Mixtures of Variational Autoencoders

Fei Ye and Adrian G. Bors

Department of Computer Science, University of York, York YO10 5GH, UK

***Abstract*—In this paper, we develop a new deep mixture learning framework, aiming to learn underlying complex data structures. Each component in the mixture model is implemented using a Variational Autoencoder (VAE). VAE is a well known deep learning model which models a latent space data representation on a variational manifold. The mixing parameters are estimated from a Dirichlet distribution modelled by each encoder. In order to train this mixture model, named M-VAE, we derive a mixture evidence lower bound on the sample log-likelihood, which is optimized in order to jointly estimate all mixture components. We further propose to use the d-variables Hilbert-Schmidt Independence Criterion (dHSIC) as a regularization criterion in order to enforce the independence among the encoders' distributions. This criterion encourages the proposed mixture components to learn different data distributions and represent them in the latent space. During the experiments with the proposed M-VAE model we observe that it can be used for discovering disentangled data representations which can not be achieved with a single VAE.**

***Index Terms*—Mixture models, Variational autoencoder, Hilbert-Schmidt Independence Criterion.**

I. Introduction

Deep generative models have emerged as an efficient artificial intelligence computational framework. Generative Adversarial Network (GAN) [1] is one of the most popular generative models which can directly yield new images from passing noise through their structure. Nevertheless, GANs do not have appropriate inference mechanisms and therefore have a limited capacity for representation learning. Variational Autoencoders (VAEs) represent another kind of generative models, which are trained for optimizing the evidence lower bound (ELBO) on the data's log-likelihood. During VAE's training, the parameters of the generative model are updated based on sampling from the latent variable space, while the inference model learns the posterior in order to approximate the prior data distribution. VAE was shown to be able to provide an approximate inference mechanism that would benefit many down-stream tasks such as semi-supervised classification [2], [3], and attribute manipulation, [4], among others.

Although the VAE yields promising results in various applications, it tends to ignore the complexity of the latent space when using a powerful decoder. On the other hand, VAEs use a simple prior distribution such as a Normal distribution while aiming to minimize the Kullback-Leibler (KL) divergence between a prior and its posterior, which prevents it from capturing complex underlying structures behind data. In this paper, we develop a new deep latent generative model, namely the Mixture of Variational Autoencoders (M-VAE), which is built from a mixture of VAEs. We propose to maximize a lower bound on the sample log-likelihood as the objective function, which is used to train jointly all components. The goal of the resulting mixture model is to learn a rich data representation while also enforcing the separability between different encoding distributions, each modelled by a component using a d-variables Hilbert-Schmidt Independence Criterion (dHSIC). dHSIC can be easily incorporated into the objective function to encourage each component to capture data in different ways. We perform a series of experiments to demonstrate that the proposed mixture model not only learns several distinct clusters in the latent space but can also enhance the generative ability of the model. In the rest of the paper, in Section II we discuss the related research approaches. In Section III we discuss the proposed Mixture of Variational Autoencoders (M-VAE) model while in Section IV we present its regularization mechanism. In Section V we provide the experimental results and in Section VI we draw the conclusions of this study.

II. Related research work

In this section, we provide a brief review of relevant methods, starting with the classical Gaussian Mixture Models (GMM), and then following with discussing deep learning models, such as the Variational Autoencoder (VAE).

A. The Gaussian Mixture Model (GMM)

Gaussian Mixture Models (GMMs) have been shown to approximate any continuous probability density function (pdf) by using a mixture of Gaussian functions, [5], [6]. They can also be seen as an extension of kernel density estimation (kdf) nonparametric modeling. GMMs have been embedded into Radial Basis Functions (RBF) networks [7], which are shallow two-layer networks, with each neuron from the first layer implementing a Gaussian function, while the second layer has fully connected linear processing units. The processing units from the first layer represent a localized data space within an elliptically defined region characterized by a center and a covariance matrix, while the second layer would map together multiple distinct regions in the feature space. Backpropagation algorithm was used for training RBF networks in [7], while variational expectation maximization was employed for modelling of uncertainty in GMMs in [8]–[10].

There have been some attempts to extend the GMM model into the deep learning framework. The Gaussian mixture variational autoencoder which considers mainly unsupervised

978-1-7281-8750-1/20/$31.00 ©2020 IEEE

clustering problems was proposed in [11]. The Infinite Variational Autoencoder, which consists of a collection of VAEs, where each VAE is treated as a distinct component, was proposed in [12]. The output of the mixture model combines the generative results of all VAEs with the mixing parameters sampled from a Dirichlet process where additional VAEs can be added when appropriate.

