Disentangled Variational Autoencoder based Multi-Label Classification with Covariance-Aware Multivariate Probit Model

Junwen Bai[*] , **Shufeng Kong** and **Carla Gomes**
Department of Computer Science, Cornell University
{jb2467, sk2299}@cornell.edu, gomes@cs.cornell.edu

Abstract

Multi-label classification is the challenging task of predicting the presence and absence of multiple targets, involving representation learning and label correlation modeling. We propose a novel framework for multi-label classification, Multivariate Probit Variational AutoEncoder (MPVAE), that effectively learns latent embedding spaces as well as label correlations. MPVAE learns and aligns two probabilistic embedding spaces for labels and features respectively. The decoder of MPVAE takes in the samples from the embedding spaces and models the joint distribution of output targets under a Multivariate Probit model by learning a shared covariance matrix. We show that MPVAE outperforms the existing state-of-the-art methods on important computational sustainability applications as well as on other application domains, using public real-world datasets[1]. MPVAE is further shown to remain robust under noisy settings. Lastly, we demonstrate the interpretability of the learned covariance by a case study on a bird observation dataset.

1 Introduction

Multi-label classification (MLC) concerns the simultaneous prediction of the presence and absence of multiple labels for each sample of a given sample set. Unlike in the conventional classification task, more than one label or target could be associated with each sample in MLC [Zhang and Zhou, 2013; Zhang *et al.*, 2018]. This setting is important for the study of key biodiversity applications in computational sustainability [Gomes *et al.*, 2019], e.g., concerning joint species distributions and wildlife mapping [Chen *et al.*, 2017], as well as in a variety of other scenarios, such as protein site localization [Alazaidah *et al.*, 2015] and drug side effects [Kuhn *et al.*, 2015]. Furthermore, understanding the label correlations is also important. For instance, in biodiversity applications, species correlation modeling is critical to address core ecological concerns like interactions of species with each other, which could affect species monitoring, protection and policy-making [Evans *et al.*, 2017].

Some early work simply decomposes the multi-label classification into multiple single-label classification problems [Boutell *et al.*, 2004]. Though these methods can be adapted from single-label predictors, they ignore the correlation among labels. To improve this, classifier chains [Read *et al.*, 2009] stack the binary classifiers into a chain and reuse the outputs of previous classifiers as extra information to improve the prediction of the current label. Followup works extend the classifier chains to recurrent neural networks [Wang *et al.*, 2016] to increase capacity and better model the label correlation. Label ordering is critical to these methods since long-term dependencies are typically weaker than short-term dependencies. The model structure also restricts parallel computation. Another straightforward method is to find nearest neighbors in the feature space and assign labels to test samples by Bayesian inference [Zhang and Zhou, 2007; Chiang *et al.*, 2012]. However, either the predefined metric space or the prior may heavily affect the model performance.

Latent embedding learning is a recent technique to match features and labels in the latent space. Pioneer studies [Yu *et al.*, 2014; Chen and Lin, 2012; Bhatia *et al.*, 2015] make low-rank assumptions of labels and features, and transform labels to label embeddings, by dimensionality reduction techniques such as canonical correlation analysis (CCA). Benefiting from the capacity of deep neural networks, more recent latent embedding methods for MLC employ neural networks to build and align the latent spaces for both labels and features [Yeh *et al.*, 2017; Chen *et al.*, 2019a]. The constraints in the conventional dimension reduction models are relaxed and embedded into the deep latent space. For example, C2AE relaxes the orthogonality constraint to the minimization of an ℓ_2 distance. These embedding methods are believed to implicitly encode the label correlations in the embedding space.

Some other state-of-the-art models initiate the research on using graph neural networks (GNN) to explicitly encode the label correlations [Chen *et al.*, 2019b; Lanchantin *et al.*, 2019]. A graph neural network for labels can build dependencies among labels through learned or given edges between them. Though GNN brings a new way to embed the correlations, the number of stacked GNNs or the iterations of message passing may require extra effort to fine-tune.

We propose the **Multivariate Probit Variational Autoen-**

[*] Contact Author

[1] Our code is available on https://github.com/JunwenBai/MPVAE

coder (MPVAE), which *improves both the embedding space learning and label correlation encoding*. In particular, **(1)** MPVAE learns probabilistic latent spaces for both labels and features, unlike most autoencoder (AE) based multi-label models. The probabilistic latent space learned by the VAE can provide three major advantages. First, it gives more control to the latent space [Chung *et al.*, 2015]. In many AE models, one can often observe the label-encoder-decoder branch gives much better performance than feature-encoder-decoder branch. Imposing the VAE structure in the latent space helps balance the difficulty of the learning and aligning of the two subspaces. Second, smoothness in the latent space is often desired [Wu *et al.*, 2018]. Probabilistic models like the VAE naturally bring smoothness on a local scale since the decoder decodes a sample rather than a specific embedding. Third, the VAE model and its variations learn representations with **disentangled** factors [van Steenkiste *et al.*, 2019]. If both latent spaces for features and labels learn disentangled factors, not only is it helpful to aligning two spaces, but it is also beneficial for the decoding process. **(2) MPVAE explicitly learns a shared covariance matrix to build dependencies among labels** by adopting the Multivariate Probit (MP) probabilistic model, which is inspired by some recent work in joint distribution modeling [Chen *et al.*, 2018]. The MP assumes an underlying latent multivariate Gaussian distribution. We show that the MP model is a simple and straightforward component of the overall probabilistic generative framework compared to other more complex models such as GNNs. More importantly, the MP model improves the prediction performance and provides the interpretability of the learned covariance matrix. By using the Cholesky decomposition and t-SNE, we demonstrate visually the value of the learned covariance, in applications like species correlation modeling in the computational sustainability domain. **(3) MPVAE is optimized with respect to a three-component loss function**, which includes a Kullback–Leibler (KL) divergence component for jointly learning and aligning the label and feature embeddings and a cross-entropy and ranking loss for the multi-label prediction in the MP model. **(4)** We thoroughly test MPVAE with multiple public datasets on a variety of metrics. **MPVAE outperforms (or is comparable to) other state-of-the-art multi-label prediction models.** We further illustrate MPVAE is still robust even if the training labels are noisy.

2 Other Related Work

Covariance matrices are commonly seen in non-deep multi-label classification models [Bi and Kwok, 2014; Zhang and Yeung, 2013]. Non-deep models often assume a matrix-variate normal distribution on features, weights or labels. Covariances thus play a key role in these models. But the scalability issue limits their applications in large-scale problems. However, these assumptions are not necessary in deep neural networks given their powerful expressiveness.

A recent success of combining deep learning and covariance based paradigms is the deep Multivariate Probit model (DMVP) [Chen *et al.*, 2018]. It is a deep generalization of the classic Multivariate Probit model. An efficient sampling process was proposed in the paper to avoid the heavy-duty Markov chain Monte Carlo (MCMC) sampling process. Though DMVP performs well on the joint likelihood measure, it lacks enough predictive power for presence-absence (0/1) classification. MPVAE makes one step further and introduces the cross-entropy loss as well as the ranking-loss under the Multivariate Probit paradigm.

Some prior work also applied deep generative models in multi-label classification. [Chu *et al.*, 2018] proposes a deep sequential generative model. Though the model is effective in the setting of missing labels, the concern of unstable training of stacked generative models should not be overlooked.

