Joint Generative Moment-Matching Network for Learning Structural Latent Code

Hongchang Gao, Heng Huang
Department of Electrical and Computer Engineering, University of Pittsburgh, USA
hongchanggao@gmail.com, heng.huang@pitt.edu

Abstract

Generative Moment-Matching Network (GMMN) is a deep generative model, which employs maximum mean discrepancy as the objective to learn model parameters. However, this model can only generate samples, failing to infer the latent code from samples for downstream tasks. In this paper, we propose a novel Joint Generative Moment-Matching Network (JGMMN), which learns the structural latent code for unsupervised inference. Specifically, JGMMN has a generation network for the generation task and an inference network for the inference task. We first reformulate this model as the two joint distributions matching problem. To solve this problem, we propose to use the Joint Maximum Mean Discrepancy (JMMD) as the objective to learn these two networks simultaneously. Furthermore, to enforce the consistency between the sample distribution and the inferred latent code distribution, we propose a novel multi-modal regularization to enforce this consistency. At last, extensive experiments on both synthetic and real-world datasets have verified the effectiveness and correctness of our proposed JGMMN.

1 Introduction

In recent years, deep generative models have attracted much attention among machine learning and computer vision communities. Due to deep learning techniques' flexibility and powerful capability on fitting nonlinear functions, deep generative models have shown promising capabilities in characterizing the distribution of complex datasets.

Among various generative models, Generative Adversarial Network (GAN) [Goodfellow *et al.*, 2014] has achieved much progress in many tasks, such as image generation [Goodfellow *et al.*, 2014; Chen *et al.*, 2016; Reed *et al.*, 2016], image translation [Zhu *et al.*, 2017; Kim *et al.*, 2017]. Specifically, given a dataset $X \sim Q_X$ where Q_X might be very complicated, GAN learns a transformation $g_\theta(\cdot)$ by a deep neural network, which operates on a simple distribution $Z \sim P_Z$, such that $P_{g_\theta(Z)}$ is a good approximation of the sample distribution Q_X. To measure the similarity between $P_{g_\theta(Z)}$ and Q_X, GAN employs a min-max optimization schema where another deep neural network $f_\phi(\cdot)$ is trained to measure the similarity of these two distributions. Although this schema has shown success in many tasks, yet it is difficult to optimize because it requires special techniques [Arjovsky *et al.*, 2017; Gulrajani *et al.*, 2017] to converge to a good solution. Instead, Generative-Moment Matching Network (GMMN) [Li *et al.*, 2015] employs Maximum Mean Discrepancy (MMD) as the measurement for the similarity of two distributions, which is much simpler than the objective of GAN models and easy to optimize. Specifically, it employs the kernel embedding [Gretton *et al.*, 2012] technique to match all orders of statistics between $P_{g_\theta(Z)}$ and Q_X. In this paper, we will focus on the GMMN model.

Although the GMMN model can successfully learn the approximation of the underlying sample distribution Q_X, yet it lacks the inference capability, failing to learn the posterior distribution of the latent code. Since many downstream tasks heavily depend on the latent code, such as the classification and clustering, it is important and necessary to endow GMMN model with inference capability.

To address the inference problem, we propose a Joint Generative Moment Matching Network (JGMMN) as shown in Figure 1. Specifically, it includes two networks: the generation network and the inference network. The generation network works as a generative model $g_\theta(\cdot)$ to generate samples: $\hat{X} = g_\theta(Z) \sim P_{X|Z}, Z \sim P_Z$. The inference network defines an inference model $h_\beta(\cdot)$ to infer latent codes from observed samples: $\hat{Z} = h_\beta(X) \sim Q_{Z|X}, X \sim Q_X$. Importantly, this problem can be formulated as the matching of two joint distributions $Q_{X,h_\beta(X)}$ and $P_{g_\theta(Z),Z}$. If they are matched, corresponding marginal distributions and conditional distributions also can be guaranteed to match [Dumoulin *et al.*, 2016]. Especially, the desired posterior distribution $P_{Z|X}$ can be matched by the conditional distribution $Q_{Z|X}$ [Dumoulin *et al.*, 2016].

However, it is challenging to match the two joint distributions $P_{g_\theta(Z),Z}$ and $Q_{X,h_\beta(X)}$. Since the marginal distributions Q_X and $Q_{h_\beta(X)}$ are not independent, we cannot match two marginal distributions (Q_X with $P_{g_\theta(Z)}$ and $Q_{h_\beta(X)}$ with P_Z) respectively. Thus, how to match the two joint distributions is challenging. Another challenge is that the learned $Q_{Z|X}$ should be consistent with the sample distribution Q_X. Specifically, the inferred latent code distribution

