# Gaussian Mixture Variational Autoencoder with Contrastive Learning for Multi-Label Classification

**Junwen Bai**
Cornell University
jb2467@cornell.edu

**Shufeng Kong**
Cornell University
sk2299@cornell.edu

**Carla Gomes**
Cornell University
gomes@cs.cornell.edu

## Abstract

Multi-label classification (MLC) is a prediction task where each sample can have more than one label. We propose a novel contrastive learning boosted multi-label prediction model based on a Gaussian mixture variational autoencoder (C-GMVAE), which learns a multimodal prior space and employs a contrastive loss. Many existing methods introduce extra complex neural modules to capture the label correlations, in addition to the prediction modules. We find that by using contrastive learning in the supervised setting, we can exploit label information effectively, and learn meaningful feature and label embeddings capturing both the label correlations and predictive power, without extra neural modules. Our method also adopts the idea of learning and aligning latent spaces for both features and labels. More specifically, C-GMVAE imposes a Gaussian mixture structure on the latent space, to alleviate posterior collapse and over-regularization issues, in contrast to previous works based on a unimodal prior. C-GMVAE outperforms existing methods on multiple public datasets and can often match other models' full performance with only 50% of the training data. Furthermore, we show that the learnt embeddings provide insights into the interpretation of label-label interactions.

## 1 Introduction

In many machine learning tasks, an instance can have several labels. The task of predicting multiple labels is known as multi-label classification (MLC). MLC is common in domains like vision [1], natural language [2], biology [3]. Unlike the single-label scenario, label correlations are more important in MLC. Early works capture the correlations through classifier chains [4], bayesian inference [5], and dimensionality reduction [6].

Boosted by the huge capacity of neural networks (NN), many previous methods can be improved by their neural extensions. For example, classifier chains can be naturally enhanced by RNN [1]. The non-linearity of NN alleviates the design complexity of feature mapping and many deep models can therefore focus on the loss function, feature-label and label-label correlation modeling.

One trending direction is to learn a deep latent space shared by features and labels. The samples from the latent space are then decoded to targets. One typical example is C2AE [7], which learns latent codes for both features and labels. The latent codes are passed through a decoder to obtain target labels. C2AE minimizes an $\ell_2$ distance between feature and label codes, together with a relaxed orthogonality regularization. However, the learnt deterministic latent space lacks smoothness and structures. Small perturbations in this latent space can lead to totally different decoding results. Even if the corresponding feature and label codes are close, we cannot guarantee the decoded targets are similar. To address this concern, MPVAE [8] proposes to replace the deterministic latent space with a probabilistic subspace under a variational autoencoder (VAE) framework. The Gaussian latent spaces are aligned with KL-divergence, and the sampling process enforces smoothness. Similar ideas can be found in [9]. However, these methods assume a unimodal Gaussian latent space, which is

NeurIPS 2021 Workshop on Deep Generative Models and Downstream Applications

known to cause over-regularization and posterior collapse [10, 11]. A better strategy would be to learn a multimodal latent space. It is more intuitive to assume the observed data are generated from a multimodal subspace rather than from a unimodal one.

Another popular group of methods targets on better label correlation modeling. Their idea is straightforward: some labels should be more correlated if they co-appear often while others should be less relevant. Existing methods adopt pairwise ranking loss, covariance matrix, conditional random field or even graph neural nets (GNN) [12, 13, 14, 15]. These methods often either constrain the learning through a predefined structure (which requires larger space), or aren't powerful enough to capture the correlations (such as pairwise ranking loss).

Our idea is simple: we learn embeddings for each label class and the inner product between embeddings should reflect the similarity. We further learn feature embeddings whose inner products with label embeddings correspond to feature-label similarity and can be used for prediction. We assume these embeddings are generated from a probabilistic multimodal latent space shared by features and labels, where we use KL-divergence to align the feature latent distribution and label latent distribution. On the other hand, one may concern that embeddings might not be able to capture label-label, label-feature correlations, thus requiring extra GNN and covariance matrix [15, 8]. To this end, we use pure losses rather than extra structures to capture these correlations. Intuitively, if two labels co-appear often, their embeddings should be close. If two labels seldom co-appear, their embeddings should be distant. A triplet-like loss could be naturally applied in this scenario. Nevertheless, more powerful contrastive loss has shown to be more effective than the triplet loss by introducing more samples rather than just one triplet. We show that contrastive loss can pull together correlated labels, and push away unrelated labels (see Fig. 3), which performs even better than GNN-based or covariance-based methods.

Our new method, the contrastive learning boosted Gaussian mixture variational autoencoder (C-GMVAE) multi-label prediction model, alleviates the over-regularization and posterior collapse concerns, as well as learns useful feature and label embeddings. C-GMVAE is applied to nine datasets and outperforms the existing methods on five metrics. Furthermore, we show that often with only 50% of the data, our results can match the full performance of other state-of-the-art methods. Ablation studies and interpretability of learnt embeddings will also be illustrated in the experiments. Our contributions can be summarized into three aspects: **(i)** We propose to use contrastive loss instead of triplet or ranking loss to strengthen the label embedding learning. We empirically show that by using a contrastive loss, one can get rid of heavy-duty label correlation modules (e.g. covariance matrix, GNN) while achieving even better performances. **(ii)** Though contrastive learning is commonly applied in self-supervised learning, our work shows that by properly defining anchor, positive and negative samples, contrastive loss can leverage label information very effectively in the supervised MLC scenario as well. **(iii)** Unlike prior probabilistic models, C-GMVAE learns a multimodal latent space and associates the probabilistic modeling (VAE module) with embedding learning (contrastive module) synergistically.

## 2 Methods

In MLC, given the dataset containing $N$ samples (with labels) $(x, y)$, where $x \in \mathbb{R}^D$ and $y \in \{0, 1\}^L$, our goal is to find a mapping from $x$ to $y$. $N, D, L$ are sample number, feature length and label set size respectively. The binary coding indicates the labels associated with the sample $x$. Labels are correlated with each other.

### 2.1 Preliminaries

#### 2.1.1 Gaussian Mixture VAE

A standard VAE [16] pulls together the posterior distribution and a parameter-free isotropic Gaussian prior. Two losses are optimized together in training: KL-divergence between the prior and posterior, and the distance between recontructed targets and real targets. One weakness of this formulation is the unimodality of the latent space, inhibiting the learning of more complex representations. Another concern is over-regularization. If the posterior is exactly the same as the prior, the learnt representations would be uninformative of training inputs. Numerous works extend the prior to be more complex [17, 18, 10] or learn deep hierarchies of latent variables [19, 20, 21]. In our

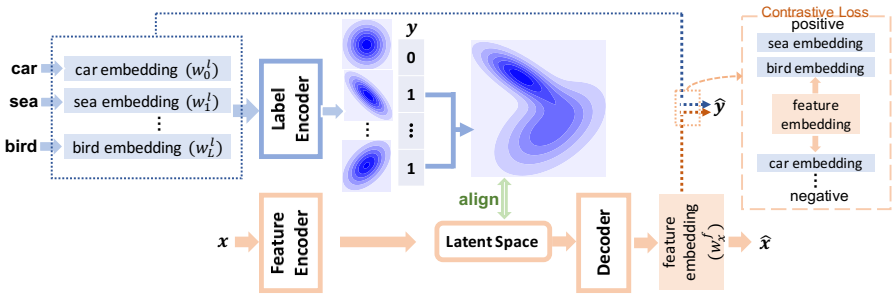


Figure 1: The full pipeline of C-GMVAE. Every label is mapped to a learnable embedding first. The label encoder transforms each embedding $w_i^l$ to a multivariate Gaussian latent space. The sample's associated label set selects the related latent label subspace and forms a Gaussian mixture distribution. Each feature is also mapped to a latent space through a feature encoder. The posterior is aligned with the prior via KL-divergence. The decoder takes in a sample from the latent space and produces a feature embedding $w_x^f$. A contrastive loss is designed to pull together the feature embedding and positive label embeddings, while separating the feature embedding from negative label embeddings. Prediction $\hat{y}$ is made by passing the feature-label embedding inner products to the sigmoid functions. In the figure, a sample with label set {sea, bird} is provided.

work, we adopt the Gausian mixture prior. The probability density can be depicted as $p(z) = \frac{1}{k}\sum_{i=1}^{k}\mathcal{N}(z|\mu_i, \sigma_i^2)$ where $i$ is the cluster index of $k$ Gaussian clusters with mean $\mu_i$ and covariance $\sigma_i^2$. Our intuition is that each label embedding could correlate to a Gaussian subspace. Given a label set, the mixture of the corresponding Gaussians forms a unique multimodal prior distribution. The label embeddings also receive the gradients from the contrastive loss and thus combine the contrastive learning and latent space construction.

