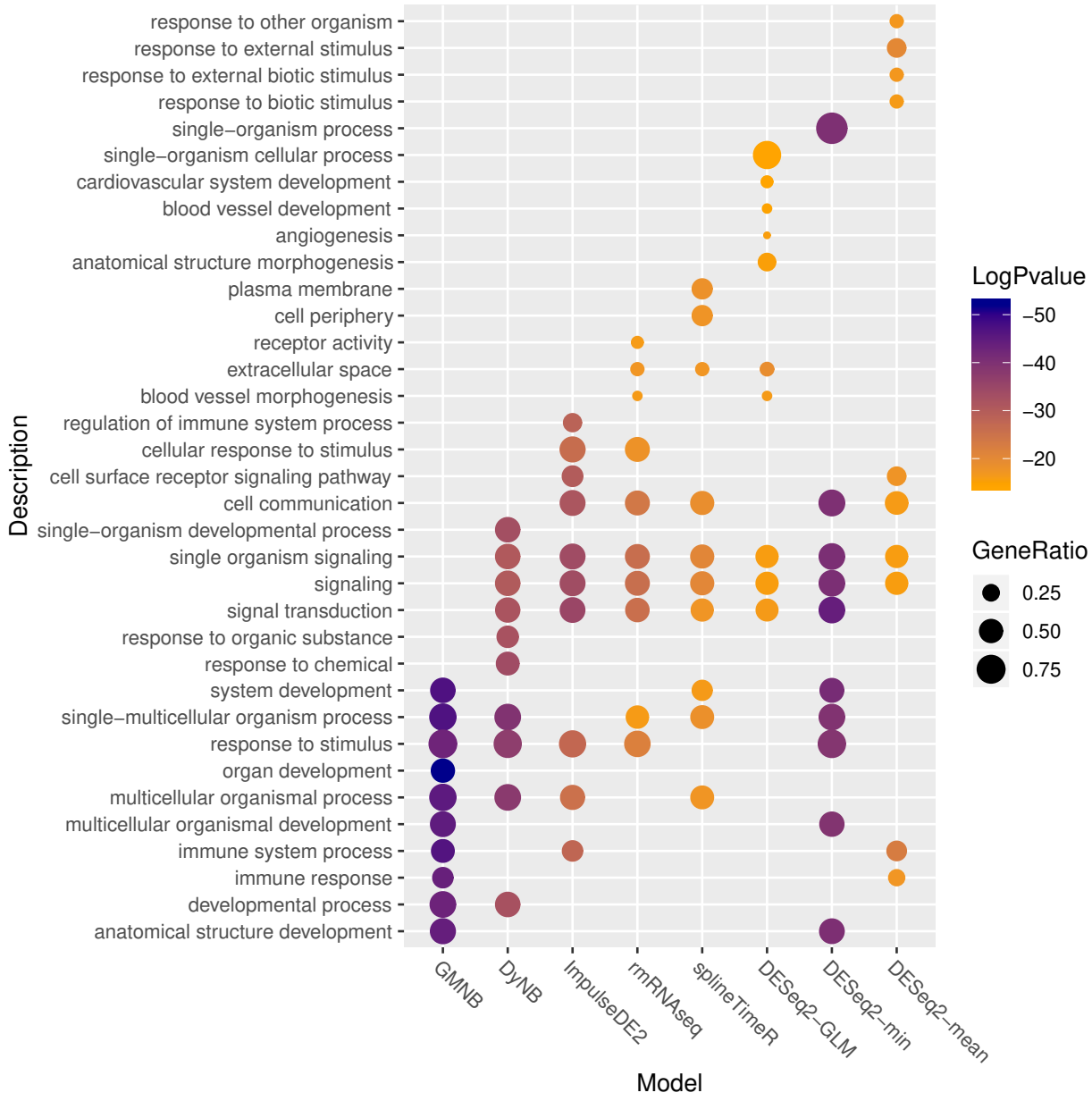


Figure C.22: Top 10 enriched GO terms of different models.