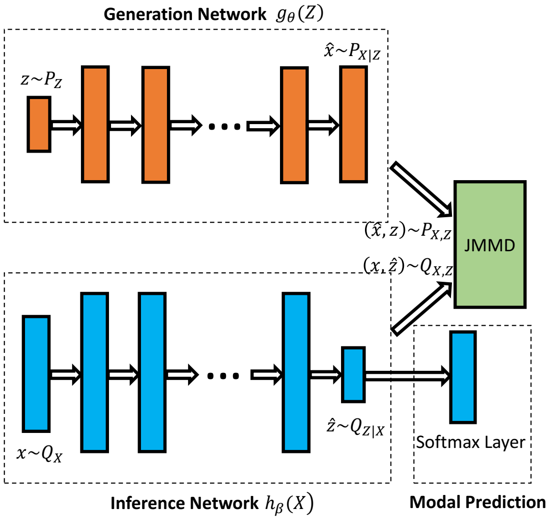
Generation Network $g_\theta(Z)$
$z\sim P_Z$
$\hat{x}\sim P_{X|Z}$
$(\hat{x},z)\sim P_{X,Z}$
JMMD
$(x,\hat{z})\sim Q_{X,Z}$
$x\sim Q_X$
$\hat{z}\sim Q_{Z|X}$
Softmax Layer
Inference Network $h_\beta(X)$
Modal Prediction

Figure 1: The architecture of our method. It includes a generation network to generate samples and an inference network to infer latent code. The JMMD module is to measure the similarity between two joint distributions. The modal prediction part is used for multi-modal regularization to guarantee the consistency between Q_X and $Q_{Z|X}$.

$Q_{Z|X}$ should have similar structures with the sample distribution Q_X. More specifically, if the sample distribution Q_X is a multi-modal distribution, $Q_{Z|X}$ should also be a multi-modal distribution such that the inferred latent code is a good representation of observed samples. Then, the downstream tasks can benefit from it.

To address the first challenge, we propose to use the Joint Maximum Mean Discrepancy (JMMD) metric to measure the similarity between two joint distributions. In detail, JMMD measures the distance of two joint distributions by their kernel embeddings in a Reproducing Kernel Hilbert Space (RKHS). This objective can be efficiently optimized by back-propagation. By minimizing this objective, we can match $P_{g_\theta(Z),Z}$ to $Q_{X,h_\beta(X)}$, learning the generative model and the inference model simultaneously. To address the second challenge, we propose a novel multi-modal regularization to enforce the latent code distribution $Q_{Z|X}$ to be consistent with the sample distribution Q_X. At last, extensive experiments are conducted to verify the effectiveness of our proposed model.

2 Related Works

In this section, we will review the most related works, analyzing their properties and giving the motivation of our proposed JGMMN.

In recent years, much progress has gone towards deep generative models for modeling complicated data distributions. Among them, Generative Adversarial Network (GAN) [Goodfellow *et al.*, 2014] has attracted much more attention in computer vision and machine learning communities, and has achieved impressive results in many tasks, such as image generation [Goodfellow *et al.*, 2014; Chen *et al.*, 2016; Reed *et al.*, 2016; Ledig *et al.*, 2016; Radford *et al.*, 2015; Denton *et al.*, 2015], image translation [Zhu *et al.*, 2017; Kim *et al.*, 2017]. Generally, GAN trains a generator, which transforms a simple latent distribution to a complicated sample distribution. Meanwhile, it also trains a discriminator, which distinguishes the generated and real samples. Mathematically, GAN is to optimize a min-max problem. However, this optimization problem is difficult to optimize, which usually needs special techniques [Arjovsky *et al.*, 2017; Gulrajani *et al.*, 2017]. As opposite to GAN, Generative Moment-Matching Network (GMMN) [Li *et al.*, 2015] employs a simpler criterion to discriminate the generated and real samples. Specifically, GMMN adopts similar generator with GAN to map a simple distribution to a complicated sample distribution. Unlike GAN using a discriminator to distinguish the generated and real samples, GMMN employs maximum mean discrepancy (MMD) as the metric to measure the distance between the generated and real sample distributions. This objective is easy to optimize as long as the kernel function is smooth [Ren *et al.*, 2016].

However, both GAN and GMMN lack the inference capability. They can only generate samples, failing to infer the latent information from observed samples. As we know, the latent code is important for model explanation and downstream tasks. Thus, some researchers propose to endow generative models with the inference capability. For instance, Adversarially Learned Inference (ALI) [Dumoulin *et al.*, 2016] and Bidirectional Generative Adversarial Networks (BiGAN) [Zhang *et al.*, 2016] are two representatives that can learn generative and inference models simultaneously. The basic idea is to incorporate an inference network to infer the latent code from observed samples and learn a discriminator to discriminate two joint distributions. However, these models are based on the GAN framework. They also need to solve a min-max problem, which is difficult to optimize. Additionally, these techniques cannot be applied to GMMN framework directly, since these two frameworks employ different learning schemas. Thus, it is important and not trivial to endow GMMN framework with the inference ability.

Furthermore, existing works, such as ALI and BiGAN, only focus on matching two joint distributions, ignoring the structure of the latent code distribution. As a good inference model, it should guarantee the inferred latent code distribution to be consistent with the sample distribution. Thus, it is important and necessary to preserve this consistency in the inference model.