#### 2.1.2 Contrastive Learning

We propose to use contrastive learning to capture the correlations (feature-label, label-label). Contrastive learning [22, 23, 24] is a novel learning style. The core idea is simple: given an anchor sample, it should be close to similar samples (positive) and far from dissimilar samples (negative) in some learnt embedding space. It differs from triplet loss in the number of negative samples and the way of loss estimation. Contrastive loss is largely motivated by the noise contrastive estimation (NCE) [25] and its form is generalizable. The raw contrastive loss formulation only considers the instance-level invariance (multiple views of one instance), but with label information, we can learn category-level invariance (multiple instances per class/category) [26]. In the multi-label scenario, one can regard the feature embedding as the anchor sample, positive label embeddings as positive samples and negative label embeddings as negative samples. The formulation can fit the contrastive learning framework naturally and is one of our major contributions. Compared to pairwise ranking loss which focuses on the final digits, contrastive loss is defined on the embeddings and thus more expressive. Contrastive loss also includes more samples in estimating the NCE and therefore outperforms triplet loss. In appendix, we show triplet loss is actually a special case of our contrastive loss.

### 2.2 C-GMVAE

C-GMVAE inherits the general variational autoencoder framework, but with a learnable Gaussian mixture prior. During training, the sample's label set activates and mixes the related Gaussian clusters to derive the prior. Contrastive learning is applied to boost the embedding learning, using a contrastive loss between the feature and label embeddings. Fig. 1 provides a full illustration, and the following subsections will elaborate on the details.

#### 2.2.1 Gaussian Mixture Latent Space

Given a sample $(x, y)$ where feature $x \in \mathbb{R}^D$ and label $y \in \{0, 1\}^L$, many previous works take $y$ as the input and transform it to a dense representation through a fully-connected layer [7, 8]. This layer essentially maps each label category to an embedding and sums up all the embeddings using label

$y$ as weights (0 or 1). The final embedding is fed into the label encoder to produce a probabilistic space. In C-GMVAE, we directly map each per-category label embedding $w_i^l \in \mathbb{R}^E$ of label class $i$ to an individual Gaussian distribution $\mathcal{N}(\mu_i, diag(\sigma_i^2))$, $\mu_i \in \mathbb{R}^d, \sigma_i^2 \in \mathbb{R}^d$. $\mu_i, \sigma_i^2$ are derived from $w_i^l$ through NN. In Fig. 1, the label categories *car, sea,..., bird* are transformed to embeddings first. Embeddings are then directly passed to label encoder rather than summed up. Each label category (e.g. *car*) corresponds to a unimodal Gaussian in the latent space. $y$ activates "positive" Gaussians and forms a Gaussian mixture subspace. Given a random variable $z \in \mathbb{R}^d$, the probability density function (PDF) in the subspace is defined as

$$p_\psi(z|y) = \frac{1}{\sum_i y_i} \sum_{i=1}^{L} \mathbb{1}\{y_i = 1\} \mathcal{N}(z|\mu_i, diag(\sigma_i^2)) \tag{1}$$

$\mathbb{1}(\cdot)$ is the indicator function and the label encoder is parameterized by $\psi$ (NN). In Fig. 1, $y$ activates *sea* and *bird*, $p_\psi(z|y) = \frac{1}{2}(\mathcal{N}(z|\mu_{sea}, diag(\sigma_{sea}^2)) + \mathcal{N}(z|\mu_{bird}, diag(\sigma_{bird}^2)))$.

Most VAE-based frameworks optimize over an evidence lower bound (ELBO) [27]:

$$\text{ELBO} = \mathbb{E}_{q_\phi(z|x)}[\log p_\theta(x|z)] - D_{KL}[q_\phi(z|x)||p(z)] \tag{2}$$

The feature encoder is parameterized by $\phi$ (NN). One pitfall of this objective is owing to the minimization of KL-divergence. If the divergence between the posterior $q_\phi(z|x)$ and the prior $p_\psi(z)$ vanishes, the learnt latent codes would be non-informative. This is the so-called posterior collapse. Many recent works suggest learnable priors [28] and more sophisticated priors [29] to avoid this issue. We adopt these ideas in our design of the prior. Compared to a standard VAE, our prior is informative, learnable and multimodal.

We form a standard posterior in our model and match it with the prior. However, unlike vanilla VAE, we cannot analytically compute the KL term. Instead, we use the following estimation:

$$\begin{aligned} \mathcal{L}_{KL} \approx & \log q_\phi(z_0|x) - \log p_\psi(z_0|y) \\ = & \log \mathcal{N}(z_0|\mu_\phi(x), diag(\sigma_\phi^2(x))) - \\ & \log \frac{1}{\sum_i y_i} \sum_{i=1}^{L} \mathbb{1}\{y_i = 1\} \mathcal{N}(z_0|\mu_i, diag(\sigma_i^2)) \end{aligned} \tag{3}$$

where $z_0 \sim q_\phi(z|x)$ denotes a single latent sample.

The reconstruction loss remains to be a standard negative log-likelihood (we add the minus since the objective function is to be minimized. $\theta$ is the decoder parameters.),

$$\mathcal{L}_{recon} = -E_{q_\phi(z|x)}[\log p_\theta(x|z)] \tag{4}$$

### 2.2.2 Contrastive Learning Module

The decoder function $f_\theta^d(\cdot)$ decodes the sample from the latent space to a feature embedding $w_x^f \in \mathbb{R}^E$. We train $w_x^f$ together with label embeddings $\{w_i^l\}_{i=1}^L$. The objective function includes contrastive loss and cross-entropy loss terms.

Prior works used to explicitly capture the label-label interactions by GNN or covariance modules, which imposes the structure *a priori* and might not be the best way. Our contrastive module instead captures the correlation completely driven by data. For example, if in the majority of all samples, "beach" and "sunshine" appear together, the contrastive learning will implicitly pull their embeddings together [24]. In other words, if two labels do co-appear often, their label embeddings become similar (Fig. 3). On the other hand, if they only co-appear occasionally, their relations are not significant and our module will not optimize for their similarity.

Original contrastive learning [22] augments inputs and learns instance-level invariance, but may not generalize to category-level invariance. In the supervised setting, however, the learning can benefit from labels and discover category-level invariance [24]. Let $A \equiv \{1...L\}$. We define $P(y) \equiv \{i \in A : y_i = 1\}$ for sample $(x, y)$. Suppose we have a batch of samples, $\mathcal{B}$, the contrastive loss can be written as

$$\mathcal{L}_{CL} = \frac{1}{|\mathcal{B}|} \sum_{(x,y)\in\mathcal{B}} \frac{1}{|P(y)|} \sum_{p\in P(y)} -\log \frac{\text{sim}(w_x^f, w_p^l)}{\sum_{t\in A} \text{sim}(w_x^f, w_t^l)} \tag{5}$$

$\text{sim}(\cdot)$ is a function measuring the similarity between two embeddings, and $w_x^f$, $w_i^l$ denote the feature and label embeddings respectively. Eq. 5 is built on top of noise-contrastive estimation [25], and the equation is equivalent to a categorical cross-entropy of correctly predicting positive labels. The choice of $\text{sim}(\cdot)$ can be a log-bilinear function [22], or a more complicated neural metric function [23]. In our experiments, we found it is simple and effective to take $\text{sim}(w_1, w_2) = \exp(w_1 \cdot w_2/\tau)$ where $\cdot$ means inner product and $\tau$ is a temperature parameter controlling the scale of the inner products. In SupCon [24], if only 1 class is positive, all other classes are contrastive to it. However, in multi-label, if "beach" is positive in the label while "sea" isn't for one particular sample, we cannot say these two classes are contrastive. Their correlation will be captured implicitly by all the samples. Therefore, we do not assume contrastive relations between labels and preserve the label correlations. We instead choose the feature embedding to be the anchor and label embeddings to be positive/negative samples. If two label embeddings co-appear often as positive samples, they would implicitly become similar (see Fig. 3). Eq. 5 saves the effort of manually configuring the positive and negative samples, and is totally data-driven. The number of positives or negatives could be greater than one. Note that though $L$ limits the max samples we can have, this formulation has already used many more samples compared to triplet loss, and we will show in experiments that this formulation is very effective.

The triplet loss often used in multi-label learning [30] can be seen as a special case of Eq. 5 with only one positive and one negative. We illustrate the connection in the appendix. Furthermore, one desired property of embedding learning is that when a good positive embedding is already close enough to our anchor embedding, it contributes less to the gradients, while poorly learnt embeddings contribute more to improve the model performance. It has been shown that the contrastive loss can implicitly achieve this goal and a full derivation of the gradients [24].