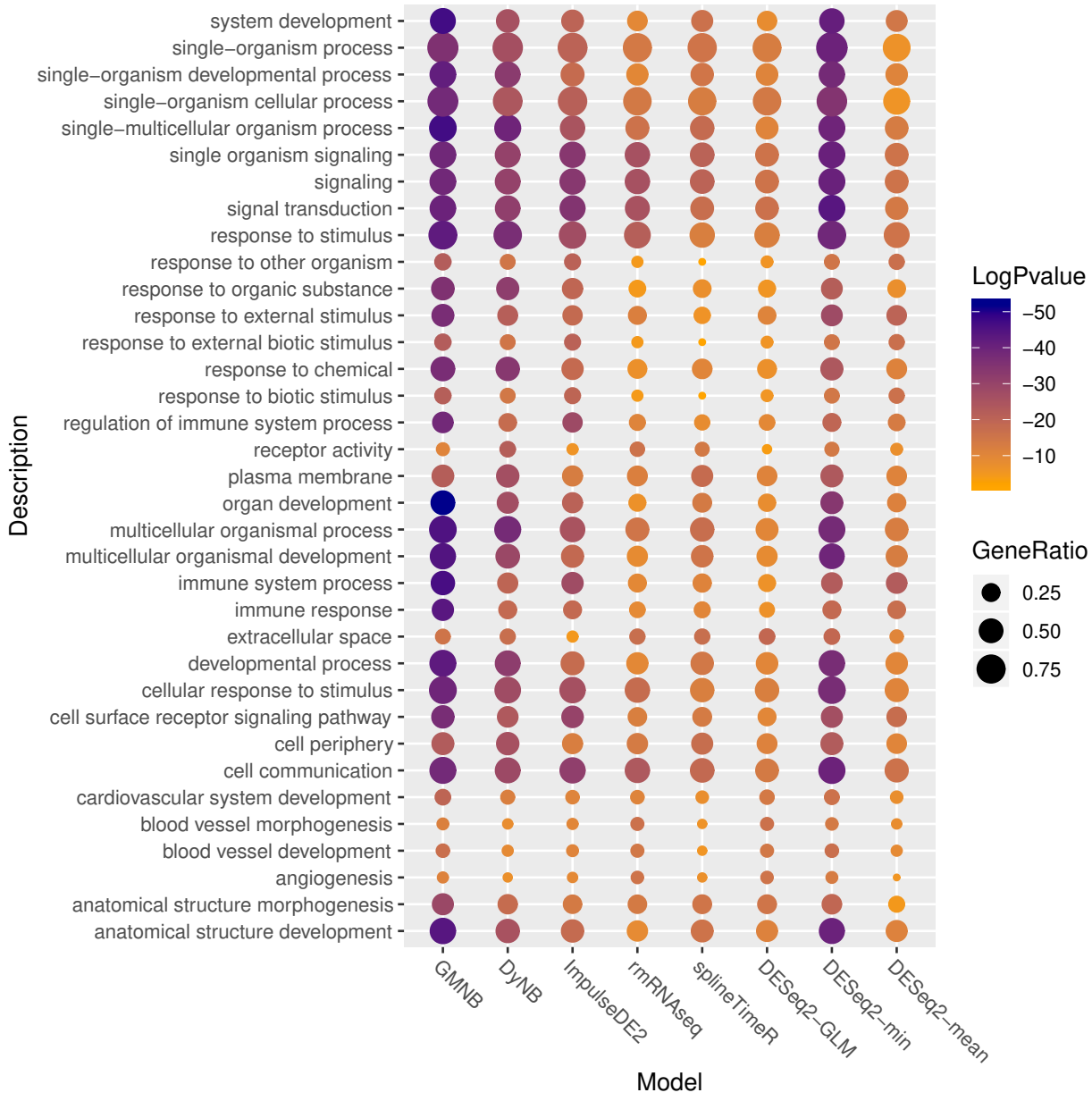


Figure C.23: Comparison different models based on gene ratio and p-value for top 10 enriched GO terms of different models.

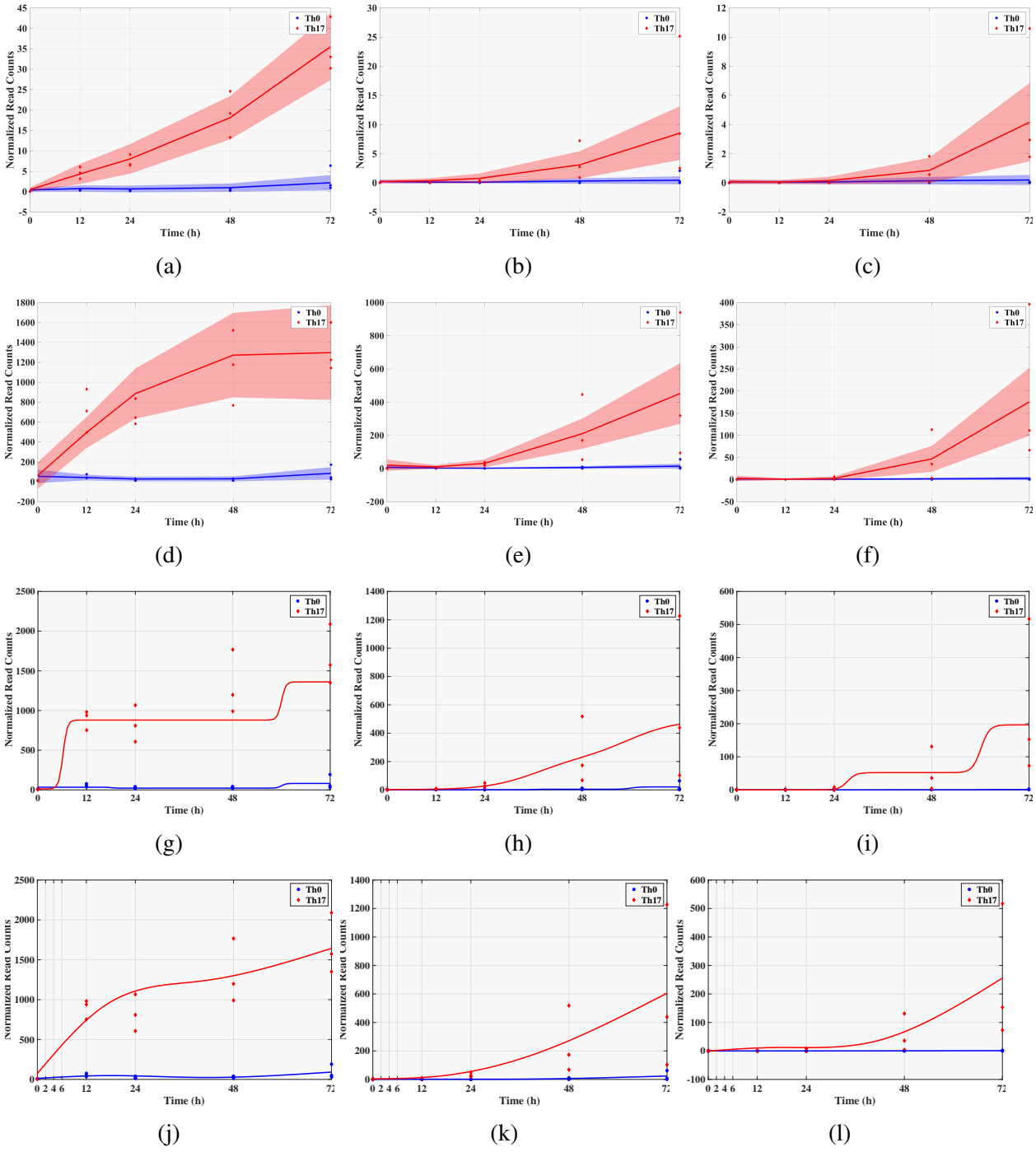


Figure C.24: Inferred expression profiles of genes *RORC*, *IL17F*, and *IL17A*. The normalized gene expression profile of *RORC* (left column), *IL17F* (middle column), and *IL17A* (right column) over time estimated by (a-c) GMNB, (d-f) DyNB, (g-i) ImpulseDE2, and (j-l) splineTimeR.