3 Background

In this section, we will give some preliminary knowledge about kernel embedding of distributions and how to measure the distance between two distributions based on the kernel embedding.

3.1 Kernel Embedding of Distributions

To measure the similarity of two distributions, a widely used approach is to compute their distance based on their embeddings in a reproducing kernel Hilbert space (RKHS). Here, RKHS is a Hilbert function space in which each probability distribution is an element. As a Hilbert function space, RKHS denoted by $\mathcal{H}$ is associated with the inner product $\langle \cdot, \cdot \rangle_{\mathcal{H}}$ as well as a kernel $k(x, x')$, satisfying $f(x) = \langle f(\cdot), k(x, \cdot) \rangle_{\mathcal{H}}$ for all elements f in this function space. Here, $k(x, \cdot)$ denotes a feature map $\phi(x)$ implicitly such that $k(x, x') = \langle \phi(x), \phi(x') \rangle_{\mathcal{H}}$.

The kernel embedding of a distribution P is defined to take expectation on its feature map [Sriperumbudur *et al.*, 2010; Gretton *et al.*, 2012]:

$$\mu_P := E_X[\phi(X)] = \int \phi(x) dP(x) \ , \tag{1}$$

where $x \in X \sim P$. Importantly, the kernel embedding μ_P is guaranteed to be an element of RKHS if $E_X[k(x, x')] \leq \infty$. More importantly, if the kernel is characteristic, all of the statistical features of a distribution will be preserved by the kernel embedding. Thus, kernel embedding is a good representation of the probability distribution.

3.2 Maximum Mean Discrepancy

Maximum Mean Discrepancy (MMD) is a basic tool to conduct two-sample test [Gretton *et al.*, 2012]. Specifically, given two distributions P and Q, two datasets $X = \{x_i\}_{i=1}^n$ and $Y = \{y_i\}_{i=1}^n$ drawn from P and Q respectively. MMD is to test whether $P = Q$ based on observed samples X and Y. Formally, MMD is defined as follows:

Definition 1. *(Maximum Mean Discrepancy (MMD)) [Gretton* et al.*, 2012] Given the unit ball $\mathcal{F} \subset \mathcal{H}$, and samples X and Y drawn from distributions P and Q respectively, the MMD is*

$$MMD[\mathcal{F}, P, Q] := \sup_{f \in \mathcal{F}} (E_X[f(X)] - E_Y[f(Y)]) \ . \tag{2}$$

According to [Gretton *et al.*, 2012], under mild conditions, MMD can be reformulated as the difference between their kernel embedding:

$$\text{MMD}[\mathcal{F}, P, Q] = \|\mu_P - \mu_Q\|_{\mathcal{F}}^2 \ . \tag{3}$$

In practical applications, we can use the empirical kernel embedding to approximate it as follows:

$$\begin{aligned}
&\hat{\text{MMD}}[\mathcal{F}, P, Q] = \|\hat{\mu}_P - \hat{\mu}_Q\|_{\mathcal{F}}^2 \\
&= \|\frac{1}{n}\sum_{i=1}^{n} \phi(x_i) - \frac{1}{m}\sum_{i=1}^{m} \phi(y_i)\|_{\mathcal{F}}^2 \\
&= \frac{1}{n^2}\sum_{i,j}^{n} k(x_i, x_j) + \frac{1}{m^2}\sum_{i,j}^{m} k(y_i, y_j) - \frac{2}{mn}\sum_{i,j}^{n,m} k(x_i, y_j) \ ,
\end{aligned} \tag{4}$$

where $\{x_i\}_{i=1}^n$ are drawn i.i.d from P and $\{y_i\}_{i=1}^m$ are drawn i.i.d from Q. After obtaining the value of MMD, we can conclude the two-sample test result according to the following Theorem.

Theorem 1. *[Gretton* et al.*, 2012] For a characteristic kernel $\mathcal{F}$, MMD$[\mathcal{F}, P, Q] = 0 \iff P = Q$.*

Thus, MMD is a reasonable metric to measure the similarity of two distributions.

4 Method

In this section, we will present our proposed model, including a joint generative moment-matching network, and a multimodal regularization to guarantee the consistency between the sample distribution and the latent code distribution.

4.1 Joint Generative Moment-Matching Network

At first, we introduce Joint Maximum Mean Discrepancy (JMMD), which will be used for computing the distance between two joint distributions.

Joint Maximum Mean Discrepancy

Similar with kernel embedding of ordinary distributions, we can define the kernel embedding of joint distributions. Formally, for a joint distribution $P_{X,Y}$, its kernel embedding is defined as follows:

$$\mu_P := E_{X,Y}[\phi(X) \otimes \psi(Y)] = \int \phi(x) \otimes \psi(y) dP(x, y) \ , \tag{5}$$

where $\otimes$ denotes the tensor product, $\phi(x) \otimes \psi(y)$ denotes the feature map in the tensor product Hilbert space [Song *et al.*, 2009; 2010; Long *et al.*, 2016]. Similarly, it has the following property:

$$\langle \phi(x) \otimes \psi(y), \phi(x') \otimes \psi(y') \rangle_{\mathcal{H}} = k_\phi(x, x') k_\psi(y, y') \ . \tag{6}$$

Based on the kernel embedding of joint distributions, the Joint Maximum Mean Discrepancy (JMMD) can be defined as follows.

