Variational Autoencoders for Sparse and Overdispersed Discrete Data

He Zhao[*] **Piyush Rai**[†] **Lan Du**[*]

Wray Buntine[*] **Dinh Phung**[*] **Mingyuan Zhou**[‡]

[*]Faculty of Information Technology, Monash University, Australia
[†]Department of Computer Science and Engineering, IIT Kanpur, India
[‡]McCombs School of Business, The University of Texas at Austin, USA

Abstract

Many applications, such as text modelling, high-throughput sequencing, and recommender systems, require analysing sparse, high-dimensional, and overdispersed discrete (count or binary) data. Recent deep probabilistic models based on variational autoencoders (VAE) have shown promising results on discrete data but may have inferior modelling performance due to the insufficient capability in modelling overdispersion and model misspecification. To address these issues, we develop a VAE-based framework using the negative binomial distribution as the data distribution. We also provide an analysis of its properties vis-à-vis other models. We conduct extensive experiments on three problems from discrete data analysis: text analysis/topic modelling, collaborative filtering, and multi-label learning. Our models outperform state-of-the-art approaches on these problems, while also capturing the phenomenon of overdispersion more effectively.[1]

1 INTRODUCTION

Discrete data are ubiquitous in many applications. For example, in text analysis, a collection of documents can be represented as a word-document count matrix; in recommender systems, users' shopping history can be represented as a binary (or count) item-user matrix, with each entry indicating whether or not a user has bought an item (or its purchase count); in extreme multi-label learning problems, data samples can be tagged with a large set of labels, presented as a binary label matrix. Such kinds of data are often characterised by high-dimensionality and extreme sparsity. With the ability to handle high-dimensional and sparse matrices, Probabilistic Matrix Factorisation (PMF) (Mnih and Salakhutdinov, 2008) has been a key method of choice for such problems. PMF generates data from a suitable probability distribution, parameterised by some low-dimensional latent factors. For modelling discrete data, Latent Dirichlet Allocation (LDA) (Blei et al., 2003) and Poisson Factor Analysis (PFA) (Canny, 2004; Zhou et al., 2012) are two representative models that generate data samples using the multinomial and Poisson distributions, respectively. Originally, LDA and PFA can be seen as single-layer models, whose modelling expressiveness may be limited. Several prior works have focused on extending them with hierarchical/deep Bayesian priors (Blei et al., 2010; Paisley et al., 2015; Gan et al., 2015b; Zhou et al., 2016; Lim et al., 2016; Zhao et al., 2018a). However, increasing model complexity with hierarchical priors can also complicate inference, which hinders their usefulness in analysing large-scale data.

The recent success of deep generative models such as Variational Autoencoders (VAEs) (Kingma and Welling, 2014; Rezende et al., 2014) on modelling real-valued data such as images has motivated machine learning practitioners to adapt VAEs to discrete data, as done in recent works (Miao et al., 2016, 2017; Krishnan et al., 2018; Liang et al., 2018). Instead of using the Gaussian distribution as the data distribution for real-valued data, the multinomial distribution has been used for discrete data (Miao et al., 2016; Krishnan et al., 2018; Liang et al., 2018). Following Liang

[1] Code at `https://github.com/ethanhezhao/NBVAE`

Proceedings of the 23$^{\text{rd}}$International Conference on Artificial Intelligence and Statistics (AISTATS) 2020, Palermo, Italy. PMLR: Volume 108.

et al. (2018), we refer to these VAE-based models as "MultiVAE" (Multi for multinomial)[2]. MultiVAE can be viewed as a deep nonlinear PMF model, with nonlinearity modelled by deep neural networks. Compared with conventional hierarchical Bayesian models, MultiVAE enjoys better modelling capacity without sacrificing the scalability, with the use of amortized variational inference (AVI) (Rezende et al., 2014). In this work, we address some key shortcomings of these existing models; in particular, **1)** insufficient capability of modelling overdispersion in count-valued data, and **2)** model misspecification for binary data.

Specifically, *overdispersion* (i.e., variance larger than mean) describes the phenomenon that the data variability is large, a key aspect in large-scale count-valued data. For example, overdispersion in text data can behave as *word burstiness* (Church and Gale, 1995; Madsen et al., 2005; Doyle and Elkan, 2009; Buntine and Mishra, 2014): When a word is seen in a document, it may excite both itself and related ones. Burstiness can cause overdispersion in text data because in a document, it is common that a few bursty words occur multiple times while other words only show up once or never, resulting in high variance in the word counts of the document. As shown in Zhou (2018), the deep-seated cause of the insufficient capability of modelling overdispersion in existing PMF models is their limited ability to handle *self-* and *cross-excitation*. Specifically, in the text data example, self-excitation captures the effect that if a word occurs in a document, it is likely to occur more times in the same document; while cross-excitation models the effect that if a word such as "puppy" occurs, it will likely excite the occurrences of the related words such as "dog." It can be shown that existing PMF models (including VAEs) with Poisson/multinomial likelihood cannot explicitly capture self- and cross-excitations.

Besides count-valued observations, binary-valued data are also prevalent in many applications, such as in collaborative filtering and graph analysis. It may not be proper to directly use multinomial or Poisson to model binary data, which is a common misspecification in many existing models. This is because a multinomial or Poisson may assign more than one count to one position, ignoring the binary nature of data. The misspecification could result in inferior modelling performance (Zhou, 2015; Zhao et al., 2017a).

To address the aforementioned challenges, we propose a Bayesian approach using the negative binomial (NB) distribution as the data distribution in a VAE-based deep generative model, so as to exploit its power in learning nonlinearity from high-dimensional and complex data spaces. We show that using NB as the likelihood can explicitly capture self-excitation, which existing VAE-based methods for discrete data are unable to deal with properly. The use of deep structures further boosts our model capacity for capturing cross-excitation. By sufficiently capturing both kinds of excitations, our proposed method is able to better handle overdispersion. Moreover, the use of NB (instead of multinomial as in Liang et al. (2018); Krishnan et al. (2018)) facilitates developing a link function between the Bernoulli and NB distributions, which enjoys better modelling performance and efficiency for binary data. Putting this together, our resulting **N**egative-**B**inomial **V**ariational **A**uto**E**ncoder (**NBVAE** for short) is a VAE-based framework generating data with a NB distribution. This can be efficiently trained and achieves superior performance on various tasks on discrete data, including text analysis, collaborative filtering, and multi-label learning.

2 PROPOSED METHOD

We start with the introduction of our proposed NBVAE model for count-valued data in Section 2.1, and then give a detailed analysis on how self- and cross-excitations are captured in different models and why NBVAE is capable of better handling them in Section 2.2. Finally, we describe the variants of NBVAE for modelling binary data and for multi-label learning in Sections 2.3 and 2.4, respectively.

2.1 Negative-Binomial Variational Autoencoder (NBVAE)

Like the standard VAE model, NBVAE consists of two components: a decoder for the generative process and an encoder for the inference process. Here we focus on the generative process and discuss the inference procedure in Section 3. Without loss of generality, we present our model in text analysis on word counts, but the model can work with any kind of count-valued matrices. Suppose the word counts of a text corpus are stored in a $V \times N$ count matrix $\mathbf{Y} \in \mathbb{N}^{V\times N} = [\boldsymbol{y}_1, \cdots, \boldsymbol{y}_N]$, where $\mathbb{N} = \{0, 1, 2, \cdots\}$, N and V are the number of documents and size of the vocabulary, respectively. To generate the occurrences of the words for the j^{th} ($j \in \{1, \cdots N\}$) document, $\boldsymbol{y}_j \in \mathbb{N}^V$, we draw a K dimensional latent representation $\boldsymbol{z}_j \in \mathbb{R}^K$ from a standard multivariate normal prior. Given $\boldsymbol{z}_j$, $\boldsymbol{y}_j$ is drawn from a (multivariate) negative-binomial distribution with $\boldsymbol{r}_j \in \mathbb{R}_+^V$ ($\mathbb{R}_+ = \{x : x \geq 0\}$) and $\boldsymbol{p}_j \in (0, 1)^V$ as the parameters. Moreover, $\boldsymbol{r}_j$ and $\boldsymbol{p}_j$ are obtained by transforming $\boldsymbol{z}_j$ from two nonlinear functions, $f_{\theta^r}(\cdot)$ and $f_{\theta^p}(\cdot)$,

[2] The generative process (decoder) in Miao et al. (2016); Krishnan et al. (2018); Liang et al. (2018) is similar, while the inference process (encoder) is different.

parameterised by θ^r and θ^p, respectively. To generate valid NB parameters, the output of $f_{\theta^r}(\cdot)$ and $f_{\theta^p}(\cdot)$ is fed into $\exp(\cdot)$ and $\mathrm{sigmoid}(\cdot)$, respectively. The above generative process can be formulated as follows:

$$\begin{aligned} \boldsymbol{z}_j &\sim \mathcal{N}(\mathbf{0}, \mathbf{I}_K), \\ \boldsymbol{r}_j = \exp\left(f_{\theta^r}(\boldsymbol{z}_j)\right)\ ,\ \boldsymbol{p}_j &= \mathrm{sigmoid}\left(f_{\theta^p}(\boldsymbol{z}_j)\right), \\ \boldsymbol{y}_j &\sim \mathrm{NB}(\boldsymbol{r}_j, \boldsymbol{p}_j). \end{aligned} \tag{1}$$

Due to the equivalence of the draws between a multivariate negative-binomial distribution and a Dirichlet-multinomial distribution (Zhou, 2018, Theorem 1), the proposed NBVAE can be viewed as a deep nonlinear generalisation of the models built upon the Dirichlet-Multinomial distribution, which effectively capture word burstiness in topic modelling (Doyle and Elkan, 2009; Buntine and Mishra, 2014).