3 Methods

3.1 Preliminaries

Let $\mathcal{D}$ denote the dataset $\{(\mathbf{x}_i, \mathbf{y}_i)\}_{i=1}^N$, where $\mathbf{x}_i \in \mathbb{R}^S$ and $\mathbf{y}_i \in \{0,1\}^L$. $\mathbf{x}_i$ is the feature vector and $\mathbf{y}_i$ represents the presence (1) or absence (0) of targets.

Variational Autoencoder (VAE)

A VAE assumes a generative process for the observed datapoints X: $P(X) = \int p_\theta(X|z;\theta)P(z)dz$, by introducing latent variables z. Since most z's contribute little to $P(X)$, Monte Carlo sampling would be inefficient. We instead learn a function $q_\phi(z|X)$ to approximate the intractable $P(z|X)$ for efficient sampling. The KL divergence ($\mathcal{D}$) between $q_\phi(z|X)$ and $P(z|X)$ is given by $\mathcal{D}(q_\phi(z|X)||P(z|X)) = \mathbb{E}_{z\sim q_\phi}[\log q_\phi(z|X) - \log P(z|X)]$. Applying Bayes rule and transforming the equation yield: $\log P(X) - \mathcal{D}[Q(z|X)||P(z|X)] = \mathbb{E}_{z\sim q_\phi}[\log p_\theta(X|z)] - \mathcal{D}[q_\phi(z|X)||P(z)]$. The right hand side of the equation is the tractable evidence lower bound (ELBO) to maximize. $P(z)$ is the prior, a standard multivariate normal distribution. The first term in the ELBO encourages the reconstruction of X and the second term penalizes the KL divergence between the approximate distribution and the prior to impose structure on the latent space. Both p_θ and q_ϕ can be parameterized by neural networks. With the help of the reparameterization trick, the whole model can be trained with back-propagation. VAE and its variations can learn disentangled factors by controlling the capacity of the information bottleneck. For example, β-VAE [Higgins *et al.*, 2017] is a known disentangled VAE, which is able to learn abstract concepts like size and shape, only with a slight modification to the objective, $\mathbb{E}_{z\sim q_\phi}[\log p_\theta(X|z)] - \beta\mathcal{D}[q_\phi(z|X)||P(z)]$. MPVAE follows the structure of β-VAE with $\beta = 1.1$.

Multivariate Probit (MP) Model

Consider a single sample $(\mathbf{x}, \mathbf{y})$ where $\mathbf{x}$ is the input feature vector and $\mathbf{y} \in \{0,1\}^L$ is the label. The Multivariate Probit model introduces auxiliary latent variables $\mathbf{y}^* \in \mathbb{R}^L$, which follow a multivariate normal distribution $\mathcal{N}(\mathbf{x}\gamma, \Sigma)$ where γ is the weight parameter and Σ is the covariance matrix. $\mathbf{y}$ is viewed as the indicator for whether $\mathbf{y}^*$ is positive or not: $y_i = \mathbb{1}\{y_i^* > 0\}, i = 1, .., L$. The probability of observing $\mathbf{y}$ is given by $P(\mathbf{y}|\mathbf{x}\gamma, \Sigma) = \int_{A_L} \cdots \int_{A_1} p(\mathbf{y}^*|\mathbf{x}\gamma, \Sigma)dy_1^* \ldots dy_L^*$, where $A_j = (-\infty, 0]$ if $y_j = 0$ or $A_j = (0, \infty)$ otherwise. $p(\cdot)$ is the probability density function of the normal distribution. The framework can be generalized to a deep model

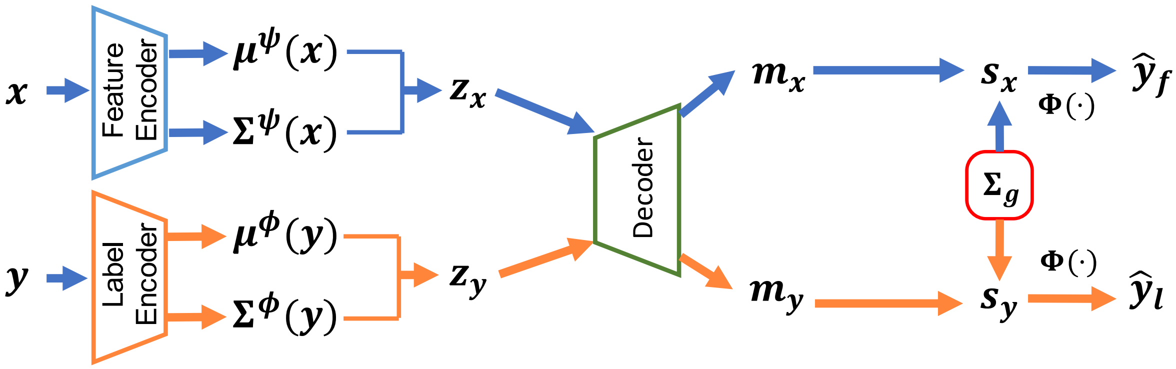

Figure 1: Network structure of MPVAE. The feature encoder encodes x to a probabilistic latent subspace with a neural network parameterized by ψ. Similarly, another label encoder with parameter ϕ maps y to another probabilistic latent subspace with the same dimensionality. Two samples z_x, z_y from the subspaces are fed into the shared decoder and deciphered as the means m_x, m_y in the Multivariate Probit model. With the help of the global covariance matrix Σ_g, we sample s_x, s_y from $\mathcal{N}(m_x, \Sigma_g), \mathcal{N}(m_y, \Sigma_g)$ to derive the final 0/1 predictions $\hat{y}_f, \hat{y}_l$. Note that during testing, only $\hat{y}_f$ is the prediction for the test instances.

simply by replacing the mean $\mathbf{x}\gamma$ with a neural network $f(\mathbf{x})$. In MPVAE, the mean is given by the decoder of VAE.

3.2 MPVAE

We propose Multivariate Probit Variational Autoencoder, a novel disentangled Variational Autoencoder based framework with covariance-aware Multivariate Probit model for MLC. The illustration of the framework is shown in Fig. 1. The whole model can be viewed as a two-stage generative process. The first stage maps the features and labels to Gaussian subspaces where the means and variances are learned by multi-layer perceptrons. The key task in this stage is to match the two subspaces. The second stage decodes the sample from each subspace and feeds the outputs into a Multivariate Probit module as the means. A global covariance matrix is learned separately. The final output of the second stage gives the predicted labels.

Learning and Aligning Probabilistic Subspaces

Given an input pair of feature vector and label $(\mathbf{x}, \mathbf{y})$ where $\mathbf{x} \in \mathbb{R}^S, \mathbf{y} \in \{0,1\}^L$, the feature encoder maps $\mathbf{x}$ to a Gaussian subspace $\mathcal{N}(\mu^{\psi}(\mathbf{x}), \Sigma^{\psi}(\mathbf{x}))$ and the label encoder maps $\mathbf{y}$ to another Gaussian subspace $\mathcal{N}(\mu^{\phi}(\mathbf{y}), \Sigma^{\phi}(\mathbf{y}))$. ϕ, ψ are trainable parameters in the encoders. $\mu^{\psi}(\mathbf{x}), \mu^{\phi}(\mathbf{y}) \in \mathbb{R}^d$ and $\Sigma^{\psi}(\mathbf{x}), \Sigma^{\phi}(\mathbf{y}) \in \mathbb{R}^{d \times d}_{\geq 0}$, where d is the dimensionality of the latent space. $\mathbf{z_x}, \mathbf{z_y}$ denote samples from each of these two distributions respectively.