Definition 2. *(Joint Maximum Mean Discrepancy (JMMD)) Given samples $(X, Y) \sim P_{X,Y}$ and $(X', Y') \sim Q_{X',Y'}$, and the unit ball $\mathcal{F} \subset \mathcal{H}$, $\mathcal{G} \subset \mathcal{H}$ in a RKHS, then the JMMD is*

$$\begin{aligned}
&JMMD[\mathcal{F}, \mathcal{G}, P, Q] \\
&:= \sup_{f \in \mathcal{F}, g \in \mathcal{G}} (E_{X,Y}[f(X)g(Y)] - E_{X',Y'}[f(X')g(Y')]) \ .
\end{aligned} \tag{7}$$

Therefore, JMMD can be represented by the kernel embedding as follows:

$$\text{JMMD}[\mathcal{F}, \mathcal{G}, P, Q] = \|\mu_P - \mu_Q\|_{\mathcal{F} \otimes \mathcal{G}}^2 \ . \tag{8}$$

For real-world applications, we can compute the empirical JMMD as follows:

$$\begin{aligned}
&\text{J}\hat{\text{MMD}}[\mathcal{F}, \mathcal{G}, P, Q] = \|\hat{\mu}_P - \hat{\mu}_Q\|_{\mathcal{F} \otimes \mathcal{G}}^2 \\
&= \|\frac{1}{n}\sum_{i=1}^{n} \phi(x_i) \otimes \psi(y_i) - \frac{1}{m}\sum_{i=1}^{m} \phi(x_i') \otimes \psi(y_i')\|_{\mathcal{F} \otimes \mathcal{G}}^2 \\
&= \frac{1}{n^2}\sum_{i,j}^{n} k_\phi(x_i, x_j) k_\psi(y_i, y_j) + \frac{1}{m^2}\sum_{i,j}^{m} k_\phi(x_i', x_j') k_\psi(y_i', y_j') \\
&\quad - \frac{2}{mn}\sum_{i,j}^{n,m} k_\phi(x_i, x_j') k_\psi(y_i, y_j') \ .
\end{aligned} \tag{9}$$

Afterwards, we can use J$\hat{\text{MMD}}[\mathcal{F}, \mathcal{G}, P, Q]$ to compute the distance between two joint distributions empirically.

Joint GMMN

Now we are ready to present the Joint Generative Moment-Matching Network (JGMMN) model and how to train it by the JMMD metric.

As shown in Figure 1, JGMMN includes two networks: the generation network and the inference network. Generally, the generation network works as a generative model to generate samples. The inference network defines an inference model to infer latent codes from observed samples. According to [Dumoulin *et al.*, 2016], this model can be formulated as a joint distribution matching problem. After matched, the desired distribution can be learned.

Specifically, the generation network works similarly with the original GMMN [Li *et al.*, 2015] model. The basic idea of this model is that there is a map g_θ between a regularly simple distribution P_Z and a complicated sample distribution Q_X such that $g(Z) \sim Q_X$ where $Z \sim P_Z$, and such a map can be represented by a deep neural network due to its flexibility and powerful capability on fitting nonlinear functions. Based on this idea, the generation network starts from a stochastic layer where each unit independently follows a uniform distribution as follows [Li *et al.*, 2015]:

$$P_Z(z) = \prod_{i=1}^{d} U(z_i) \ , \tag{10}$$

where $U(z) = I(-1 \le z \le 1)$ is an uniform distribution in $[-1, 1]$. Afterwards, several non-linear hidden layers follow this stochastic layer, mapping z to a point $\hat{x}$ in the sample space as follows:

$$\hat{x} = g_\theta(z), z \sim P_Z(z) \ , \tag{11}$$

where $\hat{x} \in \hat{X}$ is the generated sample. This process defines the conditional distribution $P_{X|Z}$. Here, g_θ denotes a multi-layer deep neural network, which is parameterized by θ. Due to the powerful capability of the deep neural network in fitting nonlinear functions, this process can characterize $P_{X|Z}$ well. As a result, we can obtain the joint distribution with respect to the prior Z and the generated $\hat{X}$ as follows:

$$P_{X,Z} = P_{X|Z} P_Z \ . \tag{12}$$

However, the process defined in Eq. (11) can only generate samples. In many tasks, we need to infer the latent code from observed samples X for interpretation and downstream tasks. Hence, we propose the inference network as shown in Figure 1. Specifically, the inference network learns a transformation h_β which maps observed samples $X \sim Q_X$ to latent codes $\hat{Z} \sim Q_{Z|X}$. Formally,

$$\hat{z} = h_\beta(x), x \sim Q_X(x) \ . \tag{13}$$

Then, we will have the joint distribution from the inference network as follows:

$$Q_{X,Z} = Q_{Z|X} Q_X \ . \tag{14}$$

The proposed JGMMN model is to match these two joint distributions defined in Eq. (12) and Eq. (14). If they are matched, the corresponding marginal distributions and conditional distributions also can be guaranteed to match [Dumoulin *et al.*, 2016]. Especially, the conditional distribution $Q_{Z|X}$ defined by the inference network will match the desired posterior distribution $P_{Z|X}$.

