IEEE Access
Multidisciplinary | Rapid Review | Open Access Journal

Received July 19, 2020, accepted August 16, 2020, date of publication August 20, 2020, date of current version September 1, 2020.

Digital Object Identifier 10.1109/ACCESS.2020.3018151

Variations in Variational Autoencoders - A Comparative Evaluation

RUOQI WEI, CESAR GARCIA, AHMED EL-SAYED, (Member, IEEE), VIYALETA PETERSON, (Member, IEEE), AND AUSIF MAHMOOD, (Member, IEEE)
Department of Computer Science and Engineering, University of Bridgeport, Bridgeport, CT 06604, USA
Corresponding author: Ausif Mahmood (mahmood@bridgeport.edu)

ABSTRACT Variational Auto-Encoders (VAEs) are deep latent space generative models which have been immensely successful in many applications such as image generation, image captioning, protein design, mutation prediction, and language models among others. The fundamental idea in VAEs is to learn the distribution of data in such a way that new meaningful data can be generated from the encoded distribution. This concept has led to tremendous research and variations in the design of VAEs in the last few years creating a field of its own, referred to as unsupervised representation learning. This paper provides a much-needed comprehensive evaluation of the variations of the VAEs based on their end goals and resulting architectures. It further provides intuition as well as mathematical formulation and quantitative results of each popular variation, presents a concise comparison of these variations, and concludes with challenges and future opportunities for research in VAEs.

INDEX TERMS Deep learning, variational autoencoders (VAEs), data representation, generative models, unsupervised learning, representation learning, latent space.

I. INTRODUCTION

Data generation, due to the scarcity of training data, is a fundamental problem in many areas of artificial intelligence such as computer vision pattern recognition and natural language processing [1]. In recent years, deep generative models have gained a lot of attention due to numerous applications in deep learning. Among them, VAEs [2] and Generative Adversarial Networks (GANs) [3] are regarded as the two most popular approaches to generative modeling.

The VAE can be regarded as a mixture of an encoder and a decoder Bayesian network. The encoder maps an input data (e.g., an image) x to a latent vector z, and then the decoder maps the latent vector z back to image or data space [4]. VAEs[1] enhance a normal Autoencoder (AE) by adding a Bayesian component that learns the parameters representing the probability distribution of the data. This is achieved by imposing a prior on the probability of the input, modeled typically as a unit Gaussian random variable. This implicitly results in a regularization that can be used to explain the probability of the input. Thus, the VAE is a generative model that can sample from the latent distribution produced by the encoder and generate new input data via the decoder.

VAEs do not suffer problems encountered in GANs, mainly: non-convergence causing mode collapse, and are hard to evaluate [3], [5], [6]. What's more, VAEs have decent theoretical guarantee: first, by introducing the variational lower bound, the complicated calculation of the marginal likelihood probability is avoided. Second, by the reparameterization trick, the complicated Markov chain sampling process of latent variable is avoided. A key benefit of VAEs is the ability to control the distribution of the latent representation vector z, which can combine VAEs with representation learning to further improve the downstream tasks. VAEs are able to learn the smooth latent representations of the input data [7] and thus can generate new meaningful samples in an unsupervised manner. These properties have allowed VAEs to enjoy success especially in computer vision, e.g., static images generation [8], zero shot learning [9]–[11], image super-resolution [12], [13], and semantic image inpainting [14], [15].

Despite the above-mentioned advantages of VAEs, they do have some constraints: 1) the generated images tend to be blurry, 2) latent representation does not have an interpretable meaning, 3) the popularly used Gaussian distribution

The associate editor coordinating the review of this manuscript and approving it for publication was Feng Shao.

[1]https://github.com/VAEs-Tutorial/paper

as priori has limitations because the learnt representations are unimodal, and do not allow for different or mixed data distributions, and 4) the Gaussian definition is based on the L_2-norm that suffers from the curse of dimensionality. In order to solve the above problems, researchers have proposed many variations of the VAEs based on different task requirements such as feature learning and deep clustering with the goal of greatly improving the quality of the generated data.

Current VAE research focuses primarily in three directions: 1) improving the disentanglement for VAEs, 2) applying custom VAEs to real-world applications, and 3) improving the quality of generated images. Many VAE-variants have been proposed in the following categories: 1) architecture-variant, such as VAE-GAN and CVAE, 2) regularizing posterior-variant, posterior regularization to improve disentanglement capability, 3) prior-variant, prior-variance based on data distributions to improve the Bayesian VAE model. In the following sections, we provide details on the VAE-variants implemented with the above categorizations.

In this paper, we focus on the recent advances in VAEs as these provide an elegant statistical approach to meaningful data generation resulting in an entire field of its own referred to as unsupervised representation learning. We study the existing VAE-variants and provide a comprehensive analysis and comparisons between different approaches. The rest of the paper is organized as follows:

–We present an overview of the conventional VAE.

–Variations of the VAE are described mathematically along with their differences, pros and cons.

–We conduct experiments on MNIST dataset and perform comparative analysis.

–We conclude this review with some future directions for advancement in this area.

The structure of our paper is organized as follow: Section II describes some background work about VAEs. Section III explains variants of VAEs in detail. Section IV provides comparative analysis of experimental results and analysis on the MNIST dataset. Section V describes summary of Variations of the VAE along with their differences, pros and cons. Conclusion and future work is given in Section VI and references are delineated at the end.

II. PRELIMINARIES

The following sub-sections introduce the theory behind Autoencoders, deep generative models, and conventional VAE. Additionally, we discuss the variational bound and the reparameterization trick.

1) AUTOENCODERS

An Autoencoder (AE) is an unsupervised learning system where during training the expected output is an approximation of the input. AE is primarily applied to data dimensionality reduction, image classification, object detection, and image denoising [8], [16]–[18]. An AE consists of the following parts [19]:

2) ENCODER

A neural network that produces a compressed latent space representation of input data.

3) LATENT SPACE

Captures input to a knowledge representation, that is, to reduce the dimensionality of input such that maximum information is preserved in it.

4) DECODER

A reconstruction of the input data from the compressed latent space.

As shown in Figure 1, the encoder h encodes the original input X into a latent space Z. The decoder f decodes the latent space Z to recreate an approximation of the original data X' such that $X' = f(Z) = f(h(X))$. After repeated training, the AE attempts to reproduce a copy of the input as the output. The application of an AE has two main aspects, the first is data denoising, and the second is dimensionality reduction for removing redundant or unimportant features. In other words, the output is made approximately equal to the input with some constraints on the AE algorithm. These constraints force the encoder to consider which parts of the input need to be preserved and which parts can be discarded. Therefore, the Autoencoder can often learn the meaningful features of the data and discard the irrelevant features. It is well known that an AE accomplishes dimensionality reduction similar to a nonlinear PCA. In classification application of an AE, the decoder section is removed after AE is trained, and replaced by a classifier network. An AE is not capable of generating new data as the latent space it produces is not regularized to aid in new data synthesis.

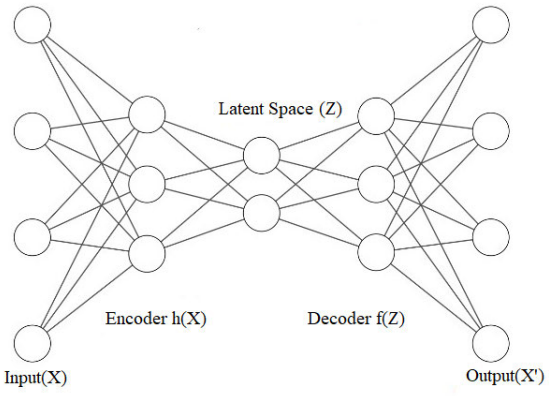

FIGURE 1. Architecture of Autoencoder.

A. DEEP GENERATIVE MODELS

The most common model in machine learning is the discriminant model. The discriminant model [20] refers to the inference of certain features of the data based on the original dataset, and then use these features to construct the corresponding application model e.g., a classifier. On the other

hand, a generative model aims to learn the features of the input and recover the original data or generate similar data from a latent space distribution.

Deep Generative models use distribution estimation and sampling to achieve generation of new data [21]. To explain this further, suppose in a continuous or discrete high-dimensional space, there is a data x obeying some unknown distribution $P_{\text{data}}(x)$, and it is necessary to estimate the unknown distribution $P_{\text{model}}(x)$ by observing part of the data samples of the set X. The deep generative model generates an estimated distribution by approximating and learning the unknown distribution $P_{\text{data}}(x)$ from some training data and allows new data to be generated from the estimated distribution $P_{\text{model}}(x)$.