Our objective function also includes a supervised cross-entropy loss term to further facilitate the training. With the label embeddings $w_i^l$ and the feature embedding $w_x^f$, the cross entropy loss is given by

$$\mathcal{L}_{CE} = \frac{1}{|\mathcal{B}|} \sum_{(x,y)\in\mathcal{B}} \sum_{i=1}^{L} y_i \log s(w_x^f w_i^l) + (1 - y_i) \log(1 - s(w_x^f w_i^l)) \tag{6}$$

where function $s(\cdot)$ is the sigmoid function. In self-supervised learning, the contrastive loss typically helps the pretraining stage and the learnt representations are applied to downstream tasks. In the supervised setting, though some models [24] stick to the two-stage training process where the model is trained with contrastive loss in the first stage and cross-entropy loss in the second stage, we didn't observe its superiority to the one-stage scheme where we train the model with an objective function incorporating all losses. This is partly because we also learn a latent space that is closely connected to label embeddings. A joint training strategy reconciles different modules. We show in the experiments that the learnt embeddings are semantically meaningful and can reveal the label correlations.

#### 2.2.3 Objective Function

The final objective function is simply the summation of different losses,

$$\mathcal{L} = \mathcal{L}_{KL} + \mathcal{L}_{recon} + \alpha\mathcal{L}_{CL} + \beta\mathcal{L}_{CE} \tag{7}$$

where $\alpha, \beta$ are trade-off weights. The model is trained with Adam [31]. Our model is optimized with $\mathcal{L}$ but will be tested on five different metrics. This is different from the methods that optimize specific metrics [32, 33].

#### 2.2.4 Prediction

During the testing phase, the input sample $x$ will be passed to the feature encoder and decoder to obtain its embedding $w_x^f$. Label embeddings $w_i^l$ are fixed during testing. The inner products between $w_x^f$ and $w_i^l$ will be passed through a sigmoid function to obtain prediction probability for class $i$.

### 2.3 Insights behind C-GMVAE

C2AE and MPVAE have shown the importance of learning a shared latent space for both features and labels. These methods share the same high-level insight similar to teacher-student regime: we map labels (teacher) to a latent space with some certain structure, which preserves the label information and is easier to be decoded back to labels. Then the features (student) are expected to be mapped to

| Metric | example-F1 | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|
| Dataset | *eBird* | *mirflickr* | *nus-vec* | *yeast* | *scene* | *sider* | *reuters* | *bkms* | *delicious* |
| BR | 0.365 | 0.325 | 0.343 | 0.630 | 0.606 | 0.766 | 0.733 | 0.171 | 0.174 |
| MLKNN | 0.510 | 0.383 | 0.342 | 0.618 | 0.691 | 0.738 | 0.703 | 0.213 | 0.259 |
| HARAM | 0.510 | 0.432 | 0.396 | 0.629 | 0.717 | 0.722 | 0.711 | 0.216 | 0.267 |
| SLEEC | 0.258 | 0.416 | 0.431 | 0.643 | 0.718 | 0.581 | 0.885 | 0.363 | 0.308 |
| C2AE | 0.501 | 0.501 | 0.435 | 0.614 | 0.698 | 0.768 | 0.818 | 0.309 | 0.326 |
| LaMP | 0.477 | 0.492 | 0.376 | 0.624 | 0.728 | 0.766 | 0.906 | 0.389 | 0.372 |
| MPVAE | 0.551 | 0.514 | 0.468 | 0.648 | 0.751 | 0.769 | 0.893 | 0.382 | 0.373 |
| C-GMVAE | **0.576** | **0.534** | **0.481** | **0.656** | **0.777** | **0.771** | **0.917** | **0.392** | **0.381** |
| std (±) | 0.001 | 0.002 | 0.000 | 0.001 | 0.002 | 0.001 | 0.001 | 0.001 | 0.002 |
| Metric | micro-F1 | | | | | | | | |
| Dataset | *eBird* | *mirflickr* | *nus-vec* | *yeast* | *scene* | *sider* | *reuters* | *bkms* | *delicious* |
| BR | 0.384 | 0.371 | 0.371 | 0.655 | 0.706 | 0.796 | 0.767 | 0.125 | 0.197 |
| MLKNN | 0.557 | 0.415 | 0.368 | 0.625 | 0.667 | 0.772 | 0.680 | 0.181 | 0.264 |
| HARAM | 0.573 | 0.447 | 0.415 | 0.635 | 0.693 | 0.754 | 0.695 | 0.230 | 0.273 |
| SLEEC | 0.412 | 0.413 | 0.428 | 0.653 | 0.699 | 0.697 | 0.845 | 0.300 | 0.333 |
| C2AE | 0.546 | 0.545 | 0.472 | 0.626 | 0.713 | 0.798 | 0.799 | 0.316 | 0.348 |
| LaMP | 0.517 | 0.535 | 0.472 | 0.641 | 0.716 | 0.797 | 0.886 | 0.373 | 0.386 |
| MPVAE | 0.593 | 0.552 | 0.492 | 0.655 | 0.742 | 0.800 | 0.881 | 0.375 | 0.393 |
| C-GMVAE | **0.633** | **0.575** | **0.510** | **0.665** | **0.762** | **0.803** | **0.890** | **0.377** | **0.403** |
| std (±) | 0.001 | 0.001 | 0.000 | 0.002 | 0.002 | 0.000 | 0.001 | 0.001 | 0.002 |

Table 1: The example-F1 (ex-F1) and micro-F1 (mi-F1) scores of different methods on all datasets. C-GMVAE's numbers are averaged over 3 seeds. The standard deviation (std) is also shown. 0.000 means an std$< 0.0005$.

| Metric | macro-F1 | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|
| Dataset | *eBird* | *mirflickr* | *nus-vec* | *yeast* | *scene* | *sider* | *reuters* | *bkms* | *delicious* |
| BR | 0.116 | 0.182 | 0.083 | 0.373 | 0.704 | 0.588 | 0.137 | 0.038 | 0.066 |
| MLKNN | 0.338 | 0.266 | 0.086 | 0.472 | 0.693 | 0.667 | 0.066 | 0.041 | 0.053 |
| HARAM | 0.474 | 0.284 | 0.157 | 0.448 | 0.713 | 0.649 | 0.100 | 0.140 | 0.074 |
| SLEEC | 0.363 | 0.364 | 0.135 | 0.425 | 0.699 | 0.592 | 0.403 | 0.195 | 0.142 |
| C2AE | 0.426 | 0.393 | 0.174 | 0.427 | 0.728 | 0.667 | 0.363 | 0.232 | 0.102 |
| LaMP | 0.381 | 0.387 | 0.203 | 0.480 | 0.745 | 0.668 | 0.520 | 0.286 | 0.196 |
| MPVAE | 0.494 | 0.422 | 0.211 | 0.482 | 0.750 | 0.690 | 0.545 | 0.285 | 0.181 |
| C-GMVAE | **0.538** | **0.440** | **0.226** | **0.487** | **0.769** | **0.691** | **0.582** | **0.291** | **0.197** |
| std (±) | 0.000 | 0.001 | 0.001 | 0.002 | 0.002 | 0.002 | 0.001 | 0.001 | 0.001 |
| Metric | Hamming Accuracy | | | | | | | | |
| Dataset | *eBird* | *mirflickr* | *nus-vec* | *yeast* | *scene* | *sider* | *reuters* | *bkms* | *delicious* |
| BR | 0.816 | 0.886 | 0.971 | 0.782 | 0.901 | 0.747 | 0.994 | 0.990 | 0.982 |
| MLKNN | 0.827 | 0.877 | 0.971 | 0.784 | 0.863 | 0.715 | 0.992 | 0.991 | 0.981 |
| HARAM | 0.819 | 0.634 | 0.971 | 0.744 | 0.902 | 0.650 | 0.905 | 0.990 | 0.981 |
| SLEEC | 0.816 | 0.870 | 0.971 | 0.782 | 0.894 | 0.675 | 0.996 | 0.989 | 0.982 |
| C2AE | 0.771 | 0.897 | 0.973 | 0.764 | 0.893 | 0.749 | 0.995 | 0.991 | 0.981 |
| LaMP | 0.811 | 0.897 | 0.980 | 0.786 | 0.903 | 0.751 | 0.997 | 0.992 | 0.982 |
| MPVAE | 0.829 | 0.898 | 0.980 | 0.792 | 0.909 | 0.755 | 0.997 | 0.991 | 0.982 |
| C-GMVAE | **0.847** | **0.903** | **0.984** | **0.796** | **0.915** | **0.767** | **0.997** | **0.992** | **0.983** |
| std (±) | 0.001 | 0.000 | 0.000 | 0.002 | 0.001 | 0.003 | 0.000 | 0.000 | 0.000 |