B. The Variational Autoencoder

The variational autoencoder (VAE) was proposed in [13] and consists of two networks which work in tandem with each other: Encoder and Decoder. Let us consider a data vector $\mathbf{x}$ and a vector of stochastic latent variables $\mathbf{z}$. The learning goal of a VAE is to maximize the average marginal log-likelihood for the given observations. However, optimizing this objective function is challenging because of the intractability of its marginal distribution which is integrated on the entire latent variable space. Variational inference was then proposed, resulting in the following objective function :

$$L_{ELBO} = \frac{1}{N}\sum_{i=1}^{N} E_{\mathbf{x}_i\sim q(\mathbf{z}|\mathbf{x})}[\log p(\mathbf{x}_i|\mathbf{z})] - D_{KL}(q(\mathbf{z}|\mathbf{x})||p(\mathbf{z})) \tag{1}$$

where the two conditional distributions $p(\cdot)$ and $q(\cdot)$, characterizing the decoder and the encoder, respectively, are parameterized by convolution neural networks (CNNs), while N represents the size of the data. The objective function L_{ELBO} from equation (1) has two components, with the first term corresponding to the expectation of the negative reconstruction error which is obtained by sampling from the encoder distribution. A latent representation is sampled from the characteristic distribution of the Encoder, considered as Gaussian, by using the reparametrization trick. The reconstruction term encourages the decoder to learn how to reconstruct the data. A VAE can also be seen as a standard autoencoder (AE) if we would only optimize the first term from (1). Meanwhile, the second term from (1) corresponds to the Kullback-Leibler (KL) divergence between the prior and the distribution of the encoder. This term is also called the regularization term which encourages the posterior to match the prior distribution $p(\mathbf{z})$. The prior is usually modelled by a multi-dimensional Gaussian distribution with the covariance matrix as the identity matrix.

III. The Mixture of VAEs (M-VAE) model

The proposed M-VAE mixture model is made off a collection of Variational Autoencoder components. The generation process for M-VAE is defined as:

$$w_i \sim \ Dir\,(\mathbf{a}) \tag{2}$$

$$\mathbf{z}_i \sim \mathcal{N}(\mu_i(\mathbf{x};\delta_i), \sigma_i^2(\mathbf{x};\delta_i)) \tag{3}$$

$$\mathbf{x}_i' \sim \mathcal{N}(\mu_i(\mathbf{z};\theta_i), \sigma_i^2(\mathbf{z};\theta_i)) \tag{4}$$

where w_i, for $i = 1,\ldots,K$, is the mixing parameter for the i-th VAE, where K represents the number of mixture's components. w_i is sampled from its characteristic Dirichlet distribution $Dir(\mathbf{a})$, which is modelled by the parameter vector $\mathbf{a}$. Meanwhile, each entry a_i from the vector $\mathbf{a} = \{a_1,\ldots,a_K\}$ is estimated by its corresponding encoder. $\mathbf{z}_i$ is the latent vector sampled from the Gaussian distribution whose mean and variance are given by the i-th encoder, $i = 1,\ldots,K$. The reconstructed data $\mathbf{x}_i'$ is also sampled from the Gaussian distribution modelled by the i-th decoder. Finally, the outputs of all decoders are multiplied by the associated weights w_i, $i = 1,\ldots,K$, resulting in :

$$E(\mathbf{x}') = \sum_{i=1}^{K} p_{\delta_i}(w_i|\mathbf{x}) E_{p_{\delta i}(z_i|\mathbf{x})} q_{\theta_i}(\mathbf{x}|\mathbf{z}_i), \tag{5}$$

$$\sum_{i=1}^{K} w_i = 1, \tag{6}$$

where $q_{\theta_i}(\cdot)$ represents the approximate posterior characterizing the decoder of the i-th VAE component.