Traditional popular deep generative models belong to Boltzmann family i.e. Deep Belief Networks (DBNs) [22] and Deep Boltzmann Machines (DBMs) [23]. However, one major limitation of them is high computational cost during inference process [24]. Latest deep generative networks are VAEs and GANs. In this paper, we focus on VAEs and its variants.

B. VARIATIONAL AUTOENCODER (VAE)

A Variational Autoencoder (VAE) is a special autoencoder based on the variational Bayes inference originally proposed by Kingma and Welling [2], Doersch [4]. The goal of a VAEs is to be able to learn the distribution of the training data so that by sampling from it, we can generate new data. Since the training data may not necessarily have a well-defined mathematical distribution, we force the distribution of the output of the encoder (known as the latent space) to follow a known distribution e.g., normal distribution. Figure 2 shows the architecture of a VAE that has an encoder and a variational inference network, followed by the decoder that samples from the latent space to generate the output. The main difference between AE and VAE is the AE learns the compressed representation of the input, and its decompression to match the given input. In contrast, the VAE is a Bayesian model which learns the compressed representation of the AE, and constructs the parameters representing the probability distribution of the data. It can sample from this distribution and generate new input data samples. Therefore, VAE is a generative model, where as an AE which just does reconstruction does not have an obvious generative interpretation.

If the original dataset is $X = \{x_i\}_{i=1}^{N}$, then each data sample x_i is a randomly generated, independent, continuous or discrete distribution variable, and the regenerated dataset at the output is $X' = \{x_i'\}_{i=1}^{N}$. Suppose the encoding process produces a latent variable z, then, the observable variable X' is a random vector in a high-dimensional space, and the unobservable variable Z is a random vector in a relatively low-dimensional space.

In the implementation of the VAE, the encoder is a neural network whose input is a datapoint x, its output is a latent

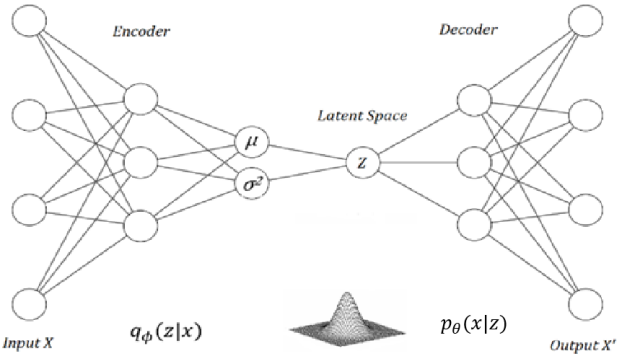

FIGURE 2. Architecture of Variational Autoencoder (VAE).

representation z. We represent its weights and biases as a model ϕ. The decoder is another neural net whose input is the latent representation z and outputs the parameters of the probability distribution for the data. The decoder's weights and biases are represented as the model θ. Suppose we want to approximate a distribution $p(Z|X)$ with some $q(Z|X)$ distribution via the Kullback-Leibler (KL) divergence, then by definition of KL,

$$D_{KL}\left[q(Z|X)\,\|p(Z|X)\right] = \sum_{Z} q(Z|X)\log\left[\frac{q(Z|X)}{p(Z|X)}\right] \tag{1}$$

$$= \mathrm{E}\left[\log\left[\frac{q(Z|X)}{p(Z|X)}\right]\right] = \mathrm{E}\left[\log\left[q(Z|X) - p(Z|X)\right]\right] \tag{2}$$

If we minimize the KL divergence as follows: using Bayes' rule:

$$p(Z|X) = \frac{p(X|Z)\,p(z)}{p(x)} \tag{3}$$

$$D_{KL}\left[q(Z|X)\,\|p(Z|X)\right] = \mathrm{E}\left[\log q(Z|X) - \log\frac{p(X|Z)\,p(Z)}{p(X)}\right] \tag{4}$$

$$= \mathrm{E}\left[\log q(Z|X) - (\log p(X|Z) - \log p(Z) + \log p(X))\right] \tag{5}$$

Since expectation is with respect to Z,

$$= \log p(X) + \mathrm{E}\left[\log q(Z|X) - \log p(X|Z) - \log p(Z)\right] \tag{6}$$

$$\log p(X) - D_{KL}\left[q(Z|X)\,\|p(Z|X)\right] = E[\log p(X|Z) - D_{KL}\left[q(Z|X)\,\|p(Z)\right] \tag{7}$$

$$D_{KL}\left[q(Z|X)\,\|p(Z|X)\right] = \log p(X) - \left[\mathrm{E}[\log p(X|Z)] - D_{KL}\left[q(Z|X)\,\|p(Z)\right]\right] \tag{8}$$

Since D_{KL} is always positive, we can conclude that:

$$\log p(X) \geq E[\log p(X|Z)] - D_{KL}\left[q(Z|X)\,\|p(Z)\right] \tag{9}$$

Equation 9 is an important result and is known as the Evidence Lower Bound (ELBO). In a deep neural network implementation of a VAE, equation 9 is used as the loss

function during training of the network. The E[$\log p(X|Z)$] term denotes the reconstruction i.e., the generation of output from the latent representation z. The $D_{KL}[q(Z|X) \| p(Z)]$ measures the similarity of the distribution of the latent space with the target distribution $p(z)$. Thus, the two components of equation 9 try to make the output similar to the input while keeping the distribution of the latent space as close to the target distribution $p(z)$ as possible.

The ELBO is tight if $q(z) = p(z|x)$, indicating that $q(z)$ is optimized to approximate the true posterior. For scalability to larger datasets, we do not optimize $q(z)$ for every data point X. Instead an inference network $q(z|x)$ is introduced that is parameterized by a neural network that outputs a probability distribution for each data point X. Therefore, the final objective is to maximize:

$$\mathcal{L}(\theta, \phi) = E_{q_\phi(z|x)}\left[\log p_\theta(x|z)\right] - D_{KL}(q_\phi(z|x)||p_\theta(z)) \quad (10)$$

1) REPARAMETERIZATION TRICK

According to the objective described in equation (10), after we introduced $q_\phi(z|x)$ to approximate $p_\theta(z|x)$, if we want to sample Z from $q_\phi(z|x)$, an easy choice is to assume that $q_\phi(z|x)$ obeys the Gaussian distribution and that the sampling of Z can be done in the following reparameterization way [25]:

$$z^i = \mu^i + \sigma^i * \epsilon^i$$

where ϵ is an auxiliary noise variable such that $\epsilon \sim N(0, 1)$ i.e., let $q(z|x)$ be a Gaussian with parameters $\mu(x)$ and $\Sigma(x)$.

Then the KL divergence between $q(z|x)$ and $p(z)$ can be computed in closed form as follows:

$$D_{KL}[N(\mu(x), \Sigma(x)) \| N(0, 1)] = \frac{1}{2}\left(tr\Sigma(x) + \mu(x)^T\mu(x) - k - \log[\det(\Sigma(x))]\right) \quad (11)$$

$$D_{KL}[N(\mu(x), \Sigma(x)) \| N(0, 1)] = \frac{1}{2}\left(\sum_k \Sigma(x) + \sum_k \mu^2(x) - \sum_k 1 - \log\prod_k \Sigma(x)\right) \quad (12)$$

$$= \frac{1}{2}\left(\sum_k \Sigma(x) + \sum_k \mu^2(x) - \sum_k 1 - \sum_k \log\Sigma(x)\right) \quad (13)$$

$$= \frac{1}{2}\sum_k\left(\Sigma(x) + \mu^2(x) - 1 - \log\Sigma(x)\right) \quad (14)$$

Replacing $\Sigma(x)$ with $e^{\Sigma(x)}$

$$D_{KL}[N(\mu(x), \Sigma(x)) \| N(0, 1)] = \frac{1}{2}\sum_k\left((\exp\Sigma(x)) + \mu^2(x) - 1 - \log\Sigma(x)\right) \quad (15)$$

The reparameterization can make the relationship between latent variable Z, σ and μ change from sampling to a numerical calculation such that it can be optimized directly by using stochastic gradient descent [26]. The main purpose of the reparameterization trick is to make back propagation possible. Conditional distribution $p_\theta(z|x)$ obeys Gaussian distribution and the mean and standard deviation can be calculated by the neural network; thus, each component of the lower bound of the variation can be directly calculated, and the model structure can be determined.