To match these two joint distributions, we train the generation and inference network by optimizing the JMMD objective as follows:

$$\min_{\theta,\beta} J_1 \triangleq \frac{1}{n^2}\sum_{i,j}^{n} k_\phi(x_i, x_j) k_\psi(\hat{z}_i, \hat{z}_j) + \frac{1}{m^2}\sum_{i,j}^{m} k_\phi(\hat{x}_i, \hat{x}_j) k_\psi(z_i, z_j) - \frac{2}{mn}\sum_{i,j}^{n,m} k_\phi(x_i, \hat{x}_j) k_\psi(z_i, \hat{z}_j) \ , \tag{15}$$

where $\{(x_i, \hat{z}_i)\}_{i=1}^{n} \sim Q_{X,Z}$ is from the inference network, $\{(\hat{x}_i, z_i)\}_{i=1}^{m} \sim P_{X,Z}$ is from the generation network. By optimizing this objective function, we can learn model parameters for both generation network and inference network simultaneously. Then, our model will have both generative and inference capabilities.

4.2 Structural Latent Code

Although matching two joint distributions can guarantee the match between the conditional distribution $Q_{Z|X}$ and the desired posterior distribution $P_{Z|X}$, it cannot guarantee the learned $Q_{Z|X}$ consistent with the sample distribution Q_X. Specifically, the inferred latent distribution $Q_{Z|X}$ should have similar structures with the sample distribution Q_X. More specifically, if the sample distribution Q_X is a multi-modal distribution, $Q_{Z|X}$ should also be a multi-modal distribution such that the inferred latent code is a good representation of observed samples. Additionally, the multi-modal property is very common in real-world applications. Discovering this property is helpful for many downstream tasks, such as clustering on the latent code. Thus, it is important to discover this multi-modal structure and preserve it in the latent space.

To address this problem, we propose an unsupervised multi-modal regularization to guarantee the consistency between Q_X and $Q_{Z|X}$. To this end, we need to discover the multi-modal distribution in both Q_X and $Q_{Z|X}$ first, and then align them. Formally, we define two auxiliary distributions p and q, where p denotes the modality assignment distribution of the latent code and q denotes that of observed samples. Specifically, assume that there are c modalities, then p_{ij} denotes the probability of the i-th latent code belonging to the j-th modality, and q_{ij} has the same meaning for the i-th observed sample. By matching these two auxiliary distributions p and q, we can guarantee the consistency between $Q_{Z|X}$ and Q_X implicitly. To this end, we employ KL-divergence to match them as follows:

$$J_2 = KL(q||p) = \frac{1}{n}\sum_{i=1}^{n}\sum_{j=1}^{c} q_{ij} \log \frac{q_{ij}}{p_{ij}} \ . \tag{16}$$

Now, the naturally following question is how to represent p and q seamlessly with previous networks' architectures.

For the modality assignment distribution p of the latent code, we propose to compute it as follows:

$$p_{ij} = \frac{\exp(w_j^T z_i)}{\sum_{j'} \exp(w_{j'}^T z_i)} \ , \tag{17}$$

where z_i is the latent code of the i-th samples, w_j is the model parameter associated with the j-th modality which can be learned jointly with network parameters. Actually, this is a softmax operation, which predicts the probability of the i-th latent code belonging to the j-th modality. Interestingly, this operation is to add a softmax layer on the top of the inference network, just as shown in Figure 1, which can be learned jointly with the other two networks' parameters.

For the modality assignment distribution q of observed samples, it is usually not available in real-world applications. Thus, how to find q is challenging. In this paper, we propose a novel strategy to compute q dynamically. Specifically, since q_{ij} denotes the probability of the i-th sample belonging to the j-th modality, it can be initialized by K-means. In this way, each cluster corresponds to one modality, and the clustering assignment can be viewed as the modality assignment. Note that we didn't perform K-means on the original samples. Instead, we pretrain the inference network in the layer-by-layer way, obtaining new representations of observed samples, then we perform K-means on the new representation to get the desired initialization. Afterward, we refine q iteratively since such an initialized distribution q is not perfect. Specifically, after obtaining network parameters and p, we fix them and optimize Eq. (16) to update q, which is easy to solve by setting the gradient to zero. Then, we fix q to update network parameters and p.