In the following we derive the optimization function for training the M-VAE model. In order to maximize the sample log-likelihood we derive a lower bound, called the Mixture Evidence Lower Bound Optimization (MELBO) :

$$\begin{aligned}\log p(\mathbf{x}) =& \log \int\int p(\mathbf{x},\mathbf{w},\mathbf{z}) d\mathbf{w} d\mathbf{z} \\ \geq& E_{q(\mathbf{z},\mathbf{w}|\mathbf{x})} \log\left[\frac{p(\mathbf{x},\mathbf{z},\mathbf{w})}{q(\mathbf{z},\mathbf{w}|\mathbf{x})}\right] = L_{MELBO}\end{aligned} \tag{7}$$

This can be rewritten as:

$$\begin{aligned}L_{MELBO} = & \sum_{i=1}^{K} p_{\delta_i}(w_i|\mathbf{x}) E_{p_{\delta i}(\mathbf{z}_i|\mathbf{x})}(q_{\theta_i}(\mathbf{x}|\mathbf{z}_i)) \\ & - \tfrac{1}{K}\sum_{i=1}^{K} D_{KL}^{i}(p_{\delta_i}(\mathbf{z}_i|\mathbf{x})||p(\mathbf{z}_i)) - \\ & - D_{KL}(p_{\delta_i, i=1,\ldots,K}(\mathbf{w}|\mathbf{x})||\pi(\mathbf{w})),\end{aligned} \tag{8}$$

where the first term represents the reconstruction error and the following terms are the KL divergence between the prior and posterior. This loss function is used to jointly train all mixing components.

IV. Kernel based independent estimation for regularization

If the optimization function is defined as in equation (8), the result can be that each component from the mixture model captures identical data structures. This would lead to overlaps between the latent spaces of different mixture components, making them redundant. In this section, we propose to add an additional penalty term to the objective function L_{MELBO} from equation (8) in order to enforce the independence of the mixing components :

$$L_{Obj} = -L_{MELBO} + \beta r(\mathbf{z}), \tag{9}$$

where $r(\mathbf{z})$ represents the penalty term and β is the parameter controlling the significance of the regularization.

In the following we use a regularization function depending on the kernel-based distance between two distributions calculated over the latent space $\mathbf{z}$. This distance, defined via the canonical distance characterizing the Hilbert space $\mathcal{H}$

embeddings, is similar to the maximum mean discrepancy, [14]. The inner product between the latent space features is calculated by a kernel function:

$$k(\mathbf{z}_1, \mathbf{z}_2) = \langle \theta(\mathbf{z}_1), \theta(\mathbf{z}_2) \rangle \tag{10}$$

where $\theta(\mathbf{z}_i)$ is a feature mapping on $\mathbf{z}_i$. The joint distribution $P(\mathbf{z}_1, \mathbf{z}_2)$ is represented using the covariance operator $C_{z_1 z_2}$:

$$C_{z_1 z_2} := E_{z_1 z_2}[(\theta_1(z_1) - \mu_{z_1}) \otimes (\theta_2(z_2) - \mu_{z_2})] \tag{11}$$

where $\mu_{z_i} = E_{z_i}[\theta_i(z_i)]$, and $\theta_i(\cdot)$ represent feature mappings, $i = 1, 2$, while $\otimes$ denotes the tensor product. This represents the generalization of the covariance matrix between random vectors, [15], in the tensor space. The largest eigenvalue of the operator $C_{z_1 z_2}$ measures the dependence between two data distributions. We want this eigenvalue to be zero in order to achieve the independence between the mixing components. Let $\mathbf{z}_i = \{z_{i_1}, z_{i_2}, \ldots, z_{i_K}\}$ be a set of iid random vectors $i = 1, \ldots, n$. We can derive a measure of independence using the Hilbert-Schmidt norm of the cross-covariance operator. This criterion is called the Hilbert-Schmidt Independence Criterion (HSIC) :

$$\begin{aligned} HSIC(\mathbf{z}) \quad &= \tfrac{1}{n^2} \sum_{ij}^{n} k(z_i, z_j) l(z'_i, z'_j) + \\ &\sum_{i,j,k,l}^{n} k(z_i, z_j) l(z'_k, z'_l) - \tfrac{2}{n^3} \sum_{i,j,k}^{n} k(z_i, z_j) l(z'_i, z'_k) \end{aligned} \tag{12}$$

where $\mathbf{z}'_i$ are independent and identical distributed (iid) copies of $\mathbf{z}_i$, and $l(\cdot, \cdot)$ is a kernel defined on $\mathcal{H}$, similarly to $k(\cdot, \cdot)$.