2) DISENTANGLEMENT AND REPRESENTATION LEARNING

Although our world is inundated with data, a large part of the data is still unlabeled and unorganized. One of the challenges of artificial intelligence is to learn useful representations using unsupervised learning methods. The performance of models can be improved by selecting different representations to adjust the difficulty of machine learning [27]. Feature engineering [28] is one of the methods that can refine the representations from raw data. Feature engineering refers to transforming raw data into advanced training data representations. However, in machine learning, manually selected features rely on human and professional knowledge, which is part of the most time-consuming and energy-intensive work, and its weakness is the inability to extract and organize discriminant information from the data. Therefore, in order to improve the scope and ease of use progress in artificial intelligence, we need to promote the work of feature engineering more quickly and effectively by relying less on feature engineering. Representation learning can learn useful disentangled representations automatically.

Representation learning is done by the meta-priors proposed by Bengio *et al.* [7]. The goal of representation learning is to be useful for downstream tasks. At present, research on successful representation learning includes speech recognition [29], signal processing [30], object recognition [31], and natural language processing [32]. The most important meta-prior is called "disentanglement" which is an unsupervised learning technique that breaks down, or disentangles, each feature into narrowly defined variables and encodes them as separate dimensions [7]. Assuming that the data is generated from independent factors of variation, and if the VAE is trained to reconstruct the sample well, then the latent space between the encoder and decoder keeps the important information of the original data.

Intuitively, a factorial code disentangles the individual elements that were originally mixed in the sample, just as humans recognize complex things by disentangling independent elements. If the dimensions of the latent vector are independent of each other, it is factorial disentangled, i.e., a good representation.

III. VARIATIONS OF VAEs

A. InfoVAE

Regularization of the encoding distribution is often used to encourage disentanglement representations of the latent variables z. The fundamental approach taken in recent research on disentanglement is to augment the VAE loss with regularizers, such as reweighting the ELBO. InfoVAE [33], also known as MMD-VAE, is a variant of VAEs that can lead to improved unsupervised representation learning based on regularization of the largest mean difference between distributions. The goal

of InfoVAE is to do the representation learning by encouraging a large mutual information between Z and X by adding a regularizer of maximum mean discrepancy. The maximum mean discrepancy (MMD) [34] was first proposed for the two-sample test problem to determine if the two distributions p and q are the same. Its basic assumption is to define unspecified function classes F to measure the disparity between p and q. If enough samples generated by the two distributions have equal mean on F, then the two distributions are similar.

The MMD is taken as a test statistic to determine whether the two distributions are similar. Such MMD-based regularization can lead to disentangled latent representation resulting in the following modified form of objective function [35]:

$$\mathcal{L}_{\text{VAE}}(\theta,\phi)+\lambda_1\mathbb{E}_{\text{p(x)}}\left[R_1\left(q_\phi\left(z|x\right)\right)\right]+\lambda_2 R_2\left(q_\phi\left(z\right)\right) \quad (16)$$

where $R1$ and $R2$ are regularizes and $\lambda_1, \lambda_2 > 0$ the corresponding hyperparameter weights. The MMD-VAE starts from an alternative way of writing $\mathcal{L}_{\text{VAE}}$:

$$\begin{aligned}&\mathcal{L}_{\text{VAE}}(\theta,\phi)\\&=D_{KL}\left(q_\phi(z)\|p(z)\right)+\mathbb{E}_{p(x)}\left[D_{KL}\left(q_\phi(x\mid z)\|p_\theta(x\mid z)\right)\right]\end{aligned} \quad (17)$$

Zhao *et al.* [33] suggest to boost a major mutual message between $z{\sim}q\left(x|z\right)$ and x by putting in a regularizer $I_{q_\phi(x,z)}$ to the formula above and reweight the first term, resulting in the final objective as:

$$\begin{aligned}&\mathcal{L}_{\text{Info VAE}}(\theta,\phi)\\&\quad=\mathcal{L}_{\text{VAE}}(\theta,\phi)+\lambda_1\mathbb{E}_{p(x)}\left[D_{KL}\left(q_\phi(z\mid x)\|p(z)\right)\right]\\&\quad\quad+\lambda_2 D_{KL}\left[q_\phi(z)\|p_\theta(z)\right]\end{aligned} \quad (18)$$

Figure 3 shows the architecture of the convolutional neural network (CNN) VAE model for the MNIST dataset which has been utilized for MMD-VAE. This structure is based on Deep Convolutional networks which includes fully connected layers (FC) and convolutional (Conv) layers. The size of the input image of the encoder neural network is $28 \times 28 \times 1$, and the input image passes through two Conv layers and the last FC layer till the latent variable space is reached. The two convolutional layers in the encoder network achieve feature maps dimensionality reduction using stride of 2 and a kernel size of 4×4. The two parallel feature vectors obtained by flattening the feature map of the second convolutional layer are μ and σ^2, respectively. For a general implementation, the number of neurons in the fully connected layer is a model decision and represents the dimension of the latent space.

The generative model $p\left(x|z\right)$ takes the sampled latent variables z received by μ and σ^2 and using the reparameterization trick feeds it through two FC layers and one Conv layer until a reconstructed output is obtained. The FC layers in the decoder reshape the latent variable z to $7 \times 7 \times 128$, and finally use a stride of 2 and a kernel size of 4×4 in the deconvolution to obtain the reconstruction image.

Disentanglement quality of inference models is typically evaluated based on the ground truth factors of variation

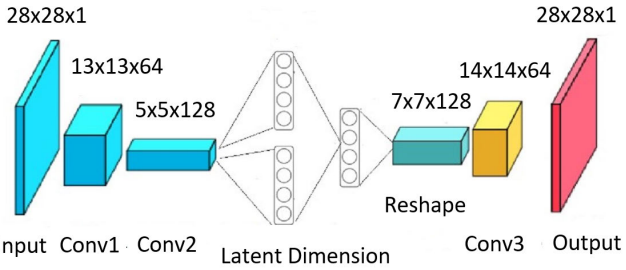

FIGURE 3. Architecture of InfoVAE for MNIST dataset.

FIGURE 4. Samples generated by InfoVAE.

(if available). Specifically, disentanglement metrics measure how predictive the individual latent factors are for the ground-truth factors [36]. By comparing different models on metrics of performance, stability and training speed, and evaluating and comparing possible types of divergences, InfoVAE with MMD regularization had better performance metrics and demonstrated stability over traditional VAE [33]. InfoVAE-MMD provides a good way to handle latent code ignorance issues [33]. However, some of the drawbacks include the often-blurred image generation, as shown in Figure 4, samples generated by InfoVAE of MINST data.

B. β-VAE

Another unsupervised method that can automatically discover disentangled factors in latent variable space based on VAE framework is β-VAE [36]. The basic principle of β-VAE is to reweight the ELBO of the model with additional parameter β as the D_{KL} weight. The ELBO can be expressed as:

$$\mathcal{L}(\theta,\phi,\beta)=\mathbb{E}_{q_\phi(z|x)}\left[\log p_\theta\left(x|z\right)\right]-\beta D_{KL}(q_\phi(z|x)||p_\theta(z)) \quad (19)$$

This constraint limits the ability of latent information channels and emphasize learning the statistically independent

latent factors. Combining the maximum likelihood objective function with the generated model, allows the model to obtain the most useful latent features of the input data. If the data was generated by some independent dimension of variation, it will be disentangled.

Compared to the unmodified VAE framework, this easy revise permits β-VAE to remarkably enhance the performance of disentanglement in learning representation [35]:

$$\mathcal{L}_{\beta-\mathrm{VAE}}(\theta,\phi) = \mathcal{L}_{\mathrm{VAE}}(\theta,\phi) + \lambda_1 \mathbb{E}_{p(x)}\left[D_{KL}\left(q_\phi(z\mid x)||p(z)\right)\right] \quad (20)$$

where $\lambda_1 = \beta - 1 > 1$ is the corresponding weight.

