Deep Clustering with Variational Autoencoder

Kart-Leong Lim and Xudong Jiang, *Senior Member, IEEE* and Chenyu Yi

***Abstract*—An autoencoder that learns a latent space in an unsupervised manner has many applications in signal processing. However, the latent space of an autoencoder does not pursue the same clustering goal as Kmeans or GMM. A recent work of Song et al proposes to artificially re-align each point in the latent space of an autoencoder to its nearest class neighbors during training. The resulting new latent space is found to be much more suitable for clustering, since clustering information is used. Inspired by Song et al, in this paper we propose several extensions to this technique. First, we propose a probabilistic approach to generalize Song's approach, such that Euclidean distance in the latent space is now represented by KL divergence. Second, as a consequence of this generalization we can now use probability distributions as inputs rather than points in the latent space. Third, we propose using Bayesian Gaussian mixture model for clustering in the latent space. We demonstrated our proposed method on digit recognition datasets, MNIST, USPS and SHVN as well as scene datasets, Scene15 and MIT67 with interesting findings.**

I. Introduction

Deep clustering networks that exploit autoencoder (AE) for clustering have been found in many recent signal processing applications such as computer vision and pattern recognition [1], [39], [14], [15], [3], [12], speech and audio recognition [7], [18], [40], [17], [27], [22], wireless communication [2], [32], [10], text classification [36], [4], [30] and etc. Deep clustering network [37], [31] typically trains a clustering algorithm e.g. Kmeans on the latent space of AE. However, the latent space of an AE may not be suitable for clustering. We can view this problem from the probabilistic perspective of the variational autoencoder (VAE) [19]. The main difference between AE and variational autoencoder (VAE) [19], [18] is the way the latent space is represented. In AE, an encoded image is represented as a point in the latent space, while in VAE an encoded image is represented by the sample draw from a Gaussian distribution. The latter is described by VAE's random variable, mean and variance associated with the image. The problem of clustering faced by VAE is that when we have a multiclass dataset such as MNIST, the underlying Gaussian distribution assumption may not be sufficient to separate different classes in the latent space. This is especially true when two different digit classes share very similar mean and variance. There is simply no mechanism in VAE that enforces samples from different classes to have different mean and variance. Unless the underlying data layout is inherently class discriminative, there is no way AE or VAE can generate a latent space suitable for clustering.

K. Lim, X. Jiang and C. Yi are with the Rapid-Rich Object Search lab, School of Electrical and Electronic Engineering, Nanyang Technological University, Singapore 639798 (email: lkartl@yahoo.com.sg, exdjiang@ntu.edu.sg, yich0003@e.ntu.edu.sg)

In order to solve VAE's clustering problem, at least two groups of researchers have converged to the same idea of using categorial distribution for VAE since the underlying distribution is discrete [11], [25]. Fortunately, there is an easier way to solve the problem. A recent approach by Song et al [31] focuses on minimizing the difference between the original latent space learnt by AE and the feature space learnt over it by traditional machine learning (ML) techniques. In such approach, there are two objectives to be solved in each iteration, the **network weights** ϕ and the **ML parameters** θ. The standard way to learn it is to alternate between each optimization while fixing the other. Our work mainly follows Song's approach [31] which we named as autoencoder with distance (AED). We further extend it to using VAE [19] which we call variational autoencoder with distance (VAED).

There are some challenges faced when using AED:

i) AE may not be the most ideal tool for training compact representation since, unlike VAE it cannot model latent space using random variable.
ii) The distance error function of AED only takes points in the latent space as inputs. It is not so straightforward to extend this function to using random variables as inputs.
iii) Kmeans assumes a spherical Gaussian distribution for each cluster. However, this is a strong assumption for most datasets.

Novel contributions in this work include:

i) Inputs to the distance error function are now probability distributions, rather than points in the latent space.
ii) The second order term (variance) of network (VAE) and ML (GMM) are now optimized by the distance error function.
iii) Bayesian GMM [5] is used to improve the clustering. More hidden variables and hyperparameters can better capture the latent space over Kmeans alone.