Similarly to (12), the definition of the cross-covariance operator can be extended into the dHSIC criterion for assessing the independence of d variables, [16], [17]. For the M-VAE model, we consider all possible pairs of distributions associated with the M-VAE's components. dHSIC is null only if the components of the random vector $\mathbf{z}$ are mutually independent. The K components are mutually independent if their joint distribution is equal to the tensor product of their marginal distributions, [18]. dHSIC is defined as:

$$\begin{aligned} \text{dHSIC}(\mathbf{z}) = \quad & \tfrac{1}{n^2} \sum_{M_2(n)} \prod_{j=1}^{K} k^j(z^j_{i_1}, z^j_{i_2}) \\ & + \tfrac{1}{n^{2K}} \sum_{M_{2K}(n)} \prod_{j=1}^{K} k^j(z^j_{i_{2j-1}}, z^j_{i_{2j}}) - \\ & - \tfrac{2}{n^{K+1}} \sum_{M_{K+1}(n)} \prod_{j=1}^{K} k^j(z^j_{i_1}, z^j_{i_{j+1}}) \end{aligned} \tag{13}$$

where $M_2(n)$, $M_{2K}(n)$ and $M_{K+1}(n)$ represent the sums of kernel products for the given n variables.

The objective function, when considering dHSIC as the regularization is then defined as :

$$L_{obj} = -L_{MELBO} + \beta \, \text{dHSIC}(\mathbf{z}). \tag{14}$$

By employing the dHSIC measure as a regularizer has many advantages. Firstly, dHSIC is a positive measure, which can be easily incorporated into the objective function, still representing a lower bound on the sample log-likelihood. Moreover, dHSIC can be easily optimized in the low-dimensional latent space. Finally, we estimate dHSIC by using the mini-batch learning procedure, where the latent variables are sampled from various components of the mixture model.

V. Experimental results

In the following we evaluate the feature learning capabilities and generative performance for the proposed Mixture of VAEs (M-VAE) model on various image databases. All encoders and decoders are implemented using parameterized convolution neural networks (CNNs). The number of convolution layers and that of the fully connected layers are chosen depending on the complexity and the size of each database. The prior for each component is considered as the Gaussian distribution $\mathcal{N}(0, I)$ while each encoder represents the hyperparameters corresponding to a Gaussian distribution. We set $\beta = 1$ in the objective function from equation (14) for all experiments. We have used Tensorflow platform and Python programming language.

(a) Randomly selected images. (b) Reconstructed results by M-VAE.

Fig. 1. Image reconstruction results by M-VAE for the MNIST dataset.

A. Results on image databases showing digits

In this section, we evaluate the performance of the proposed M-VAE mixture model on the MNIST database [19]. We consider contains 60,000 and 100,000 of training and testing images from MNIST database, respectively. Each image has a size of 28×28 pixels and represents examples of handwritten digits. In the experiments we consider four VAE components for the mixture model. We use 2 fully connected layers to create the networks for the encoder and decoder while the number of neurons on each layer is set to 500. The softplus activation function is used in the encoder in order to estimate the hyperparameters corresponding to a Gaussian function and one of the parameters for the Dirichlet distribution. We consider 20 dimensional latent codes for learning the data representation. We use the Adam optimization algorithm with a learning rate of 0.001 to train the M-VAE model on the MNIST database and the number of epochs is set to 100. Randomly selected images from the MNIST database are shown in Figure 1a while their corresponding reconstructions by the M-VAE model are shown in Figure 1b. We observe that the proposed mixture model can reconstruct well simple images of handwritten digits. The results for each component of the mixture are shown in Figure 2, where Figure 2a

shows random test images from the MNIST database, and Figures 2b-e provide the reconstructed results provided by each of the mixture components $iin\{1, 2, 3, 4\}$, by setting the mixing parameters to 1 for each component while the others are zero. The bottom row from Figure 2f shows the results provided by the M-VAE model. From these results we can find that the M-VAE model can yield better results than any of the individual VAE components. Table I summarizes the reconstruction error, calculated as the mean square error for the proposed M-VAE approach and for a single VAE. As it can be seen, the reconstruction error for M-VAE is reduced when increasing the number of mixture components.

(a) Real test samples.

(b) Results of component 1.

(c) Results of component 2.

(d) Results of component 3.

(e) Results of component 4.

(f) Results of mixture model.

Fig. 2. The reconstructed results by the proposed approach on the MNIST database.

TABLE I
RECONSTRUCTION ERROR ON THE MNIST DATABASE.