This regularization causes $q_\phi(z|x)$ to better match the a priori $p(z)$ which conversely restricts the implicit capacity of the latent feature z$\sim$ $q_\phi(z|x)$ and causes it to be disentangled. Note that the β-VAE with $\beta = 1$ is equivalent to a standard VAE. β-VAE implements the representation of disentanglement by selecting the appropriate hyperparameter β. This simple penalty has proven to be able to obtain models with a high degree of disentanglement. However, it is not explicitly stated why using the factor a priori penalty on $KL(q(z|x)||p(z))$ helps in encoding latent variables with a disentangled representation of the data. Recently, the authors in [37] found that ELBO has a decomposition that can be used to explain the success of β-VAE in learning to solve disentangled representations. Specifically, the total correlation (TC) penalty in the loss function encourages the model to find statistically independent factors in the data distribution. In information theory, TC is a kind of generalizations of mutual message and is the amount of information shared between variables in the collection. It is also referred to as multiple message or multivariate constraint. TC quantifies the dependency or redundancy between a group of stochastic variables. In β-VAE, the penalty of TC forces the model to find statistically independent factors in the data distribution. This leads to the learning of latent variables that exhibit a disentangled transformation of all data samples, and thus the existence of the term is the reason for the success of β-VAE.

The β-VAE has a relation to Info-VAE because the Info-VAE family generalizes β-VAEs [33]. β-VAE can be transformed from INFO-VAE by setting λ_2 in equation (18) to 0. The disadvantage of β-VAE over previous INFO-VAE is that the β-VAE model cannot effectively penalize the weights and information preferences of X and Z, resulting in under-fitting or ignoring the latent variables. Specifically, for each λ, INFO-VAE can choose a unique value. If we choose a larger value of $\lambda \geq 1$ to balance the importance of the observation space and the latent space X and Z,we must also choose $\alpha \leq 0$, which forces the model to penalize mutual information, thus avoiding under-fitting or ignoring the latent variables.

After the latent variable generation factor is known and disentangled, the indicator for evaluating the disentangled performance requires a supervised classifier-based evaluation metric. Overall, β-VAE tends to find more latent factors consistently and learns more clearly the characterization of disentanglement (as shown in Figure 5) [38]. In addition, β-VAE does not require a hypothesis of the distribution of the data, and the training procedure is very steady.

FIGURE 5. Latent features learnt by β-VAE on MNIST Dataset.

C. VQ-VAE

In machine learning, in addition to learning based on continuous features [22], [39]–[41], there is also learning based on discrete representations [23], [42]–[44]. Discrete representations are naturally suitable for complex reasoning, planning, and predictive learning. Although the use of discrete latent variables in deep learning has proven challenging, powerful autoregressive models have been developed for modeling distributions on discrete variables [45].

The main purpose of VQ-VAE is to learn discrete latent variables. VQ-VAE is implemented using a vector quantization (VQ) algorithm. We know that quantization can divided into scalar quantization and vector quantization (VQ). Scalar quantization samples signal values and quantizes them one by one. Vector quantization divides several sampled signals into a group, thus simplifying the amount of data. Specifically, for each latent variable, we look for points within a certain range around it to represent it, so that we can treat the latent variables as a k-dimensional vector. Vector quantization is an extremely important method of signal compression, which is widely used in speech coding, speech recognition and synthesis, image compression and other fields.

In VQ-VAE, each latent embedding vector e_i is a vector in a d-dimension latent space, and the size of the discrete latent space of k such vectors are learnt, together with the rest of the model parameters (as shown in Figure 6).

The posterior $q_\phi(z|x)$ is implemented as one hot vectors $\{e_k\}_{k=1}^k$:

$$q_\phi\left(z_j = e_k|x\right) = \begin{cases} 1 & if\ k = argmin_j - ||z_e(x) - e_j||_2 \\ 0 & otherwise. \end{cases} \quad (21)$$

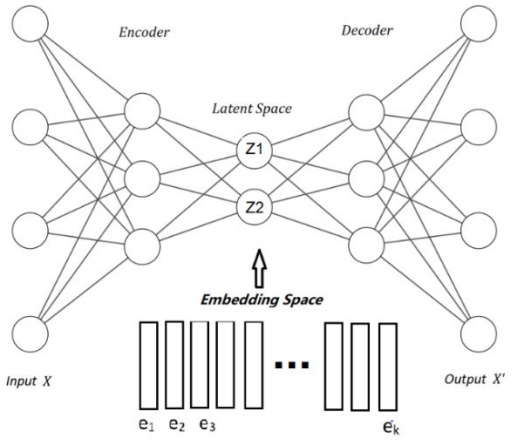

FIGURE 6. Architecture of VQ-VAE.

FIGURE 7. Left: MNIST original images, right: Reconstructions from a VQ-VAE.

where $z_e(x)$ is the output of the encoder, the embeddings e_k can be learned individually for each latent variable z_j.

The principle of the VQ-VAE sampling procedure is based on autoregressive distribution[45]. In the autoregressive model, the target variable is predicted based on a combination of historical data of the target variables. After training, the autoregressive distribution is fitted over z, $p(z)$ to generate X by an ancestral sampling.

VQ-VAE can achieve good reconstructions [45] (Figure 7) as compared to conventional VAEs. In addition, the image contains a lot of redundant information, because most pixels are correlated and noisy, so a pixel-level learning model can be wasteful. When applied to training language data, VQ-VAE learns the basic phoneme-grade speech model in a fully unsupervised manner for controlled speech generation and phoneme classification[46].

Structures with discrete latent variables are greatly reduced by discrete coding, and reconstructions appear to be slightly blurry compared to the original input. However, in some well-trained VQ-VAEs (i.e. high-entropy), parts of the codebook may be lost. The model will suffer from codebook crashes and will no longer use the full capacity of the discrete bottlenecks, leading to worse likelihoods and poor reconstruction. The reason for this phenomenon is not clear, it can be noted that the K-means and Gaussian Mixture model algorithms may have similar problems [47].

D. CLUSTERING VAE

Cluster analysis is an unsupervised learning method which aims to learn training samples without classification markers and to reveal the intrinsic properties and laws of the data. Mathematical methods are used to study and deal with the classification of given objects and the degree of closeness between the categories. Specifically, cluster analysis divides the data set into several subsets, and the elements in each subset have higher similarity to the elements in the subset under certain metrics. The subsets that are divided in this way are ''clustered'', each of which represents a potential category. The distinction between classification and clustering is that classification is to first determine the category and then divide the data; clustering is to first divide the data and then determine the category.

From a machine learning perspective, cluster analysis is an unsupervised learning method where the classes are not given in advance but are created according to the similarity and distance of the data [48]–[50]. The structure of the clusters is not presupposed, but the number of clusters can be proposed. The purpose of the clustering algorithm is to find potential natural grouping structures and relationships of interest in the data. Clustering has been widely used in various fields of engineering and science. In general, the clustering method is mainly the measurement of data's groups based on similarity or dissimilarity [51], which can be divided into direct and indirect methods. The direct method is based on the similarity clustering of the original input, mainly by measuring a certain metric between the samples to achieve clustering. The indirect method applies the metrics on the features generated from the original data.

Recently, deep clustering has become one of the popular approaches to achieving good learning representations. Deep embedded clustering (DEC) [52] among others have been proposed to make deep clustering a popular research field. Deep embedded clustering (DEC) uses deep neural networks to learn the representations, and then uses clustering algorithms to perform cluster analysis on the generated features. The data is usually mapped to the representation space and then fed straight into the clustering model. In order to generate meaningful data samples, the generative models need to have two purposes: one is to seize the statistical architecture of the data, and the other is to generate data samples. DEC acts fine in clustering, however, in some cases it poorly models the generative procedure of data, so it does not generate good quality samples. Therefore, there is a need is to develop a better deep clustering model which: 1) learns to capture a good representation of the statistical structure of the data, and 2) is able to generate samples.

E. VARIATIONAL DEEP EMBEDDING (VaDE)

The Variational Deep Embedding (VaDE) [53] is one of the techniques utilized for both data clustering and generation. It is an extension of the variational autoencoder that applies the Gaussian Mixture Model (GMM) [54] on the latent

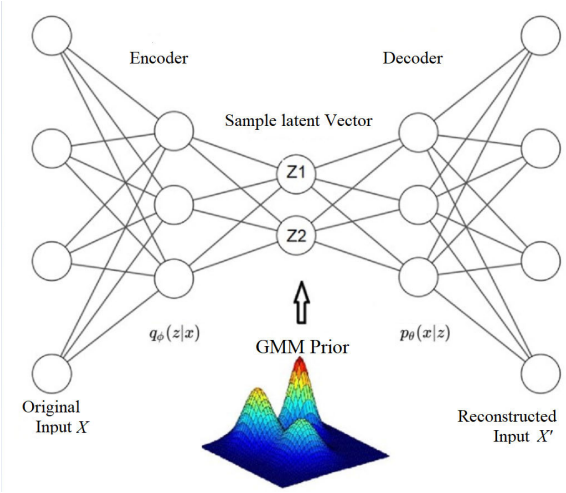

FIGURE 8. Architecture of VAE-based GMM Deep Clustering Model.

variables for clustering purpose(as show in Figure 8). The GMM defines the probability density function as multiple Gaussian density weighted sums. One of the most common ways to estimate GMM parameters is Maximum Likelihood Estimation or Expectation Maximization (EM) [55]. The benefit of GMM is that it can generate samples by estimating the data density.