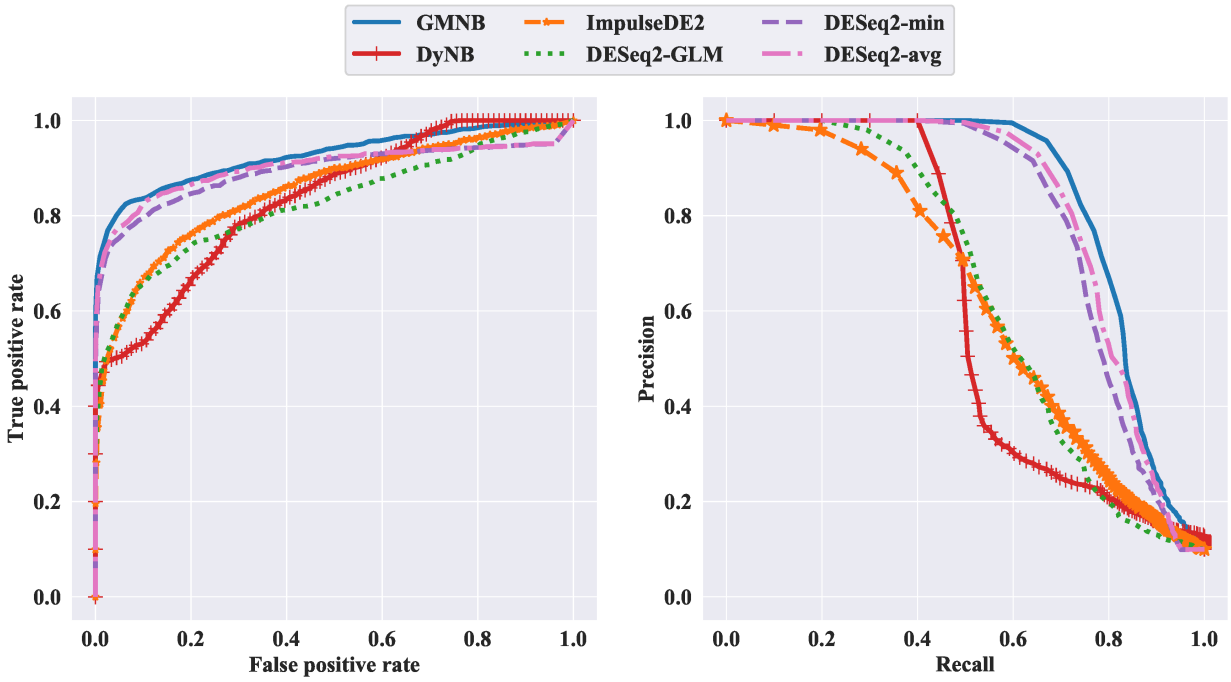


Figure C.25: Left column: Precision-recall (PR) curves, Right column: Receiver operating characteristic (ROC) curves. Performance comparison of different methods for differential gene expression over time based on the GMNB generative model. Area-under-the-curves (AUCs) are given in the corresponding legends in the plots.

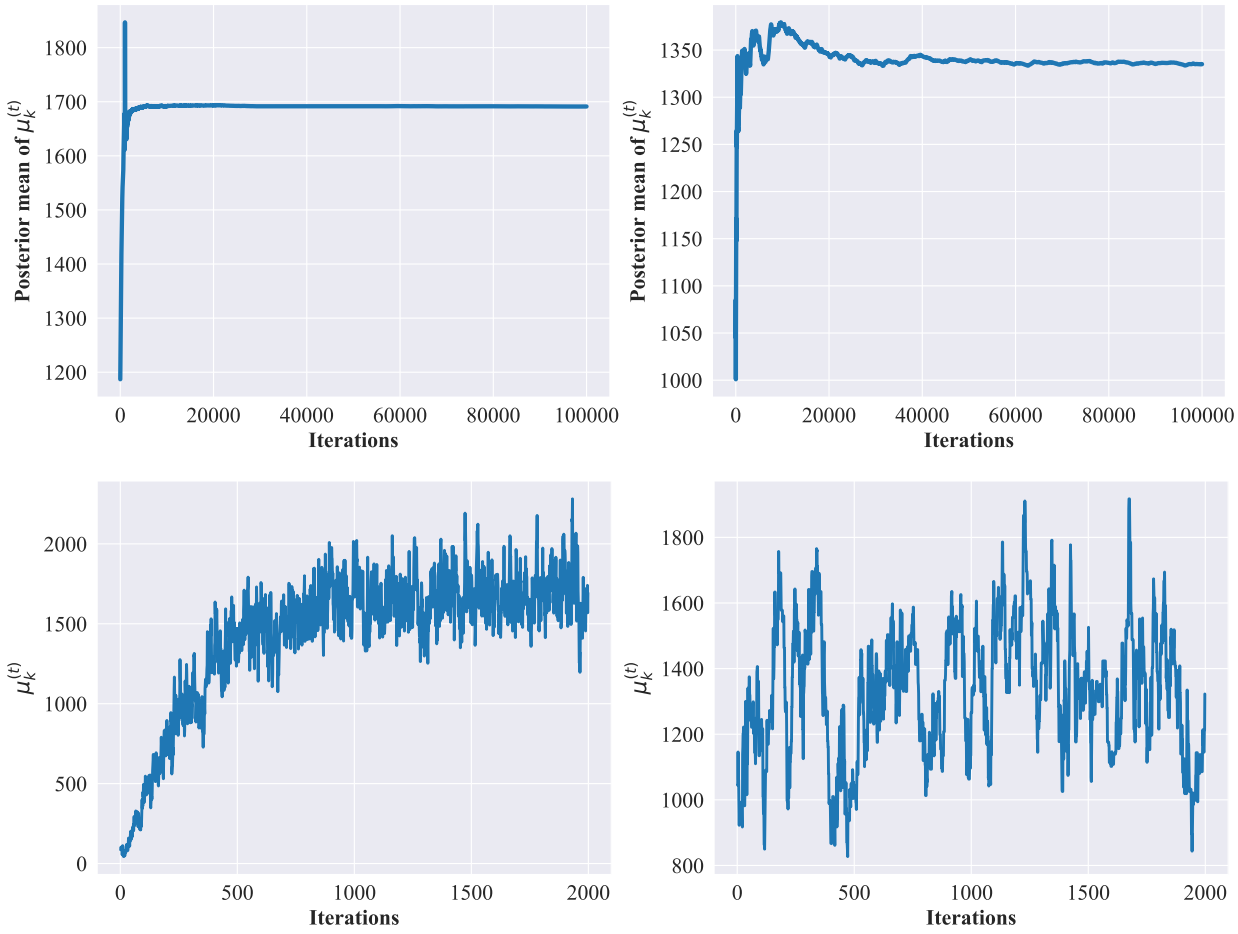


Figure C.26: Trace plots of MCMC samples for gene expression parameter of the GMNB (left column) and DyNB (right column) methods, applied to the Human-activated T- and Th17 cells data in [1].