Model	No. of Components	MSE
M-VAE	7	11.54
M-VAE	8	10.20
M-VAE	10	9.17
VAE	1	13.44

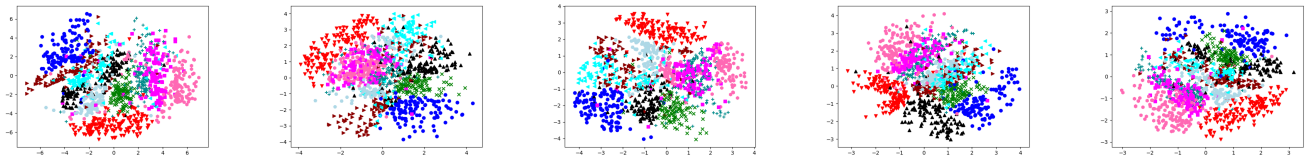
In the following we also assess the variability in the information captured by the M-VAE model from the data. We train the mixture model in order to learn two-dimensional latent representations which can be easily analyzed through graphic projections of the latent space modelled by the encoder components. We project the resulting latent space variables corresponding to each VAE component onto a 3D space representation and the results are shown in Figure 3, where different colors identify the digit classes 1-9 from the MNIST database. We observe that the latent variables learned by each individual VAE component would capture specific characteristics of the digit images. The reason for this is that the dHSIC, used as the regularization in the loss function from (14), encourages each VAE component to learn various characteristics of data in different ways. Consequently, each mixture component learns a specific region from the latent space for each data class. Then the output of the M-VAE represents the combination of all such regions resulting in a complete overall model.

B. Testing the representation ability of the features inferred by the M-VAE encoders

In this section, we investigate the usefulness of the representation learned by M-VAE. We show that the proposed M-VAE mixture model can capture aspects of data in different ways when compared to what it can be achieved when using a single VAE. Firstly, by using a single VAE has limitations when capturing the details of images due to its simple posterior and prior assumptions. Moreover, different components in the proposed M-VAE mixture model are encouraged to learn different aspects of the images, according to the dHSIC criterion, as explained in Section IV, thus providing a powerful latent representation. In the following we train the proposed M-VAE model considering four encoders. We consider the codes produced by the encoders as features for a series of simple classifiers, such as: Multi-Layer Perceptron (MLP), Linear Support Vector Machine (SVM) and the k-nearest neighbours (KNN). We train these classifiers considering the latent variables produced by the encoders. For the M-VAE model, we combine all the latent variables corresponding to the four VAE components as a vector of 80 dimensions. The classification results on the MNIST database are presented in Table II where C_i denotes the classification accuracy for the classifiers trained on the latent variables learned and represented by the i-th VAE component. From this table we observe that when combining the latent variables sampled from all four components, we achieve better results than when using any of the VAE components, by any of the classifiers considered. This demonstrates that the mixture model can provide more powerful latent representations. We also investigate the relationship between the performance and the complexity of the mixture model. We train the M-VAE model considering different number of components, while the KNN is considered on the latent variables produced by the encoders. The classification results when increasing the number of VAE components from 4 to 10, are provided in the plot from Figure 4. From this plot we observe that the performance is improved when increasing the complexity of the M-VAE model. Each VAE component capture characteristics of the data in different ways, by using the regularization as explained in Section IV, resulting in comprehensive data representations. When combining different regions of the latent variable's manifold, we provide additional useful information for the classifiers, improving their performance.

TABLE II
RESULTS ON THE MNIST DATABASE CONSIDERING A 20 DIMENSIONAL LATENT SPACE AND SIMPLE CLASSIFIERS.

Classifier	Mixture	C_1	C_2	C_3	C_4	VAE
MLP	96.59	95.15	94.67	94.68	94.30	95.35
Linear SVM	93.37	90.78	90.11	89.83	89.72	90.23
KNN	97.17	96.56	96.39	96.63	96.45	96.34



(a) Component 1. (b) Component 2. (c) Component 3. (d) Component 4. (e) VAE.

Fig. 3. The representation of the latent spaces produced by each of the four encoders of the M-VAE mixture are shown in (a)-(d), while that corresponding to a single VAE is provided in (e), when considering MNIST database for training. Each colour represents the class associated with a specific digit 1-9.