A. Related work

AED [31] first proposes to integrate both reconstruction error and the error between Kmeans and the encoded image (a.k.a. distance error or $L3$) into a single objective. Backpropagation on this objective will adjust the AE weights to minimize the within class latent space representation of the encoded image. Many recent works [31], [23], [34], [35], [37], [12] including our paper follow this strategy. DCN [37] offers a concise study of AED but both use identical $L3$. DC-Kmeans [34] use the alternating directed method of multiplier to train AED. The authors of DEC [35] proposed using a Student's t-distribution kernel for $L3$. DBC [23] combines a convolutional autoencoder with DEC. Instead of Euclidean distance, in Sun et al [33], $L3$ is learnt using a sparse representation. Similarly, PARTY [28] proposed a sparsity prior to $L3$. NSC-AGA [12] applies the self-expressiveness model to $L3$. The inputs to the $L3$ represented by sparse coding, NSC-AGA and PARTY are essentially point estimates. Furthermore, when sparse representation or self-expressiveness in $L3$ are negligible, we can effectively treat $L3$ as Euclidean distance much like in AED. The most related work to ours is VaDE [16]. Where the goal of VaDE is to decode images of different classes from its respective cluster representation in the latent space, our goal is to minimize the difference between a VAE's latent space and the cluster representation learnt over it using traditional machine learning approach. VaDE represents the relationship between VAE and GMM using a jointly distributed probability distribution, a.k.a. the variational lower bound in [19]. However, the intuition of VaDE is not easily understood in [16]. We offer a different perspective. The same relationship between VAE and GMM is initially inspired by AED [31]. We then further shows that this relationship can be further refined using a probabilistic approach, e.g. KL divergence. We discuss why such a probabilistic approach is necessary to correctly represent both VAE and GMM. Following that, we show that under a condition, the KL divergence will revert back to AED. Furthermore, we use a Bayesian GMM over the standard GMM. More hidden variables and hyperparameters can better capture the latent space over Kmeans alone.

II. Background

A. Gaussian mixture model (GMM)

GMM [5] models a set of N observation $z = \{z_n\}_{n=1}^{N} \in \mathbb{R}^D$, using a linear superposition of K Gaussian components. The total dimension of each observed instance is denoted by D. The GMM mean is denoted by $\eta = \{\eta_k\}_{k=1}^{K} \in \mathbb{R}^D$. When assuming diagonal covariance $\varSigma_k = \tau_k^{-1} I$, we define GMM precision as $\tau = \{\tau_k\}_{k=1}^{K} \in \mathbb{R}^D$. The GMM cluster assignment is denoted by $\zeta = \{\zeta_n\}_{n=1}^{N}$ where ζ_n is a $1-of-K$ binary vector, subjected to $\sum_{k=1}^{K} \zeta_{nk} = 1$ and $\zeta_{nk} \in \{0,1\}$. In the Bayesian approach to GMM, we treat each hidden variable as a posterior distribution and we introduce a prior distribution for each hidden variable. The Bayesian GMM posterior is formulated below, where $\theta = \{\eta, \tau, \zeta\}$

$$\begin{gathered} p(z) = \int p(z|\theta)p(\theta)d\theta, \\ p(z \mid \zeta, \eta, \tau) = \prod_{n=1}^{N} \prod_{k=1}^{K} \mathcal{N}\left(z_n \mid \eta_k, \tau_k^{-1}\right)^{\zeta_{nk}}, \\ p(\eta, \tau) = \prod_{k=1}^{K} \mathcal{N}\left(\eta_k \mid m_0, \lambda_0 \tau_k^{-1}\right) Gamma\left(\tau_k \mid a_0, b_0\right) \end{gathered} \tag{1}$$

Due to the intractable integral in $p(z)$, an approximation such as variational inference is required to perform learning of the hidden variables θ. We will discuss this in later section and we refer the readers to [5] for more details.