The generative model for VaDE can be formulated as [56]:

$$p(x, z, c) = p(x|z)\, p(z|c)\, p(c)\,,$$

where

$$\begin{aligned} c &\sim Cat\left(\frac{1}{K}\right) \\ \mathcal{N}z &\sim (\mu_c, \sigma_c^2 I) \\ \mathcal{N}x &\sim (\mu_x(z), \sigma_x^2(z)I), orBer(\mu_x(z)) \end{aligned} \tag{22}$$

where $c\epsilon\ [1, K]$ is the distribution of the weights of the Gaussian terms in the GMM (parametrized by π), and K is the number of classes which are predetermined, μ and σ^2 are parameters of the elements in the clusters. $Ber\left(x|\mu_x\right)$ and $\mathcal{N}(x|\mu_x, \sigma_x^2 I)$ are multivariate Bernoulli distribution and Gaussian distribution parameterized by μ, and σ^2. The encoder model can be stated as:

$$q(z, c|x) = q(z|x)\, q(c|x) \tag{23}$$

VaDE maximizes the evidence lower bound (ELBO) using Jensen's inequality:

$$\begin{aligned} &\log p(x) \\ &= \log \int_z \sum_c p(x, z, c)\, dz \\ &\geq \mathrm{E}_q(z, c|x)\left[\log \frac{p(x, z, c)}{q(z, c|x)}\right] = L_{ELBO}(x) \\ &= \mathrm{E}_{q(z,c|x)}[\log(p(x|z))] - D_{KL}(q(z, c|x)||p(z, c)) \end{aligned} \tag{24}$$

where $q(z, c|x)$ is the group member probability of observed variable x to class c. The first term in L_{ELBO} is the reconstruction loss L_n, and the second term is the clustering loss L_c, which is the Kullback-Leibler divergence between the distribution of the observed sample and the Mixture of Gaussian (MoG) prior. After training, the class can be inferred from the MoG latent space.

VaDE is an unsupervised clustering model. The number of clusters of a VaDE can be set to the number of classes in each dataset, or a different number of clusters K can be selected. If K is less than the total number of classes in the dataset, numbers with similar appearances will be grouped together. On the other hand, if K is greater than the number of classes, some numbers with the same appearance will be divided into subclasses.

Samples of generated digits from MNIST dataset is shown in Figure 9.

FIGURE 9. The digits generated by VaDE.

F. GAUSSIAN MIXTURE VAE (GMVAE)

Although VaDE is simple and performs GMM on the latent space for clustering, it cannot be considered as a real GMM for data generation due to having independent gaussian distributions as the prior. However, because the best choice of prior distribution is the one with the ability to describe the distribution of clustering latent structures, the authors in [57] proposed the prior distribution $p(z)$ to be a GMM that depends on another two latent spaces w, and c. This approach has advantages over the regular VAE as the GMM can capture the clustering representations of data that are not necessarily unimodal.

GMVAE uses Gaussian mixture model as a priori for latent encoding space and defines a generative procedure that formulates a variational Bayes optimization objective (as shown in Figure 10). It supposes that the sample is generated by a Gaussian mixture and can infer the class of data points from the latent variable spaces. After optimizing the ELBO, the learned GMM model can infer the cluster allocation from the latent spaces.

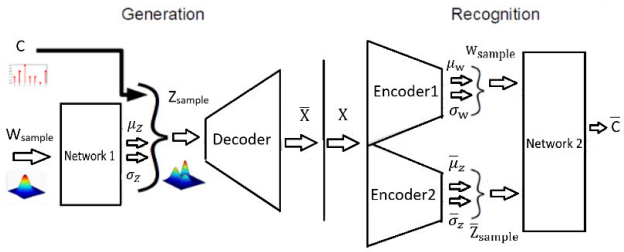

FIGURE 10. Architecture of VAE-based GMM Deep Clustering Model.

This GMVAE algorithms can cluster the given data, and generate images, but because of the overhead of the extra latent variables, it typically has high computational complexity than other deep clustering techniques. The generative model for the GMVAE can be expressed as [56]:

$$p(x, z, w, c) = p(x|z)\, p(z|c, w)\, p(w)\, p(c)\,,$$

where

$$\begin{aligned} c &\sim Cat\left(\frac{1}{K}\right) \\ w &\sim \mathcal{N}(0, I) \\ z &\sim \mathcal{N}\left(\mu_c(w), \sigma_c^2(w)\right) \\ x &\sim \mathcal{N}((\mu_x(z), \sigma_x^2(z)I) or Ber(\mu_x(z)) \end{aligned} \tag{25}$$

where, K is the number of clusters, w is the regular latent variable, c is the label latent variable, z is the GMM latent variable, and x is the generated data.

For the recognition (cluster inference) step, the trained networks (Encoder 1 (*E1*), Encoder 2 (*E2*), and Network 2 (*N2*)) are working to approximate the posterior distribution $q(z, w, c|x)$ which can be factories for each network parameters as:

$$q(z, w, c|x) = \prod_i q_{E1}(w_i|x_i) q_{E2}(z_i|x_i)\, p_{N2}(c_i|w_i, z_i) \tag{26}$$

where i is the index for training data, E_1 produces latent space w, E_2 generates latent space z, and N_2 is the classification network as shown in Figure 10. The training loss for the GMVAE can be expressed as:

$$\begin{aligned} &\mathcal{L}(N_1, D, E_1, E_2, N_2) \\ &= \mathbb{E}_{q(z|x)}[log\, p_D(x|z)] - D_{KL}(q_{E1}(w|x)\,||p(w)) \\ &\quad -\mathbb{E}_{q(w|x)p(c|z,w)}[D_{KL}(q_{E2}(z|x)||p_{N1}(z|w, c))] \\ &\quad -\mathbb{E}_{q(z|x)q(w|x)}[D_{KL}(p_{N2}(c|z, w)||p(c))] \end{aligned} \tag{27}$$

where the loss terms are composed of: reconstruction term, w-prior term, conditional prior term, and c-prior term respectively. Comparing the loss function of the GMVAE to the VaDE, it can be seen that the VaDE is slightly less complex because there is no need to sample an additional w.

As seen in Figure 11, images generated from the GMVAE have better quality than the ones generated from VaDE algorithm.

FIGURE 11. Generative results By GMVAE on MNIST.

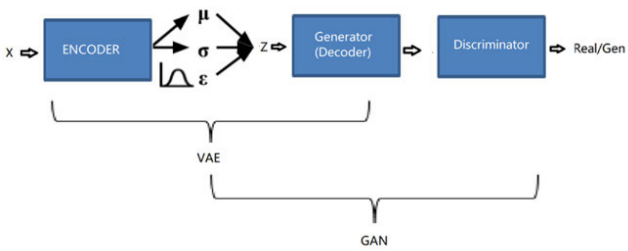

FIGURE 12. Overview of VAE-GAN network.

G. VAE-GAN

VAE-GAN [58] is a combination of the VAEs and GANs into an unsupervised generative model. VAE-GAN transforms the features of the image learned by the discriminator into the reconstruction error of the VAE. The basic idea of this model is to improve the fidelity of the output of VAEs. Since images generated by a VAE are usually blurred, the GAN component can ensure the trueness of the generated image. VAE-GAN is built on the VAE structure with a GAN discriminator [59] added after the decoder to ensure that the samples generated by the VAE have high quality (as shown in Figure 12).