Finally, the objective function of our model is:

$$\min_{\theta,\beta,w,q} J_1 + J_2, \tag{18}$$

where θ and β are model parameters of the generation and inference network respectively, w is the model parameter of the softmax layer. This model can be optimized easily by stochastic gradient descent method.

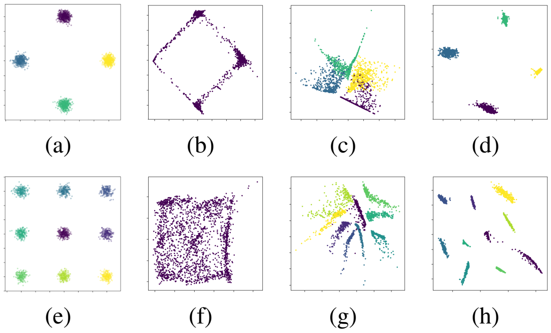

Figure 2: 1st-row: Synthetic Data-1. 2nd-row: Synthetic Data-2. In each row, they are: Real Samples, Generated Samples, Latent Code learned without Regularization (*i.e.* Eq. (15)), Latent Code Learned with Regularization (*i.e.* Eq. (18)).

5 Experiments

5.1 Synthetic Data

At first, we conduct two experiments on the synthetic dataset. Specifically, we construct two synthetic datasets. The first dataset shown in Figure 2(a) is drawn from a 2D Gaussian mixture distribution Q_X^1 which has 4 components. The second dataset shown in Figure 2(e) is more complicated than the first one. It is drawn from a 2D Gaussian mixture distribution Q_X^2 which has 9 components. For both of them, we draw 2,000 samples.

Both the generation and inference network of the first synthetic dataset is a 3-layer MLP: $[100, 100, 2]$. Similarly, that of the second synthetic dataset is $[100, 300, 2]$. The activation function employed is ReLU [Nair and Hinton, 2010]. Note that the last layer employs the linear activation. The size of mini-batch is set as 500. The kernel employed in JMMD is the Gaussian kernel. Here, we use a mixture of several Gaussian kernels, that is $k(x_i, x_j) = \sum_{m=1}^{M} k_m(x_i, x_j)$ where different kernels have different bandwidth parameters. In this paper, the bandwidth employed is $\{2.0, 5.0, 10.0, 20.0, 40.0, 80.0\}$.

The result is shown in Figure 2. From Figure 2(b) and 2(f), we can find our proposed method can generate similar samples with the real ones, which means our approach has learned a good generation network. Figure 2(d) and 2(h) show that the inferred latent code follows a multi-modal distribution as the sample distribution. Thus, the inference network learned from our model can capture the intrinsic latent structure in the dataset. In Figure 2(c), the latent code is learned without regularization, that is Eq. (15). While, in Figure 2(d), the latent code is learned from our proposed method with multi-modal regularization, that is Eq. (18). Comparing these two figures, we can find that the latent code learned with regularization has a better structure than that without regularization. Specifically, the margin among different modalities is larger and each modality is more compact, which verifies the effectiveness of our proposed regularization technique. Similar observations can be found from Figure 2(g) and 2(h). In summary, all these observations have confirmed the effectiveness of our proposed JGMMN.

5.2 Real-World Data

To further show the performance of our proposed model, we conduct experiments on five real-world datasets, which includes 3 image datasets: MNIST [LeCun *et al.*, 1998], USPS [Cai *et al.*, 2011] , ExtendYaleB (EYB), and 2 text datatsets: Reuters-10K [Xie *et al.*, 2016], 20News[1]. We summarize the statistics of these datasets in Table 2. The network architectures (from the 2nd layer) for these datasets are summarized in Table 3. Both the generation and the inference network consist of a 4-layer MLP. In our experiments, the generation network employs ReLU in all layers except the last layer. Sigmoid activation function is used in the last layer. The inference network also utilizes ReLU in all layers other than the last layer. The linear activation function is used in its last layer. The number of modalities c is set as the number of classes. The used kernels are same as those of the synthetic dataset.

The Generative Network: In Figure 3, we show the real and generated samples of three image datasets. Specifically, the fourth column in Figure 3 is the generated samples of our proposed JGMMN. The second and third columns are those

[1] http://qwone.com/~jason/20Newsgroups/

Methods	MNIST		USPS		EYB		Reuters-10K		20News	
	ACC	NMI	ACC	NMI	ACC	NMI	ACC	NMI	ACC	NMI
Kmeans	0.5323	0.4997	0.6743	0.6148	0.0994	0.1300	0.5487	0.42902	0.4055	0.3975
AE+Kmeans	0.8212	0.7546	0.7348	0.7105	0.3318	0.5009	0.7156	0.4809	0.4752	0.4479
VAE+Kmeans	0.7281	0.6094	0.4432	0.3667	0.1700	0.3631	0.6590	0.3975	0.3830	0.4147
DEC	0.8588	0.8261	0.7713	**0.7925**	0.3144	0.4533	0.6990	0.5108	0.4392	0.4593
JGMMN	**0.8667**	**0.8434**	**0.7714**	0.7913	**0.3417**	**0.5077**	**0.7454**	**0.5577**	**0.5020**	**0.4776**

Table 1: Clustering performance of different methods

Dataset	#Samples	#Classes	#Features
MNIST	70,000	10	28 × 28 × 1
USPS	9,298	10	16 × 16 × 1
EYB	2,414	38	32 × 32 × 1
Reuters-10K	10,000	4	2000
20News	18,774	20	2000

Table 2: The description of real-world datasets.