APPENDIX D

NOTES ON RELATED WORKS OF VGRNN

Several dynamic graph embedding methods have been developed using various techniques such as matrix factorization [154, 155], random walk [156, 157], deep learning [79, 74, 95, 158], and stochastic process [159, 160, 76]. The shortcomings of the existing methods can be categorized as follows:

- Most of these existing methods either capture topological evolution or node attribute changes to learn dynamic node embeddings [161, 162]. But only a few of them model both changes simultaneously [76].

- Some of the existing methods, such as the ones in [159, 74, 155], assume that the temporal patterns of evolving processes are of short duration and fail to capture long-range temporal dependencies in dynamic networks.

- A common assumption in the literature is that the topological changes are smooth. The methods with this assumption [74, 155] usually use a regularization term to avoid abrupt changes, which limits their flexibility. Deep learning based models, such as the ones in [158, 95], have been proposed to address this shortcoming; however, these methods only care about the topological changes over time but do not model node attribute dynamics or complex dependencies between two evolving processes.

- Many of the existing methods, such as [79, 76], cannot model the deletion of nodes or edges which limits their generalizability and flexibility.

- While generative models in form of parametric temporal point processes [160] and deep temporal point processes [76] have been used for modeling dynamic graphs, none of the existing methods are capable of modeling the uncertainty of the latent representations.

Our proposed (SI-)VGRNN is the first variational based deep generative model for representation learning of dynamic graphs. On the contrary to existing methods, (SI-)VGRNN is capable of inferring the uncertainty of latent representations which is the key in modeling non-smooth changes in dynamic graphs. Moreover, (SI-)VGRNN can capture long-term dependencies in node attribute dynamics as well as topological evolution. Furthermore, (SI-)VGRNN can handle node and edge addition/deletion.

APPENDIX E

LOWER BOUND FOR ELBO IN SI-VGRNN

SI-VGRNN posterior can be derived by marginalizing out the mixing distribution as follows,

$$\mathbf{Z}^{(t)} \sim q(\mathbf{Z}^{(t)} \,|\, \boldsymbol{\psi}_t), \qquad \boldsymbol{\psi}_t \sim q_{\boldsymbol{\phi}}(\boldsymbol{\psi}_t \,|\, \mathbf{A}^{(\leq t)}, \mathbf{X}^{(\leq t)}, \mathbf{Z}^{(<t)}) = q_{\boldsymbol{\phi}}(\boldsymbol{\psi}_t | \mathbf{A}^{(t)}, \mathbf{X}^{(t)}, \mathbf{h}_{t-1}),$$
$$g_{\boldsymbol{\phi}}(\mathbf{Z}^{(t)} \,|\, \mathbf{A}^{(t)}, \mathbf{X}^{(t)}, \mathbf{h}_{t-1}) = \int_{\boldsymbol{\psi}_t} q(\mathbf{Z}^{(t)} \,|\, \boldsymbol{\psi}_t)\, q_{\boldsymbol{\phi}}(\boldsymbol{\psi}_t \,|\, \mathbf{A}^{(t)}, \mathbf{X}^{(t)}, \mathbf{h}_{t-1})\, d\boldsymbol{\psi}_t \,.$$

Based on the first theorem in [89], which shows that

$$\mathbf{KL}(\mathbb{E}_{\psi \sim q_{\boldsymbol{\phi}}(\psi \,|\, \mathbf{X}, \mathbf{A})}[q(\mathbf{Z} \,|\, \psi)] \,||\, p(\mathbf{Z})) \leq \mathbb{E}_{\psi \sim q_{\boldsymbol{\phi}}(\psi \,|\, \mathbf{X}, \mathbf{A})}[\mathbf{KL}(q(\mathbf{Z} \,|\, \psi) \,||\, p(\mathbf{Z}))],$$