B. Autoencoder with distance (AED)

The AED error function first appeared in [31]. η^* refers to the nearest cluster to z in the latent space. T and y refer to target and network output respectively. We use cluster assignment to find out the k^{th} cluster membership η_k belongs to. The parameter λ_3 is set to $[0,1]$ where a smaller λ_3 reduces the effect of distance error function on network weight updates

$$\mathcal{L}^{AED} = \min_{w,b} \|T - y\|^2 - \lambda_3 \|z - \eta^*\|^2 \tag{2}$$

C. Variational autoencoder (VAE)

A standard VAE [19] has a network structure including the encoder and decoder as defined below

$$\begin{aligned} & h_j = f_1\left(\textstyle\sum_i w_{ij} \cdot x_i\right), \quad && \mu = \textstyle\sum_j w_{j\mu} \cdot h_j \\ & \ln \sigma^2 = \textstyle\sum_j w_{j\sigma} \cdot h_j, \quad && h_k = f_4\left(\textstyle\sum_z w_{zk} \cdot z\right) \\ & y_l = f_5\left(\textstyle\sum_k w_{kl} \cdot h_k\right) \end{aligned} \tag{3}$$

The latent variable z generates an output from the encoder network using $z = \mu + \sigma \cdot \epsilon$ where $\varepsilon \sim \mathcal{N}(0,1)$. For the activations, we mainly use $f() = tanh()$. The error function of VAE consists of the standard reconstruction error and KL divergence as follows.

$$\begin{aligned} \mathcal{L}^{VAE} &= \ln p(x|z) - D_{KL}\left(q(z|x) \parallel p(z)\right) \\ &= -\tfrac{1}{2}\left(T - y\right)^2 - \tfrac{1}{2}\left(\sigma_1^2 + \mu_1^2 - \ln \sigma_1^2 - 1\right) \end{aligned} \tag{4}$$

VAE represents both encoder $q(z|x)$ and decoder $p(x|z)$ using diagonal covariance Gaussian distribution. The KL divergence above is expressed in terms of VAE's random variables μ, σ which are in turn expressed in terms of network parameters w. Weight training for VAE's error function is performed using stochastic gradient ascent (SGA).

III. Proposed: VAE with distance (VAED)

A. Naive approach, AED

A naive way to represent the distance error function in VAE is to use $\|z - \eta^*\|^2$ such as in [33]. The reduced complexity is that we can exploit the reparameterization trick $z = \mu + \sigma \cdot \epsilon$ to represent mean and variance in VAE. However, problems will arise:

i) The GMM variance term τ, cannot be optimized by $\|z - \eta^*\|^2$
ii) The network gradient $\frac{\partial \mathcal{L}^{VAED}}{\partial \sigma}$ is essentially a factor of $\frac{\partial \mathcal{L}^{VAED}}{\partial \mu}$ weighted by the randomly generated noise $\varepsilon \sim \mathcal{N}(0,1)$ as seen in eqn (5)

$$\begin{aligned} \frac{\partial \mathcal{L}^{AED}}{\partial \mu} &= z - \eta^* \\ \frac{\partial \mathcal{L}^{AED}}{\partial \sigma} &= (z - \eta^*)\,\epsilon \end{aligned} \tag{5}$$

A severse issue is when $\epsilon \to 0$, the naive approach suffers from the vanishing gradient problem. Fortunately, this problem can be elevated by the proposed method in eqn (10) and (11).

B. Proposed approach, VAED

A VAE representation in the latent space is the mean and variance. In a ML approach, a GMM contains K independent Gaussian distributions that models the latent space of VAE. We can use the KL divergence to measure the distance between these two probability distributions. We introduce our VAED objective as follows

$$\mathcal{L}^{VAED} = \mathcal{L}^{VAE} - \lambda_3 \cdot D_{KL}\left(p(z_n \mid \theta) \parallel q(z_n \mid x_n)\right) \tag{6}$$

We will show an illustration of eqn (6) in Fig 1. Also, the validity of eqn (6) should prevail to cases where both $p(z_n \mid \theta)$ and $q(z_n \mid x_n)$ are no longer Gaussian distributed. We introduce the KL divergence term as our new distance error which measures the distance between the distributions of GMM and VAE encoder. We can further re-express the GMM term as

$$p(z_n \mid \theta) = \prod_{k=1}^{K} \mathcal{N}(z_n \mid \eta_k, (\tau_k)^{-1})^{\zeta_{nk}} = \mathcal{N}(z_n \mid \eta^*, (\tau^*)^{-1}) \tag{7}$$