Dataset	Generation	Inference
MNIST	$[64, 256, 256, 784]$	$[500, 500, 2000, 10]$
USPS	$[64, 256, 256, 256]$	$[100, 100, 500, 10]$
EYB	$[64, 256, 256, 1024]$	$[100, 100, 500, 10]$
Reuters-10K	$[64, 256, 256, 2000]$	$[500, 500, 2000, 10]$
20News	$[64, 256, 256, 2000]$	$[500, 500, 2000, 20]$

Table 3: The network architecture of real-world datasets.

of ALI [Dumoulin *et al.*, 2016] and GMMN [Li *et al.*, 2015] respectively. Although the generated samples of GMMN and JGMMN is a little obscure than those of ALI due to the intrinsic property of kernels [Li *et al.*, 2017], most generated samples of JGMMN are very similar with real samples and those of baseline methods, which confirms the correctness of our JGMMN.

The Inference Network: To show the performance of the inference network of JGMMN, we perform clustering on the obtained latent code. Here, we directly use the output of the softmax layer defined in Eq. (17) as the clustering result. Then, we compare our method with four baseline methods: K-means, AE+K-means which performs K-means on the learned feature from autoencoder, VAE+K-means which performs K-means on the learned feature from variational autoencoder and DEC [Xie *et al.*, 2016]. Note that, to make a fair comparison, we didn't compare it with those CNN-based deep clustering methods. Here, we only focus on the DNN structure and ensure these DNN-based methods have consistent network architecture. Here, we use Clustering Accuracy (ACC) and Normalized Mutual Information (NMI) to evaluate the performance the clustering performance. From Table 1, we can find our proposed JGMMN has achieved the best result for most cases. In conclusion, the inference network of our proposed JGMMN can learn meaningful latent code from samples.

Additionally, to verify the effect of the multi-modal regularization, we compare our method with JGMMN-NR which corresponds to Eq.(15). The clustering result is shown in Figure 4. Here, we only show the result of MNIST dataset. The other datasets have similar results. We can find that JGMMN-NR has very bad clustering performance, which means that its inference network fails to learn discriminative latent representations. Thus, the proposed modal regularization is very critical to learn a good inference network.

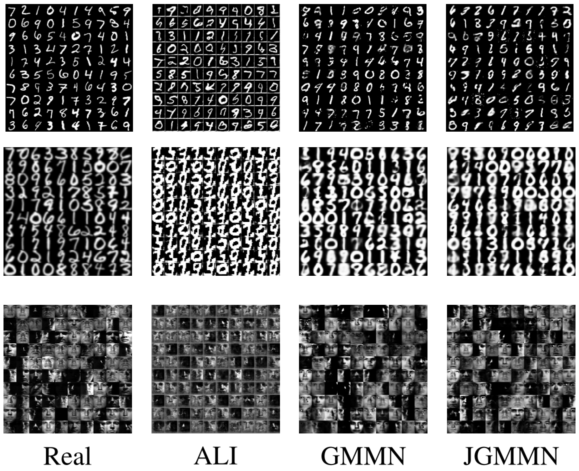

Figure 3: 1st-row: MNIST. 2nd-row: USPS. 3rd-row: EYB.

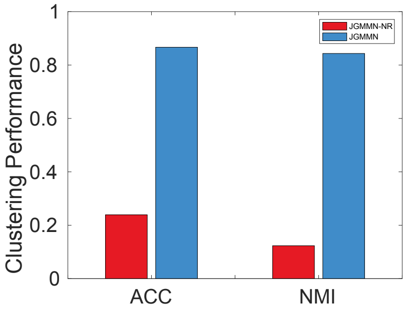

Figure 4: ACC and NMI of JGMMN and JGMMN-NR for MNIST.

6 Conclusion

In this paper, we propose a novel Joint Generative Moment-Matching Network, which can simultaneously learn a generative network and an inference one. To learn them, we employ the Joint Maximum Mean Discrepancy to match two joint distributions. Furthermore, we propose a novel multi-modal regularization to enforce the latent distribution to be consistent with the sample distribution. Extensive experiments have verified the effectiveness and correctness of our proposed JGMMN model.

Acknowledgements

This work was partially supported by the following grants: NSF-IIS 1302675, NSF-IIS 1344152, NSF-DBI 1356628, NSF-IIS 1619308, NSF-IIS 1633753, NIH R01 AG049371.

References

[Arjovsky *et al.*, 2017] Martin Arjovsky, Soumith Chintala, and Léon Bottou. Wasserstein gan. *arXiv preprint arXiv:1701.07875*, 2017.