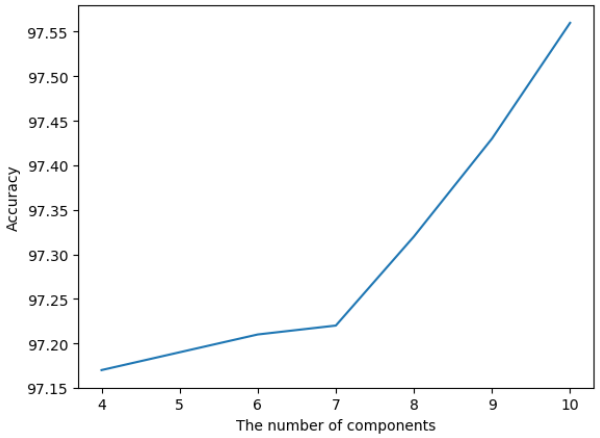


Fig. 4. The classification accuracy on the MNIST database, when increasing the number of components.

C. Results on the CIFAR10 database

In this section, we evaluate the generative performance on CIFAR10 database [20], which contains 50,000 images representing a variety of complex scenes. We consider 10,000 images for training and the same number for testing. Each image from this database is 32×32 pixels in size, labelled in 10 classes. We consider five convolution layers with 3×3 kernels for each mixture's encoder and decoder, respectively. The dimension of the latent variables is considered as 256 and we use the softplus activation function in order to estimate the variance of the Gaussian distribution and the parameter of the Dirichlet distribution, produced by each encoder. We consider $K = 4$ VAE components for the mixture model. The Adam optimization algorithm with a learning rate of 0.001 is used to train this mixture model using the backpropagation algorithm with batch normalization. A random selection of images from CIFAR10 database and its reconstruction by the proposed M-VAE model are shown in Figures 5a and 5b, respectively.

D. Assessment of the quality of generated images

We consider the Inception score (IS), which was proposed in [21], in order to assess the quality of generated images :

$$IS = \exp\left(\mathbb{E}_{\mathbf{x}}[D_{KL}(p(\mathbf{y}|\mathbf{x})||p^*(\mathbf{y}))]\right) \quad (15)$$

where $\mathbf{x}$ denotes the given image, $p(\mathbf{y}|\mathbf{x})$ is the probability represented by the output of the softmax layer of the trained classifier which indicates the ability to generate distinct images, while $p^*(\mathbf{y})$ is the overall label distribution and measures the ability to generate diverse classes of images. IS was proposed for measuring how realistic are the images generated by Generative Adversarial Networks (GANs) and assesses both the quality of the generated images as well as their diversity [22]. The Inception scores for the images generated by the proposed M-VAE and the results are shown in Table III, where we also provide the root mean square error (RMSE) evaluation. The results from Table III indicate that the generated image quality for the M-VAE model is better than that provided by the GAN based models, considered for comparison. M-VAE can yield realistic images demonstrating that the mixture model not only benefits representation learning, but also has a good generative performance.

TABLE III
INCEPTION SCORE FOR VARIOUS DEEP LEARNING MODELS ON CIFAR10 DATABASE.

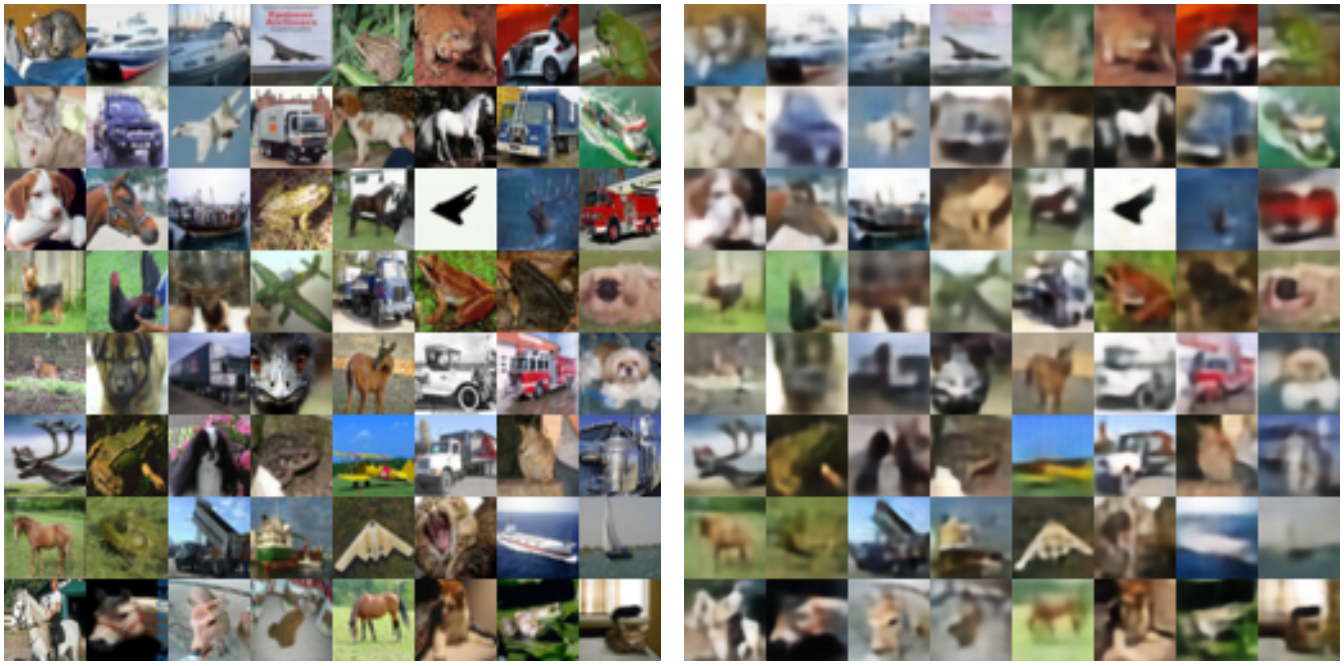
Model	RMSE	Inception score
MIX+Wasserstein GAN [23]	-	4.04
DCGAN [24] in [25]	-	4.89
ALI [26] in [25]	14.53	4.97
PixelCNN++ [27] in [25]	3.289	5.51
WGAN [23]	-	3.82
M-VAE	3.17	5.96