the lower bound for ELBO can be derived as follows,

$$\begin{aligned}
\underline{\mathcal{L}} &= \sum_{t=1}^{T} \underline{\mathcal{L}} \left(q(\mathbf{Z}^{(t)} \,|\, \boldsymbol{\psi}_t),\, q_{\boldsymbol{\phi}}(\boldsymbol{\psi}_t \,|\, \mathbf{A}^{(t)}, \mathbf{X}^{(t)}, \mathbf{h}_{t-1}) \right) \\
&= \sum_{t=1}^{T} \mathbb{E}_{\boldsymbol{\psi}_t \sim q_{\boldsymbol{\phi}}(\boldsymbol{\psi}_t \,|\, \mathbf{A}^{(t)}, \mathbf{X}^{(t)}, \mathbf{h}_{t-1})} \mathbb{E}_{\mathbf{Z}^{(t)} \sim q(\mathbf{Z}^{(t)} \,|\, \boldsymbol{\psi}_t)} \log \left(\frac{p(\mathbf{A}^{(t)} \,|\, \mathbf{Z}^{(t)}, \mathbf{h}_{t-1})\, p(\mathbf{Z}^{(t)} \,|\, \mathbf{h}_{t-1})}{q(\mathbf{Z}^{(t)} \,|\, \boldsymbol{\psi}_t)} \right) \\
&= -\sum_{t=1}^{T} \mathbb{E}_{\boldsymbol{\psi}_t \sim q_{\boldsymbol{\phi}}(\boldsymbol{\psi}_t \,|\, \mathbf{A}^{(t)}, \mathbf{X}^{(t)}, \mathbf{h}_{t-1})} \mathbf{KL} \left(q(\mathbf{Z}^{(t)} \,|\, \boldsymbol{\psi}_t) \,||\, p(\mathbf{Z}^{(t)} \,|\, \mathbf{h}_{t-1}) \right) \\
&\qquad + \mathbb{E}_{\boldsymbol{\psi}_t \sim q_{\boldsymbol{\phi}}(\boldsymbol{\psi}_t \,|\, \mathbf{A}^{(t)}, \mathbf{X}^{(t)}, \mathbf{h}_{t-1})} \mathbb{E}_{\mathbf{Z}^{(t)} \sim q(\mathbf{Z}^{(t)} \,|\, \boldsymbol{\psi}_t)} \log p(\mathbf{A}^{(t)} \,|\, \mathbf{Z}^{(t)}, \mathbf{h}_{t-1}) \\
&\leq -\sum_{t=1}^{T} \mathbf{KL} \left(\mathbb{E}_{\boldsymbol{\psi}_t \sim q_{\boldsymbol{\phi}}(\boldsymbol{\psi}_t \,|\, \mathbf{A}^{(t)}, \mathbf{X}^{(t)}, \mathbf{h}_{t-1})} q(\mathbf{Z}^{(t)} \,|\, \boldsymbol{\psi}_t) \,||\, p(\mathbf{Z}^{(t)} \,|\, \mathbf{h}_{t-1}) \right) \\
&\qquad + \mathbb{E}_{\boldsymbol{\psi}_t \sim q_{\boldsymbol{\phi}}(\boldsymbol{\psi}_t \,|\, \mathbf{A}^{(t)}, \mathbf{X}^{(t)}, \mathbf{h}_{t-1})} \mathbb{E}_{\mathbf{Z}^{(t)} \sim q(\mathbf{Z}^{(t)} \,|\, \boldsymbol{\psi}_t)} \log p(\mathbf{A}^{(t)} \,|\, \mathbf{Z}^{(t)}, \mathbf{h}_{t-1}) \\
&= \sum_{t=1}^{T} \mathbb{E}_{\mathbf{Z}^{(t)} \sim g_{\boldsymbol{\phi}}(\mathbf{Z}^{(t)} \,|\, \mathbf{A}^{(t)}, \mathbf{X}^{(t)}, \mathbf{h}_{t-1})} \log \left(\frac{p(\mathbf{A}^{(t)} \,|\, \mathbf{Z}^{(t)}, \mathbf{h}_{t-1})\, p(\mathbf{Z}^{(t)} \,|\, \mathbf{h}_{t-1})}{g_{\boldsymbol{\phi}}(\mathbf{Z}^{(t)} \,|\, \mathbf{A}^{(t)}, \mathbf{X}^{(t)}, \mathbf{h}_{t-1})} \right) \\
&= \mathbb{E}_{\mathbf{Z} \sim q(\mathbf{Z}^{(\leq t)} \,|\, \mathbf{A}^{(\leq t)}, \mathbf{X}^{(\leq t)})} \left[\log p(\mathbf{A}^{(\leq t)}, \mathbf{X}^{(\leq t)}, \mathbf{Z}^{(\leq t)}) - \log q(\mathbf{Z}^{(\leq t)} \,|\, \mathbf{A}^{(\leq t)}, \mathbf{X}^{(\leq t)}) \right] \\
&= \mathcal{L}
\end{aligned}$$

While a Monte Carlo estimation of $\underline{\mathcal{L}}$ only requires $q_{\boldsymbol{\phi}}(\mathbf{Z}^{(t)} \mid \boldsymbol{\psi}_t)$ to have an analytic density functions and $q_{\boldsymbol{\phi}}(\boldsymbol{\psi}_t \mid \mathbf{X}^{(t)}, \mathbf{h}_{t-1})$ to be convenient to sample from, the marginal posterior $g_{\boldsymbol{\phi}}(\mathbf{Z}^{(t)} \mid \mathbf{X}^{(t)}, \mathbf{h}_{t-1})$ is often intractable and so the Monte Carlo estimation of the ELBO $\mathcal{L}$ is prohibited. SI-VGRNN evaluates the lower bound separately from the distribution sampling. This captures the idea that combining an explicit $q_{\boldsymbol{\phi}}(\mathbf{Z}^{(t)} \mid \boldsymbol{\psi}_t)$ with an implicit $q_{\boldsymbol{\phi}}(\boldsymbol{\psi}_t \mid \mathbf{X}^{(t)}, \mathbf{h}_{t-1})$ is as powerful as needed, but makes the computation tractable.

As discussed in [89], if optimizing the variational parameter by climbing $\underline{\mathcal{L}}$, without stopping the optimization algorithm early, $q_{\boldsymbol{\phi}}(\boldsymbol{\psi}_t \mid \mathbf{X}^{(t)}, \mathbf{h}_{t-1})$ could converge to a point mass density, making SI-VGRNN degenerate to VGRNN. To prevent this problem and inspired by SIVI, we add a regularization term to the lower bound as follows,

$$\underline{\mathcal{L}}_K = \underline{\mathcal{L}} + B_K,$$

where

$$B_K = \sum_{t=1}^{T} \mathbb{E}_{\boldsymbol{\psi}_t, \boldsymbol{\psi}_t^{(1)}, \ldots, \boldsymbol{\psi}_t^{(K)} \sim q_{\boldsymbol{\phi}}(\boldsymbol{\psi}_t \mid \mathbf{A}^{(t)}, \mathbf{X}^{(t)}, \mathbf{h}_{t-1})} \mathbf{KL}(q(\mathbf{Z}^{(t)} \mid \boldsymbol{\psi}_t) \,||\, \tilde{g}_K(\mathbf{Z}^{(t)} | \mathbf{A}^{(t)}, \mathbf{X}^{(t)}, \mathbf{h}_{t-1})),$$

$$\tilde{g}_K(\mathbf{Z}^{(t)} \mid \mathbf{A}^{(t)}, \mathbf{X}^{(t)}, \mathbf{h}_{t-1})) = \frac{q_{\boldsymbol{\phi}}(\boldsymbol{\psi}_t \mid \mathbf{A}^{(t)}, \mathbf{X}^{(t)}, \mathbf{h}_{t-1}) + \sum_{k=1}^{K} q_{\boldsymbol{\phi}}(\boldsymbol{\psi}_t^{(k)} \mid \mathbf{A}^{(t)}, \mathbf{X}^{(t)}, \mathbf{h}_{t-1})}{K+1}.$$

The lower bound leads to an asymptotically exact ELBO that satisfies $\underline{\mathcal{L}}_0 = \underline{\mathcal{L}}$ and $\lim_{K \to \infty} \underline{\mathcal{L}}_K = \mathcal{L}$.

APPENDIX F

ADDITIONAL EXPERIMENTAL RESULTS FOR VGRNN

F.1 Additional dataset details

Enron emails (Enron). This graph constructed from 500,000 email messages exchanged between 184 Enron employees from 1998 to 2002 [163]. The nodes represent the employees and the edges are emails exchanged between two employees. Following the same procedure as in [164, 165] we clean the data to get 10 temporal snapshots of the graph. This graph does not have any node or edge attribute.