2.2 How NBVAE Captures Self- and Cross-Excitations

We now compare our NBVAE model and other PMF models in terms of their ability of capturing self- and cross-excitations in count-valued data. For easy comparison, we present the related models with a unified framework and show our analysis in the case of generating word counts. In general, $\boldsymbol{y}_j$ can be explicitly generated from the data distribution by $\boldsymbol{y}_j \sim \mathrm{p}(\boldsymbol{y}_j \mid \boldsymbol{l}_j)$, where $\boldsymbol{l}_j \in \mathbb{R}_+^V$ is the model parameter. Alternatively, one can first generate the token of each word in document j, $\{w_j^i\}_{i=1}^{y_{\cdot j}}$ ($w_j^i \in \{1, \cdots, V\}$), and then we count the occurrences of different tokens by $y_{vj} = \sum_{i=1}^{y_{\cdot j}} \mathbf{1}(w_j^i = v)$, where $\mathbf{1}(\cdot)$ is the indicator function. It can be seen that the two ways are equivalent. As shown later, in both ways, $\boldsymbol{l}_j$ takes a factorised form. Now if we look at the predictive distribution of a word, w_j^i, conditioned on the other words' occurrences in the corpus, $\mathbf{Y}^{-ij}$, it can be presented as follows:

$$\begin{aligned} &\mathrm{p}(w_j^i = v \mid \mathbf{Y}^{-ij}) \propto \\ \int &\mathrm{p}(w_j^i = v \mid \boldsymbol{l}'_j)\, \mathrm{p}(\boldsymbol{l}'_j \mid \mathbf{Y}^{-ij}) \mathrm{d}\boldsymbol{l}'_j = \\ &\mathbb{E}_{\mathrm{p}(\boldsymbol{l}'_j \mid \mathbf{Y}^{-ij})} \left[\mathrm{p}(w_j^i = v \mid \boldsymbol{l}'_j)\right], \end{aligned} \tag{2}$$

where $\boldsymbol{l}'_j$ is the predictive rate computed with the parameters obtained from the posterior.

With the above notation, we can relate (as special cases) various existing models for discrete data to our proposed NBVAE model, such as PFA (Canny, 2004; Zhou et al., 2012), LDA (Blei et al., 2003), MultiVAE (Miao et al., 2016; Krishnan et al., 2018; Liang et al., 2018), and Negative-Binomial Factor Analysis (NBFA) (Zhou, 2018), as follows:

PFA: It is easy to see that PFA directly fits into this framework, where $\mathrm{p}(\boldsymbol{y}_j \mid \boldsymbol{l}_j)$ is the Poisson distribution and $\boldsymbol{l}_j = \boldsymbol{\Phi}\boldsymbol{\theta}_j$. Here $\boldsymbol{\Phi} \in \mathbb{R}_+^{V \times K} = [\boldsymbol{\phi}_1, \cdots, \boldsymbol{\phi}_K]$ is the factor loading matrix and $\boldsymbol{\Theta} \in \mathbb{R}_+^{K \times N} = [\boldsymbol{\theta}_1, \cdots, \boldsymbol{\theta}_N]$ is the factor score matrix. Their linear combinations determine the probability of the occurrence of v in document j.

LDA: Originally, LDA explicitly assigns a topic $z_j^i \in \{1, \cdots, K\}$ to w_j^i, with the following process: $z_j^i \sim \mathrm{Cat}(\boldsymbol{\theta}_j/\theta_{\cdot j})$ and $w_j^i \sim \mathrm{Cat}(\boldsymbol{\phi}_{z_{ij}})$, where $\theta_{\cdot j} = \sum_k^K \theta_{kj}$ and "Cat" is the categorical distribution. By collapsing all the topics, we can derive an equivalent representation of LDA, in line with the general framework: $\boldsymbol{y}_j \sim \mathrm{Multi}(y_{\cdot j}, \boldsymbol{l}_j)$, where $\boldsymbol{l}_j = \boldsymbol{\Phi}\boldsymbol{\theta}_j/\theta_{\cdot j}$.

MultiVAE: MultiVAE generates data from a multinomial distribution, whose parameters are constructed by the decoder: $\boldsymbol{y}_j \sim \mathrm{Multi}(y_{\cdot j}, \boldsymbol{l}_j)$, where $\boldsymbol{l}_j = \mathrm{softmax}(f_\theta(\boldsymbol{z}_j))$. As shown in Krishnan et al. (2018), MultiVAE can be viewed as a nonlinear PMF model.

NBFA: NBFA uses a NB distribution as the data distribution, the generative process of which can be represented as: $\boldsymbol{y}_j \sim \mathrm{NB}(\boldsymbol{l}_j, p_j)$, where $\boldsymbol{l}_j = \boldsymbol{\Phi}\boldsymbol{\theta}_j$.

The above comparisons on the data distributions and predictive distributions of those models are shown in Table 1. In particular, y_{vj}^{-i} denotes the number of occurrences of word v in document j, excluding the i^{th} word. Now we can show a model's capacity of capturing self- and cross-excitations by analysing its predictive distribution. **1) Self-excitation:** If we compare PFA, LDA, and MultiVAE against NBFA and NBVAE, it can be seen that the latter two models with NB as their data distributions explicitly capture self-excitation via the term y_{vj}^{-i} in the predictive distributions. Specifically, if v appears more in document j, y_{vj}^{-i} will be larger, leading to larger probability that v shows up again. Therefore, the latter three models directly capture word burstiness with y_{vj}^{-i}. However, PFA, LDA, and MultiVAE cannot do this because they predict a word purely based on the interactions of the latent representations and pay less attention to the existing word frequencies. Therefore, even with deep structures, their potential for modelling self-excitation is still limited. **2) Cross-excitation:** For the models with NB, i.e., NBVAE and NBFA, as self-excitation is explicitly captured by y_{vj}^{-i}, the interactions of the latent factors are only responsible to model cross-excitation. Specifically, NBFA applies a single-layer linear combination of the latent representations, i.e., $\sum_k^K \phi_{vk}\theta_{kj}$, while NBVAE can be viewed as a deep extension of NBFA, using deep neural networks to conduct multi-layer nonlinear combinations of the latent representations, i.e, $r_{vj} = \exp\left(f_{\theta^r}(\boldsymbol{z}_j)\right)_v$ and $p_{vj} = \mathrm{sigmoid}\left(f_{\theta^p}(\boldsymbol{z}_j)\right)_v$. Therefore, NBVAE enjoys richer modelling capacity than NBFA on capturing cross-excitation. **3) Summary:** We summarise our

Table 1: Comparison of the data distributions, model parameters, predictive rates, and posteriors. q(·) denotes the encoder in VAE models, which will be introduced in Section 3.

Model	Data distribution	Model parameter	Predictive rate	Posterior
PFA	$\boldsymbol{y}_j \sim \text{Poisson}(\boldsymbol{l}_j)$	$\boldsymbol{l}_j = \boldsymbol{\Phi}\boldsymbol{\theta}_j$	$l'_{vj} \propto \sum_k^K \phi_{vk}\theta_{kj}$	$\boldsymbol{\Phi}, \boldsymbol{\theta}_j \sim \text{p}(\boldsymbol{\Phi}, \boldsymbol{\theta}_j \mid \mathbf{Y}^{-ij})$
LDA	$\boldsymbol{y}_j \sim \text{Multi}(y_{\cdot j}, \boldsymbol{l}_j)$	$\boldsymbol{l}_j = \boldsymbol{\Phi}\boldsymbol{\theta}_j / \theta_{\cdot j}$	$l'_{vj} \propto \sum_k^K \phi_{vk}\theta_{kj}/\theta_{\cdot j}$	$\boldsymbol{\Phi}, \boldsymbol{\theta}_j \sim \text{p}(\boldsymbol{\Phi}, \boldsymbol{\theta}_j \mid \mathbf{Y}^{-ij})$
MultiVAE	$\boldsymbol{y}_j \sim \text{Multi}(y_{\cdot j}, \boldsymbol{l}_j)$	$\boldsymbol{l}_j = \text{softmax}(f_\theta(\boldsymbol{z}_j))$	$l'_{vj} \propto \text{softmax}(f_\theta(\boldsymbol{z}_j))_v$	$\boldsymbol{z}_j \sim \text{q}(\boldsymbol{z}_j \mid \mathbf{Y}^{-ij})$
NBFA	$\boldsymbol{y}_j \sim \text{NB}(\boldsymbol{l}_j, p_j)$	$\boldsymbol{l}_j = \boldsymbol{\Phi}\boldsymbol{\theta}_j$	$l'_{vj} \propto (y_{vj}^{-i} + \sum_k^K \phi_{vk}\theta_{kj})p_j$	$\boldsymbol{\Phi}, \boldsymbol{\theta}_j, p_j \sim \text{p}(\boldsymbol{\Phi}, \boldsymbol{\theta}_j, p_j \mid \mathbf{Y}^{-ij})$
NBVAE	$\boldsymbol{y}_j \sim \text{NB}(\boldsymbol{r}_j, \boldsymbol{p}_j)$	$\boldsymbol{r}_j = \exp(f_{\theta^r}(\boldsymbol{z}_j))$ $\boldsymbol{p}_j = \text{sigmoid}\,(f_{\theta^p}(\boldsymbol{z}_j)$	$l'_{vj} \propto (y_{vj}^{-i} + \exp(f_{\theta^r}(\boldsymbol{z}_j))_v) \cdot \text{sigmoid}\,(f_{\theta^p}(\boldsymbol{z}_j)_v$	$\boldsymbol{z}_j \sim \text{q}(\boldsymbol{z}_j \mid \mathbf{Y}^{-ij})$

Table 2: How self- and cross-excitation are captured.