Table 2: The macro-F1 (ma-F1) and Hamming accuracy (HA) scores of different methods on all datasets. C-GMVAE's numbers are averaged over 3 seeds.

this latent space to facilitate the label prediction. Two general concerns exist for these methods: 1) the uni-Gaussian space previously used is too restrictive to impose sophisticated structures for prior, 2) how to properly capture label correlations with embeddings. For the first, we learn a modality for each label class to form a mixture latent space. For the second, we replace the commonly used ranking and triplet losses with contrastive loss since contrastive loss involves more samples than triplet loss and has a larger capacity than ranking loss.

| Dataset | *eBird* | *mir.* | *nus-vec* | *yeast* | *scene* | *sider* | *reuters* | *bkms* | *del.* |
|---|---|---|---|---|---|---|---|---|---|
| BR | 0.598 | 0.582 | 0.443 | 0.745 | 0.700 | 0.573 | 0.752 | 0.301 | 0.485 |
| MLKNN | 0.772 | 0.491 | 0.456 | 0.730 | 0.675 | 0.916 | 0.753 | 0.310 | 0.460 |
| MLARAM | 0.768 | 0.350 | 0.404 | 0.682 | 0.722 | 0.930 | 0.679 | 0.312 | 0.419 |
| SLEEC | 0.656 | 0.623 | 0.531 | 0.745 | 0.730 | 0.882 | 0.908 | 0.415 | 0.676 |
| C2AE | 0.753 | 0.705 | 0.569 | 0.749 | 0.703 | 0.923 | 0.845 | 0.407 | 0.609 |
| LaMP | 0.737 | 0.685 | 0.456 | 0.740 | 0.746 | 0.937 | 0.927 | 0.420 | 0.663 |
| MPVAE | 0.820 | 0.726 | 0.587 | 0.743 | 0.777 | 0.958 | 0.930 | 0.437 | 0.696 |
| C-GMVAE | **0.825** | **0.732** | **0.595** | **0.751** | **0.788** | **0.962** | **0.939** | **0.465** | **0.707** |
| std (±) | 0.001 | 0.002 | 0.001 | 0.000 | 0.000 | 0.002 | 0.001 | 0.003 | 0.001 |

Table 3: The precision@1 scores of different methods on all datasets. “mir." stands for mirflickr and “del." means delicious dataset.

# 3 Related Work

Learning a shared latent space for features and labels is a common and useful idea. In single-label prediction tasks, CADA-VAE [34] learns and aligns latent label and feature spaces through distribution alignment losses. Similar ideas can be seen in out-of-distribution detection as well [9]. In multi-label scenarios, methods adopting this idea typically have a similar module that directly maps the multi-hot labels to embeddings [7, 35, 8]. This is a rather difficult learning task. Suppose we have 30 label categories. There could be up to $2^{30}$ label sets. For probabilistic models like MPVAE, that means one latent label space has to represent up to $2^{30}$ label combinations. In contrast, C-GMVAE learns per-category subspaces and forms a mixture prior distribution based on the observed samples' label sets. Learning a mixture latent space has been explored in prior works [10, 36]. But none of them applied their methods on multi-label prediction tasks.

Contrastive learning has become one of the most popular self-supervised learning techniques. It has also drawn attention in supervised tasks. SupCon [24] first demonstrated the effectiveness of supervised contrastive loss in image classification. It was soon generalized to other domains like visual reasoning [37]. Nevertheless, these methods depend on vision-specific augmentation techniques. Another related work is multi-label contrastive learning [38]. But the work does not deal with MLC. Instead, it extends the contrastive learning to identifying more than 1 positive sample, which resembles a multi-label scenario.

Some earlier works also attempted metric learning or triplet loss in MLC [39]. Triplet loss typically only takes one pair of positive and negative samples for one anchor, while contrastive loss uses many more negative/positive samples. Recent papers found that more samples can greatly boost the performance [23, 26]. Note that though our contrastive module is constrained by the maximum number of label classes, it has already used many more samples than triplet loss, and our observations support that more samples help with the performance.

# 4 Experiments

We have various setups to validate the performance of C-GMVAE. First, we compare the example-F1, micro-F1 and macro-F1 scores, Hamming accuracies and precision@1 of different methods. Second, we compare their performances when fewer training data are available. Third, an ablation study shows the importance of the proposed modules. Finally, we demonstrate the interpretability of label embeddings on an eBird dataset.

## 4.1 Setup

For the main evaluation experiments, we use nine datasets, including image datasets *mirflickr*, *nuswide*, *scene* [40, 41, 42], biology datasets *sider*, *yeast* [43, 44], ecology dataset *eBird* [45], text datasets *reuters, bookmarks, delicious* [46, 47, 48]. All features are collected in the vector format [15, 8]. The feature pre-processing is standard following previous works [15, 8] and the datasets are publicly available[1]. Each of them is separated into training (80%), validation (10%) and testing (10%) splits. The datasets are also preprocessed to fit the input formats of different methods. We use mini-batch training with batch size 128. Each batch is randomly sampled from the dataset.

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

| | variations | eb-F1 | mi-F1 | ma-F1 |
|---|---|---|---|---|
| *ebird* | uni-Gaussian | 0.545 | 0.583 | 0.490 |
| | GM only | 0.561 | 0.603 | 0.511 |
| | contrastive only | 0.558 | 0.594 | 0.515 |
| | GM+contrastive | 0.576 | 0.633 | 0.538 |
| *mirflickr* | uni-Gaussian | 0.510 | 0.541 | 0.413 |
| | GM only | 0.521 | 0.561 | 0.429 |
| | contrastive only | 0.526 | 0.565 | 0.428 |
| | GM+contrastive | 0.534 | 0.575 | 0.440 |
| *nus-vec* | uni-Gaussian | 0.461 | 0.479 | 0.203 |
| | GM only | 0.472 | 0.505 | 0.218 |
| | contrastive only | 0.470 | 0.501 | 0.213 |
| | GM+contrastive | 0.481 | 0.510 | 0.226 |

Table 4: Ablation study on the contrastive learning module and Gaussian mixture module. Note that both contrastive learning module and mixture Gaussian space are contributions in this work. GM can bring improvements consistently. Contrastive module can further boost the performance.

| | method (data %) | HA | ex-F1 | mi-F1 | ma-F1 |
|---|---|---|---|---|---|
| *ebird* | MPVAE (100%) | 0.829 | 0.551 | 0.593 | 0.494 |
| | C-GMVAE (50%) | 0.842 | 0.557 | 0.615 | 0.521 |
| *mirflickr* | MPVAE (100%) | 0.898 | 0.514 | 0.552 | 0.422 |
| | C-GMVAE (50%) | 0.899 | 0.512 | 0.553 | 0.412 |
| *nus-vec* | MPVAE (100%) | 0.980 | 0.468 | 0.492 | 0.211 |
| | C-GMVAE (50%) | 0.975 | 0.465 | 0.494 | 0.201 |

Table 5: Comparisons between MPVAE and C-GMVAE using 100% and 50% respectively.

The evaluation metrics are three F1 scores, hamming accuracy and precision@1. The evaluation process, model selection and preprocessing strictly follow previous works [49, 15, 8]. Most numbers are also directly quoted from their papers for comparison. Our method is compared against MPVAE [8], LaMP [15], C2AE [7], SLEEC [6], HARAM [50], MLKNN [5], and BR [51]. MPVAE is a novel method which learns and aligns the probabilistic feature and label subspaces. Label correlations are captured by a Multivariate Probit module. LaMP adopts attention-based neural message passing to handle label correlations, which is a neural extension to previous CRF-based methods. C2AE is one of the first papers which use neural networks to learn and align latent spaces. C2AE imposes a CCA constraint on latent space. SLEEC explores the low-rank assumption in MLC. Low-rank assumption also builds the foundation for other deep methods. HARAM is one of the first methods which introduced neural nets to MLC. Lastly, MLKNN is a classic MLC method using K-nearest neighbors (KNN).

### 4.2 Metrics

We evaluate our method trained with objective Eq. 7 on several commonly used multi-label metrics. Suppose the ground-truth label is $y$ and the predicted label is $\hat{y}$. We denote true positives, false positives, false negatives by $tp_j, fp_j, fn_j$ respectively for the $j$-th of $L$ label categors. (i) HA: $\frac{1}{L}\sum_{j=1}^{L} \mathbb{1}[y_j = \hat{y}_j]$ (ii) example-F1: $\frac{2\sum_{j=1}^{L} y_i \hat{y}_i}{\sum_{j=1}^{L} y_i + \sum_{j=1}^{L} \hat{y}_i}$ (iii) micro-F1: $\frac{\sum_{j=1}^{L} tp_j}{\sum_{j=1}^{L} 2tp_j + fp_j + fn_j}$ (iv) macro-F1: $\frac{1}{L}\sum_{j=1}^{L} \frac{2tp_j}{2tp_j + fp_j + fn_j}$

Furthermore, precision@1 is the proportion of correctly predicted labels in the top-1 predictions.