Collaboration (COLAB). This dataset represents collaborations between 315 authors. Each node in this dynamic graph is an author and the edges represent co-authorship relationships. The data, provided by [165], are collected from years 2000-2009 with a total of 10 snapshots considering each year as a time stamp. This COLAB graph does not have any node or edge attribute.

Facebook. The Facebook wall posts dynamic graph, provided by [166], has 9 time stamps. Following the same data cleaning procedure as in [164, 165], we get 663 nodes at each snapshot. No node or edge attribute is provided for this graph.

HEP-TH. The original dataset [167] covers all the citations of the papers in High Energy Physics Theory conference from January 1993 to April 2003 [168]. For each month, we create a citation graph using all the papers published up to that month. We only consider the first ten months leading to 10 snapshots in this dynamic graph. The graph has 1199 nodes at the first month and 2462 at the last one. This graph also has no node or edge attributes.

Cora. The Cora dataset is another citation graph consists of 2708 scientific publications [169]. The nodes in the graph represent the publications and the edges indicate the citation relations. Each node is provided with a 1433-dimensional binary attribute vector. Each dimension of the attribute vector indicates the presence of a word in the publication from a dictionary. Originally, Cora is a static graph dataset, therefore in order to use it in a dynamic fashion, we preprocess the data as follows in

the same manner as in [170]. We take the indices of nodes as their arriving order in the dynamic graph and add 200 nodes with their corresponding edges, at each temporal snapshot. The dynamic graph includes 11 snapshots, starting with 708 nodes and reaches to 2708 nodes at the last snapshot.

Social evolution. The social evolution dataset is collected from Jan 2008 to June 30, 2009 and released by MIT Human Dynamics Lab [76]. For this dataset, we consider Calls and SMS records between users as node attributes and all Close Friendship records and Proximity as graph topology. We consider the collected information from Jan 2008 until Sep 10, 2008 (i.e. survey date) to form the initial network. We used cumulative data for 10 days periods of to form a snapshot of dynamic network for 27 snapshots.

F.2 Details on the experimental setup and hyper-parameters selection

Dynamic autoencoder (DynAE) [95]. This autoencoder model uses multiple fully connected layers for both encoder and decoder to capture highly non-linear interactions between nodes at each time step and across multiple time steps. It can take a set of graphs with different adjacency matrices. This model has $\mathcal{O}(nld_1)$ parameters, where n, l, and d_1 are the number of nodes, autoregressive lag, and dimension of the first hidden layer, respectively. Learning to optimize this huge number of parameters can be challenging for sparse graphs [95], which is often the case when studying real-world datasets. The input to this model at each node is the neighborhood vector of that node.

Dynamic recurrent neural network (DynRNN) [95]. This model uses LSTM networks as both encoder and decoder to capture the long-term dependencies in dynamic graphs. Comparing to DynAE, the number of parameters is reduced and the model is capable of learning complex temporal patterns more efficiently. The input to this model at each node is the neighborhood vector of that node.

Dynamic autoenncoder recurrent neural network (DynAERNN) [95]. Instead of passing the input adjacency matrices into LSTM, DynAERNN uses a fully connected encoder to initially acquire low dimensional hidden representations and then pass them as the input of LSTM to learn the embedding. The decoder of this model is a fully connected network similar to DynAE. The input to this model at each node is the neighborhood vector of that node.

Experimental setups. For VGAE at each snapshot, we use two GCN layers with 32 and 16 units for $\mathbf{GCN}_\mu$ and $\mathbf{GCN}_\sigma$. Since VGAE is a method for static graph embedding, we start training with the first snapshot and use the inferred parameters as initialization for the next snapshot. We continue this process until the last training snapshot. In all VGAE experiments, the learning rate is set to be 0.01. We learn the model for 500 training epochs and use the validation set for the early stopping. We use the code provided by the author [72] in our experiments. For DynAE, DynRNN, and DynAERNN, we chose the dimension and number of layers of the encoder and decoder such that the total numbers of parameters is comparable to (SI-)VGRNN. For these methods, we use the source code published by the authors. In these methods, the learning rate is set to be 0.01 and the learning procedure converges in 250 training epochs. The *look back* parameter in these models, which indicates how much in the past the model looks to learn the embedding, is set to be 2. In all of the experiments in the chapter four, the embedding dimension is set to 16 except for HEP-TH where embedding dimension is 32.

All of the node embedding methods for link prediction performance comparison are run on a single cluster node with dual-GPU Tesla K80 accelerator and 128GB RAM. For running each epoch on the HEP-TH dataset using one of the GPUs on this cluster, SI-VGRNN, VGRNN, DynRNN, DynAERNN, and DynAE take around 36, 12, 40, 5, and 1 seconds, respectively. This is expected as DynRNN has two 2-layer LSTMs as decoders and encoders. On the other hand, the number of parameters in DynAERNN, which includes just one 2-layer LSTM, is less than that of DynRNN. DynAE are faster as they do not have LSTM units.

F.3 Additional experimental results on interpretability of latent representations

Here, we include the latent representations of the simulated graph (in Section 4.2 of the main text) learned by DynAERNN (shown in Figure F.1). Compared to the latent representation learned by VGRNN, not only DynAERNN is not capable of modeling uncertainty of representations, but also it fails to separate the communities of the graph at different time steps, which VGRNN has successfully accomplished.

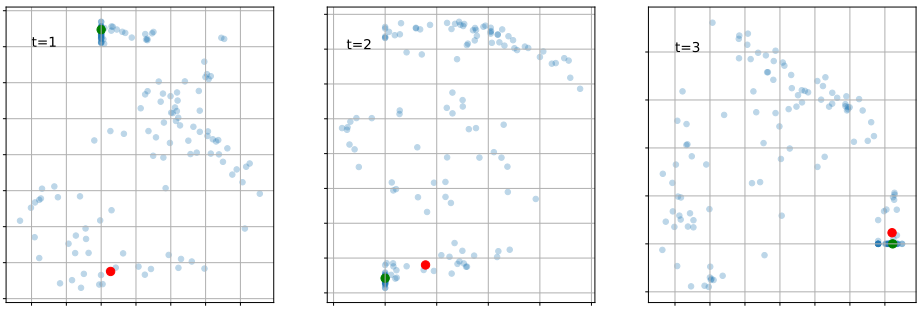


Figure F.1: Latent representation of the simulated graph in different time steps in 2-d space using DynAERNN.