Model	Self-excitation	Cross-excitation
PFA	Single-layer structure	
LDA	Single-layer structure	
MultiVAE	Multi-layer neural networks	
NBFA	y_{vj}^{-i}	Single-layer structure
NBVAE	y_{vj}^{-i}	Multi-layer neural networks

analysis on how self- and cross-excitations are captured in related models in Table 2.

2.3 NBVAE for Binary Data

In many problems, discrete data are in binary form. For example, suppose a binary matrix $\mathbf{Y} \in \{0,1\}^{V \times N}$ stores the buying history of N users on V items, where $y_{vj} = 1$ indicates that user j has brought item v, and vice versa. Previous models like MultiVAE (Krishnan et al., 2018; Liang et al., 2018) treat such binary data as counts, which is a model misspecification and is likely to result in inferior performance. To better model binary data, we develop a simple yet effective method that links the generative process of NBVAE with the Bernoulli distribution. Specifically, inspired by the link function used in Zhou (2015), we first generate a latent discrete intensity vector, $\boldsymbol{m}_j \in \mathbb{N}^V$, from the generative process of NBVAE. Next, we generate the binary vector, $\boldsymbol{y}_j$ by thresholding $\boldsymbol{m}_j$ at one as:

$$
\begin{aligned}
\boldsymbol{z}_j &\sim \mathcal{N}(\mathbf{0}, \mathbf{I}_K), \\
\boldsymbol{r}_j = \exp\left(f_{\theta^r}(\boldsymbol{z}_j)\right)\ &,\ \boldsymbol{p}_j = \text{sigmoid}\left(f_{\theta^p}(\boldsymbol{z}_j)\right), \\
\boldsymbol{m}_j \sim \text{NB}(\boldsymbol{r}_j, \boldsymbol{p}_j)\ &,\ \boldsymbol{y}_j = \mathbf{1}(\boldsymbol{m}_j \geq 1).
\end{aligned} \tag{3}
$$

Intuitively, m_{vj} can be viewed as the (latent) interest of user j on item v and the user will buy this item if and only if $m_{vj} > 0$. As $\boldsymbol{m}_j$ is drawn from NB, we do not have to explicitly generate it. We can marginalise it out and get the following data likelihood: $\boldsymbol{y}_j \sim \text{Bernoulli}\left(1 - (1 - \boldsymbol{p}_j)^{\boldsymbol{r}_j}\right)$, where $\boldsymbol{r}_j$ and $\boldsymbol{p}_j$ have the same construction of the original NBVAE. Here we refer to this extension of NBVAE as NBVAE$_\text{b}$ (b for binary). Note that the NB distribution is a gamma mixed Poisson distribution, so the elements of $\boldsymbol{m}_j$ (from a multivariate NB) can be viewed to be individually generated from a Poisson distribution. This is an important property that makes the link function applicable to NBVAE. In contrast, it is inapplicable to MutiVAE (Liang et al., 2018; Krishnan et al., 2018) as multinomial generates elements dependently.

2.4 NBVAE for Multi-label Learning

Our model admits easy extensions for other problems that require modeling discrete data. In particular, we show how to extend NBVAE for solving the multi-label learning problem, which can be formulated into a supervised task of modelling binary matrices. In a multi-label learning task, there is a large set of labels in a dataset while an individual sample is associated with a small subset of the labels. Suppose there are N samples, each of which is associated with a D dimensional feature vector $\boldsymbol{x}_j \in \mathbb{R}^D$ and a binary label vector $\boldsymbol{y}_j \in \{0,1\}^V$. V is the number of labels that can be very large, and $y_{vj} = 1$ indicates sample j is labelled with v. The goal is to predict the labels of a sample given its features. Here the label matrix $\mathbf{Y} \in \{0,1\}^{V \times N}$ is a large-scale, sparse, binary matrix.

Inspired by the idea of Conditional VAE (Sohn et al., 2015) and corresponding linear methods (Zhao et al., 2017a,b), we develop a conditional version of NBVAE, named NBVAE$_\text{c}$ (c for conditional), for extreme multi-label learning[3]. The general idea is that instead of drawing the latent representation of a sample from an uninformative prior (i.e., standard normal) as in NBVAE$_\text{b}$, we use a prior constructed with the sample's feature in NBVAE$_\text{c}$. Specifically, we introduce a parametric function $f_\psi(\cdot)$ to transform the features of sample j to the mean and variance of the normal prior, formulated as follows:

$$
\begin{aligned}
\boldsymbol{\mu}_j, \boldsymbol{\sigma}_j = f_\psi(\boldsymbol{x}_j), &\boldsymbol{z}_j \sim \mathcal{N}(\boldsymbol{\mu}_j, \text{diag}\{\boldsymbol{\sigma}_j^2\}), \\
\boldsymbol{r}_j = \exp\left(f_{\theta^r}(\boldsymbol{z}_j)\right), &\boldsymbol{p}_j = \text{sigmoid}\left(f_{\theta^p}(\boldsymbol{z}_j)\right), \\
\boldsymbol{m}_j \sim \text{NB}(\boldsymbol{r}_j, \boldsymbol{p}_j), &\boldsymbol{y}_j = \mathbf{1}(\boldsymbol{m}_j \geq 1).
\end{aligned} \tag{4}
$$

Note that $f_\psi(\cdot)$ defines $\text{p}(\boldsymbol{z}_j \mid \boldsymbol{x}_j)$, which encodes the

[3] Other approaches that incorporate supervised/conditional information in VAEs can also be used in our model. The main purpose here is to demonstrate our model's appealing potential for modelling binary data as well its model flexibility.

features of a sample into the prior of its latent representation. Intuitively, we name it the *feature encoder*. With the above construction, given the feature vector of a testing sample j^*, we can feed $\boldsymbol{x}_{j^*}$ into the feature encoder to sample the latent representation, $\boldsymbol{z}_{j^*}$, then feed it into the decoder to predict its labels.

3 VARIATIONAL INFERENCE

The inference of NBVAE and NBVAE_b follows the standard amortized variational inference procedure of VAEs, where a data-dependent variational distribution $\text{q}(\boldsymbol{z}_j \mid \boldsymbol{y}_j)$ (i.e., encoder) is used to approximate the true posterior, $\text{p}(\boldsymbol{z}_j \mid \boldsymbol{y}_j)$, as follows:

$$\widetilde{\boldsymbol{\mu}}_j, \widetilde{\boldsymbol{\sigma}}_j = f_\phi(\boldsymbol{y}_j), \boldsymbol{z}_j \sim \mathcal{N}(\widetilde{\boldsymbol{\mu}}_j, \text{diag}\{\widetilde{\boldsymbol{\sigma}}_j^2\}). \quad (5)$$

Given $\text{q}(\boldsymbol{z}_j \mid \boldsymbol{y}_j)$, the learning objective is to maximise the Evidence Lower BOund (ELBO) of the marginal likelihood of the data: $\mathbb{E}_{\text{q}(\boldsymbol{z}_j|\boldsymbol{y}_j)}\left[\log \text{p}(\boldsymbol{y}_j \mid \boldsymbol{z}_j)\right] - \text{KL}\left[\text{q}(\boldsymbol{z}_j \mid \boldsymbol{y}_j) \parallel \text{p}(\boldsymbol{z}_j)\right]$, in terms of the decoder parameters θ^r, θ^p and the encoder parameter ϕ. Here the reparametrization trick (Kingma and Welling, 2014; Rezende et al., 2014) is used to sample from $\text{q}(\boldsymbol{z}_j \mid \boldsymbol{y}_j)$.