We refer to η^* as the optimal η_k computed by GMM's cluster assignment $E\left[\zeta_{nk}\right]$ shown in the next section and vice versa for τ^*. Thus, the KL divergence for distance error now becomes a function between two Gaussian distributions. Under such assumption, KL divergence is well defined as follows

$$\begin{aligned} & D_{KL}\left(\mathcal{N}(z_n \mid \eta^*, (\tau^*)^{-1}) \parallel \mathcal{N}(z_n \mid \mu, \sigma)\right) \\ & = \ln \tau^* + \ln \sigma + \tfrac{(\tau^*)^{-1} + (\eta^* - \mu)^2}{2\sigma^2} - \tfrac{1}{2} \end{aligned} \tag{8}$$

Interestingly, when we assume spherical Gaussian i.e. unit variance, the VAED distance error reverts back to the AED distance error

$$D_{KL}\left(\mathcal{N}(z_n \mid \eta^*) \parallel \mathcal{N}(z_n \mid \mu)\right) = \frac{1}{2}\left(\eta^* - \mu\right)^2 \tag{9}$$

C. *Optimization of VAED*

The optimization of VAED is achieved by i) using SGA to learn VAED weights $w_{ij}, w_{j\mu}, w_{j\sigma}, w_{kl}, w_{zk}$ and ii) using variational inference to learn GMM parameters $E\left[\eta_k\right], E\left[\tau_k\right], E\left[\zeta_{nk}\right]$.

1) Weights learning: It is straightforward to obtain the network gradient terms for eqn (8) as follows

$$\frac{\partial \mathcal{L}^{VAED}}{\partial \mu} = -\frac{(\eta^* - \mu)}{\sigma^2} \tag{10}$$

$$\frac{\partial \mathcal{L}^{VAED}}{\partial \sigma} = \frac{1}{\sigma} - \frac{(\tau^*)^{-1} + (\eta^* - \mu)^2}{\sigma^3} \tag{11}$$

After that, the goal is to update all the following weights for eqn (6) using SGA where Δ is the weight change of the hidden layers and γ is the learning rate.

$$\begin{aligned} w_{j\mu} &= w_{j\mu} + \gamma \Delta w_{j\mu}, \quad w_{kl} = w_{kl} + \gamma \Delta w_{kl} \\ w_{j\sigma} &= w_{j\sigma} + \gamma \Delta w_{j\sigma}, \quad w_{zk} = w_{zk} + \gamma \Delta w_{zk} \\ w_{ij} &= w_{ij} + \gamma \Delta w_{ij} \end{aligned} \tag{12}$$

2) GMM learning: The GMM parameters (or Bayesian posteriors of the hidden variables) can be learnt in the latent space using the variational inference. In variational inference, the hidden variables of a mixture model are maintained as posterior distributions. When performing iterative learning, we update the mixture model by estimating the expected value of each posterior per iteration. In Bayesian GMM [5], $E\left[\tau_k\right]$, $E\left[\eta_k\right]$, $E\left[\zeta_{nk}\right]$ are the expected value of the Bayesian posteriors of GMM precision, mean and cluster assignment respectively. Using the maximization-maximization algorithm [24], closed solution for the expectations are shown below. The hyperparameters a_0, b_0, λ_0, m_0 are treated as constants.

$$E\left[\zeta_{nk}\right] = \arg\max_{\zeta_{nk}} \left\{ \ln E\left[\tau_k\right] - E\left[\tau_k\right]\left(z_n - E\left[\eta_k\right]\right)^2 \right\} \zeta_{nk} \tag{13}$$

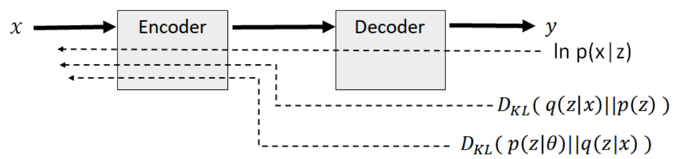
x
Encoder
Decoder
y
ln p(x|z)
$D_{KL}(\, q(z|x)||p(z)\,)$
$D_{KL}(\, p(z|\theta)||q(z|x)\,)$