VI. Conclusion

In this paper, we propose a new deep learning model, called M-VAE, which is built using a mixture of VAEs. During the training, each VAE component is guided to capture different aspects of the data. In order to achieve this we consider a regularization term in the objective function enforcing the independence between the latent spaces represented by the mixture components. The experimental results show that the proposed mixture model can learn a rich data representation and can improve the quality of generated images when compared with that yielded by a single VAE model. In future research we will aim to determine the optimal number of VAE components to be used in the mixture when solving a given task or when learning a series of different tasks.

Acknowledgement

The authors would like to thank NVIDIA for granting a Titan XP GPU, which was used for the experiments.



(a) Randomly selected images. (b) Reconstructed results by M-VAE.

Fig. 5. Qualitative results of the proposed M-VAE model for CIFAR10 dataset.

REFERENCES

[1] I. Goodfellow, J. Pouget-Abadie, M. Mirza, B. Xu, D. Warde-Farley, S. Ozair, A. Courville, and Y. Bengio, "Generative adversarial nets," in *Proc. Advances in Neural Inf. Proc. Syst. (NIPS)*, 2014, pp. 2672–2680.

[2] D. P. Kingma, S. Mohamed, D. J. Rezende, and M. Welling, "Semi-supervised learning with deep generative models," in *Proc. Advances in Neural Inf. Proc. Systems (NIPS)*, 2014, pp. 3581–3589.

[3] A. Makhzani, J. Shlens, N. Jaitly, I. Goodfellow, and B. Frey, "Adversarial autoencoders," in *Proc. Int. Conf. on Learning Representations (ICLR)*, 2016. [Online]. Available: https://arxiv.org/abs/1511.05644

[4] J. Klys, J. Snell, and R. Zemel, "Learning latent subspaces in variational autoencoders," in *Proc. Advances in Neural Inf. Proc. Systems (NIPS)*, 2018, pp. 6445–6455.

[5] T. Poggio and F. Girosi, "Networks for approximation and learning," *Proc. of the IEEE*, vol. 78, no. 9, pp. 1481–1497, 1990.

[6] J. Park and J. Sandberg, "Universal approximation using radial basis functions network," *Neural Comput.*, vol. 3, no. 2, pp. 246–257, 1991.

[7] A. G. Bors and M. Gabbouj, "Minimal topology for a radial basis functions neural network for pattern classification," *Digital Signal Processing*, vol. 4, no. 3, pp. 173–188, 1994.

[8] Z. Ghahramani and M. Beal, "Variational inference for bayesian mixtures of factor analysers," in *Proc. Advances in Neural Inf. Proc. Systems (NIPS)*, 2000, pp. 449–455.

[9] H. Attias, "A variational Bayesian framework for graphical models," in *Proc. Advances in Neural Inf. Proc. Systems (NIPS)*, 2000, pp. 209–215.

[10] N. Nasios and A. G. Bors, "Variational learning for Gaussian mixture models," *IEEE Transactions on Systems, Man, and Cybernetics, Part B*, vol. 36, no. 4, pp. 849–862, 2006.