For NBVAE_c, there are two differences of inference from the above two models: **1)** Due to the use of the informative prior conditioned on features, the Kullback-Leiber (KL) divergence on the RHS of the ELBO is calculated between two non-standard multivariate normal distributions: $\text{KL}\left[\text{q}(\boldsymbol{z}_j \mid \boldsymbol{y}_j) \parallel \text{p}(\boldsymbol{z}_j \mid \boldsymbol{x}_j)\right]$. Thus, the KL divergence of the ELBO does not only serve as a regularizer for $\text{q}(\boldsymbol{z}_j \mid \boldsymbol{y}_j)$ but also helps learn the feature encoder, $f_\psi(\cdot)$. **2)** In NBVAE_c, the feature encoder plays an important role, as it is the one that generates the latent representations for test samples. To get more opportunities to train it, when sampling $\boldsymbol{z}_j$ in the training phase, we propose to alternatively draw it between $\boldsymbol{z}_j \sim \text{q}(\boldsymbol{z}_j \mid \boldsymbol{y}_j)$ and $\boldsymbol{z}_j \sim \text{p}(\boldsymbol{z}_j \mid \boldsymbol{x}_j)$. This enables the feature encoder to directly contribute to the generation of labels, which improves its performance in the testing phase.

4 RELATED WORK

Probabilistic matrix factorisation models for discrete data. Several well-known models fall into this category, including LDA (Blei et al., 2003) and PFA (Zhou et al., 2012), as well as their hierarchical extensions such as those in Blei et al. (2010); Teh et al. (2012); Paisley et al. (2015); Gan et al. (2015a); Ranganath et al. (2015); Henao et al. (2015); Lim et al. (2016); Zhou et al. (2016); Zhao et al. (2017b, 2018b). Among various models, the closest ones to ours are NBFA (Zhou, 2018) and nonparametric LDA (Buntine and Mishra, 2014), which generate data with the negative-binomial distribution and Dirichlet-multinomial distribution, respectively. Our models can be viewed as a deep generative extensions to them, providing better model expressiveness, flexibility, as well as inference scalability.

VAEs for discrete data. Miao et al. (2016) proposed Neural Variational Document Model (NVDM), which extended the standard VAE with multinomial likelihood for document modelling. Miao et al. (2017) further built a VAE to generate the document-topic distributions in the LDA framework. Srivastava and Sutton (2017) developed an AVI algorithm for the inference of LDA, which can be viewed as a VAE model. Card et al. (2018) introduced a general VAE framework for topic modelling with meta-data. Grønbech et al. (2019) recently proposed a Gaussian mixture VAE with negative-binomial for gene-expression data, which has a different construction to ours and does not consider binary data, multi-label learning, or in-depth analysis. Burkhardt and Kramer (2019) proposed a VAE framework for topic modelling, the latent representations of which are drawn from Dirichlet instead of Gaussian. Krishnan et al. (2018) found that using the standard training algorithm of VAEs in large sparse discrete data may suffer from model underfitting and proposed a stochastic variational inference (SVI) (Hoffman et al., 2013) algorithm initialised by AVI to mitigate this issue. In the collaborative filtering domain, Liang et al. (2018) noticed a similar issue and alleviated it by proposing MultiVAE with a training scheme based on KL annealing (Bowman et al., 2016). Note that NVDM, NFA, and MultiVAE are the closest ones to ours, their generative processes are very similar but their inference procedures are different. NFA is reported to outperform NVDM on text analysis (Krishnan et al., 2018) while MultiVAE is reported to have better performance than NFA on collaborative filtering tasks (Liang et al., 2018). Compared with them, we improve the modelling performance in a different way, i.e., by better capturing self- and cross-excitations so as to better handle overdispersion. Moreover, NFA and MultiVAE use the multinomial likelihood, which may not work well for binary data.

VAEs for multi-label learning. In various applications, such as image/document tagging, recommender system, ad-placement, multi-label learning/extreme classification problems have drawn a significant attention (Yu et al., 2014; Mencia and Fürnkranz, 2008; Yen et al., 2016; Bhatia et al., 2015; Rai et al., 2015; Jain et al., 2016; Prabhu et al., 2018a; You et al., 2018; Prabhu et al., 2018b). Most of the existing state-of-the-art methods adopt either multiple steps of processing of the labels and features or complex optimisation algorithms. In contrast, NBVAE_c achieves comparable performance but with much simpler model structures. To our knowledge, the adaptation of VAEs to the multi-label learning area is rare, because most existing VAE models are unable to deal with large-scale sparse binary label matrices effectively.

5 EXPERIMENTS

In this section, we evaluate the proposed models on three important applications of large-scale discrete data: text analysis, collaborative filtering, and multi-label learning. We run our models multiple times and report the average results. The details of the datasets, evaluation metrics, experimental settings, and running time comparison are provided in the appendix.

5.1 Experiments on Text Analysis

Our first set of experiments is on text analysis. We consider three widely-used corpora (Srivastava et al., 2013; Gan et al., 2015a; Henao et al., 2015; Cong et al., 2017): 20 News Group (20NG), Reuters Corpus Volume (RCV), and Wikipedia (Wiki). Following Wallach et al. (2009), we report per-heldout-word perplexity of all the models, which is a widely-used metric for text analysis. Note that we use the same perplexity calculation for all the compared models, detailed in the appendix. We compare our proposed NBVAE with the following three categories of state-of-the-art models for text analysis: **1)** Bayesian deep extensions of PFA and LDA: Deep latent Dirichlet allocation (DLDA) (Cong et al., 2017), Deep Poisson Factor Modelling (DPFM) (Henao et al., 2015), Deep Poisson Factor Analysis (DPFA) (Gan et al., 2015b) with different kinds of inference algorithms such as Gibbs sampling, stochastic variational inference, and stochastic gradient MCMC (SGMCMC) (Chen et al., 2014); **2)** NBFA (Zhou, 2018), a recently-proposed single-layer factor analysis model with negative-binomial likelihood, where the truncated version is used and the inference is done by Gibbs sampling. **3)** MultiVAE (Liang et al., 2018; Krishnan et al., 2018), a recent VAE model for discrete data with the multinomial distribution as the data distribution. We use the implementation in Liang et al. (2018) for MultiVAE.

The perplexity results are shown in Table 3. Following Gan et al. (2015a); Henao et al. (2015); Cong et al. (2017), we report the performance of DLDA, DPFM, and DPFA with two and/or three hidden layers, which are the best results reported in their papers. For the VAE-based models, we vary the network architecture with one and two hidden layers and vary the depths and widths of the layers, as shown in Table 3. We observe the following from the results: **1)** If we compare NBFA with DLDA, DPFM, and DPFA, it can be seen that modelling self-excitation with the NB distribution in NBFA has a large contribution to the modelling performance. **2)** It can be observed that the single-layer VAE models (i.e., MultiVAEs with one layer) achieve no better results than NBFA. However, when multi-layer structures were used, VAE models largely improve their performance. This shows the increased model capacity with deeper neural networks is critical to getting better modelling performance via cross-excitation. **3)** Most importantly, our proposed NBVAE significantly outperforms all the other models, which demonstrates the necessity of modelling self-excitation explicitly and modelling cross-excitation with deep structures.

Table 3: Perplexity comparisons. "Layers" indicate the architecture of the hidden layers (for VAE models, it is the hidden layer architecture of the encoder.). Best results are in boldface. TLASGR and SGNHT are the algorithms of SGMCMC, detailed in the papers of DLDA (Cong et al., 2017) and DPFA (Gan et al., 2015a). Some results of the models with Gibbs sampling on RCV and Wiki are not reported because of the scalability issue. All the experimental settings here are consistent with those in Gan et al. (2015a); Henao et al. (2015); Cong et al. (2017).

Model	Inference	Layers	20NG	RCV	Wiki
DLDA	TLASGR	128-64-32	757	815	786
DLDA	Gibbs	128-64-32	752	802	-
DPFM	SVI	128-64	818	961	791
DPFM	MCMC	128-64	780	908	783
DPFA-SBN	Gibbs	128-64-32	827	-	-
DPFA-SBN	SGNHT	128-64-32	846	1143	876
DPFA-RBM	SGNHT	128-64-32	896	920	942
NBFA	Gibbs	128	690	702	-
MultiVAE	VAE	128-64	746	632	629
MultiVAE	VAE	128	772	786	756
NBVAE	VAE	128-64	**688**	**579**	**464**
NBVAE	VAE	128	714	694	529

Table 4: Perplexity comparisons with larger layer width.

Model	Inference	Layers	RCV	Wiki
DLDA	TLASGR	256-128-64	710	682
NBFA	Gibbs	256	649	-
MultiVAE	VAE	256-128	587	589
NBVAE	VAE	256-128	535	451
DLDA	TLASGR	512-256-128	656	602
MultiVAE	VAE	512-256	552	558
NBVAE	VAE	512-256	**512**	**445**

In Table 3, we use 128 as the maximum layer width, following Gan et al. (2015a); Henao et al. (2015); Cong et al. (2017). Given the sizes of RCV and Wiki, we increase the maximum to 256 and 512 to study how the width of layer matters, the results of which are shown in Table 4. By looking at Tables 3 and 4, we find: **1)** With the increased layer widths, all the models gained significant improvements; **2)** Among all the models,

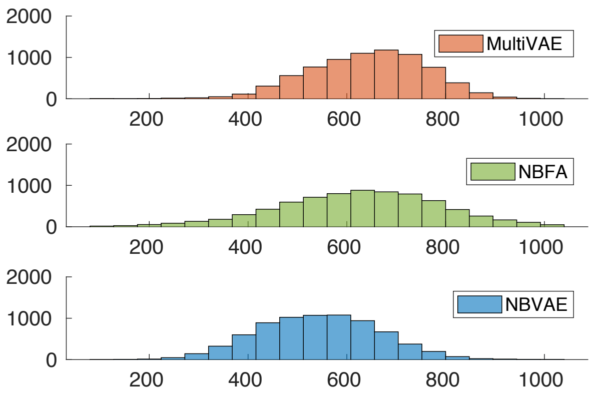


Figure 1: Comparisons of the entropy histograms on the 20NG dataset with 2,000 as the vocabulary and 7,531 test documents. The horizontal axis: the value of entropy. The vertical axis: the number of the test documents that are with a specific entropy value.

regardless of each layer's width, NBVAE outperforms the others with a significant margin; **3)** With smaller model structures (e.g., [256-128]), NBVAE is able to achieve comparable results to the other models with larger structures (e.g., MultiVAE [512-256]), demonstrating our models' expressiveness.