The objective function of VAE-GAN is to minimize the loss function $\mathcal{L}$ that is comprised of the VAE components and the GAN components as:

$$\mathcal{L} = \mathcal{L}_{\text{prior}} + \mathcal{L}_{\text{llike}}^{\text{D}_l} + \mathcal{L}_{\text{GAN}} \tag{28}$$

where, $\mathcal{L}_{\text{prior}}$ represents the KL divergence of the prior in a VAE with the latent distribution $q(x|z)$:

$$\mathcal{L}_{\text{prior}} = D_{KL}(q(z|x)||p(z)) \tag{29}$$

The second term $\mathcal{L}_{\text{llike}}^{\text{D}_l}$ represents the reconstruction loss. It replaces the typical VAE reconstruction loss (expected log likelihood) with a reconstruction error expressed in the GAN discriminator. $\text{D}_l(x)$ denote the hidden representation of the l th layer of the discriminator. Therefore, a Gaussian observation algorithm for Discriminator (x) with identity covariance and mean Discriminator (x') is proposed.

$$p(\text{Dis}_l(x \mid z)) = \mathcal{N}\left(Dis_l(x) \mid Dis_l\left(x'\right), I\right) \tag{30}$$

where x'$\sim$ Decoder(z) is the sample from the generative model of x. Thus the reconstruction loss becomes:

$$\mathcal{L}_{llike}^{D_l} = -\mathbb{E}_{q(z|x)}[logp((Dis_l(x|z))] \tag{31}$$

The above equations assume that the lth layer of the discriminator produces outputs that differ in a Gaussian manner. Thus, the mean squared error (MSE) between the lth layer outputs gives us the VAE's loss function.

The third term in equation (28) is the loss in the GAN part of VAE-GAN. The goal of a conventional GAN is to find a binary classifier that distinguishes between the real data and generated data while encouraging the generator to fit the real data distribution, i.e. traditional GAN loss is defines as:

$$\mathcal{L}_{\text{Gan}} = log\left(Dis\left(x\right)\right) + \log\left(1 - Dis\left(Gen\left(z\right)\right)\right) \quad (32)$$

However, since the GAN in VAE-GAN receives input from the encoder $q\left(z|x\right)$, the GAN loss becomes:

$$\mathcal{L}_{\text{Gan}} = log\left(Dis\left(x\right)\right) + \log\left(1 - Dis\left(Gen\left(z\right)\right)\right) + log\left(1 - Dis\left(Dec\left(Enc\left(x\right)\right)\right)\right) \quad (33)$$

Since VAE-GAN combines the VAE and the GAN, it has a good effect in image synthesis, effectively overcoming the fuzziness generated by regular VAEs (as shown in Figure 13).

FIGURE 13. Create a MNIST dataset by VAE-GAN.

H. F-VAEGAN-D2

The human visual system is superior to the spectral camera system most of the time due to its physical and physiological characteristics. These great features are built on at least two foundations. The first is the brain: about half of our brain is directly involved in the processing of visual information [60]. Second, basic visual skills are learned in a long process that runs through the first few years of life [61]. For example, newborns can distinguish certain patterns based on statistical features such as space or contour. Infants can notice simple rough geometric relationships, and they do not always focus on contours and shapes. At about two years old, children begin to discover higher-order geometric relationships. Here, the term ''visual feature learning'' refers to basic features (e.g., color, shape) and non-basic features (e.g., different directions).

In deep learning, due to its powerful ability to learn general visual features at different levels, deep neural networks have been used as the basic structure of many visual feature learning on Computer Vision, such as object detection [62]–[64] semantic segmentation [65]–[67], etc. Among the deep learning models, with complex architectures and large-scale data sets, convolutional neural network models such as AlexNet [68], VGG [69], GoogLeNet [70], ResNet [71], and DenseNet [72] constantly break through the latest level of many Visual Feature Learning tasks [73]–[77] in computer vision. They are based on learning Visual features of images through CNN and rely on pairs of image features and class attributes. However, the collection and annotation of large-scale data sets is time-consuming and expensive. Therefore, in order to avoid time-consuming and expensive data annotation, in recent years, many studies have emerged through unsupervised learning methods that can learn CNN visual features from large-scale unseen images without using any annotation, such as zero-shot /one-shot/few-shot learning.

Zero-shot learning is when features won't be available in training samples. An important theoretical basis of zero-shot learning is to use high-dimensional semantic features instead of low-dimensional features of samples, so that the trained model is transferable. Most of recent zero-shot learning works [78]–[81] learn a compatibility function between the image and semantic embedding spaces. Few-Shot / One-Shot Learning refers to small sample learning. The purpose is to overcome the problem of massive data required for training models in machine learning. It is expected that enough knowledge can be obtained with a small amount of data. The general approach is to train the model on classes with sufficient training samples and generalize to classes with few samples without learning new parameters [82]–[86]. However, it is suspected that these generated features from small sample learning cannot represent complex features well. f-VAEGAN-D2 [87] generates enough visual features utilized in any-shot learning. The goal is to infer rich features from limited data samples i.e., generate rich features from 0 shots (unseen pictures) to few shots (only a few pictures per class) to many-shots (each class has many pictures).

Figure 14 shows the architecture of the f-VAEGAN-D2 model. It proposes to enhance the feature generator by combining VAEs and GANs with shared decoder and generator and adding another discriminator to distinguish real or generated features from unseen samples.

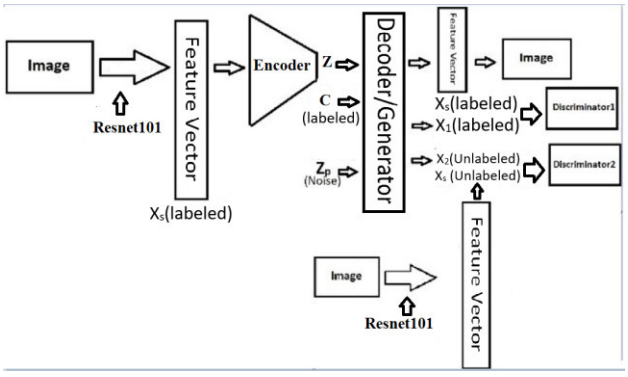

FIGURE 14. Overview of f-VAEGAN-D2 network.

To train the VAE section, Restnet101 is fed with a labeled image and outputs an embedded 2048-dim x_s. This feeds to the Encoder generating the latent variable z. To this latent variable a class label is appended for the sample and fed to the Decoder/Generator. An embedded space $\bar{x}_s$ sampled from the Decoder/Generator is compared to x_s to obtain a loss for the VAE.

The WGAN training utilizes sampling from a latent space $\mathcal{N}z_p \sim (0, 1)$ and a concatenated class label in same manner as the VAE section. This is fed into the Decoder/Generator to create fake embeddings ($\hat{x}_s$). This fake embedding along with $\bar{x}_s$ is utilized by the discriminator to distinguish between real and fake data.

Unseen images are processed by resnet101 and produce an unseen embedding x_u which is utilized along a generated unseen embedding $\hat{x}_u$ in Discriminator 2. This discriminator helps train the model to recognize unseen categories, which can achieve zero-shot learning.

The final objective function of the f-VAEGAN-D2 network can be stated as following:

$$\min_{G,E} \max_{D_1,D_2} \mathcal{L}^{S}_{VAEGAN} + \mathcal{L}^{n}_{WGAN} \tag{34}$$

where G is the VAE decoder and the WGAN generator, D_1, D_2 are the discriminators of both seen and unseen groups respectively. $\mathcal{L}^{s}_{VAEGAN}$ is the loss function of the VAEGAN for the seen samples and can be formulated as:

$$\mathcal{L}^{\mathrm{s}}_{\mathrm{VAEGAN}} = \mathcal{L}^{\mathrm{s}}_{\mathrm{VAE}} + \gamma \mathcal{L}^{\mathrm{s}}_{\mathrm{WGAN}} \tag{35}$$

where γ is a hyperparameter to control the weight of VAE loss $\mathcal{L}^{s}_{VAE}$ and the WGAN loss $\mathcal{L}^{s}_{WGAN}$. These losses functions can be expressed as:

$$\mathcal{L}^{s}_{VAE} = D_{KL}(q(z|x, c)||p(z|c) - \mathbb{E}_{q(z|x,c)}\left[\log p(x|z, c)\right] \tag{36}$$

which is similar to the original VAE loss with addition to the class embedding variable c, and the WGAN loss function can be expressed as:

$$\mathcal{L}^{s}_{WGAN} = \mathbb{E}\left[D_1(x, c)\right] - \mathbb{E}\left[D_1(Dec(z, c), c)\right] - \lambda\mathbb{E}[(\left\|\nabla_{\hat{x}} D_1\left(\hat{x}, c\right)\right\|_2 - 1)^2] \tag{37}$$

where $\hat{x} = \alpha x + (1 - \alpha)Dec(z, c)$ with $\alpha \sim U(0, 1)$, and λ is the penalty coefficient. Finally, the unseen WGAN loss function $\mathcal{L}^{n}_{WGAN}$ can be stated as:

$$\mathcal{L}^{n}_{WGAN} = \mathbb{E}\left[D_2(x_n)\right] - \mathbb{E}\left[D_2\left(G\left(z_p, c_n\right)\right)\right] - \lambda\mathbb{E}[(\left\|\nabla_{\hat{x}} D_2\left(\hat{x}_n\right)\right\|_2 - 1)^2] \tag{38}$$

where again $\hat{x}_n = \alpha x_n + (1 - \alpha)G\left(z_p, c_n\right)$ with $\alpha \sim U(0, 1)$, and λ is the penalty coefficient.