Fig. 1. Proposed VAED: The weights of the encoder is updated via backpropagation using the reconstruction error, VAE's regularization error, and the proposed VAED distance error.

$$E\left[\tau_k\right] = \frac{\frac{1}{2}\sum_{n=1}^{N} E\left[\zeta_{nk}\right] + (a_0 - 1)}{b_0 + \sum_{n=1}^{N} \frac{E\left[\zeta_{nk}\right]}{2}\left(z_n - E\left[\eta_k\right]\right)^2 + \frac{\lambda_0}{2}\left(E\left[\eta_k\right] - m_0\right)^2} \tag{14}$$

$$E\left[\eta_k\right] = \frac{\sum_{n=1}^{N} z_n E\left[\zeta_{nk}\right] + \lambda_0 m_0}{\sum_{n=1}^{N} E\left[\zeta_{nk}\right] + \lambda_0} \tag{15}$$

D. *Proposed algorithm for VAED*

We introduce our proposed algorithm for VAED in Algo. 1. The first part of VAED trains a GMM in the latent space of VAED. The learnt GMM parameters are in turn used to perform VAED weight updating. Finally, the updated weights of VAED replace the weights from the previous iteration. The process repeats itself until enough iterations have passed. A way to check the convergence of VAED is to run GMM training accuracy each iteration. When normalized mutual information (NMI) and accuracy (ACC) of GMM clustering have converged, we can stop the training of VAED. We refer to [6] for NMI and ACC computations.

Algorithm 1 VAED

```
Input: $x$
Output:
a) VAED weights, $\phi = \{w_{ij}, w_{j\mu}, w_{j\sigma}, w_{zk}, w_{kl}\}$
b) GMM parameters $\theta = \{E[\eta_k], E[\tau_k], E[\zeta_{nk}]\}$
Initialization: $\phi$, $\gamma$, $\lambda_3$, $\theta$
Main: Repeat till convergence
% —-This is GMM optimization—-
    1) run forward pass to obtain $z$ given the raw input
       $x$
    2) update GMM parameters using eqn (13-15)
%—-This is VAED optimization—-
    3) given a random sample $z_n$, compute $E[\zeta_n]$ to get
       corresponding $\eta^*$ and $\tau^*$
    4) perform SGA on VAED in eqn (6) and (8)
```

IV. Experiments

A. *Comparison of end-to-end clustering*

We compare our method with recent clustering methods [31], [34], [35], [23], [37], [13], [38] in Table 1. The most commonly used digit datasets are: i) USPS [8] with 7291 train and 2007 test images, ii) MNIST [21] with 50,000 train and 10,000 test images. For USPS, we use raw image pixel as

Method	USPS NMI	USPS ACC	MNIST NMI	MNIST ACC
Kmeans [34]	0.4503	0.4585	0.5163	0.5618
AED [31]	0.5449	0.6111	0.6615	0.734
DC-Kmeans [34]	0.5737	0.6442	0.7448	0.8015
DCN [37]	-		0.81	0.83
DC-GMM [34]	0.6939	0.6476	0.8318	0.8555
DEC [35]	0.6191	0.6246	0.8273	0.8496
DBC [23]	0.724	0.743	**0.917**	**0.964**
VaDE [16]	-		-	0.945
NSC-AGA [13]	**0.7727**	0.7256	-	
VAED (ours)	0.6233	**0.7613**	0.819	0.8875

TABLE I
PROPOSED METHOD VS STATE-OF-THE-ARTS (USING RAW PIXEL)

Method	SCENE15 NMI	SCENE15 ACC	SVHN NMI	SVHN ACC	MIT67 NMI	MIT67 ACC
Original	0.7754	71.20	0.6397	68.15	0.6610	48.61
AED	0.8016	80.27	0.7080	73.73	0.6650	49.12
VAE	0.8150	82.41	0.7371	76.63	0.6516	48.62
VAED (ours)	**0.8332**	**88.12**	**0.8111**	**91.53**	**0.67098**	**58.96**

TABLE II
PROPOSED METHOD VS BASELINES (USING RESNET18)

	SCENE15	SVHN	MIT67
AED	1080	2297	7704
VAE	122	122	120
VAED	1858	3393	13200

TABLE III
COMPUTATIONAL TIME (IN SECONDS) FOR 50 ITERATIONS

feature vector, hence, the encoder network is 256-196-128. For MNIST, we also use raw image pixel as feature vector and the encoder network we use is 784-512-256. We rerun our experiments at least 10 rounds and take the average result. The GMM hyperparameters we use are $a_0 = 1.25$, $b_0 = 0.25$, $m_0 = 1$ and $\lambda_0 = 0.5$. The VAED parameter is $\lambda_3 = 0.1$.