APPENDIX G

SUPPLEMENTARY MATERIALS FOR BAYREL

G.1 BayReL model

To further clarify the model and workflow of our proposed BayReL, we provide a schematic illustration of BayReL in Figure S1, where we only include two views for clarity. We again note that the framework and training/inference algorithms can be generalized to multiple (> 2) views though we have focused on the examples with two views due to the availability of the corresponding data and validation sets.

G.2 Additional experimental results for acute myeloid leukemia (AML) data

Figure G.2 shows the inferred bipartite network with the top 200 interactions by BayReL. Among the genes involved in these identified interactions, Secreted Phosphoprotein 1 (SPP1) has been considered as a prognostic marker of AML patients [171] for their sensitivity to different AML drugs. CD163 has been reported to be over-expressed in AML cells [172]. BayReL in fact has also identified corresponding drugs that have been proposed to target this gene. In addition, AML has been studied with the evidence that dysregulation of several pathways, including down-regulation of major histocompatibility complex (MHC) class II genes, such as human leukocyte antigen (HLA)-DPA1 and HLA-DQA1, involved in antigen presentation, may change immune functions influencing AML development [173]. BayReL has identified the interactions between these genes and some of the validated AML drugs as appropriate therapeutics to reverse the corresponding epigenetic changes.

To further demonstrate the biological relevance of the inferred drug-gene interactions by BayReL, gene ontology (GO) enrichment analysis with the genes among the top 200 interactions has been performed using Fisher's exact test. Table G.1 shows that the enriched GO terms by these genes agree with the reported mechanistic understanding of AML disease development. The most significantly enriched GO terms include the MHC class II receptor activity (p-value $= 0.00099$),

chemokine activity (p-value $= 0.00175$), MHC class II protein complex (p-value $= 0.00273$), chemokine receptor binding (p-value $= 0.00404$), and regulation of leukocyte tethering or rolling (p-value $= 0.00030$). The top molecular function (MF) GO term, the MHC class II receptor activity, encoded by human leukocyte antigen (HLA) class II genes, plays important roles in antigen presentation and initiation of immune responses [174].

For comparison, we have performed the same GO enrichment analysis for the top 200 drug-gene interactions inferred by BCCA. The most significantly enriched GO terms are telomere cap complex (p-value $= 0.00016$), nuclear telomere cap complex (p-value $= 0.00016$), positive regulation of telomere maintenance (p-value $= 0.00041$), telomeric DNA binding (p-value $= 0.00044$), and SRP-dependent cotranslational protein targeting to membrane (p-value $= 0.00058$). To the best of our knowledge, these enriched GO terms are not directly related to AML disease mechanisms.

G.3 Negative accuracy threshold for cystic fibrosis (CF) data

While we only discussed the results for one specified threshold value for negative accuracy (97%) in the chapter five for brevity, we here provide additional results with other threshold values, which show similar improvements by BayReL (see Figure G.3). We note that there is a trade-off between positive and negative accuracies, and the optimal point can be chosen depending on the application.

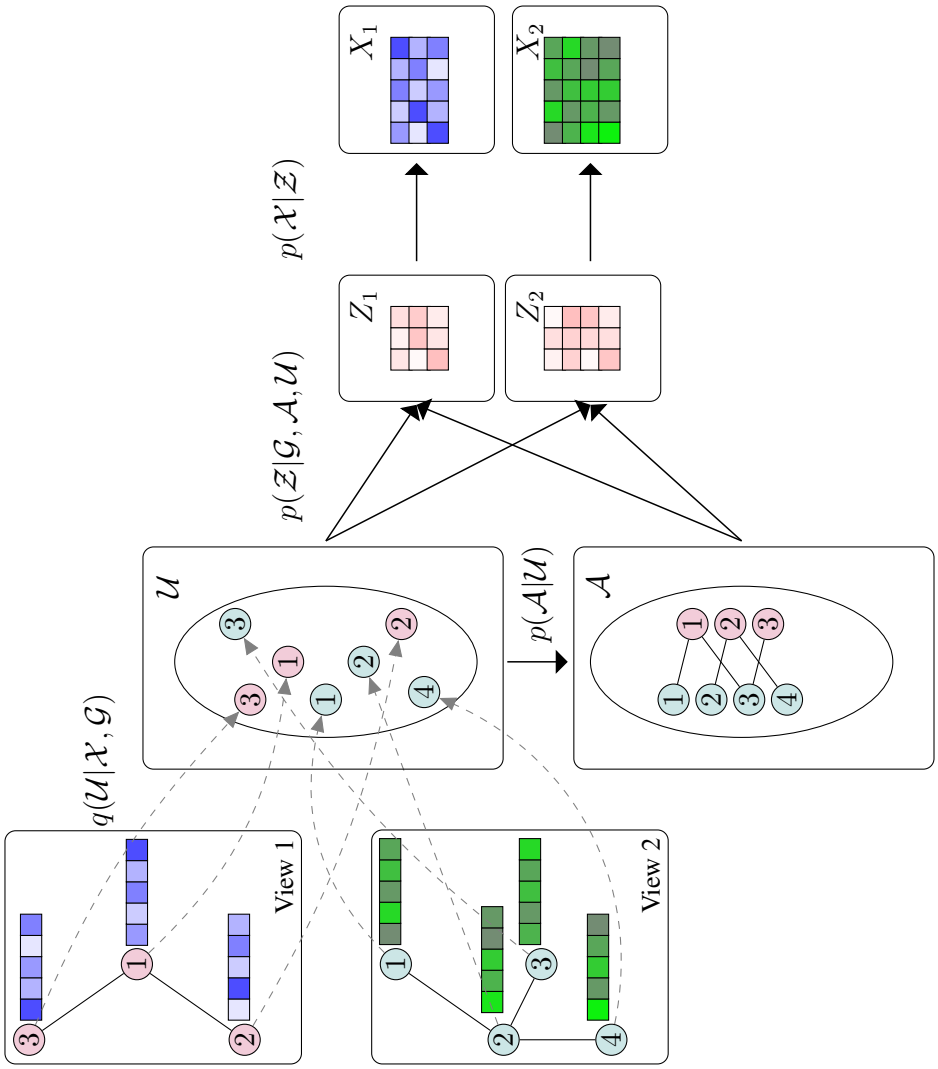


Figure G.1: Schematic illustration of BayReL.