I. ZERO-VAE-GAN

Zero-shot learning (ZSL) is a challenging task due to the lack of unseen class data during training. Existing works attempt to establish a mapping between the visual and class spaces through a common intermediate semantic space. The main limitation of existing methods is a strong bias towards seen classes, known as the domain shift problem. This leads to unsatisfactory performance in both conventional and generalized ZSL tasks. Zero-VAE-GAN [88] tackles this challenge by converting ZSL to a conventional supervised learning by generating features for unseen classes. Zero-VAE-GAN is a joint generative model that couples variational autoencoder (VAE) and generative adversarial network (GAN). The main ideas of this model are:1) generate more seen CNN features 2) labeled unseen CNN features.

The Zero-VAE-GAN model consists of four components:1) Encoder E, and 2) Generator G: by combining two generative models, the model is capable of synthesizing high-quality features, 3) Discriminator D: for discriminating real features and fake generated features, 4) Categorizer C: a classifier to help the model generate more discriminative features for the classification task. The generator G and the discriminator D learn the distribution of features through a two-player minimax competition. G tries to minimize the following loss:

$$\mathcal{L}_{G,D} = -\mathbb{E}[logD(G(z, s))] - \mathbb{E}[logD(G(z', s))] \tag{39}$$

where $x \sim p(x)$, $s \sim p(s)$ and $\mathcal{N}z \sim (0, 1)$, $p(x)$ and $p(s)$ denote the prior distributions of real features and semantic embeddings, respectively. $z' = E(x, s) \in \mathbb{R}^d$ denotes the d-dimensional latent representation generated by the encoder E. Compared with z', $z \in \mathbb{R}^d$ is the arbitrary representation drawn from a Gaussian distribution, which is used as the input for the GAN along with the semantic embedding. On the other hand, D tries to minimize the following loss:

$$\mathcal{L}_D = -\mathbb{E}[\log D(x))] - \mathbb{E}[\log(-D(G(z, s)))] - \mathbb{E}[log(1 - D(G(z', s)))] \tag{40}$$

Unlike f-VAEGAN-D2, Zero-VAE-GAN uses feedback classification probabilities generated from pretrained multi-layers-perceptron (MLP) or k-nearest neighbor classifiers to generate pseudo labels for the real unseen CNN features. These classifiers are trained on the synthesized (fake) data generated from the trained generator G in the first step. The classifiers' pseudo labels probabilities are used for self-training-refinement of the generator G to improve the generation of the features of unseen data.

J. HYPERSPHERICAL VAE

One way to improve any Bayesian model is to change the prior distribution based on the data [89]. The prior distribution does not need to have an objective basis, so it can be based in part or completely on subjective beliefs. Further, an ideal latent space should separate clusters for each class [90]. However, in normal VAEs, due to the Gaussian prior, there are limitations in the latent space, e.g., the Gaussian prior leads to improper clustering in high dimensional data, and further cannot effectively represent directional data such as spanning from protein structure [91]. Therefore, to improve the clusters in the latent space in high dimensions and learn useful representations on directional data, there is a need to

replace the Gaussian prior to a prior that separates the classes over the entire latent space. One solution is to use von Mises-Fisher (vMF) for the prior.

The vMF distribution [92] refers to a continuous probability distribution model on a circle, which is also called a circular normal distribution. Some views regard it as an approximation to the wrapped normal distribution, as it is a cyclic simulation of the normal distribution. This is a normal distribution in hyperspherical space [93]. Figure 15 shows sets of points sampled from VMF distributions on the 3D sphere.

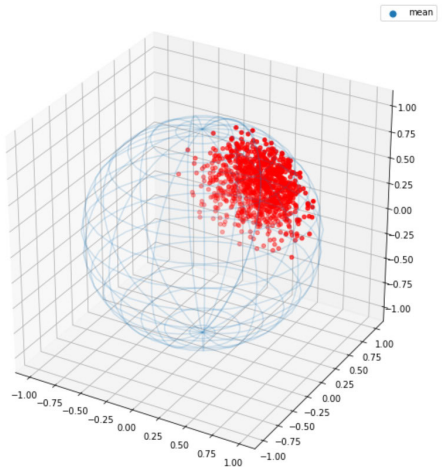

FIGURE 15. Sets of points sampled from VMF distributions on the 3D sphere.

Hyperspherical VAE (S-VAE) [94] uses the vMF distribution as an alternative to the Gaussian distribution. This replacement leads to a hyperspherical latent space as opposed to a hyperplanar one, where the Uniform distribution on the hypersphere is conveniently recovered as a special case of the vMF. Let $z \in \mathbb{R}^m$, then we can define the vMF distribution of latent variable as:

$$q\left(z|\mu,\kappa\right) = C_m\left(\kappa\right)e^{\kappa\mu^T z} \tag{41}$$

where parameters μ and κ are called the mean direction and concentration parameter, respectively. The greater the value of κ, the higher the concentration of the distribution around the mean direction μ. The distribution is unimodal for $\kappa > 0$ and is uniform on the sphere for $\kappa = 0$ where $||\mu^2|| = 1$. $C_m\left(\kappa\right)$ is the normalization constant and is equal to

$$C_m\left(\kappa\right) = \frac{\kappa^{m/2-1}}{\left(2\pi\right)^2 \mathcal{J}_{m/2-1}\left(\kappa\right)} \tag{42}$$

where $\mathcal{J}_{\mathrm{m}/2-1}\left(\kappa\right)$ is modified Bessel function of the first kind at order v.

The authors use a special case of KL divergence such that uniform prior is placed on the latent space.

Given Gamma function,

$$\Gamma\left(z\right) = \left(z-1\right)! = \int_0^\infty t^{z-1}e^{-t}dt \tag{43}$$

Steifel manifold area is

$$\tau\left(d,r\right) = \frac{2^r\pi^{mr/2}}{\pi^{r(r-1)/4}\Pi_{j=1}^r\Gamma\frac{p-r+1}{2}} \tag{44}$$

Von Mises-Fisher distribution is a case of Steifel manifold with radius $r = 1$. This is actually the surface area of the n-sphere of radius 1. Thus, uniform distribution of vMF, a case where the $\kappa = 0$ is

$$C_m\left(0\right) = \tau\left(d,1\right) = \frac{2^1\pi^{m/2}}{\pi^{1(1-1)/4}\Pi_{j=1}^1\Gamma\frac{m-1+1}{2}} = \frac{2\pi^{m/2}}{\Gamma\left(m/2\right)} \tag{45}$$

Then, in this case of KL Divergence derivation to uniform distribution, posterior is vMF $= q\left(z|\mu,\kappa\right)$ and prior is $U\left(S^{m-1}\right) = p\left(z\right)$. Then

$$q\left(z|\mu,\kappa\right) = C_m\left(\kappa\right)e^{\kappa\mu^T z} = \frac{\kappa^{m/2-1}}{\left(2\pi\right)^{2I_{m/2-1}}\left(\kappa\right)}e^{\kappa\mu^T z} \tag{46}$$

and

$$p\left(z\right) = \left(\frac{2\pi^{m/2}}{\Gamma\left(m/2\right)}\right)^{-1} = \frac{\Gamma\left(m/2\right)}{2\pi^{m/2}} \tag{47}$$

Finally, the KL Divergence with vMF term $KL(\mathrm{vMF}(\mu,\kappa))||U\left(S^{m-1}\right)$ to be optimized is:

$$\begin{aligned} &KL[q\left(z|\mu,\kappa\right))||p(z))] \\ &= \kappa\frac{\mathcal{J}_{m/2}\left(\kappa\right)}{\mathcal{J}_{m/2-1}\left(\kappa\right)} + \log C_m\left(\kappa\right) + \log\left(\frac{\Gamma\left(m/2\right)}{2\left(\pi^{m/2}\right)}\right) \end{aligned} \tag{48}$$

Since the KL term does not depend on μ, this parameter is only optimized in the reconstruction term. One difficulty is that the modified Bessel function in $C_m\left(\kappa\right)$ in the above expression cannot be handled by automatic differentiation packages. Thus, to optimize this term, the gradient is derived with respect to the concentration parameter κ:

$$\begin{aligned} &\nabla_\kappa KL(\mathrm{vMF}(\mu,\kappa))||U\left(S^{m-1}\right) \\ &= \frac{1}{2}\kappa(\frac{\mathcal{J}_{m/2+1}\left(\kappa\right)}{\mathcal{J}_{m/2-1}\left(\kappa\right)} - \frac{\mathcal{J}_{m/2}\left(\kappa\right)\left(\mathcal{J}_{m/2-2}\left(\kappa\right)+\mathcal{J}_{m/2}\left(\kappa\right)\right)}{\mathcal{J}_{m/2-1}\left(\kappa\right)^2}+1) \end{aligned} \tag{49}$$

In the S-VAE all digits occupy the entire space. S-VAE is naturally suited to capture data with a hyperspherical latent structure, while outperforming a normal VAE, in low dimensions. However, the available spherical surface area can be limited and may collapse in higher dimensions. Figure 16 shows the visualization of latent space representation of MNIST for S-VAE. Visualization of latent space representation of MNIST for S-VAE.