If we only consider the label encoder-decoder branch (orange branch in Fig. 1), it models a standard (β-)VAE. The ELBO to optimize can be written as,

$$\mathbb{E}_{\mathbf{z} \sim q_{\phi}}[\log p_{\theta}(\mathbf{y}|\mathbf{z})] - \beta \mathcal{D}[q_{\phi}(\mathbf{z}|\mathbf{y}) || P(\mathbf{z})] \tag{1}$$

The issue with this label autoencoder is the lack of connections between $\mathbf{x}$ and $\mathbf{z}$. Even if a good generative model is learned, prediction given $\mathbf{x}$ is impossible since the prior $P(\mathbf{z})$ is unrelated to $\mathbf{x}$. Our simple fix is to replace $P(\mathbf{z})$ with a prior distribution dependent on $\mathbf{x}$, $q_{\psi}(\mathbf{z}|\mathbf{x})$. That's where the feature encoder comes from. The feature encoder is also a neural network parameterized by learnable ψ. With the feature encoder, given the input $\mathbf{x}$, we can sample from $q_{\psi}(\mathbf{z}|\mathbf{x})$, which is approximately equal to $q_{\theta}(\mathbf{z}|\mathbf{y})$. The challenges thereafter are twofold: a) how to learn ψ and b) how to align $\mathcal{N}(\mu^{\psi}(\mathbf{x}), \Sigma^{\psi}(\mathbf{x}))$ and $\mathcal{N}(\mu^{\phi}(\mathbf{y}), \Sigma^{\phi}(\mathbf{y}))$. We propose to extend the objective function to:

$$\frac{1}{2}(\mathbb{E}_{\mathbf{z} \sim q_{\phi}}[\log p_{\theta}(\mathbf{y}|\mathbf{z})] + \mathbb{E}_{\mathbf{z} \sim q_{\psi}}[\log p_{\theta}(\mathbf{y}|\mathbf{z})]) - \beta \mathcal{D}[q_{\phi}(\mathbf{z}|\mathbf{y}) || q_{\psi}(\mathbf{z}|\mathbf{x})]$$

The first two terms will be handled by the Multivariate Probit model in the next subsection. The last term is simply the KL divergence between two multivariate normal distributions. Since both distributions have diagonal covariance matrices, we can derive the KL loss term for $\mathcal{D}[q_{\phi}(\mathbf{z}|\mathbf{y}) || q_{\psi}(\mathbf{z}|\mathbf{x})]$:

$$\begin{aligned} L_{\mathrm{KL}}(\mathbf{x}, \mathbf{y}) = & \frac{1}{2}\beta[\sum_{i=1}^{d} \log \frac{\Sigma^{\psi}_{i,i}(\mathbf{x})}{\Sigma^{\phi}_{i,i}(\mathbf{y})} - d + \sum_{i=1}^{d} \frac{\Sigma^{\phi}_{i,i}(\mathbf{y})}{\Sigma^{\psi}_{i,i}(\mathbf{x})} + \\ & \sum_{i=1}^{d} \frac{(\mu^{\psi}_i(\mathbf{x}) - \mu^{\phi}_i(\mathbf{y}))^2}{\Sigma^{\psi}_{i,i}(\mathbf{x})}] \end{aligned} \tag{2}$$

As in the conditional VAE, our implementation further adds an improvement to concatenate each of $\mathbf{y}, \mathbf{z_y}, \mathbf{z_x}$ with $\mathbf{x}$, which basically does not affect the derivations above, but uses extra information for inference.

Reconstruction and Prediction

The Multivariate Probit (MP) is a classic latent variable model for data with presence-absence relationships. Unlike the typical softmax transformation in most deep methods, the model maps a sample from a multivariate normal distribution to its own cumulative distribution (CDF), to bound the output range within $[0, 1]$. The major drawback of the MP model is that the integration step is intractable for large-scale data and is usually approximated with MCMC. [Chen *et al.*, 2018] provides an alternative parallelizable sampling process for estimating the CDF. GPUs can thus expedite the estimation. We adopt this sampling method in our model, and find it works well in MPVAE.

Label information is available for both the training and testing phases in [Chen *et al.*, 2018]. However, as a predictive model, MPVAE does not have access to labels until the final predicted targets are given, when the loss can be computed between the ground-truth and predicted labels for training, or the predicted targets are directly given as the output during testing. In this case, we let MPVAE calculate the integral only w.r.t. the $(0, \infty)$ region for each target(dimension). If the corresponding target is present, the CDF should be close to 1. Otherwise, it should be near 0.

Sampling Process. The shared decoder reads the samples $\mathbf{z_x}$, $\mathbf{z_y}$, and outputs $\mathbf{m_x}$, $\mathbf{m_y} \in \mathbb{R}^L$ respectively as the means in the MP model. The covariance matrix $\Sigma_g \in \mathbb{R}^{L\times L}$ in the MP only models the dependencies among targets and is unrelated to the features. Thus Σ_g can be learned as a shared parameter. This stabilizes the training and helps interpret the label correlation (shown in experiments). Instead of directly sampling $\mathbf{y}^*$ from $\mathcal{N}(\mathbf{m}, \Sigma_g)$, we sample twice by decomposing Σ_g to $V + \Sigma_r$ where V is a diagonal positive definite matrix and Σ_r is the residual. For 2 random variables $\mathbf{w} \sim \mathcal{N}(\mathbf{0}, V), \mathbf{s} \sim \mathcal{N}(\mathbf{m}, \Sigma_r)$, the difference $(\mathbf{w} - \mathbf{s}) \sim \mathcal{N}(-\mathbf{m}, \Sigma_g)$. Note that since V is diagonal, the different dimensions of $\mathbf{w}$ are independent. To estimate the probability of presence $P(y_i^* \geq 0|\mathbf{m}, \Sigma_g), i \in [1, L]$,

$$\begin{aligned}
&P(y_i^* \geq 0|\mathbf{m}, \Sigma_g) = P(y_i^* \leq 0| -\mathbf{m}, \Sigma_g)\\
=&P(w_i - s_i \leq 0) \quad \mathbf{w} \sim \mathcal{N}(\mathbf{0}, V),\ \mathbf{s} \sim \mathcal{N}(\mathbf{m}, \Sigma_r)\\
=&\mathbb{E}_{\mathbf{s}\sim\mathcal{N}(\mathbf{m},\Sigma_r)}[P(w_i \leq s_i|\mathbf{s})] \quad \mathbf{w} \sim \mathcal{N}(\mathbf{0}, V)\\
=&\mathbb{E}_{\mathbf{s}\sim\mathcal{N}(\mathbf{m},\Sigma_r)}[\Phi(\frac{s_i}{\sqrt{V_{i,i}}})]
\end{aligned} \tag{3}$$

where Φ is the CDF of a univariate standard normal distribution. Since V is simply a scaling factor, w.l.o.g., V can be set to identity I. Under this mechanism, $P(y_i^* \geq 0|\mathbf{m}, \Sigma_g)$ for each i can be computed in parallel and independently with one sample $\mathbf{s}$ or the average of multiple samples. Note that with this derivation, Σ_r is learned rather than Σ_g. But they only differ by I. The 0/1 outputs given $\mathbf{m_x}, \mathbf{m_y}$ are denoted by $\hat{\mathbf{y}}_f$ and $\hat{\mathbf{y}}_l$. If $P(y_i^* \geq 0|\mathbf{m}, \Sigma_g)$ is higher than a certain threshold, $\hat{y}_i$ gives 1. Otherwise, $\hat{y}_i$ is set to 0. During testing, $\hat{\mathbf{y}}_f$ is regarded as the final result.