[Cai *et al.*, 2011] Deng Cai, Xiaofei He, Jiawei Han, and Thomas S Huang. Graph regularized nonnegative matrix factorization for data representation. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 33(8):1548–1560, 2011.

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

[Denton *et al.*, 2015] Emily L Denton, Soumith Chintala, Rob Fergus, et al. Deep generative image models using a laplacian pyramid of adversarial networks. In *Advances in neural information processing systems*, pages 1486–1494, 2015.

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

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

[Gretton *et al.*, 2012] Arthur Gretton, Karsten M Borgwardt, Malte J Rasch, Bernhard Schölkopf, and Alexander Smola. A kernel two-sample test. *Journal of Machine Learning Research*, 13(Mar):723–773, 2012.

[Gulrajani *et al.*, 2017] Ishaan Gulrajani, Faruk Ahmed, Martin Arjovsky, Vincent Dumoulin, and Aaron Courville. Improved training of wasserstein gans. *arXiv preprint arXiv:1704.00028*, 2017.

[Kim *et al.*, 2017] Taeksoo Kim, Moonsu Cha, Hyunsoo Kim, Jungkwon Lee, and Jiwon Kim. Learning to discover cross-domain relations with generative adversarial networks. *arXiv preprint arXiv:1703.05192*, 2017.

[LeCun *et al.*, 1998] Yann LeCun, Léon Bottou, Yoshua Bengio, and Patrick Haffner. Gradient-based learning applied to document recognition. *Proceedings of the IEEE*, 86(11):2278–2324, 1998.

[Ledig *et al.*, 2016] Christian Ledig, Lucas Theis, Ferenc Huszár, Jose Caballero, Andrew Cunningham, Alejandro Acosta, Andrew Aitken, Alykhan Tejani, Johannes Totz, Zehan Wang, et al. Photo-realistic single image super-resolution using a generative adversarial network. *arXiv preprint arXiv:1609.04802*, 2016.

[Li *et al.*, 2015] Yujia Li, Kevin Swersky, and Rich Zemel. Generative moment matching networks. In *Proceedings of the 32nd International Conference on Machine Learning (ICML-15)*, pages 1718–1727, 2015.

[Li *et al.*, 2017] Chun-Liang Li, Wei-Cheng Chang, Yu Cheng, Yiming Yang, and Barnabás Póczos. Mmd gan: Towards deeper understanding of moment matching network. In *Advances in Neural Information Processing Systems*, pages 2200–2210, 2017.

[Long *et al.*, 2016] Mingsheng Long, Han Zhu, Jianmin Wang, and Michael I Jordan. Deep transfer learning with joint adaptation networks. *arXiv preprint arXiv:1605.06636*, 2016.

[Nair and Hinton, 2010] Vinod Nair and Geoffrey E Hinton. Rectified linear units improve restricted boltzmann machines. In *Proceedings of the 27th international conference on machine learning (ICML)*, pages 807–814, 2010.

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

[Reed *et al.*, 2016] Scott Reed, Zeynep Akata, Xinchen Yan, Lajanugen Logeswaran, Bernt Schiele, and Honglak Lee. Generative adversarial text to image synthesis. *arXiv preprint arXiv:1605.05396*, 2016.

[Ren *et al.*, 2016] Yong Ren, Jun Zhu, Jialian Li, and Yucen Luo. Conditional generative moment-matching networks. In *Advances in Neural Information Processing Systems*, pages 2928–2936, 2016.

[Song *et al.*, 2009] Le Song, Jonathan Huang, Alex Smola, and Kenji Fukumizu. Hilbert space embeddings of conditional distributions with applications to dynamical systems. In *Proceedings of the 26th Annual International Conference on Machine Learning*, pages 961–968, 2009.

[Song *et al.*, 2010] Le Song, Byron Boots, Sajid M Siddiqi, Geoffrey J Gordon, and Alex Smola. Hilbert space embeddings of hidden markov models. 2010.

[Sriperumbudur *et al.*, 2010] Bharath K Sriperumbudur, Arthur Gretton, Kenji Fukumizu, Bernhard Schölkopf, and Gert RG Lanckriet. Hilbert space embeddings and metrics on probability measures. *Journal of Machine Learning Research*, 11(Apr):1517–1561, 2010.

[Xie *et al.*, 2016] Junyuan Xie, Ross Girshick, and Ali Farhadi. Unsupervised deep embedding for clustering analysis. In *International Conference on Machine Learning*, pages 478–487, 2016.

[Zhang *et al.*, 2016] Richard Zhang, Phillip Isola, and Alexei A Efros. Colorful image colorization. In *European Conference on Computer Vision*, pages 649–666, 2016.

[Zhu *et al.*, 2017] Jun-Yan Zhu, Taesung Park, Phillip Isola, and Alexei A Efros. Unpaired image-to-image translation using cycle-consistent adversarial networks. *arXiv preprint arXiv:1703.10593*, 2017.