On USPS in Table 1, Kmeans [34] obtained NMI=0.4503, ACC=0.4585 on the original feature space. All deep clustering methods outperforms Kmeans by a large margin. AED obtains better overall result than DC-GMM and DC-Kmeans. The ACC of DEC is the poorest amongst all deep methods. Overall, our proposed method obtains the best ACC but our NMI suffers. We believe this is due to VAED using randomly initalized weights and USPS having a smaller training sample size.

On MNIST in Table 1, Kmeans on the original space was only able to obtain NMI=0.5163 and ACC=0.5618. In comparison, VAED obtained better result than most methods at NMI=0.819, ACC=0.8817 except DBC. Reason could be that 3 layers for VAED's encoder may not be enough for state-of-the-art clustering on MNIST. In comparison, VaDE, DCN, DBC and DEC use 5 layers for encoder while AED, NSC-AGA, DC-Kmeans and DC-GMM use 4 layers.

B. More challenging datasets

Our next goal is to evaluate VAED on real datasets such as datasets having larger classes (MIT67) and more difficult image content such as scene categories (Scene15 and MIT67). These datasets are rarely used by deep clustering algorithms such as [31], [34], [35], [23]. As a result, we implemented AED and VAE as our baselines for comparison. For the latter, VAE is first learnt on the dataset and then Kmeans is applied in the latent space. All methods here use ResNet18 as the input and they have the same network dimensions as VAED. Our VAED encoder uses a $512 - 384 - 256$ network structure whereby 512 refers to the output dimension of Resnet18 [9] as our image feature extraction, 384 is the dimension of our 2nd layer and our latent space has 256 neurons.

Two scene recognition datasets are used in our experiments. Scene15 [20] has 15 classes, 4485 images and MIT67 [29] has 67 classes, 15,620 images. We also include SVHN [26] which is a more complex and challenging dataset than MNIST. The images in SVHN are natural and not clean which have large variance in illumination and distraction. For each dataset, we start with Kmeans on the original ResNet18 feature space. From Table 2, we see that AED is able to outperform Kmeans by a large gain in both Scene15 and SVHN. VAE is able to obtain minor performance gain over AED. However, both AED and VAE do not perform any better than Kmeans on MIT67. In fact, VAE performes worse than Kmeans on MIT67. We suspect that the poor performance of both methods on MIT67 is due to the large class number. Fortunately, VAED does not suffer from this issue. The performance of VAED is significantly much better than both AED and VAE on all three datasets. In Table 3, we compare the complexity of VAED with AED and VAE using CPU time. Overall, VAED is the most expensive. In VAE, the only requirement is to perform weight updates. It is also consistent across all datasets. In comparison, we see that AED and VAED are much slower due to ensuring Kmeans or GMM converged. Also, the class number and sample size also affect Kmeans and GMM and hence the computational time.

V. CONCLUSION

We have discussed about training an AE or VAE for clustering. One of the main problems is how to improve the multiclass representation in the latent space. A recent approach known as AED attempts to solve this problem, where D refers to the distance error function between AE and Kmeans in the latent space. We found several issues with the original AED. Firstly, AED suffers from the constraint of using points in the latent space as inputs. Secondly, AED cannot be optimized for both VAE and GMM since it does not treat variance as useful information. Lastly, when using the reparametrization trick for AED, the network gradient for VAE may suffer from the vanishing gradient problem. We proposed VAED to overcome all these problems of AED. In fact, AED is a specific case of VAED when assuming spherical Gaussian. We showed significant improvements using VAED over AED and VAE on the digit and scene recognition datasets as well as on par results or better results than recent published best methods on deep clustering networks.