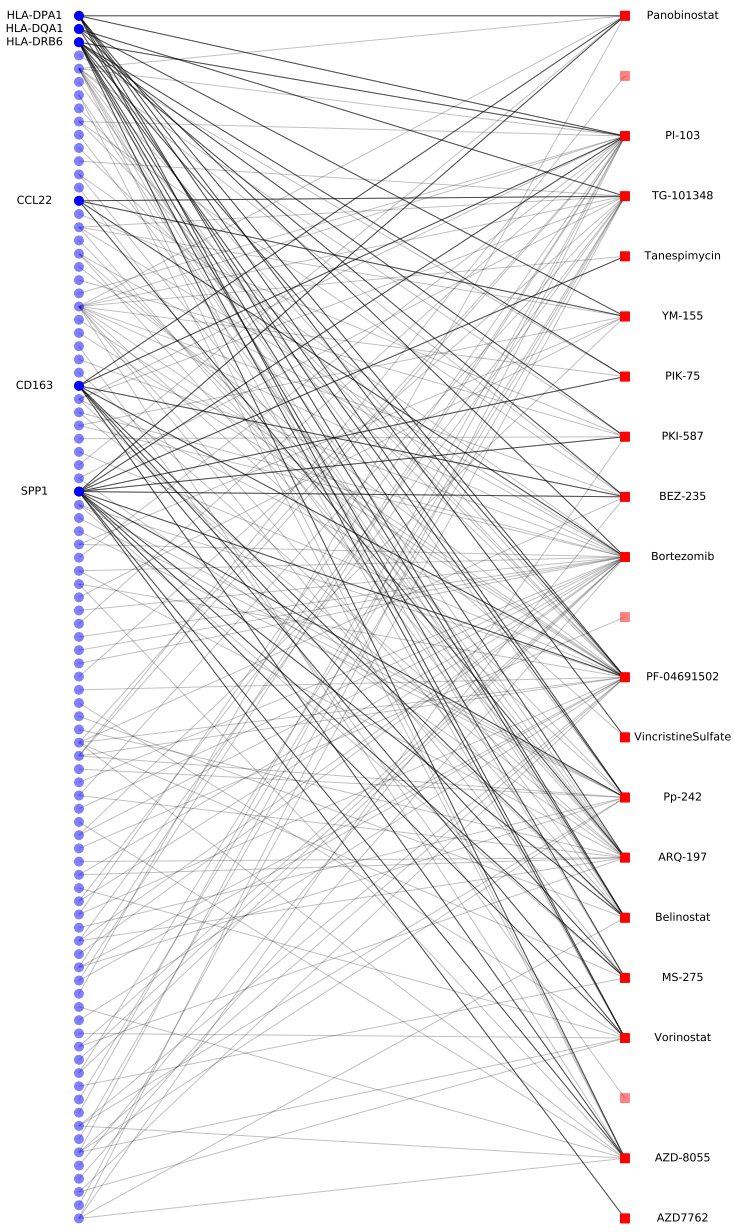


Figure G.2: The bipartite sub-network with the top 200 interactions inferred by BayReL in AML data, where only the top six genes and their associated drugs are labelled in the figure for better visualization. Genes and drugs are shown as blue and red nodes, respectively.

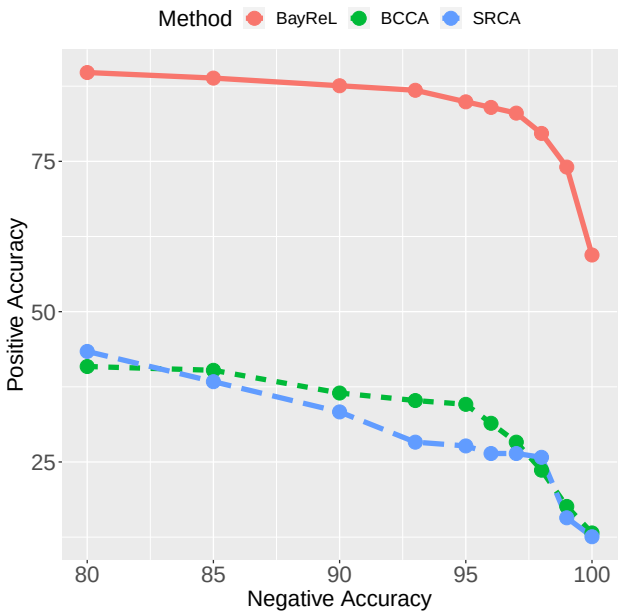


Figure G.3: Positive vs negative accuracy in CF data.

G.4 Details on the experimental setups, hyper-parameter selection, and run time

BayReL. In all cystic fibrosis (CF) and acute myeloid leukemia (AML) experiments, the learning rate is set to be 0.01. For the TCGA breast cancer (BRCA) dataset, the learning rates are 0.0005 when using all and half of samples and 0.005 when using 25% of all samples. We learn the model for 1000 training epochs and use the validation set for early stopping. All of the experiments are run on a single GPU node GeForce RTX 2080. Each training epoch for CF, BRCA, and AML took 0.01, 0.42, and 0.23 seconds, respectively.

BCCA. In all experiments, we used CCAGFA R package as the official implementation of BCCA. We got the default hyper-parameters using the function 'getDefaultOpts' as suggested by the authors. We then construct the bi-partite graph using Spearman's rank correlation between the mean projections of views. We report the results based on four independent runs.

Table G.1: Enriched GO terms for the top 200 interactions in AML data.

GO ID	**GO class**	**Description**	**p-value**
GO:0006084	BP	acetyl-CoA metabolic process	0.00051
GO:0009404	BP	toxin metabolic process	0.00099
GO:0032395	MF	MHC class II receptor activity	0.00099
GO:0033004	BP	negative regulation of mast cell activation	0.00099
GO:0048240	BP	sperm capacitation	0.00099
GO:0008009	MF	chemokine activity	0.00175
GO:0042613	CC	MHC class II protein complex	0.00273
GO:0042379	MF	chemokine receptor binding	0.00404