Binary Cross Entropy (BCE) Loss in MP. With the efficient sampling scheme, the reconstruction losses $\mathbb{E}_{\mathbf{z}\sim q_\phi}[\log p_\theta(\mathbf{y}|\mathbf{z})]$ and $\mathbb{E}_{\mathbf{z}\sim q_\psi}[\log p_\theta(\mathbf{y}|\mathbf{z})]$ can be concretized. For a binary prediction task for each target, a Bernoulli likelihood assumption is valid, which leads to the binary cross entropy loss between the labels and predictions:

$$\begin{aligned}
&\mathcal{L}_{\text{CE}}(\mathbf{y}, \mathbf{z}) = -\log p_\theta(\mathbf{y}|\mathbf{z}, \Sigma_g) = -\log p_\theta(\mathbf{y}|\mathbf{m}, \Sigma_r)\\
=& -\log \mathbb{E}_{\mathbf{s}\sim\mathcal{N}(\mathbf{m},\Sigma_r)}[\prod_{i=1}^{L} \Phi(s_i)^{y_i}(1 - \Phi(s_i))^{1-y_i}]\\
=& -\log \mathbb{E}_{\mathbf{s}\sim\mathcal{N}(\mathbf{m},\Sigma_r)} \exp[\sum_{i=1}^{L} y_i \log \Phi(s_i) + (1 - y_i) \log(1 - \Phi(s_i))]\\
\approx& -\log \frac{1}{M}\sum_{k=1}^{M} \exp[\sum_{i=1}^{L} y_i \log \Phi(s_i^k) + (1 - y_i) \log(1 - \Phi(s_i^k))]
\end{aligned}$$

The last step is a Monte Carlo approximation. M is the preset number of samples. The log-sum-exp trick is applied for computation to avoid overflow issues. Since $\mathbb{E}_{\mathbf{z}\sim q_\phi}[\log p_\theta(\mathbf{y}|\mathbf{z})]$ is typically approximated by a single sample from q_ϕ and minimization is preferred for training, the objective function can be written as $-\mathbb{E}_{\mathbf{z}\sim q_\phi}[\log p_\theta(\mathbf{y}|\mathbf{z})] \approx -\log p_\theta(\mathbf{y}|\mathbf{z_y}) = \mathcal{L}_{\text{CE}}(\mathbf{y}, \mathbf{z_y})$ where $\mathbf{z_y} \sim q_\phi(\mathbf{z}|\mathbf{y})$. Similarly, $-\mathbb{E}_{\mathbf{z}\sim q_\psi}[\log p_\theta(\mathbf{y}|\mathbf{z})] \approx -\log p_\theta(\mathbf{y}|\mathbf{z_x}) = \mathcal{L}_{\text{CE}}(\mathbf{y}, \mathbf{z_x})$ where $\mathbf{z_x} \sim q_\psi(\mathbf{z}|\mathbf{x})$.

Algorithm 1 Training MPVAE

```
Input: ${(\mathbf{x}_i, \mathbf{y}_i)}_{i=1}^N$, batch size $b$
for # of iterations do
    Sample a minibatch $\{(\mathbf{x}^{(k)}, \mathbf{y}^{(k)})\}_{k=1}^b$
    Compute $\{(\mu^\psi(\mathbf{x}^{(k)}), \Sigma^\psi(\mathbf{x}^{(k)})), (\mu^\phi(\mathbf{y}^{(k)}), \Sigma^\phi(\mathbf{y}^{(k)}))\}$
    Derive $\mathcal{L}_{\text{KL}}$ by Eq (2)
    Sample latent variables $\{\mathbf{z}_\mathbf{x}^{(k)}\}, \{\mathbf{z}_\mathbf{y}^{(k)}\}$
    Decode $\mathbf{z}$'s and obtain $\{\mathbf{m}_\mathbf{x}^{(k)}\}, \{\mathbf{m}_\mathbf{y}^{(k)}\}$
    Calculate the mean loss by Eq (5) for the batch
    Compute gradients and update parameters with Adam
return $\phi, \psi, \theta, \Sigma_r$
```

Ranking Loss in MP. Ranking loss [Zhang and Zhou, 2013] is also widely used in many multi-label tasks. It is a loss to measure the correlations between positive labels and negative labels. The idea behind the ranking loss is simple: the gap between the logits for positive and negative labels should be as large as possible. Suppose $\mathbf{y}$ is the ground-truth label set. Let $\mathbf{y}^0$ denote the set of indices of negative labels and $\mathbf{y}^1$ the positive labels. $\mathbf{s}$ denotes the sample from $\mathcal{N}(\mathbf{m}, \Sigma_r)$, which depends on $\mathbf{z}$. Thus the ranking loss on $\mathbf{s}$ can also be viewed as a loss on $\mathbf{z}$. We define the ranking loss in MP as

$$\begin{aligned}
&\mathcal{L}_{\text{RL}}(\mathbf{y}, \mathbf{z}) = \mathcal{L}'_{\text{RL}}(\mathbf{y}, \mathbf{s})\\
=&\mathbb{E}_{\mathbf{s}\sim\mathcal{N}(\mathbf{m},\Sigma_r)}[\frac{1}{|\mathbf{y}^0||\mathbf{y}^1|}\sum_{(i,j)\in\mathbf{y}^1\times\mathbf{y}^0} \exp(-(\Phi(s_i) - \Phi(s_j)))]\\
=&\frac{1}{M}\sum_{k=1}^{M}[\frac{1}{|\mathbf{y}^0||\mathbf{y}^1|}\sum_{(i,j)\in\mathbf{y}^1\times\mathbf{y}^0} \exp(-(\Phi(s_i^k) - \Phi(s_j^k)))]
\end{aligned}$$

The ranking loss can be defined for both $\mathbf{z_y}$ and $\mathbf{z_x}$.

Entropy Loss in MP. Some multi-label datasets have very sparse positive labels even if there exist multiple labels for one feature. For example, in the *mirflickr* dataset, the positive label rate is as low as 12%. Therefore, for sparse datasets, we add an extra entropy loss (define $p_i = \frac{\exp(\Phi(s_i))}{\sum_{j=1}^{L}\exp(\Phi(s_i))}$),

$$\mathcal{L}_{\text{Ent}}(\mathbf{z}) = \mathcal{L}'_{\text{Ent}}(\mathbf{s}) = \mathbb{E}_{\mathbf{s}\sim\mathcal{N}(\mathbf{m},\Sigma_r)}[-\sum_{i=1}^{L} p_i \log p_i] \tag{4}$$

Entropy loss acts as a self-regularizer and only depends on the predicted values.