References

[1] M. Abavisani and V. M. Patel. Deep sparse representation-based classification. *IEEE Signal Processing Letters*, 26(6):948–952, 2019. I

[2] A. Ali and F. Yangyu. Automatic modulation classification using deep learning based on sparse autoencoders with nonnegativity constraints. *IEEE Signal Processing Letters*, 24(11):1626–1630, 2017. I

[3] S. Amini and S. Ghaemmaghami. A new framework to train autoencoders through non-smooth regularization. *IEEE Transactions on Signal Processing*, 67(7):1860–1874, April 2019. I

[4] B. O. Ayinde and J. M. Zurada. Deep learning of constrained autoencoders for enhanced understanding of data. *IEEE transactions on neural networks and learning systems*, 29(9):3969–3979, 2017. I

[5] C. M. Bishop. *Pattern recognition and machine learning*. springer, 2006. I, II-A, II-A, III-C2

[6] D. Cai, X. He, and J. Han. Document clustering using locality preserving indexing. *IEEE Transactions on Knowledge and Data Engineering*, 17(12):1624–1637, December 2005. III-D

[7] J. Deng, X. Xu, Z. Zhang, S. Frühholz, and B. Schuller. Universum autoencoder-based domain adaptation for speech emotion recognition. *IEEE Signal Processing Letters*, 24(4):500–504, 2017. I

[8] T. Hastie, R. Tibshirani, and J. Friedman. The elements of statistical learning. springer series in statistics. In *:*. Springer, 2001. IV-A

[9] K. He, X. Zhang, S. Ren, and J. Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 770–778, 2016. IV-B

[10] S. Inoue, H. Kameoka, L. Li, S. Seki, and S. Makino. Joint separation and dereverberation of reverberant mixtures with multichannel variational autoencoder. In *ICASSP 2019-2019 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 96–100. IEEE, 2019. I

[11] E. Jang, S. Gu, and B. Poole. Categorical reparameterization with gumbel-softmax. *arXiv preprint arXiv:1611.01144*, 2016. I

[12] P. Ji, T. Zhang, H. Li, M. Salzmann, and I. Reid. Deep subspace clustering networks. In *Advances in Neural Information Processing Systems*, pages 24–33, 2017. I, I-A

[13] Q. Ji, Y. Sun, J. Gao, Y. Hu, and B. Yin. Nonlinear subspace clustering via adaptive graph regularized autoencoder. *IEEE Access*, 7:74122–74133, 2019. IV-A

[14] X. Jiang. Asymmetric principal component and discriminant analyses for pattern classification. *IEEE Transactions on Pattern Analysis & Machine Intelligence*, (5):931–937, 2008. I

[15] X. Jiang. Linear subspace learning-based dimensionality reduction. *IEEE Signal Processing Magazine*, 28(2):16–26, 2011. I

[16] Z. Jiang, Y. Zheng, H. Tan, B. Tang, and H. Zhou. Variational deep embedding: An unsupervised and generative approach to clustering. In *Proceedings of the 26th International Joint Conference on Artificial Intelligence*, IJCAI'17, pages 1965–1972. AAAI Press, 2017. I-A, IV-A

[17] H. Kameoka, T. Kaneko, K. Tanaka, and N. Hojo. Acvae-vc: Non-parallel voice conversion with auxiliary classifier variational autoencoder. *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, 2019. I

[18] E. Karamatli, A. T. Cemgil, and S. Kirbiz. Audio source separation using variational autoencoders and weak class supervision. *IEEE Signal Processing Letters*, pages 1–1, 2019. I

[19] D. P. Kingma and M. Welling. Stochastic gradient vb and the variational auto-encoder. In *Second International Conference on Learning Representations, ICLR*, 2014. I, I-A, II-C

[20] S. Lazebnik, C. Schmid, and J. Ponce. Beyond bags of features: Spatial pyramid matching for recognizing natural scene categories. In *Computer Vision and Pattern Recognition, 2006 IEEE Computer Society Conference on*, volume 2, pages 2169–2178. IEEE, 2006. IV-B

[21] Y. LeCun, L. Bottou, Y. Bengio, and P. Haffner. Gradient-based learning applied to document recognition. *Proceedings of the IEEE*, 86(11):2278–2324, 1998. IV-A