To explicitly show how self-excitation is captured in our model, we compare the entropy of the predictive distributions on the test documents of 20NG. The entropy of document j given the predictive distribution parameterised with $\boldsymbol{l}'_j$ is computed as follows:

$$\text{entropy}_j = \exp\left(-\sum_{v}^{V} l'_{vj}\log(l'_{vj})\right), \tag{6}$$

where $\boldsymbol{l}'_j$ is assumed to be normalised and is computed model-specifically according to Table 1. The intuition here is that, given a model, the entropy of the predictive distribution of a document can be interpreted as the effective number of unique words that the document is expected to focus on. Therefore, if a model takes self-excitation into account, the entropy of a document is expected to be small, as the model's predictive distribution will put large mass on the words that already occur in the document. In contrast, if a model does not take into account self-excitation, its predictive distribution of a document would relatively spread over all the words. Essentially, it means that a smaller entropy indicates better capture of self-excitation. After computing the entropy of all the test documents, we plot the histograms in Figure 1. It can be observed that, in our proposed NBVAE, the centre of the histogram is closer to the origin of coordinates, which shows that our model has more documents that with small entropy. This is because self-excitation is better captured by our model.

5.2 Experiments on Collaborative Filtering

Our second set of experiments is on collaborative filtering, where the task is to recommend items to users using their clicking history. We evaluate our models' performance on four user-item consumption datasets: MovieLens-10M (ML-10M), MovieLens-20M (ML-20M), Netflix Prize (Netflix), and Million Song Dataset (MSD) (Bertin-Mahieux et al., 2011). Following Liang et al. (2018), we report two evaluation metrics: Recall@R and the truncated normalized discounted cumulative gain (NDCG@R), detailed in the appendix. As datasets used here are binary, we compared NBVAE$_\text{b}$, with the recent VAE models: **1)** MultiVAE. **2)** MultiDAE (Liang et al., 2018), a denoising autoencoder (DAE) with multinomial likelihood, which introduces dropout (Srivastava et al., 2014) at the input layer. MultiVAE and MultiDAE are state-of-the-art VAE models for collaborative filtering and they have been reported to outperform several recent advances such as Wu et al. (2016) and He et al. (2017). The experimental settings are consistent with those in Liang et al. (2018), detailed in the appendix.

Figure 2 shows the NDCG@R and Recall@R of the models on the four datasets, where we used $R \in \{1, 5, 10, 20, 50\}$. In general, our proposed NBVAE$_\text{b}$ outperforms the baselines (i.e., MulitVAE and MultiDAE) on almost all of the datasets, In particular, the margin is notably large while the R value is small, such as 1 or 5. It indicates that the top-ranked items in NBVAE$_\text{b}$ are always more accurate than those in MultiVAE and MuliDAE. This fact is also supported by the large gap of NDCG@R between NBVAE$_\text{b}$ and the two baselines, as NDCG@R penalises the true items that are ranked low.

To further demonstrate the benefit of using NBVAE$_\text{b}$, we compare it with NBVAE on the ML-10M dataset, where the latter treats binary data as count-valued data. The results of NDCG@R and Recall@R on ML-10M are shown in Table 5, where NBVAE$_\text{b}$'s results are significantly better than NBVAE, showing the necessity of dealing with binary data separately from count-valued data.

5.3 Experiments on Multi-Label Learning

Finally, we compare NBVAE$_\text{c}$ with several recent advances in multi-label learning, including LEML (Yu et al., 2014), PfastreXML (Jain et al., 2016), PD-Sparse (Yen et al., 2016), and GenEML (Jain et al., 2017). We use three multi-label learning benchmark datasets: Delicious (Tsoumakas et al., 2008), Mediamill (Snoek et al., 2006), and EURLex (Mencia and Fürnkranz, 2008). We report Precision@R ($R \in \{1, 3, 5\}$), which is a widely-used evaluation metric for

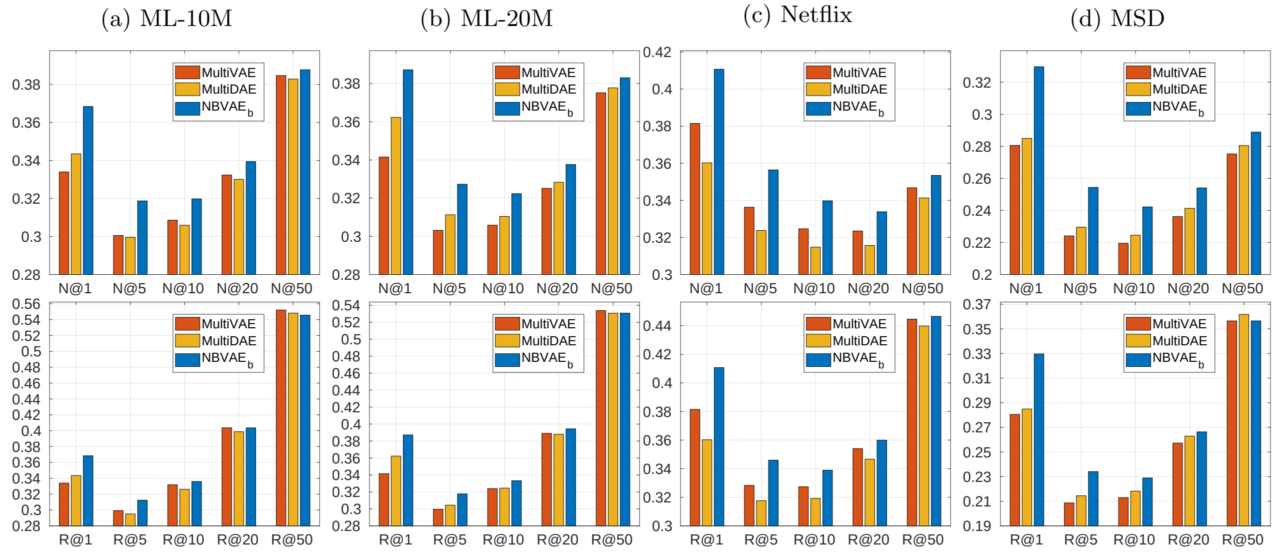


Figure 2: Comparisons of NDCG@R (N@R) and Recall$R$ (R@R). Standard errors in multiple runs are generally less than 0.003 for all the models on all the datasets, which are too tiny to show in the figures.

Table 5: NDCG@R (N@R) and Recall$R$ (R@R) of NBVAE and its variants on ML-10M. Best results are in boldface.

Model	N@1	N@5	N@10	N@20	N@50
NBVAE	0.3333	0.2951	0.3012	0.3263	0.3788
NBVAE$_b$	**0.3684**	**0.3187**	**0.3198**	**0.3394**	**0.3878**
Model	R@1	R@5	R@10	R@20	R@50
NBVAE	0.3333	0.2927	0.3224	0.3968	0.5453
NBVAE$_b$	**0.3684**	**0.3124**	**0.3360**	**0.4039**	**0.5456**

Table 6: Precision (P@R). Best results for each dataset are in boldface. The standard errors of our model are computed in multiple runs. The results of GenEML on Delicious are not reported due to the unavailability.

Datasets	Metric	LEML	PfastreXML	PD-Sparse	GenEML	NBVAE$_c$
Delicious	P@1	65.67	67.13	51.82	-	**68.49**±0.39
	P@3	60.55	**63.48**	46.00	-	62.83±0.47
	P@5	56.08	**60.74**	42.02	-	58.04±0.31
Mediamill	P@1	84.01	83.98	81.86	87.15	**88.27**±0.24
	P@3	67.20	67.37	62.52	69.9	**71.47**±0.18
	P@5	52.80	53.02	45.11	55.21	**56.76**±0.26
EURLex	P@1	63.40	75.45	76.43	77.75	**78.28**±0.49
	P@3	50.35	62.70	60.37	63.98	**66.09**±0.17
	P@5	41.28	52.51	49.72	53.24	**55.47**±0.15

multi-label learning, following Jain et al. (2017). For the baselines, the reported results are the publicly known best ones.

Table 6 shows the performance comparisons on the multi-label learning datasets. It can be observed that the proposed NBVAE$_c$ outperforms the others on the prediction precision, showing its promising potential on multi-label learning problems. Note that the baselines are specialised to the multi-label learning problem, many of which either take multiple steps of processing of the labels and features or use complex optimisation algorithms. Compared with those models, the model simplicity of NBVAE$_c$ is an appealing advantage, which gives VAE models great potential in multi-label learning and related problems.