Overall Loss Function

Let $\mathcal{L}_{prior} = \lambda_1\mathcal{L}_{\text{CE}}(\mathbf{y}, \mathbf{z_x}) + \lambda_2 L_{\text{RL}}(\mathbf{y}, \mathbf{z_x}) + \lambda_3 L_{\text{Ent}}(\mathbf{z_x})$ and $\mathcal{L}_{recon} = \lambda_1\mathcal{L}_{\text{CE}}(\mathbf{y}, \mathbf{z_y}) + \lambda_2 L_{\text{RL}}(\mathbf{y}, \mathbf{z_y}) + \lambda_3 L_{\text{Ent}}(\mathbf{z_y})$. Together with the KL loss $\mathcal{L}_{\text{KL}}$ defined in the previous subsection, the overall loss function can be described as

$$\mathcal{L} = \mathcal{L}_{\text{recon}} + \mathcal{L}_{\text{prior}} + \beta\mathcal{L}_{\text{KL}} \tag{5}$$

$\mathcal{L}_{\text{prior}}$ encompasses the losses in the feature-encoder-decoder branch (blue branch) for the prior, while $\mathcal{L}_{\text{recon}}$ encompasses the losses in the label-encoder-decoder branch (orange

Dataset	MLKNN	MLARAM	SLEEC	C2AE	DMVP	LaMP	MPVAE
eBird	0.5103	0.5101	0.2578	0.5007	0.5291	0.4768	**0.5511**
fish	0.7641	0.5072	0.7790	0.7654	0.7684	0.7844	**0.7881**
mirflickr	0.3826	0.4316	0.4163	0.5011	0.5105	0.4918	**0.5138**
nuswide	0.3420	0.3964	0.4312	0.4354	0.4657	0.3760	**0.4684**
yeast	0.6176	0.6292	0.6426	0.6142	0.6335	0.6242	**0.6479**
scene	0.6913	0.7166	0.7184	0.6978	0.6886	0.7279	**0.7505**
sider	0.7382	0.7222	0.5807	0.7682	0.7658	0.7662	**0.7687**
bibtex	0.1826	0.3530	0.4490	0.3346	0.4456	0.4469	**0.4534**
delicious	0.2590	0.2670	0.3081	0.3257	0.3639	0.3720	**0.3732**

Dataset	MLKNN	MLARAM	SLEEC	C2AE	DMVP	LaMP	MPVAE
eBird	0.5573	0.5732	0.4124	0.5459	0.5699	0.5170	**0.5933**
fish	0.7349	0.5177	0.7563	0.7387	0.7426	0.7598	**0.7648**
mirflickr	0.4149	0.4471	0.4127	0.5448	0.5499	0.5352	**0.5516**
nuswide	0.3679	0.4151	0.4277	0.4724	0.4912	0.4720	**0.4923**
yeast	0.6252	0.6350	0.6531	0.6258	0.6326	0.6407	**0.6554**
scene	0.6667	0.6927	0.6993	0.7131	0.6935	0.7156	**0.7422**
sider	0.7718	0.7535	0.6965	0.7978	0.7961	0.7977	**0.8002**
bibtex	0.1782	0.3645	0.4074	0.3884	**0.4801**	0.4733	0.4800
delicious	0.2639	0.2734	0.3333	0.3479	0.3791	0.3868	**0.3934**

Dataset	MLKNN	MLARAM	SLEEC	C2AE	DMVP	LaMP	MPVAE
eBird	0.3379	0.4735	0.3625	0.4260	0.4391	0.3806	**0.4936**
fish	0.6377	0.4272	0.6570	0.6466	0.6379	0.6865	**0.6925**
mirflickr	0.2660	0.2838	0.3636	0.3931	0.4193	0.3871	**0.4217**
nuswide	0.0863	0.1565	0.1354	0.1742	0.1633	0.2031	**0.2105**
yeast	0.4716	0.4484	0.4251	0.4272	0.4747	0.4802	**0.4817**
scene	0.6932	0.7131	0.6990	0.7284	0.7160	0.7449	**0.7504**
sider	0.6674	0.6491	0.5917	0.6674	0.6033	0.6684	**0.6904**
bibtex	0.0727	0.2267	0.2937	0.2680	0.3732	0.3763	**0.3863**
delicious	0.0526	0.0739	0.1418	0.1019	0.1806	**0.1951**	0.1814

Table 1: Top: example-F1 scores, Middle: micro-F1 scores, and Bottom: macro-F1 scores for different methods on all the datasets. The best scores are in bold. Each score is the average after 3 runs.

Dataset	MLKNN	MLARAM	SLEEC	C2AE	DMVP	LaMP	MPVAE
eBird	0.8273	0.8186	0.8156	0.7712	0.7900	0.8113	**0.8286**
fish	0.8829	0.6710	0.8905	0.8840	0.8901	0.8880	**0.8906**
mirflickr	0.8767	0.6337	0.8698	0.8973	0.8651	0.8969	**0.8978**
nuswide	0.9714	0.9711	0.9710	0.9725	0.9717	0.9801	**0.9804**
yeast	0.7835	0.7439	0.7824	0.7635	0.7808	0.7857	**0.7920**
scene	0.8633	0.9021	0.8937	0.8934	0.8748	0.9025	**0.9094**
sider	0.7146	0.6501	0.6750	0.7487	0.7387	0.7510	**0.7547**
bibtex	0.9853	0.9861	0.9818	0.9867	0.9874	**0.9876**	0.9875
delicious	0.9807	0.9811	0.9815	0.9814	0.9821	0.9822	**0.9824**

Table 2: Hamming accuracies of different methods across all the datasets. Every accuracy is the average after 3 runs.

branch) for the reconstruction. $\lambda_1, \lambda_2, \lambda_3$ control the weights of the three loss terms in each branch and are the same in each branch. β governs the information bottleneck in β-VAE. Both branches share the decoder and the Multivariate Probit module, which in turn helps and regularizes the learning of the latent subspace where $\mathbf{z}$ is embedded. The whole model can be trained end-to-end through back-propagation with Adam [Kingma and Ba, 2015] (see Alg. 1).

Interpretability of Σ_g

One thing special in MPVAE compared to other multi-label prediction methods is the global parameter Σ_g (i.e. $\Sigma_r + I$). Suppose each target/label can be represented by a vector $\mathbf{v}_i, i \in [1, L]$, the correlations can be captured by the inner product between $\mathbf{v}_i$ and $\mathbf{v}_j$. If the covariance matrix Σ_g indeed contains such correlation, by Cholesky decomposition $\Sigma_g = VV^T$, each row of V could be regarded as $\mathbf{v}_i$. By some dimension reduction tricks like t-SNE, the vectors of similar targets should be close. For illustration purposes, we show in the experiments section, using a real-world dataset (*eBird*), that Σ_g does capture such information.

4 Experiments

4.1 Datasets

MPVAE is validated on 9 real-world datasets from a variety of fields including ecology, biology, images, texts, etc. The datasets are *eBird* [Munson *et al.*, 2011], *North American fish* [Morley *et al.*, 2018], *mirflickr* [Huiskes and Lew, 2008], *NUS-WIDE*[2] [Chua *et al.*, 2009], *yeast* [Nakai and Kanehisa, 1992], *scene* [Boutell *et al.*, 2004], *sider* [Kuhn *et al.*, 2016], *bibtex* [Katakis *et al.*, 2008], and *delicious* [Tsoumakas *et al.*, 2008]. *eBird* is a crowd-sourced bird presence-absence dataset collected from birders' observations. *North American fish* (*fish*) is a fish distribution dataset collected from the trawlers in the North Atlantic. *yeast* is a biology database of the protein localization sites and *sider* is another database of drug side-effects. *mirflickr*, *NUS-WIDE* (*nuswide*), and *scene* datasets are from the image domain. Finally, the *bibtex* dataset contains a large number of BibTex files online and the *delicious* dataset contains web bookmarks.