[11] N. Dilokthanakul, P. Mediano, M. Garnelo, M. Lee, H. Salimbeni, K. Arulkumaran, and M. Shanahan, "Deep unsupervised clustering with Gaussian mixture variational autoencoders," 2016. [Online]. Available: https://arxiv.org/abs/1511.05644

[12] E. Abbasnejad, M. Dick, and A. van der Hengel, "Infinite variational autoencoder for semi-supervised learning," in *Proc. of IEEE Conf. on Computer Vision and Pattern Recog. (CVPR)*, 2017, pp. 5888–5897.

[13] D. J. Rezende, S. Mohamed, and D. Wierstra, "Stochastic backpropagation and approximate inference in deep generative models," in *Proc. Int. Conf. on Machine Learning, vol. PMLR 32(2)*, 2014, pp. 1278–1286.

[14] A. Gretton, K. M. Borgwardt, M. J. Rasch, B. Schölkopf, and A. J. Smola, "A kernel two-sample test," *Jour. of Machine Learning Research*, vol. 13, no. 233, pp. 723–773, 2012.

[15] A. Gretton, K. Fukumizu, C. H. Teo, L. Song, B. Schölkopf, and A. J. Smola, "A kernel statistical test of independence," in *Proc. Advances in Neural Inf. Proc. Systems (NIPS)*, 2008, pp. 585–592.

[16] N. Pfister, P. Bühlmann, B. Schölkopf, and J. Peters, "Kernel-based tests for joint independence," *Jour. of the Royal Statistical Society: Series B (Statistical Methodology)*, vol. 80, no. 1, pp. 5–31, 2018.

[17] Z. Szabó and B. K. Sriperumbudur, "Characteristic and universal tensor product kernels," *Journal of Machine Learning Research*, vol. 18, no. 233, pp. 1–29, 2018.

[18] R. Lopez, J. Regier, M. I. Jordan, and N. Yosef, "Information constraints on auto-encoding variational Bayes," in *Proc. Advances in Neural Inf. Proc. Systems (NIPS)*, 2018, pp. 6117–6128.

[19] Y. LeCun, L. Bottou, Y. Bengio, and P. Haffner, "Gradient-based learning applied to document recognition," *Proc. of the IEEE*, vol. 86, no. 11, pp. 2278–2324, 1998.

[20] A. Krizhevsky, "Learning multiple layers of features from tiny images," University of Toronto, Canada, Tech. Rep., 2009.

[21] T. Salimans, I. Goodfellow, W. Zaremba, V. Cheung, A. Radford, and X. Chen, "Improved techniques for training GANs," in *Proc. Advances in Neural Inf. Proc. Systems (NIPS)*, 2016, pp. 2234–2242.

[22] T. Che, Y. Li, A. Jacob, Y. Bengio, and W. Li, "Mode regularized generative adversarial networks," in *Proc. Int. Conf. on Learning Representations (ICLR)*, 2017. [Online]. Available: https://arxiv.org/abs/1612.02136

[23] S. Arora, R. Ge, Y. Liang, T. Ma, and Y. Zhang, "Generalization and equilibrium in generative adversarial nets (GANs)," in *Proc. Int. Conf. on Machine Learning (ICML), vol. PMLR 70*, 2017, pp. 224–232.

[24] A. Radford, L. Metz, and S. Chintala, "Unsupervised representation learning with deep convolutional generative adversarial networks," in *Proc. Int. Conf. on Learning Representations (ICLR)*, 2016. [Online]. Available: https://arxiv.org/abs/1511.06434

[25] Y. Pu, W. Wang, R. Henao, C. L., Z. Gan, C. Li, and L. Carin, "Adversarial symmetric variational autoencoder," in *Proc. Advances in Neural Inf. Proc. Systems (NIPS)*, 2017, pp. 4333–4342.

[26] V. Dumoulin, I. Belghazi, B. Poole, O. Mastropietro, A. Lamb, M. Arjovsky, and A. Courville, "Adversarially learned inference," in *Proc. Int. Conf. on Learning Representations (ICLR)*, 2017. [Online]. Available: https://arxiv.org/abs/1606.00704

[27] T. Salimans, A. Karpathy, X. Chen, and D. P. Kingma, "Improving the pixelcnn with discretized logistic mixture likelihood and other modifications," in *Proc. Int. Conf. on Learning Representations (ICLR)*, 2017. [Online]. Available: https://arxiv.org/abs/1701.05517