6 CONCLUSION

In this paper, we have focused on analysing and addressing two shortcomings of probabilistic modelling on large-scale, sparse, discrete data: insufficient capability of modelling overdispersion in count-valued data and model misspecification in binary data. To address this we use the NB distribution, which conquers overdispersion in count-valued data by better capturing self- and cross-excitations as well as deals with binary data in a proper and effective way.

The advantages of our methods on count-valued and binary data have been demonstrated by their superior performance on text analysis and collaborative filtering. Remarkably, due to the great model capacity and flexibility of modelling binary data, our model can be extended to multi-label learning, obtaining state-of-the-art performance in the comparison with many advanced custom-designed algorithms. Extensive experiments have shown the great potential of our models on modelling various kinds of discrete data.

Finally, we believe that our methods are excellent examples of hybridising the valuable knowledge in Bayesian probabilistic modelling with newly-developed deep learning techniques, which would inspire considerable future work in various areas.

Acknowledgements: MZ was supported in part by NSF IIS-1812699. PR acknowledges support from Visvesvaraya Young Faculty Fellowship. DP gratefully acknowledges funding support from the Australian Research Council under the ARC DP160109394 Project.

References

T. Bertin-Mahieux, D. P. Ellis, B. Whitman, and P. Lamere. The million song dataset. In *International Conference on Music Information Retrieval*, 2011.

K. Bhatia, H. Jain, P. Kar, M. Varma, and P. Jain. Sparse local embeddings for extreme multi-label classification. In *NIPS*, pages 730–738, 2015.

D. M. Blei, A. Y. Ng, and M. I. Jordan. Latent Dirichlet allocation. *JMLR*, 3:993–1022, 2003.

D. M. Blei, T. L. Griffiths, and M. I. Jordan. The nested Chinese restaurant process and Bayesian nonparametric inference of topic hierarchies. *Journal of the ACM*, 57(2):7, 2010.

S. R. Bowman, L. Vilnis, O. Vinyals, A. Dai, R. Jozefowicz, and S. Bengio. Generating sentences from a continuous space. In *CoNLL*, pages 10–21, 2016.

W. L. Buntine and S. Mishra. Experiments with nonparametric topic models. In *SIGKDD*, pages 881–890, 2014.

S. Burkhardt and S. Kramer. Decoupling sparsity and smoothness in the Dirichlet variational autoencoder topic model. *JMLR*, 20(131):1–27, 2019.

J. Canny. GaP: a factor model for discrete data. In *SIGIR*, pages 122–129, 2004.

D. Card, C. Tan, and N. A. Smith. Neural models for documents with metadata. In *ACL*, pages 2031–2040, 2018.

T. Chen, E. Fox, and C. Guestrin. Stochastic gradient Hamiltonian Monte Carlo. In *ICML*, pages 1683–1691, 2014.

K. W. Church and W. A. Gale. Poisson mixtures. *Natural Language Engineering*, 1(2):163–190, 1995.

Y. Cong, B. Chen, H. Liu, and M. Zhou. Deep latent Dirichlet allocation with topic-layer-adaptive stochastic gradient Riemannian MCMC. In *ICML*, pages 864–873, 2017.

G. Doyle and C. Elkan. Accounting for burstiness in topic models. In *ICML*, pages 281–288, 2009.

Z. Gan, C. Chen, R. Henao, D. Carlson, and L. Carin. Scalable deep Poisson factor analysis for topic modeling. In *ICML*, pages 1823–1832, 2015a.

Z. Gan, R. Henao, D. Carlson, and L. Carin. Learning deep sigmoid belief networks with data augmentation. In *AISTATS*, pages 268–276, 2015b.

C. H. Grønbech, M. F. Vording, P. N. Timshel, C. K. Sønderby, T. H. Pers, and O. Winther. scVAE: Variational auto-encoders for single-cell gene expression data. *bioRxiv*, page 318295, 2019.

X. He, L. Liao, H. Zhang, L. Nie, X. Hu, and T.-S. Chua. Neural collaborative filtering. In *WWW*, pages 173–182, 2017.

R. Henao, Z. Gan, J. Lu, and L. Carin. Deep Poisson factor modeling. In *NIPS*, pages 2800–2808, 2015.

M. D. Hoffman, D. M. Blei, C. Wang, and J. Paisley. Stochastic variational inference. *JMLR*, 14(1):1303–1347, 2013.

H. Jain, Y. Prabhu, and M. Varma. Extreme multi-label loss functions for recommendation, tagging, ranking & other missing label applications. In *KDD*, 2016.

V. Jain, N. Modhe, and P. Rai. Scalable generative models for multi-label learning with missing labels. In *ICML*, pages 1636–1644, 2017.

D. P. Kingma and M. Welling. Auto-encoding variational Bayes. In *ICLR*, 2014.

R. Krishnan, D. Liang, and M. Hoffman. On the challenges of learning with inference networks on sparse, high-dimensional data. In *AISTATS*, pages 143–151, 2018.

D. Liang, R. G. Krishncan, M. D. Hoffman, and T. Jebara. Variational autoencoders for collaborative filtering. In *WWW*, pages 689–698, 2018.

K. W. Lim, W. Buntine, C. Chen, and L. Du. Nonparametric Bayesian topic modelling with the hierarchical Pitman-Yor processes. *International Journal of Approximate Reasoning*, 78:172–191, 2016.

R. E. Madsen, D. Kauchak, and C. Elkan. Modeling word burstiness using the Dirichlet distribution. In *ICML*, pages 545–552, 2005.

E. L. Mencia and J. Fürnkranz. Efficient pairwise multilabel classification for large-scale problems in the legal domain. In *ECML/PKDD*, pages 50–65, 2008.

Y. Miao, L. Yu, and P. Blunsom. Neural variational inference for text processing. In *ICML*, pages 1727–1736, 2016.

Y. Miao, E. Grefenstette, and P. Blunsom. Discovering discrete latent topics with neural variational inference. In *ICML*, pages 2410–2419, 2017.

A. Mnih and R. R. Salakhutdinov. Probabilistic matrix factorization. In *NIPS*, pages 1257–1264, 2008.

J. Paisley, C. Wang, D. M. Blei, and M. I. Jordan. Nested hierarchical Dirichlet processes. *TPAMI*, 37 (2):256–270, 2015.

Y. Prabhu, A. Kag, S. Gopinath, K. Dahiya, S. Harsola, R. Agrawal, and M. Varma. Extreme multi-label learning with label features for warm-start tagging, ranking & recommendation. In *WSDM*, 2018a.

Y. Prabhu, A. Kag, S. Harsola, R. Agrawal, and M. Varma. Parabel: Partitioned label trees for extreme classification with application to dynamic search advertising. In *WWW*, 2018b.

P. Rai, C. Hu, R. Henao, and L. Carin. Large-scale bayesian multi-label learning via topic-based label embeddings. In *NIPS*, 2015.

R. Ranganath, L. Tang, L. Charlin, and D. Blei. Deep exponential families. In *AISTATS*, pages 762–771, 2015.

D. J. Rezende, S. Mohamed, and D. Wierstra. Stochastic backpropagation and approximate inference in deep generative models. In *ICML*, pages 1278–1286, 2014.

C. G. Snoek, M. Worring, J. C. Van Gemert, J.-M. Geusebroek, and A. W. Smeulders. The challenge problem for automated detection of 101 semantic concepts in multimedia. In *ACM MM*, pages 421–430, 2006.

K. Sohn, H. Lee, and X. Yan. Learning structured output representation using deep conditional generative models. In *NIPS*, pages 3483–3491. 2015.

A. Srivastava and C. Sutton. Autoencoding variational inference for topic models. In *ICLR*, 2017.

N. Srivastava, R. Salakhutdinov, and G. Hinton. Modeling documents with a deep Boltzmann machine. In *UAI*, pages 616–624, 2013.

N. Srivastava, G. Hinton, A. Krizhevsky, I. Sutskever, and R. Salakhutdinov. Dropout: a simple way to prevent neural networks from overfitting. *JMLR*, 15 (1):1929–1958, 2014.

Y. Teh, M. Jordan, M. Beal, and D. Blei. Hierarchical Dirichlet processes. *Journal of the American Statistical Association*, 101(476):1566–1581, 2012.

G. Tsoumakas, I. Katakis, and I. Vlahavas. Effective and efficient multilabel classification in domains with large number of labels. In *ECML/PKDD Workshop on Mining Multidimensional Data*, volume 21, pages 53–59, 2008.

H. M. Wallach, I. Murray, R. Salakhutdinov, and D. Mimno. Evaluation methods for topic models. In *ICML*, pages 1105–1112, 2009.

Y. Wu, C. DuBois, A. X. Zheng, and M. Ester. Collaborative denoising auto-encoders for top-n recommender systems. In *WSDM*, pages 153–162, 2016.

I. E.-H. Yen, X. Huang, P. Ravikumar, K. Zhong, and I. Dhillon. PD-sparse: A primal and dual sparse approach to extreme multiclass and multilabel classification. In *ICML*, pages 3069–3077, 2016.