### 4.3 Architecture and Hyperparameters

As we state in the introduction, we do not require very sophisticated neural architectures in C-GMVAE. All the neural layers are fully connected. The feature encoder is a fully connected neural network with 3 hidden layers and the activation function is ReLU. The label encoder is also fully connected comprising two hidden layers and the decoder has two hidden layers as well. More details of the model can be found in the appendix. We set $\alpha = 1, \beta = 0.5, E = 2048$ from tuning for most runs. Grid search is applied to find the best learning rate, dropout ratio, weight decay ratio for each dataset. We use one V100 GPU for all experiments. More architecture, hyper-parameter tuning and final selection (Tab. 7 in Appx), and implementation details can be found in the appendix.

### 4.4 Evaluations

**Full supervision** In the full supervision scenario which is commonly adopted by the methods we compare against, we evaluate four metrics: example-F1 (ex-F1), micro-F1 (mi-F1), macro-F1

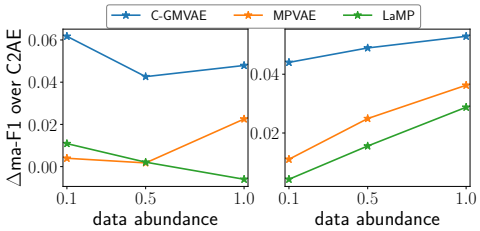


Figure 2: Relative improvements of C-GMVAE, MPVAE and LaMP on ma-F1 compared to C2AE. Left and right plots correspond to *mir-flickr* and *nus-vec* datasets respectively. Every method (including C2AE) is trained on the same amount of data (10%, 50% or 100%) for comparison.

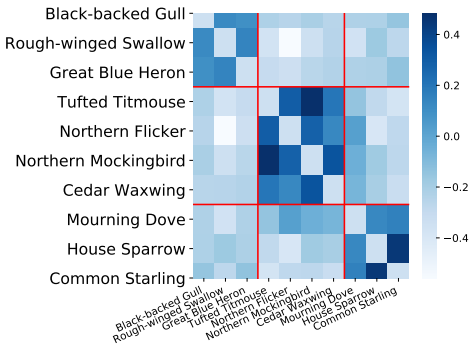


Figure 3: Label-label embedding inner-products from C-GMVAE.

(ma-F1), Hamming accuracy (HA) and precision@1. ex-F1 score is the averaged F1-score over all the samples. mi-F1 score measures the aggregated contributions of all classes. ma-F1 treats each class equally and takes the class-wise average. HA counts the correctly predicted labels regardless of samples or classes.

Tab. 1, 2 and 3 present the performance of all the methods w.r.t. the metrics. We abbreviate *nuswide-vector* to *nus-vec*, and *bookmarks* to *bkms*. C-GMVAE outperforms the existing state-of-the-art methods on all the datasets. The best numbers are marked in bold. All the numbers for C-GMVAE are averaged over 3 seeds for stability and the standard deviations are included in the table. On ex-F1, C-GMVAE improves over MPVAE by 2.5%, and LaMP by 8.8% on average across all the datasets.. Similarly, on mi-F1, C-GMVAE improves over MPVAE by 2.4% and LaMP 6.1% on average. On ma-F1, the improvements are as large as 4.1% and 11% respectively. C-GMVAE outperforms other methods consistently.

**Ablation study** To demonstrate the strength of our C-GMVAE, we compare it with a uni-Gaussian latent model, a Gaussian mixture (GM) only latent model (without contrastive module), and a contrastive only model (without KL term) in Tab. 4. Our C-GMVAE (GM+contrastive) consistently outperforms other models by a margin. For instance, on ma-F1, C-GMVAE improves over uni-Gaussian model by 7%.

**Training on fewer data** Contrastive learning learns contrastive views and thus requires less information compared to generative learning which demands a more complete representation for reconstruction. Contrastive learning has the potential to discover the intrinsic structures present in the data, and therefore is widely used in self-supervised learning since it generalizes well. We observe this with C-GMVAE as well. To demonstrate this, we shrink the size of training data by 50% or 90% and train methods on them. Surprisingly, we found C-GMVAE can often match the performance of other methods with only 50% of the training data. Tab. 5 compares MPVAE trained on full data and C-GMVAE trained on 50% of data. Their performances approximately match. We further compare several major state-of-the-art methods including ours all trained on the same **randomly** selected 10%, 50% and 100% of the data and show their performance over C2AE. Fig. 2 shows the improvements over C2AE on ma-F1. Ours clearly outperforms others on fewer data.

**Interpretability** Our work is also motivated by ecological applications [52], where it is important to understand species interactions. Fig. 3 shows a map of label-label embedding inner-product weights for the eBird dataset. The bird species on the x-axis and the y-axis are the same. The first 3 bird species are water birds. The following 4 bird species are forest birds. The last 3 bird species are residential birds. The darker the grid is, the closer two birds will be. We subtract the diagonal to exclude the self correlation. The heatmap matrix clearly form three blocks. The first block contains Black-backed Gull, Rough-winged Swallow and Great Blue Heron. These three birds are water birds living near sea or lake. The second block has Tufted Titmouse, Northern Flicker, Northern Mockingbird, and Cedar Waxwing. These birds typically live in the forest with a lot of trees. The remaining birds are commonly seen residential birds, Mourning Dove, House Sparrow and Common

Starling. They live inside or near human residences. Since human activities are wide-spread, the distribution of these birds are therefore quite broad. For example, Mourning Dove also has some correlations with forest birds in Fig. 3. But one can observe that for each group of birds, their intra-group correlations are always stronger than inter-group correlations. Therefore, the learnt embeddings do encompass semantic meanings. The derived correlaions could also help the study of wildlife protection [53].

## 5 Conclusion

In this work, we propose a contrastive learning boosted Gaussian mixture variational autoencoder (C-GMVAE) multi-label predictor, a novel method for MLC. C-GMVAE combines the learning of a Gaussian mixture latent space with the contrastive learning of feature and label embeddings. Not only does C-GMVAE achieve the state-of-the-art performance, it also provides insights into semi-supervised learning and model interpretability. Interesting future directions include the exploration of various contrastive learning mechanisms, model architecture improvements, and other latent space structures.

## References


[1] Jiang Wang, Yi Yang, Junhua Mao, Zhiheng Huang, Chang Huang, and Wei Xu. Cnn-rnn: A unified framework for multi-label image classification. In *CVPR*, 2016.

[2] Wei-Cheng Chang, Hsiang-Fu Yu, Kai Zhong, Yiming Yang, and Inderjit Dhillon. X-bert: extreme multi-label text classification with using bidirectional encoder representations from transformers. *arXiv preprint arXiv:1905.02331*, 2019.

[3] Guoxian Yu, Huzefa Rangwala, Carlotta Domeniconi, Guoji Zhang, and Zhiwen Yu. Protein function prediction using multilabel ensemble classification. *IEEE/ACM Transactions on Computational Biology and Bioinformatics*, 2013.

[4] 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*. Springer, 2009.

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

[6] Kush Bhatia, Himanshu Jain, Purushottam Kar, Manik Varma, and Prateek Jain. Sparse local embeddings for extreme multi-label classification. *Advances in neural information processing systems*, 28:730–738, 2015.

[7] Chih-Kuan Yeh, Wei-Chieh Wu, Wei-Jen Ko, and Yu-Chiang Frank Wang. Learning deep latent space for multi-label classification. In *AAAI*, 2017.

[8] Junwen Bai, Shufeng Kong, and Carla Gomes. Disentangled variational autoencoder based multi-label classification with covariance-aware multivariate probit model. *IJCAI*, 2020.

[9] Vijaya Kumar Sundar, Shreyas Ramakrishna, Zahra Rahiminasab, Arvind Easwaran, and Abhishek Dubey. Out-of-distribution detection in multi-label datasets using latent space of $\beta$-vae. *arXiv preprint arXiv:2003.08740*, 2020.

[10] Nat Dilokthanakul, Pedro AM Mediano, Marta Garnelo, Matthew CH Lee, Hugh Salimbeni, Kai Arulkumaran, and Murray Shanahan. Deep unsupervised clustering with gaussian mixture variational autoencoders. *arXiv preprint arXiv:1611.02648*, 2016.