The datasets represent a large variety of different scales. The dimensionality varies from 15 (*eBird*) to 1836 (*bibtex*). The number of labels ranges from 6 (*scene*) to 983 (*delicious*). The size of the datasets could be as high as 200,000 (*nuswide*), or as low as 1427 (*sider*). Most datasets are available on a public website[3]. The rest can be found in the related papers. If a dataset has been split *a priori*, we follow those divisions. Otherwise, we separate the dataset into training (80%), validation (10%) and testing (10%). The datasets are also preprocessed to fit the requirements of the input formats for the different methods. For example, we expand the input features with word embeddings for LaMP.

4.2 Implementation and Model Comparison

The encoders and decoder of MPVAE are parameterized by 3-layer fully connected neural networks with latent dimensionalities 512 and 256. The compared models share the same neural network structure for fair comparison, in cases where neural networks are used. The activation function in the neural networks is set to ReLU. Σ_r is a shared learnable parameter of size $L \times L$. By default, we set $\beta = 1.1, \lambda_1 = \lambda_3 = 0.5, \lambda_2 = 10.0$. These default hyperparameter values are inherited from the existing well-trained DMVP [Chen *et al.*, 2018] and β-VAE [Higgins *et al.*, 2017] models. We achieve the best performance for our own model in the neighborhood of this default set of parameters via grid search. We also use grid search to find the best learning rate, learning rate decay ratio and dropout ratio hyperparameters. Note that since not all datasets have sparse labels, λ_3 could be set to 0 or close to 0 in such scenarios. In practice, we found larger λ_2 gives better performance because the average ranking loss of mul-

[2] We only use the 128-d cVALD features as the input.

[3] http://mulan.sourceforge.net/datasets-mlc.html

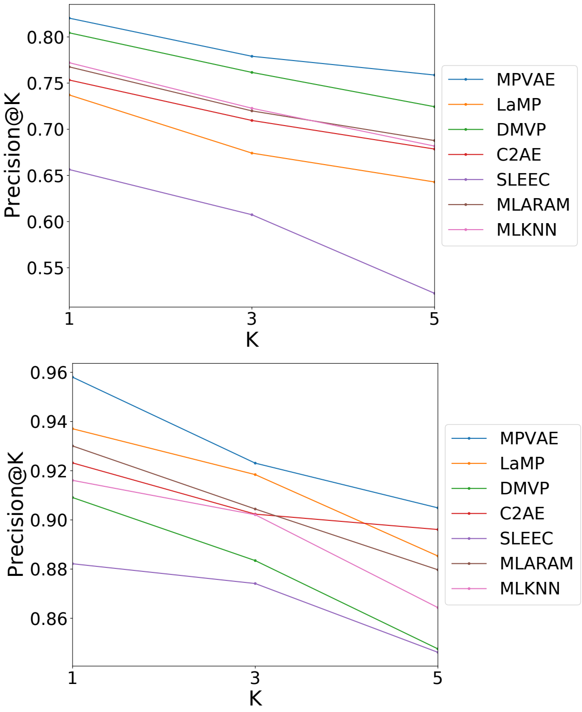

Figure 2: Top: Precision@K evaluations on *eBird*. Bottom: Precision@K evaluations on *sider*.

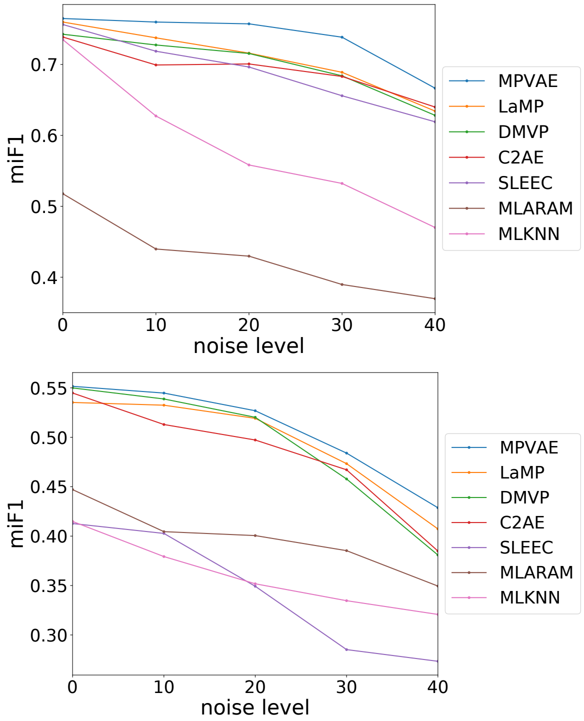

Figure 3: Micro-F1 scores of the methods under different noise levels. Top: *fish* dataset. Bottom: *mirflickr* dataset.

tiple samples could provide good guidance for training the model. β is the tradeoff between the capacity of the information bottleneck and the learnability of the decoder. In our experiments, the best values for β are in the vicinity of 1.1.

MPVAE is compared with 6 other state-of-the-art methods for multi-label prediction. MLKNN [Zhang and Zhou, 2007] is a nearest neighbor based algorithm. Bayesian inference is applied for testing. MLARAM [Benites and Sapozhnikova, 2015] is a scalable extension to the adaptive resonance associative map neural network designed for large-scale multi-label classification. SLEEC [Bhatia *et al.*, 2015] learns a small ensemble of embeddings preserving local distances. It makes low-rank assumptions and can be improved with neural networks (implemented for comparison). C2AE [Yeh *et al.*, 2017] is a recently proposed approach that learns a deep latent space through an autoencoder structure. Features and labels are encoded through deep neural networks into a latent space, where the latent embeddings for features and labels are associated by deep canonical correlation analysis (DCCA). DMVP [Chen *et al.*, 2018] is proposed for joint likelihood modeling but can be used for prediction if the trained model follows the sampling process in section 3.3 in the test phase. LaMP [Lanchantin *et al.*, 2019] is the state-of-the-art GNN-based model for multi-label prediction. It encodes the correlations among labels to a GNN, and predicts unseen instances with the trained GNN.

The major evaluation metrics for multi-label predictions are example-based F1 (example-F1), micro-averaged F1 (micro-F1) and macro-averaged F1 (macro-F1) scores.

Example-F1 measures the proportion of true positive predictions among the aggregation of the positive ground-truth labels and positive predicted labels: $\frac{1}{N_t}\sum_{i=1}^{N_t}\frac{\sum_{j=1}^{L}2y_j^i\hat{y}_j^i}{\sum_{j=1}^{L}y_j^i+\sum_{j=1}^{L}\hat{y}_j^i}$, where N_t is the number of test samples, y_j^i is the j-th actual label of test sample i and $\hat{y}_j^i$ is the j-th predicted label of test sample i. The **top** table in Table 1 shows the performances of different methods on all the datasets. Each F1 score is the average of 3 runs (same for the numbers in other tables and figures). MPVAE outperforms other methods on this metric. On average, MPVAE yields a 6% improvement compared to LaMP and a 9% improvement compared to C2AE. Micro-F1 computes the average F1 scores over all samples: $\frac{\sum_{j=1}^{L}\sum_{i=1}^{N_t}2y_j^i\hat{y}_j^i}{\sum_{j=1}^{L}\sum_{i=1}^{N_t}[2y_j^i\hat{y}_j^i+(1-y_j^i)\hat{y}_j^i+y_j^i(1-\hat{y}_j^i)]}$. The results are given in the **middle** table in Table 1. MPVAE is only slightly worse than DMVP on *bibtex*, but compared to DMVP, MPVAE performs better by 2.5% on average. The third F1-related metric is the macro-F1 score, which is the averaged F1 score over all labels: $\frac{1}{L}\sum_{j=1}^{L}\frac{\sum_{i=1}^{N_t}2y_j^i\hat{y}_j^i}{\sum_{i=1}^{N_t}[2y_j^i\hat{y}_j^i+(1-y_j^i)\hat{y}_j^i+y_j^i(1-\hat{y}_j^i)]}$. Results on the **bottom** table in Table 1 illustrate that MPVAE outperforms other methods except on *delicious* (second best).