R. You, S. Dai, Z. Zhang, H. Mamitsuka, and S. Zhu. AttentionXML: Extreme multi-label text classification with multi-label attention based recurrent neural networks. *arXiv preprint arXiv:1811.01727*, 2018.

H.-F. Yu, P. Jain, P. Kar, and I. Dhillon. Large-scale multi-label learning with missing labels. In *ICML*, pages 593–601, 2014.

H. Zhao, L. Du, and W. Buntine. Leveraging node attributes for incomplete relational data. In *ICML*, pages 4072–4081, 2017a.

H. Zhao, L. Du, W. Buntine, and G. Liu. MetaLDA: A topic model that efficiently incorporates meta information. In *ICDM*, pages 635–644, 2017b.

H. Zhao, L. Du, W. Buntine, and M. Zhou. Dirichlet belief networks for topic structure learning. In *NeurIPS*, pages 7966–7977, 2018a.

H. Zhao, L. Du, W. Buntine, and M. Zhou. Inter and intra topic structure learning with word embeddings. In *ICML*, pages 5887–5896, 2018b.

M. Zhou. Infinite edge partition models for overlapping community detection and link prediction. In *AISTATS*, pages 1135–1143, 2015.

M. Zhou. Nonparametric Bayesian negative binomial factor analysis. *Bayesian Analysis*, 2018.

M. Zhou, L. Hannah, D. B. Dunson, and L. Carin. Beta-negative binomial process and Poisson factor analysis. In *AISTATS*, pages 1462–1471, 2012.

M. Zhou, Y. Cong, and B. Chen. Augmentable gamma belief networks. *JMLR*, 17(163):1–44, 2016.

Appendix for "Variational Autoencoders for Sparse and Overdispersed Discrete Data"

He Zhao[*] **Piyush Rai**[†] **Lan Du**[*]

Wray Buntine[*] **Dinh Phung**[*] **Mingyuan Zhou**[‡]

[*]Faculty of Information Technology, Monash University, Australia
[†]Department of Computer Science and Engineering, IIT Kanpur, India
[‡]McCombs School of Business, The University of Texas at Austin, USA

1 EXPERIMENTS ON TEXT ANALYSIS

1.1 Datasets

The statistics of the datasets used in the text analysis experiments are shown in Table 1. The 20NG and RCV datasets were downloaded from the code repository of Gan et al. (2015)[1]. The Wiki dataset was downloaded from *Wikipedia* using the scripts provided in Hoffman et al. (2010).

1.2 Evaluation Metric

We report per-heldout-word perplexity of all the models, which is a widely-used metric for text analysis. Following the approach in Wallach et al. (2009), after training a model with the training documents, we randomly select some words as the observed words and use the remaining words as the unobserved words in each testing document, then use the observed words to estimate the predictive probability, and finally compute the perplexity of the unobserved words. Specifically, suppose that the matrix of the testing documents is $\mathbf{Y}^* \in \mathbb{N}^{V \times N_{\text{test}}}$, which is split into the observed word matrix $\mathbf{Y}^{*\text{o}} \in \mathbb{N}^{V \times N_{\text{test}}}$ and the unobserved word matrix $\mathbf{Y}^{*\text{u}} \in \mathbb{N}^{V \times N_{\text{test}}}$, where $\mathbf{Y}^* = \mathbf{Y}^{*\text{o}} + \mathbf{Y}^{*\text{u}}$. The predictive rates of the testing documents are estimated with $\mathbf{Y}^{*\text{o}}$ and used to compute the perplexity of $\mathbf{Y}^{*\text{u}}$, detailed as follows[2]:

$$\text{Perplexity} = \exp - \left(\frac{1}{y^{*\text{u}}_{\cdot\cdot}} \sum_{j}^{N_{\text{test}}} \sum_{v}^{V} y^{*\text{u}}_{vj} \log \frac{l_{vj}}{l_{\cdot j}} \right), \quad (1)$$

where $y^{*\text{u}}_{\cdot\cdot} = \sum_j^{N_{\text{test}}} \sum_v^V y^{*\text{u}}_{vj}$. Note that l_{vj} is the predictive rate, whose derivation is model specific shown in Table

Table 1: Statistics of the datasets in text analysis. N_{train}: number of training instances, N_{test}: number of test instances. The number of nonzeros and density are computed of each whole dataset.

Dataset	N_{train}	N_{test}	V	#Nonzeros	Density
20NG	11,315	7,531	2,000	774,984	0.0343
RCV	794,414	10,000	10,000	58,637,816	0.0074
Wiki	10,000,000	1,000	7,702	82,311,745	0.0107

1.3 Experimental Settings

In the experiments of text analysis, in terms of model settings of our proposed models, following (Liang et al., 2018), we basically use the settings as for MultiVAE. Specifically, for both MultiVAE and NBVAE,

- We apply the fully connected multi-layer perceptrons (MLP) with tanh as the nonlinear activation function between the layers of the encoder and the decoder.

[1] https://github.com/zhegan27/dpfa_icml2015

[2] Our perplexity calculation is the same with the ones in Gan et al. (2015); Henao et al. (2015); Cong et al. (2017), but different from the ones in Miao et al. (2017, 2016); Krishnan et al. (2018), which use ELBO obtained from all the words of a testing document without splitting it. The results of Miao et al. (2017, 2016); Krishnan et al. (2018) can only be compared with models with variational inference.

- We use the same network architecture for the two parametric functions in the decoder, $f_{\theta^r}(\cdot)$ and $f_{\theta^p}(\cdot)$.

- The architecture of $f_\phi(\cdot)$ is symmetric to those of $f_{\theta^r}(\cdot)$ and $f_{\theta^p}(\cdot)$. For example, if we use [32, 64, 128] as the architecture of the hidden layers for the decoder, then $K = 32$ is the dimension of the latent representations and the architecture of the hidden layers for the encoder would be [128, 64, 32].

- The output layers of the encoder and decoder have no activation function.

- We set the batch size to 500 and 2000 for 20NG and the other two larger datasets, respectively.

- The number of training epochs was set to 800 and the optimisation of the VAE models was done by Adam (Kingma and Ba, 2014) with 0.003 as the learning rate.

- We use the same KL annealing procedure mentioned in the MultiVAE paper (Liang et al., 2018).

For the baselines, we use the original model settings provided in the code published by the authors. For the VAE-based models, we report the perplexity computed with the parameters (the encoder and decoder) in the last iteration of the training phrase, whereas for models with MCMC sampling (e.g., NBFA), we report the perplexity averaged over multiple samples in the collection iterations.

2 EXPERIMENTS ON COLLABORATIVE FILTERING

2.1 Datasets

ML-10M and ML-20M are downloaded from `https://grouplens.org/datasets/movielens/`; Netflix is downloaded from `http://www.netflixprize.com/`; MSD (Bertin-Mahieux et al., 2011) is downloaded from `https://labrosa.ee.columbia.edu/millionsong/`. All the datasets are preprocessed and binarised by the Python code provided by Liang et al. (2018), using the same settings described in the paper. The statistics of the datasets are shown in Table 1. Note that following Liang et al. (2018), we also generate a validation set with the same size of the testing set.

2.2 Evaluation Metrics

Two ranking-based metrics are used, which are Recall@R and the truncated normalized discounted cumulative gain (NDCG@R). To compute those metrics, following Liang et al. (2018), we first estimate the predictive rate $\boldsymbol{l}'_j$ of user j given the observed items $\boldsymbol{y}^{*o}_j$, and then rank the unobserved items $\boldsymbol{y}^{*u}_j$ by sorting $\boldsymbol{l}'_j$. The metrics are computed as follows:

$$\text{Recall@}R = \frac{\sum_{r=1}^{R} \mathbf{1}\left(y^{*u}_{\omega(r)j} = 1\right)}{\min(R, y^{*u}_{\cdot j})}, \tag{2}$$

$$\text{DCG@}R = \sum_{r=1}^{R} \frac{2^{\mathbf{1}\left(y^{*u}_{\omega(r)j}=1\right)} - 1}{\log(r+1)}, \tag{3}$$

where $\omega(r) \in \{1, \cdots, V\}$ is the item at rank r, obtained by sorting the predictive rate of the user; $\mathbf{1}\left(y^{*u}_{\omega(r)j} = 1\right)$ indicates whether the item is actually clicked on by user j; NDCG@R is computed by linearly normalising DCG@R into $[0, 1]$. Intuitively, Recall@R measures the number of the R predicted items that are within the set of the ground-truth items but does not consider the item rank in R, while NDCG@R assigns larger discounts to lower ranked items. In the experiments, we use the code provided in (Liang et al., 2018) to compute the above two metrics. Moreover, we report the testing performance of the models with the best NDCG@50 on the validation set.

Table 2: Statistics of the datasets in collaborative filtering. N_{train}: number of training instances, N_{test}: number of test instances. The number of nonzeros and density are computed of each whole dataset.