[11] Mike Wu and Noah Goodman. Multimodal generative models for scalable weakly-supervised learning. In *Advances in Neural Information Processing Systems*, 2018.

[12] Min-Ling Zhang and Zhi-Hua Zhou. A review on multi-label learning algorithms. *IEEE transactions on knowledge and data engineering*, 2013.

[13] Wei Bi and James Kwok. Multilabel classification with label correlations and missing labels. In *Proceedings of the AAAI Conference on Artificial Intelligence*, 2014.

[14] David Belanger and Andrew McCallum. Structured prediction energy networks. In *International Conference on Machine Learning*, 2016.

[15] Jack Lanchantin, Arshdeep Sekhon, and Yanjun Qi. Neural message passing for multi-label classification. *arXiv preprint arXiv:1904.08049*, 2019.

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

[17] Junyoung Chung, Kyle Kastner, Laurent Dinh, Kratarth Goel, Aaron C Courville, and Yoshua Bengio. A recurrent latent variable model for sequential data. *Advances in neural information processing systems*, 28:2980–2988, 2015.

[18] SM Eslami, Nicolas Heess, Theophane Weber, Yuval Tassa, David Szepesvari, Koray Kavukcuoglu, and Geoffrey E Hinton. Attend, infer, repeat: Fast scene understanding with generative models. *arXiv preprint arXiv:1603.08575*, 2016.

[19] Arash Vahdat and Jan Kautz. Nvae: A deep hierarchical variational autoencoder. *arXiv preprint arXiv:2007.03898*, 2020.

[20] Lars Maaløe, Marco Fraccaro, Valentin Liévin, and Ole Winther. Biva: A very deep hierarchy of latent variables for generative modeling. *Advances in Neural Information Processing Systems*, 32:6551–6562, 2019.

[21] Rewon Child. Very deep vaes generalize autoregressive models and can outperform them on images. In *International Conference on Learning Representations*, 2020.

[22] Aaron van den Oord, Yazhe Li, and Oriol Vinyals. Representation learning with contrastive predictive coding. *arXiv preprint arXiv:1807.03748*, 2018.

[23] Ting Chen, Simon Kornblith, Mohammad Norouzi, and Geoffrey Hinton. A simple framework for contrastive learning of visual representations. *arXiv preprint arXiv:2002.05709*, 2020.

[24] Prannay Khosla, Piotr Teterwak, Chen Wang, Aaron Sarna, Yonglong Tian, Phillip Isola, Aaron Maschinot, Ce Liu, and Dilip Krishnan. Supervised contrastive learning. *arXiv preprint arXiv:2004.11362*, 2020.

[25] Michael Gutmann and Aapo Hyvärinen. Noise-contrastive estimation: A new estimation principle for unnormalized statistical models. In *AISTATS*, 2010.

[26] Feng Wang, Huaping Liu, Di Guo, and Fuchun Sun. Unsupervised representation learning by invariancepropagation. *arXiv preprint arXiv:2010.11694*, 2020.

[27] Carl Doersch. Tutorial on variational autoencoders. *arXiv preprint arXiv:1606.05908*, 2016.

[28] Jakub Tomczak and Max Welling. Vae with a vampprior. In *International Conference on Artificial Intelligence and Statistics*, pages 1214–1223. PMLR, 2018.

[29] Prince Zizhuang Wang and William Yang Wang. Neural gaussian copula for variational autoencoder. *arXiv preprint arXiv:1909.03569*, 2019.

[30] Zachary Seymour and Zhongfei Zhang. Multi-label triplet embeddings for image annotation from user-generated tags. In *Proceedings of the 2018 ACM on International Conference on Multimedia Retrieval*, pages 249–256, 2018.

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

[32] Oluwasanmi Koyejo, Nagarajan Natarajan, Pradeep Ravikumar, and Inderjit S Dhillon. Consistent multilabel classification. In *NIPS*, volume 29, pages 3321–3329, 2015.

[33] Stijn Decubber, Thomas Mortier, Krzysztof Dembczyński, and Willem Waegeman. Deep f-measure maximization in multi-label classification: A comparative study. In *Joint European Conference on Machine Learning and Knowledge Discovery in Databases*, pages 290–305. Springer, 2018.

[34] Edgar Schönfeld, Sayna Ebrahimi, Samarth Sinha, Trevor Darrell, and Zeynep Akata. Generalized zero-and few-shot learning via aligned variational autoencoders. In *CVPR*. IEEE, 2019.

[35] 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 *AAAI*, 2019.

[36] Lars Maaløe, Marco Fraccaro, and Ole Winther. Semi-supervised generation with cluster-aware generative models. *stat*, 1050:3, 2017.

[37] Mikołaj Małkiński and Jacek Mańdziuk. Multi-label contrastive learning for abstract visual reasoning. *arXiv preprint arXiv:2012.01944*, 2020.

[38] Jiaming Song and Stefano Ermon. Multi-label contrastive predictive coding. *Advances in Neural Information Processing Systems*, 33, 2020.

[39] Mauro Annarumma and Giovanni Montana. Deep metric learning for multi-labelled radiographs. *arXiv preprint arXiv:1712.07682*, 2017.

[40] Mark J Huiskes and Michael S Lew. The mir flickr retrieval evaluation. In *Proceedings of the 1st ACM international conference on Multimedia information retrieval*, pages 39–43, 2008.

[41] 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.

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

[43] 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.

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

[45] D Fink, T Auer, F Obregon, WM Hochachka, M Iliff, B Sullivan, C Wood, I Davies, and S Kelling. The ebird reference dataset version 2016 (erd2016), 2017.

[46] Franca Debole and Fabrizio Sebastiani. An analysis of the relative hardness of reuters-21578 subsets. *Journal of the American Society for Information Science and technology*, 56(6): 584–596, 2005.

[47] Ioannis Katakis, Grigorios Tsoumakas, and Ioannis Vlahavas. Multilabel text classification for automated tag suggestion. *ECML PKDD Discovery Challenge*, page 75, 2008.

[48] Grigorios Tsoumakas, Ioannis Katakis, and Ioannis Vlahavas. Effective and efficient multilabel classification in domains with large number of labels. In *Proc. ECML/PKDD 2008 Workshop on Mining Multidimensional Data (MMD'08)*, volume 21, pages 53–59, 2008.

[49] Lifu Tu and Kevin Gimpel. Learning approximate inference networks for structured prediction. *arXiv preprint arXiv:1803.03376*, 2018.

[50] 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)*. IEEE, 2015.

[51] 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.

[52] Carla Gomes, Thomas Dietterich, et al. Computational sustainability: Computing for a better world and a sustainable future. *Communications of the ACM*, 62(9):56–65, 2019.

[53] A Johnston, WM Hochachka, ME Strimas-Mackey, V Ruiz Gutierrez, OJ Robinson, ET Miller, T Auer, ST Kelling, and D Fink. Best practices for making reliable inferences from citizen science data: case study using ebird to estimate species distributions. *BioRxiv*, page 574392, 2019.

[54] Jiang Wang, Yang Song, Thomas Leung, Chuck Rosenberg, Jingbin Wang, James Philbin, Bo Chen, and Ying Wu. Learning fine-grained image similarity with deep ranking. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 1386–1393, 2014.

# A Contrastive Learning Module

## A.1 Connection with Triplet Loss

Triplet loss [54] is one of the popular ranking losses used in multi-label learning [30].

Given an anchor embedding $v_x^f$, a positive embedding $v_+$ and a negative embedding $v_-$, they form a triplet $(v_x^f, v_+, v_-)$. A triplet loss is defined as

$$\begin{aligned} &\mathcal{L}_{trip}(v_x^f, v_+, v_-) \\ =&max\{0, g + dist(v_x^f, v_+) - dist(v_x^f, v_-)\} \end{aligned} \tag{8}$$

where $g$ is a gap parameter measuring the distance between $(v_x^f, v_+)$ and $(v_x^f, v_-)$, and $dist(\cdot,\cdot)$ is a distance function. This hinge loss $\mathcal{L}_{trip}$ encourages fewer violations to "positive>negative" ranking order. Let $\tau = 1/2$. With the same triplet, we can write down a contrastive loss

$$\begin{aligned} &\mathcal{L}_{CL}(v_x^f, v_+, v_-) \\ =&-\log \frac{\exp(2 \cdot v_x^f \cdot v_+)}{\sum_{t\in\{+,-\}} \exp(2 \cdot v_x^f \cdot v_t)} \\ =&\log(1 + \frac{\exp(2 \cdot v_x^f \cdot v_-)}{\exp(2 \cdot v_x^f \cdot v_+)}) \\ \approx&1 + (2 \cdot v_x^f \cdot v_- - 2 \cdot v_x^f \cdot v_+) \\ =&1 + (-v_x^f \cdot v_x^f + 2v_x^f \cdot v_- - v_- \cdot v_- \\ &+ v_x^f \cdot v_x^f - 2 \cdot v_x^f \cdot v_+ + v_+ \cdot v_+) \\ =&||v_x^f - v_+||^2 + ||v_x^f - v_-||^2 + 1 \end{aligned} \tag{9}$$

Note that in the second to the last equation, $v_+$ and $v_-$ have the same norm due to the normalization in our contrastive learning module.