Besides the 3 most commonly used metrics, we test MPVAE on 2 other metrics: Hamming accuracy, and Precision@K. Hamming accuracy measures how many labels are predicted correctly: $\frac{1}{N_t}\frac{1}{L}\sum_{i=1}^{N_t}\sum_{j=1}^{L}\mathbb{1}[y_j^i = \hat{y}_j^i]$, regardless of pos/neg. Accuracies are collected in Table 2. The other metric Precision@K is defined as the percent-

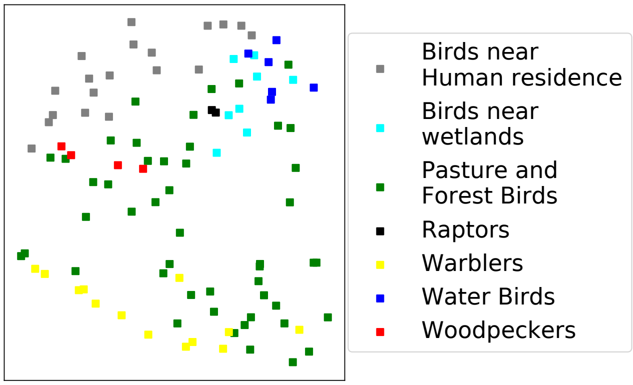

Figure 4: t-SNE plot of the decomposed vectors representing each species. Different colors denote different categories of birds.

age of correctly predicted labels in the top-K predictions. MPVAE is validated on two datasets *eBird* and *sider*, w.r.t. Precision@K (Fig. 2). The definition, implementation and threshold selection on the validation set for the evaluation metrics follow the paper [Lanchantin *et al.*, 2019].

4.3 Noisy Labels

Noisy labels are quite common in real-world datasets. For example, in trawl survey data of fish, the raw collected presence or absence of species might be misrecorded [Carton *et al.*, 2018]. Though the datasets we use have been cleaned and calibrated, the noisy setting can be reproduced by randomly flipping the labels in the training data. We tested all the methods on *fish* and *mirflickr* w.r.t. 4 noise levels: 10%, 20%, 30% and 40%. The comparisons are demonstrated in Fig. 3. As the noise level increases, MPVAE is still the most robust one. This is because when the latent dimensionality of VAE is relatively small, the model is forced to focus on the strong patterns and ignore the noise. The global covariance matrix also helps with the robustness. But as the noise level reaches 30% and beyond, all the methods perform much worse since the noise affects the whole distribution.

4.4 Interpreting the Covariance Σ_g

We validate the interpretability of Σ_g on *eBird* dataset. As we mentioned in section 3.3, Σ_g can be decomposed as VV^T. We regard each row of V as $\mathbf{v}_i$ and plot these vectors using t-SNE (see Fig. 4). One can observe that birds in the same category are clustered together. Similar clusters are also close to each other; e.g., water birds and birds near wetlands have similar embeddings. Since forest and pasture birds are the most commonly seen birds, it's not surprising that they spread across the plot. In contrast, the embeddings of rare raptors are close together. The bird categories and habits are collected from experts and professional websites[4].

5 Conclusion

In this paper, we propose a disentangled Variational Autoencoder based framework incorporating covariance-aware Multivariate Probit model (MPVAE) for multi-label prediction. MPVAE comprises a feature encoder, a label encoder, a shared decoder and a Multivariate Probit model. Encoders are learned for the features and labels respectively to map them to a probabilistic subspace. The samples from the subspaces are decoded under the Multivariate Probit model to give the prediction. The disentangled β-VAE module improves the label embedding learning as well as feature embedding learning. The Multivariate Probit module provides a simple and convenient way to capture the label correlations. More importantly, we claim that the learned covariance matrix in the MP model is interpretable as shown in a real-wolrd dataset. MPVAE performs favorably against other state-of-the-art methods on 9 public datasets and remains effective under noisy settings, which verifies the usefulness and robustness of our proposed model.

Acknowledgments

This work is supported by National Science Foundation awards OIA-1936950 and CCF-1522054. We also want to thank the Cornell Lab of Ornithology and Gulf of Maine Research Institute for providing data, resources and advice.

References

[Alazaidah *et al.*, 2015] Raed Alazaidah, Fadi Thabtah, and Qasem Al-Radaideh. A multi-label classification approach based on correlations among labels. *International Journal of Advanced Computer Science and Applications*, 2015.

[Benites and Sapozhnikova, 2015] Fernando Benites and Elena Sapozhnikova. Haram: a hierarchical aram neural network for large-scale text classification. In *2015 IEEE International Conference on Data Mining Workshop (ICDMW)*, pages 847–854. IEEE, 2015.

[Bhatia *et al.*, 2015] Kush Bhatia, Himanshu Jain, Purushottam Kar, Manik Varma, and Prateek Jain. Sparse local embeddings for extreme multi-label classification. In *Advances in neural information processing systems*, 2015.

[Bi and Kwok, 2014] Wei Bi and James T Kwok. Multilabel classification with label correlations and missing labels. In *Proceedings of the Twenty-Eighth AAAI Conference on Artificial Intelligence*, pages 1680–1686, 2014.

[Boutell *et al.*, 2004] Matthew R Boutell, Jiebo Luo, Xipeng Shen, and Christopher M Brown. Learning multi-label scene classification. *Pattern recognition*, 37(9):1757–1771, 2004.

[Carton *et al.*, 2018] James A Carton, Gennady A Chepurin, and Ligang Chen. Soda3: A new ocean climate reanalysis. *Journal of Climate*, 31(17):6967–6983, 2018.

[Chen and Lin, 2012] Yao-Nan Chen and Hsuan-Tien Lin. Feature-aware label space dimension reduction for multi-label classification. In *Advances in Neural Information Processing Systems*, pages 1529–1537, 2012.

[Chen *et al.*, 2017] Di Chen, Yexiang Xue, Daniel Fink, Shuo Chen, and Carla P Gomes. Deep multi-species embedding. In *Proceedings of the 26th International Joint*

[4] https://ebird.org/home

Conference on Artificial Intelligence, pages 3639–3646, 2017.

[Chen *et al.*, 2018] Di Chen, Yexiang Xue, and Carla Gomes. End-to-end learning for the deep multivariate probit model. In *International Conference on Machine Learning*, pages 932–941, 2018.

[Chen *et al.*, 2019a] Chen Chen, Haobo Wang, Weiwei Liu, Xingyuan Zhao, Tianlei Hu, and Gang Chen. Two-stage label embedding via neural factorization machine for multi-label classification. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 33, pages 3304–3311, 2019.

[Chen *et al.*, 2019b] Zhao-Min Chen, Xiu-Shen Wei, Peng Wang, and Yanwen Guo. Multi-label image recognition with graph convolutional networks. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 5177–5186, 2019.