Dataset	N_{train}	N_{test}	V	#Nonzeros	Density
ML-10M	49,167	10,000	10,066	4,131,372	0.0059
ML-20M	116,677	10,000	20,108	9,128,733	0.0033
Netflix	383,435	40,000	17,769	50,980,816	0.0062
MSD	459,330	50,000	36,716	29,138,887	0.0014

2.3 Experimental Settings

For NBVAE and NBVAE$_\text{b}$, we used the same settings as in the text analysis experiments, except that:

- The batch size is set to 500 for all the datasets.

- We use two hidden layers in the encoder with [200-600] (The architectures of the two parametric functions in the decoder are symmetric to that in the encoder).

- Following MultiVAE, we use the annealing cap β, which is set to 0.2, detailed in Liang et al. (2018).

Note that all the above settings are consistent with those in (Liang et al., 2018) The original code of MultiVAE and MultiDAE and their best settings provided by the authors are used in the comparison.

Table 3: The statistics of the datasets used in the experiments. N_{train}: number of training instances, N_{test}: number of test instances, D: number of features, V: number of labels.

Dataset	N_{train}	N_{test}	D	V
Delicious	12920	3185	500	983
Mediamill	30993	12914	120	101
EURLex	15539	3809	5000	3993

3 EXPERIMENTS ON MULTI-LABEL LEARNING

3.1 Datasets

All the datasets are downloaded from http://manikvarma.org/downloads/XC/XMLRepository.html and the statistics of the datasets are shown in Table 3.

3.2 Evaluation Metrics

We report Precision@R ($R \in \{1, 3, 5\}$), which is a widely-used ranking-based evaluation metric for multi-label learning, following Jain et al. (2017). To compute this metric, after training NBVAE$_{\text{c}}$, given the feature vector of a testing sample j^*, we can feed $\boldsymbol{x}_{j^*}$ into the feature encoder to sample the latent representation, $\boldsymbol{z}_{j^*}$, then feed it into the decoder to get the predictive rate $\boldsymbol{l'}_{j^*}$. With the predictive rate, we can rank the labels and compute Precision@R, which is similar to the computation of Recall and NDCG used in collaborative filtering.

3.3 Experimental Settings

For NBVAE$_{\text{c}}$ in multi-label learning, we used the same settings as NBVAE$_{\text{b}}$ in the text analysis experiments, specifically:

- In the Delicious and Mediamill datasets, we use [200-600] for two hidden layers in the encode and for EURLex, we use one hidden layer in the encoder with 600 units.

- We use relu as the activation function for Delicious and EURLex and tanh as the activation function for Mediamill.

4 RUNNING SPEED COMPARISON

Here we compare the running speeds of NBVAE and MultiVAE.

Table 4: Running time (seconds) per iteration on the text datasets.

Model	20NG	RCV	Wiki
MultiVAE	0.12	48.21	42.18
NBVAE	0.17	49.49	48.57

Table 5: Running time (seconds) per iteration on the collaborative filtering datasets.

Model	ML-10M	ML-20M	Netflix	MSD
MultiVAE	2.91	14.79	46.90	105.33
NBVAE	3.47	17.54	52.12	124.43

Analytically, NBVAE has additional computational cost over MultiVAE in two aspects: (1) there are two parameters p and r in NBVAE so we use two decoders f_{θ^r} and f_{θ^p} (one more than MultiVAE); (2) an additional term in the log likelihood for zeros, as pointed out by the reviewer. In terms of (2), according to our description under Eq. (3) of the main paper, we just need to compute the Bernoulli parameter $\text{temp} = 1 - (1 - p_j)^{r_j}$ and then compute $y_j * \log \text{temp} + (1 - y_j) * \log(1 - \text{temp})$, where the RHS is for zeros.

Empirically, as our models are implemented in TensorFlow running on GPUs, the overhead as compared to MultiVAE is not large. Here we report the running time (seconds) per iteration of NBVAE (NBVAE$_{\text{b}}$) and MultiVAE on the text and collaborative filtering datasets in Table 4 and Table 5, respectively. All the experiments are implemented in TensorFlow and with the same settings in the paper, and run on the same machine with Nvidia Tesla P100 GPU.

In addition, in Figure 1, we plot the validation set performance of NBVAE (NBVAE$_{\text{b}}$) and MultiVAE in the training phase.

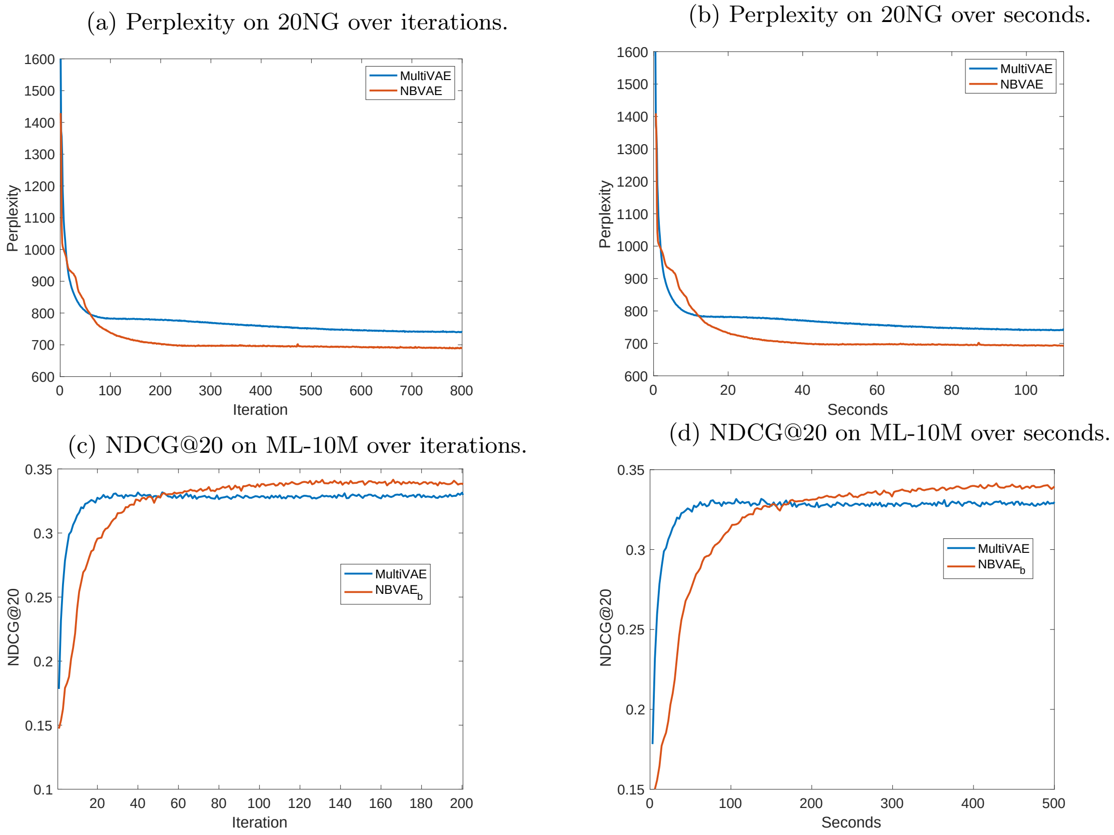
References

T. Bertin-Mahieux, D. P. Ellis, B. Whitman, and P. Lamere. The million song dataset. In *International Conference on Music Information Retrieval*, 2011.

Y. Cong, B. Chen, H. Liu, and M. Zhou. Deep latent Dirichlet allocation with topic-layer-adaptive stochastic gradient Riemannian MCMC. In *ICML*, pages 864–873, 2017.

Z. Gan, C. Chen, R. Henao, D. Carlson, and L. Carin. Scalable deep Poisson factor analysis for topic modeling. In *ICML*, pages 1823–1832, 2015.

R. Henao, Z. Gan, J. Lu, and L. Carin. Deep Poisson factor modeling. In *NIPS*, pages 2800–2808, 2015.



(a) Perplexity on 20NG over iterations.

(b) Perplexity on 20NG over seconds.

(c) NDCG@20 on ML-10M over iterations.

(d) NDCG@20 on ML-10M over seconds.

Figure 1: Performance of NBVAE and MultiVAE on the validation set during training.

M. Hoffman, F. R. Bach, and D. M. Blei. Online learning for latent Dirichlet allocation. In *NIPS*, pages 856–864, 2010.

V. Jain, N. Modhe, and P. Rai. Scalable generative models for multi-label learning with missing labels. In *ICML*, pages 1636–1644, 2017.

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

R. Krishnan, D. Liang, and M. Hoffman. On the challenges of learning with inference networks on sparse, high-dimensional data. In *AISTATS*, pages 143–151, 2018.

D. Liang, R. G. Krishncan, M. D. Hoffman, and T. Jebara. Variational autoencoders for collaborative filtering. In *WWW*, pages 689–698, 2018.

Y. Miao, L. Yu, and P. Blunsom. Neural variational inference for text processing. In *ICML*, pages 1727–1736, 2016.

Y. Miao, E. Grefenstette, and P. Blunsom. Discovering discrete latent topics with neural variational inference. In *ICML*, pages 2410–2419, 2017.

H. M. Wallach, I. Murray, R. Salakhutdinov, and D. Mimno. Evaluation methods for topic models. In *ICML*, pages 1105–1112, 2009.