By setting $dist(\cdot,\cdot)$ to commonly used $\ell_2$ distance and $g = 1$, Eq. 9 is a fair approximation of Eq. 8. Therefore, triplet loss can be viewed as a special case of our contrastive loss. But in contrastive loss, embeddings are normalized and more positives/negatives are available. As shown in [23], contrastive loss generally outperforms triplet loss.

## A.2 Gradients of Contrastive Loss

Recall our contrastive loss:

$$\mathcal{L}_{CL} = \sum_{(x,y)\in\mathcal{B}} \frac{1}{|P(y)|} \sum_{p\in P(y)} -\log \frac{\exp(v_x^f \cdot v_p^l/\tau)}{\sum_{t\in A} \exp(v_x^f \cdot v_t^l/\tau)} \tag{10}$$

For the illustration purpose, we only consider one sample $(x, y)$ instead of one batch:

$$\mathcal{L}_{CL} = \frac{1}{|P(y)|} \sum_{p\in P(y)} -\log \frac{\exp(v_x^f \cdot v_p^l/\tau)}{\sum_{t\in A} \exp(v_x^f \cdot v_t^l/\tau)} \tag{11}$$

Define $N(y) \equiv A \setminus P(y)$. We now derive the gradients w.r.t. $v_x^f$.

$$
\begin{aligned}
\frac{\partial \mathcal{L}_{CL}}{\partial v_x^f} =& \frac{1}{\tau |P(y)|} \sum_{p \in P(y)} (\frac{\sum_{t \in A} v_t^l \exp(v_x^f \cdot v_t^l / \tau)}{\sum_{t \in A} \exp(v_x^f \cdot v_t^l / \tau)} - v_p^l) \\
=& \frac{1}{\tau |P(y)|} \sum_{p \in P(y)} (\frac{\sum_{t \in P(y)} v_t^l \exp(v_x^f \cdot v_t^l / \tau)}{\sum_{t \in A} \exp(v_x^f \cdot v_t^l / \tau)} + \\
& \frac{\sum_{t \in N(y)} v_t^l \exp(v_x^f \cdot v_t^l / \tau)}{\sum_{t \in A} \exp(v_x^f \cdot v_t^l / \tau)} - v_p^l) \\
=& \frac{1}{\tau} \frac{\sum_{t \in P(y)} v_t^l \exp(v_x^f \cdot v_t^l / \tau)}{\sum_{t \in A} \exp(v_x^f \cdot v_t^l / \tau)} + \\
& \frac{1}{\tau} \frac{\sum_{t \in N(y)} v_t^l \exp(v_x^f \cdot v_t^l / \tau)}{\sum_{t \in A} \exp(v_x^f \cdot v_t^l / \tau)} - \frac{1}{\tau |P(y)|} \sum_{p \in P(y)} v_p^l \\
=& \frac{1}{\tau} [ \sum_{t \in P(y)} v_t^l (\frac{\exp(v_x^f \cdot v_t^l / \tau)}{\sum_{a \in A} \exp(v_x^f \cdot v_a^l / \tau)} - \frac{1}{|P(y)|}) + \\
& \sum_{t \in N(y)} v_t^l \frac{\exp(v_x^f \cdot v_t^l / \tau)}{\sum_{a \in A} \exp(v_x^f \cdot v_a^l / \tau)} ]
\end{aligned}
\tag{12}
$$

Further, we have the unnormalized feature embedding $w_x^f$, $v_x^f = \frac{w_x^f}{||w_x^f||}$.

$$
\begin{aligned}
\frac{\partial v_x^f}{\partial w_x^f} =& \frac{1}{||w_x^f||} (I - \frac{w_x^f {w_x^f}^T}{||w_x^f||^2}) \\
=& \frac{1}{||w_x^f||} (I - v_x^f {v_x^f}^T)
\end{aligned}
\tag{13}
$$

where $I$ is an $E \times E$ identity matrix.

The gradient of $\mathcal{L}_{CL}$ w.r.t. $w_x^f$ can be derived using chain rule,

$$
\begin{aligned}
& \frac{\partial \mathcal{L}_{CL}}{\partial w_x^f} = \frac{\partial v_x^f}{\partial w_x^f} \frac{\partial \mathcal{L}_{CL}}{\partial v_x^f} \\
=& \frac{1}{||w_x^f||} (I - v_x^f {v_x^f}^T) \frac{1}{\tau} [ \sum_{t \in P(y)} v_t^l (\frac{\exp(v_x^f \cdot v_t^l / \tau)}{\sum_{a \in A} \exp(v_x^f \cdot v_a^l / \tau)} \\
& - \frac{1}{|P(y)|}) + \sum_{t \in N(y)} v_t^l \frac{\exp(v_x^f \cdot v_t^l / \tau)}{\sum_{a \in A} \exp(v_x^f \cdot v_a^l / \tau)} ] \\
=& \frac{1}{\tau ||w_x^f||} [ \sum_{t \in P(y)} (v_t^l - (v_x^f v_t^l) v_x^f) (\frac{\exp(v_x^f \cdot v_t^l / \tau)}{\sum_{a \in A} \exp(v_x^f \cdot v_a^l / \tau)} \\
& - \frac{1}{|P(y)|}) + \sum_{t \in N(y)} (v_t^l - (v_x^f v_t^l) v_x^f) \frac{\exp(v_x^f \cdot v_t^l / \tau)}{\sum_{a \in A} \exp(v_x^f \cdot v_a^l / \tau)} ]
\end{aligned}
\tag{14}
$$

We can then observe that if $v_x^f$ and $v_t^l$ are orthogonal ($v_x^f v_t^l \to 0$), $||v_t^l - (v_x^f v_t^l) v_x^f||$ will be close to 1 and the gradients would be large. Otherwise, for weak positives or negatives ($|v_x^f v_t^l| \to 1$), the gradients would be small.

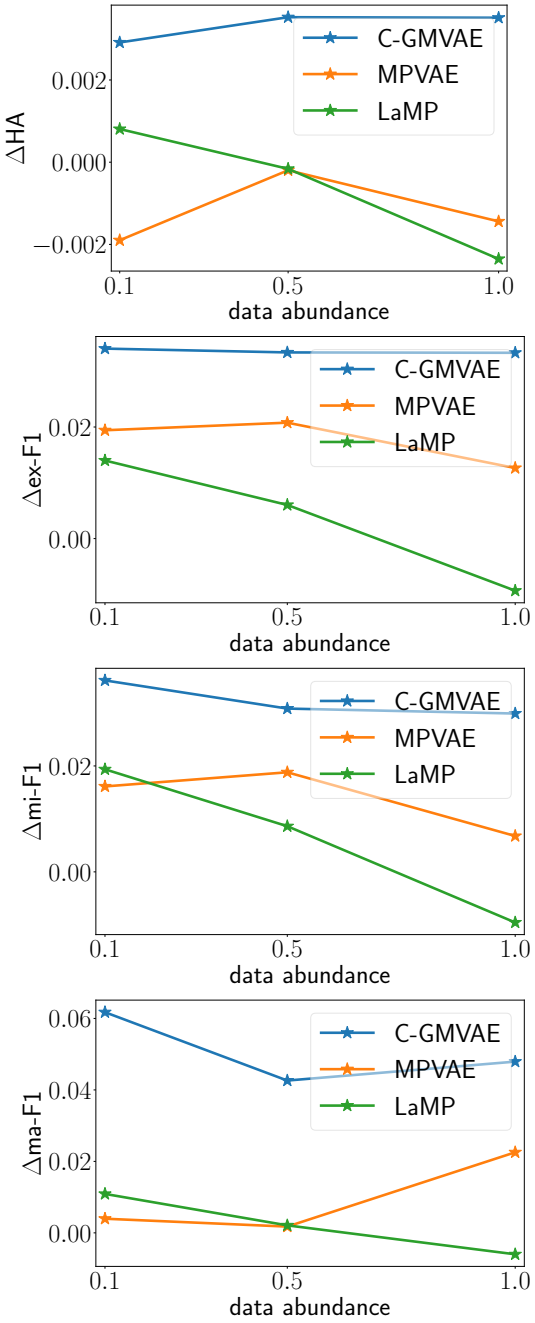


Figure 4: Relative performances w.r.t. HA, ex-F1, mi-F1 and ma-F1 on *mirflickr* dataset.