[22] S. Leglaive, U. Şimşekli, A. Liutkus, L. Girin, and R. Horaud. Speech enhancement with variational autoencoders and alpha-stable distributions. In *ICASSP 2019-2019 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 541–545. IEEE, 2019. I

[23] F. Li, H. Qiao, and B. Zhang. Discriminatively boosted image clustering with fully convolutional auto-encoders. *Pattern Recognition*, 83:161–173, 2018. I-A, IV-A, IV-B

[24] K.-L. Lim and H. Wang. Map approximation to the variational bayes gaussian mixture model and application. *Soft Computing*, pages 1–13, 2017. III-C2

[25] C. J. Maddison, A. Mnih, and Y. W. Teh. The concrete distribution: A continuous relaxation of discrete random variables. *arXiv preprint arXiv:1611.00712*, 2016. I

[26] Y. Netzer, T. Wang, A. Coates, A. Bissacco, B. Wu, and A. Y. Ng. Reading digits in natural images with unsupervised feature learning. 2011. IV-B

[27] S. Parthasarathy, V. Rozgic, M. Sun, and C. Wang. Improving emotion classification through variational inference of latent variables. In *ICASSP 2019-2019 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 7410–7414. IEEE, 2019. I

[28] X. Peng, J. Feng, S. Xiao, W.-Y. Yau, J. T. Zhou, and S. Yang. Structured autoencoders for subspace clustering. *IEEE Transactions on Image Processing*, 27(10):5076–5086, 2018. I-A

[29] A. Quattoni and A. Torralba. Recognizing indoor scenes. In *Computer Vision and Pattern Recognition, 2009. CVPR 2009. IEEE Conference on*, pages 413–420. IEEE, 2009. IV-B

[30] R. G. Soares. Effort estimation via text classification and autoencoders. In *2018 International Joint Conference on Neural Networks (IJCNN)*, pages 01–08. IEEE, 2018. I

[31] C. Song, F. Liu, Y. Huang, L. Wang, and T. Tan. Auto-encoder based data clustering. In *Iberoamerican Congress on Pattern Recognition*, pages 117–124. Springer, 2013. I, I-A, II-B, IV-A, IV-B

[32] B. Sun and H. Feng. Efficient compressed sensing for wireless neural recording: A deep learning approach. *IEEE Signal Processing Letters*, 24(6):863–867, 2017. I

[33] J. Sun, X. Wang, N. Xiong, and J. Shao. Learning sparse representation with variational auto-encoder for anomaly detection. *IEEE Access*, 6:33353–33361, 2018. I-A, III-A

[34] K. Tian, S. Zhou, and J. Guan. Deepcluster: A general clustering framework based on deep learning. In *Joint European Conference on Machine Learning and Knowledge Discovery in Databases*, pages 809–825. Springer, 2017. I-A, IV-A, IV-B

[35] J. Xie, R. Girshick, and A. Farhadi. Unsupervised deep embedding for clustering analysis. In *International conference on machine learning*, pages 478–487, 2016. I-A, IV-A, IV-B

[36] W. Xu and Y. Tan. Semisupervised text classification by variational autoencoder. *IEEE transactions on neural networks and learning systems*, 2019. I

[37] B. Yang, X. Fu, N. D. Sidiropoulos, and M. Hong. Towards k-means-friendly spaces: Simultaneous deep learning and clustering. In *Proceedings of the 34th International Conference on Machine Learning-Volume 70*, pages 3861–3870. JMLR. org, 2017. I, I-A, IV-A

[38] J. Yang, J. Liang, K. Wang, P. Rosin, and M.-H. Yang. Subspace clustering via good neighbors. *IEEE transactions on pattern analysis and machine intelligence*, 2019. IV-A

[39] T. Yu, C. Guo, L. Wang, S. Xiang, and C. Pan. Self-paced autoencoder. *IEEE Signal Processing Letters*, 25(7):1054–1058, 2018. I

[40] Q. Zhang and J. H. Hansen. Language/dialect recognition based on unsupervised deep learning. *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, 26(5):873–882, 2018. I