[Chiang *et al.*, 2012] Tsung-Hsien Chiang, Hung-Yi Lo, and Shou-De Lin. A ranking-based knn approach for multi-label classification. In *Asian Conference on Machine Learning*, pages 81–96, 2012.

[Chu *et al.*, 2018] Hong-Min Chu, Chih-Kuan Yeh, and Yu-Chiang Frank Wang. Deep generative models for weakly-supervised multi-label classification. In *Proceedings of the European Conference on Computer Vision*, 2018.

[Chua *et al.*, 2009] Tat-Seng Chua, Jinhui Tang, Richang Hong, Haojie Li, Zhiping Luo, and Yantao Zheng. Nus-wide: a real-world web image database from national university of singapore. In *Proceedings of the ACM international conference on image and video retrieval*, pages 1–9, 2009.

[Chung *et al.*, 2015] Junyoung Chung, Kyle Kastner, Laurent Dinh, Kratarth Goel, Aaron C Courville, and Yoshua Bengio. A recurrent latent variable model for sequential data. In *Advances in neural information processing systems*, pages 2980–2988, 2015.

[Evans *et al.*, 2017] Daniel M Evans, Judy P Che-Castaldo, Deborah Crouse, Frank W Davis, Rebecca Epanchin-Niell, Curtis H Flather, R Kipp Frohlich, Dale D Goble, Ya-Wei Li, and Timothy D Male. Species recovery in the united states: increasing the effectiveness of the endangered species act. *Issues in Ecology*, 2017.

[Gomes *et al.*, 2019] Carla Gomes, Thomas Dietterich, Christopher Barrett, Jon Conrad, Bistra Dilkina, Stefano Ermon, Fei Fang, Andrew Farnsworth, Alan Fern, Xiaoli Fern, et al. Computational sustainability: Computing for a better world and a sustainable future. *Communications of the ACM*, 62(9):56–65, 2019.

[Higgins *et al.*, 2017] Irina Higgins, Loic Matthey, Arka Pal, Christopher Burgess, Xavier Glorot, Matthew Botvinick, Shakir Mohamed, and Alexander Lerchner. β-vae: Learning basic visual concepts with a constrained variational framework. *The Eighth International Conference on Learning Representations*, 2(5):6, 2017.

[Huiskes and Lew, 2008] Mark J. Huiskes and Michael S. Lew. The mir flickr retrieval evaluation. In *MIR '08: Proceedings of the 2008 ACM International Conference on Multimedia Information Retrieval*, New York, NY, USA, 2008. ACM.

[Katakis *et al.*, 2008] Ioannis Katakis, Grigorios Tsoumakas, and Ioannis Vlahavas. Multilabel text classification for automated tag suggestion. *Discovery Challenge in Joint European Conference on Machine Learning and Knowledge Discovery in Databases (ECML-PKDD)*, page 75, 2008.

[Kingma and Ba, 2015] Diederik P Kingma and Jimmy Ba. Adam: A method for stochastic optimization. *International conference on learning representation*, 2015.

[Kuhn *et al.*, 2015] Michael Kuhn, Ivica Letunic, Lars Juhl Jensen, and Peer Bork. The sider database of drugs and side effects. *Nucleic acids research*, 44(D1):D1075–D1079, 2015.

[Kuhn *et al.*, 2016] Michael Kuhn, Ivica Letunic, Lars Juhl Jensen, and Peer Bork. The sider database of drugs and side effects. *Nucleic acids research*, 44(D1):D1075–D1079, 2016.

[Lanchantin *et al.*, 2019] Jack Lanchantin, Arshdeep Sekhon, and Yanjun Qi. Neural message passing for multi-label classification. *Joint European Conference on Machine Learning and Knowledge Discovery in Databases*, 2019.

[Morley *et al.*, 2018] James W. Morley, Rebecca L. Selden, Robert J. Latour, Thomas L. Frölicher, Richard J. Seagraves, and Malin L. Pinsky. Projecting shifts in thermal habitat for 686 species on the north american continental shelf. *PLOS ONE*, 13(5):1–28, 05 2018.

[Munson *et al.*, 2011] M Arthur Munson, Kevin Webb, Daniel Sheldon, Daniel Fink, Wesley M Hochachka, Marshall Iliff, Mirek Riedewald, Daria Sorokina, Brian Sullivan, Christopher Wood, et al. The ebird reference dataset. *Cornell Lab of Ornithology and National Audubon Society*, 2011.

[Nakai and Kanehisa, 1992] Kenta Nakai and Minoru Kanehisa. A knowledge base for predicting protein localization sites in eukaryotic cells. *Genomics*, 14(4):897–911, 1992.

[Read *et al.*, 2009] Jesse Read, Bernhard Pfahringer, Geoff Holmes, and Eibe Frank. Classifier chains for multi-label classification. In *Joint European Conference on Machine Learning and Knowledge Discovery in Databases*, pages 254–269. Springer, 2009.

[Tsoumakas *et al.*, 2008] Grigorios Tsoumakas, Ioannis Katakis, and Ioannis Vlahavas. Effective and efficient multilabel classification in domains with large number of labels. In *Proceedings of ECML-PKDD 2008 Workshop on Mining Multidimensional Data*, pages 53–59, 2008.

[van Steenkiste *et al.*, 2019] Sjoerd van Steenkiste, Francesco Locatello, Jürgen Schmidhuber, and Olivier Bachem. Are disentangled representations helpful for abstract visual reasoning? In *Advances in Neural*

Information Processing Systems, pages 14222–14235, 2019.

[Wang *et al.*, 2016] Jiang Wang, Yi Yang, Junhua Mao, Zhiheng Huang, Chang Huang, and Wei Xu. Cnn-rnn: A unified framework for multi-label image classification. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 2285–2294, 2016.

[Wu *et al.*, 2018] Baoyuan Wu, Fan Jia, Wei Liu, Bernard Ghanem, and Siwei Lyu. Multi-label learning with missing labels using mixed dependency graphs. *International Journal of Computer Vision*, 126(8):875–896, 2018.

[Yeh *et al.*, 2017] Chih-Kuan Yeh, Wei-Chieh Wu, Wei-Jen Ko, and Yu-Chiang Frank Wang. Learning deep latent space for multi-label classification. In *Thirty-First AAAI Conference on Artificial Intelligence*, 2017.

[Yu *et al.*, 2014] Hsiang-Fu Yu, Prateek Jain, Purushottam Kar, and Inderjit Dhillon. Large-scale multi-label learning with missing labels. In *International conference on machine learning*, pages 593–601, 2014.

[Zhang and Yeung, 2013] Yu Zhang and Dit-Yan Yeung. Multilabel relationship learning. *ACM Transactions on Knowledge Discovery from Data (TKDD)*, 7(2):7, 2013.

[Zhang and Zhou, 2007] Min-Ling Zhang and Zhi-Hua Zhou. Ml-knn: A lazy learning approach to multi-label learning. *Pattern recognition*, 40(7):2038–2048, 2007.

[Zhang and Zhou, 2013] Min-Ling Zhang and Zhi-Hua Zhou. A review on multi-label learning algorithms. *IEEE transactions on knowledge and data engineering*, 26(8):1819–1837, 2013.

[Zhang *et al.*, 2018] Min-Ling Zhang, Yu-Kun Li, Xu-Ying Liu, and Xin Geng. Binary relevance for multi-label learning: an overview. *Frontiers of Computer Science*, 12(2):191–202, 2018.