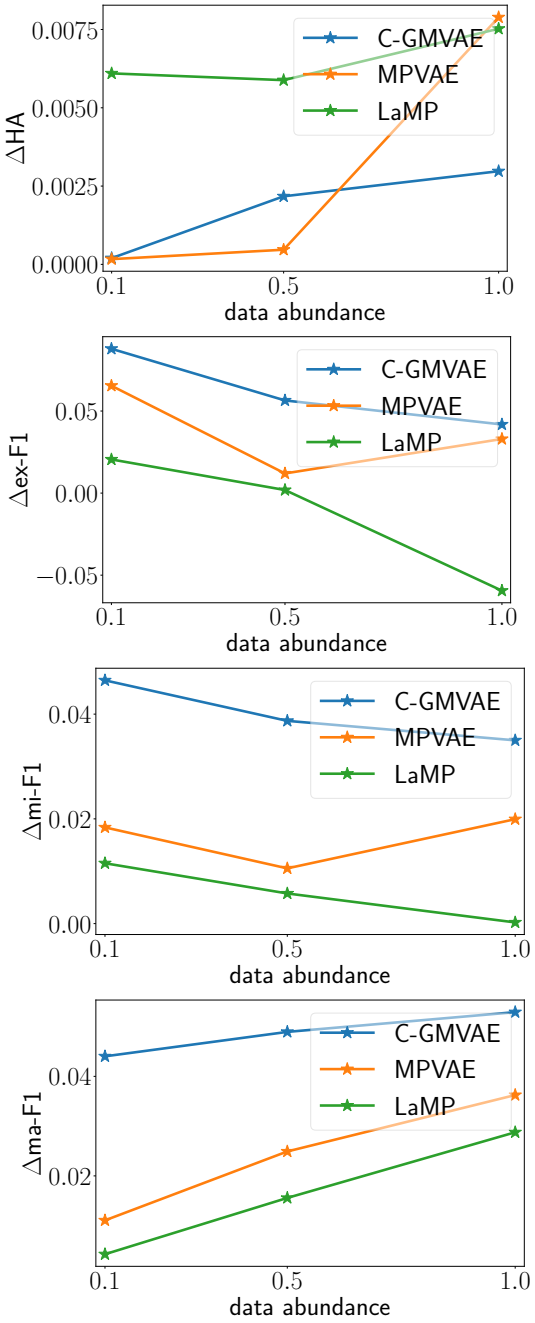


Figure 5: Relative performances w.r.t. HA, ex-F1, mi-F1 and ma-F1 on *nus-vec* dataset.

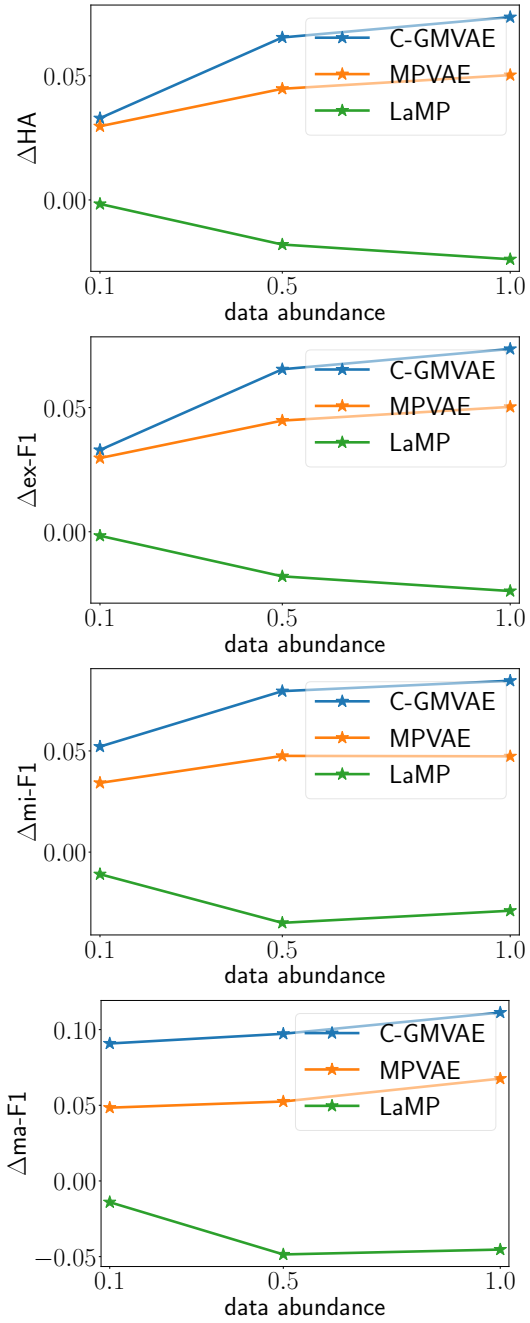


Figure 6: Relative performances w.r.t. HA, ex-F1, mi-F1 and ma-F1 on *ebird* dataset.

| | # Samples | # Labels | Mean Labels /Sample | Median Labels /Sample | Max Labels /Sample | Mean Samples /Label |
|---|---|---|---|---|---|---|
| *eBird* | 41778 | 100 | 20.69 | 18 | 96 | 8322.95 |
| *bookmarks* | 87856 | 208 | 2.03 | 1 | 44 | 584.67 |
| *nus-vec* | 269648 | 85 | 1.86 | 1 | 12 | 3721.7 |
| *mirflickr* | 25000 | 38 | 4.80 | 5 | 17 | 1247.34 |
| *reuters* | 10789 | 90 | 1.23 | 1 | 15 | 106.50 |
| *scene* | 2407 | 6 | 1.07 | 1 | 3 | 170.83 |
| *sider* | 1427 | 27 | 15.3 | 16 | 26 | 731.07 |
| *yeast* | 2417 | 14 | 4.24 | 4 | 11 | 363.14 |
| *delicious* | 16105 | 983 | 19.06 | 20 | 25 | 250.15 |

Table 6: Dataset Statistics.

| | lr | $\alpha$ | $\beta$ | $E$ | dropout | bs |
|---|---|---|---|---|---|---|
| *eBird* | 0.001 | 1 | 0.5 | 2048 | 0.5 | 128 |
| *bookmarks* | 0.002 | 1 | 1 | 2048 | 0.5 | 128 |
| *nus-vec* | 0.004 | 1 | 0.5 | 1024 | 0.5 | 256 |
| *mirflickr* | 0.001 | 2 | 0.5 | 2048 | 0.5 | 128 |
| *reuters* | 0.005 | 2 | 1 | 2048 | 0.5 | 128 |
| *scene* | 0.003 | 1 | 0.5 | 512 | 0.3 | 128 |
| *sider* | 0.002 | 1 | 0.5 | 512 | 0.5 | 128 |
| *yeast* | 0.002 | 1 | 0.5 | 512 | 0.5 | 128 |
| *delicious* | 0.001 | 1 | 0.5 | 2048 | 0.5 | 128 |

Table 7: Major hyperparameters used in training. “lr" stands for learning rate.

# B Supplementary Experimental Results

## B.1 Implementation Details

We use one Tesla V100 GPU on CentOS for every experiment. The batch size is set to 128. The latent dimensionality is 64. The feature encoder is an MLP with 3 hidden layers of sizes [256, 512, 256]. The label encoder has 2 hidden layers of sizes [512, 256]. The decoder contains 2 hidden layers of sizes [512, 512]. On *reuters* and *bookmarks*, we add one more hidden layer with 512 units to the decoder. The embedding size $E$ is 2048 (tuned within the range [512, 1024, 2048, 3072]). We set $\alpha = 1$ (tuned within [0.1, 0.5, 1, 1.5, 2]), $\beta = 0.5$ (tuned within [0.1, 0.5, 1, 1.5, 2.0]) for most runs. We tune learning rates from 0.0001 to 0.004 with interval 0.0002, dropout ratio from [0.3, 0.5, 0.7], and weight decay from [0, 0.01, 0.0001]. Grid search is adopted for tuning. The final hyper-parameter selections are shown in Tab. 7. Every batch in our experiments requires less than 16GB memory. The number of epochs is 100 by default.

## B.2 Training on Fewer Data

We provide relative performances of several major state-of-the-art methods including ours to C2AE, on HA, ex-F1, mi-F1, ma-F1 scores. All methods are trained on 10% or 50% of the data, including C2AE. The compared results have the same amount of data for training and thus the comparison is fair.

Fig. 4, Fig. 5, Fig. 6 show the relative performance of various state-of-the-art methods over C2AE, on *mirflickr*, *nus-vec*, *eBird* respectively.