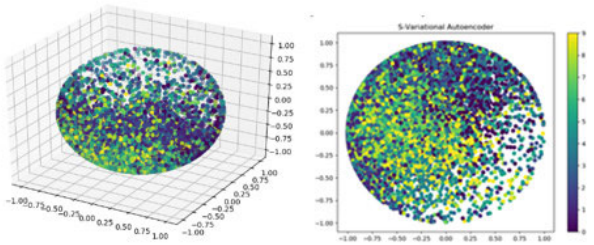

FIGURE 16. Visualization of 3D latent space representation (left) and 2D latent space representation (right) of MNIST for *S*-VAE.

IV. EXPERIMENTS AND RESULTS

A. MNIST DATASET

For all experiments, we used the Modified National Institute of Standards and Technology (MNIST) benchmark dataset. There has been extensive research on this dataset for various purposes such as image classification and generation. The dataset consists of 60,000 images for training purpose and 10,000 images for testing purpose, and both are sharing the same distribution. All images are of size 28 $\times$ 28 and the dataset contains ten label classes from [0-9]. In all experiments, Keras framework has been utilized to build all versions and variations of VAE included in this paper.

B. IMPLEMENTATION DETAILS

The hardware specifications for executing different implementations use Tesla P100 GPU with 25 GB RAM. Table 1 shows some values of hyper-parameters which are used in all experiments. For comparison purpose, some parameters for all VAE variants have been set to the same values to perform fair comparison on the MNIST dataset. All codes are available at https://github.com/VAEs-Tutorial/paper.

TABLE 1. Hyper-parameters used in all experiments.

HYPER-PARAMETERS	VALUES
EPOCHS	5000
BATCH-SIZE	100
LEARNING RATE	0.001
OPTIMIZER	RMSprop
DATASET	MNIST

C. RESULTS

1) QUALITY OF THE GENERATED IMAGES

We applied VAE, INFO VAE, β-VAE, VAE-GAN, GMVAE, VaDE, VQ-VAE and *S*-VAE on the MNIST dataset with 5000 epochs. The generated image results are shown in Fig. 17 and Fig. 18. It is clearly seen in the figures that GMVAE generated more clear digits as compared to other VAE models. After GMVAE, β-VAE and VaDE produced better digits, but some blurriness is also present in these

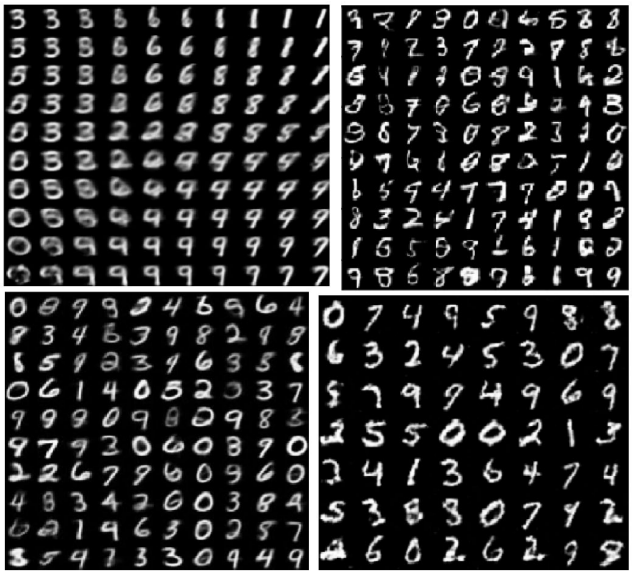

FIGURE 17. MNIST digits generated at 5000 epochs: VAE (Top Left), Info VAE (Top Right), β-VAE (Bottom Left) and VAE-GAN (Bottom Right).

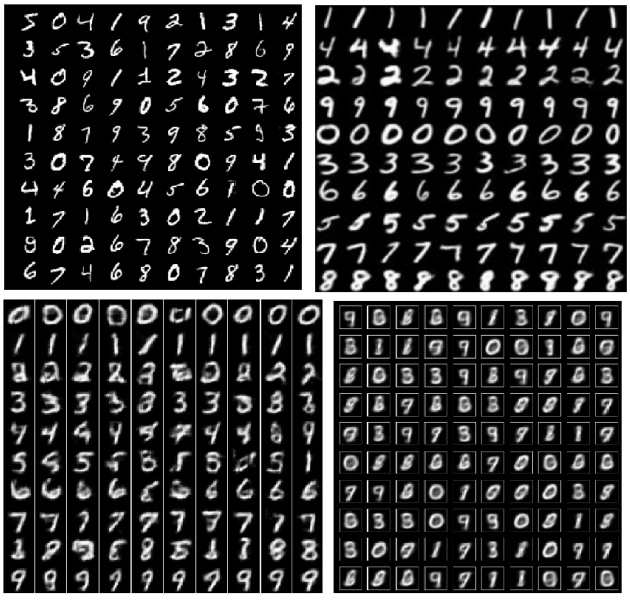

FIGURE 18. MNIST digits generated at 5000 epochs: GMVAE (Top Left), VaDE (Top Right), VQ-VAE (Bottom Left) and *S*-VAE (Bottom Right).

digits. β-VAE and INFO VAE tend to find more latent factors consistently and learn more clearly the characterization of disentanglement than other VAE models. InfoVAE provides a good way to handle latent code ignorance issues. However, its generated images have high distortion in digits as compared to β-VAE. VaDE and GMVAE are clustering models, they can perform clustering tasks in addition to generating digits better than other VAE models. Images generated from the GMVAE have better quality than the ones generated from VaDE. VAE-GAN, VQ-VAE and *S*-VAE produced noise in image and digits are not very clear. In addition, some digits generated by VQ-VAE and VAE-GAN have not clear shape

TABLE 2. Quantitative results of VAE and its variants on MNIST.

Algorithms	Accuracy of re-generated images%	Computational Time (seconds per epoch)
VAE	78.6	2
VaDE	94.0	65
GMVAE	96.3	37
InfoVAE	66.7	29
β-VAE	84.1	19
VQ-VAE	83.7	3
S -VAE	76.5	55
VAE-GAN	63.3	35

and edges. Images generated from the VAE-GAN have better quality than the ones generated from VQ-VAE. If we increase epochs, the quality of generated images can improve. However, we compared the results at 5000 epochs only for a consistent comparison.

2) QUANTITATIVE RESULTS ON THE MNIST DATASET

Table 2 shows quantitative results of all VAE variants on the MNIST dataset. The evaluation metrics are classification accuracy, loss, and computational time (seconds per epoch). We fixed 5000 epochs for all experiments. Table 1 shows other fixed hyperparameter values for these experiments to perform fair comparison. We use a pre-trained CNN classifier to calculate the classification accuracy of the re-generated images. Initially, the quality and accuracy of generated images is very low because generator does not know much about real data. But after various epochs, the generator starts to learn and generates more accurate images. As we can see, the highest classification accuracy is obtained by GMVAE i.e. 96.3%. The computational time (seconds per epoch) to perform this experiment is also less i.e. two seconds, which shows the efficiency of VAE. The lowest classification accuracy score is 51.87 % by VAE-GAN. Figure 19 shows comparison of classification accuracy of VAE and its variants on the MNIST dataset. If we analyze computational time, then

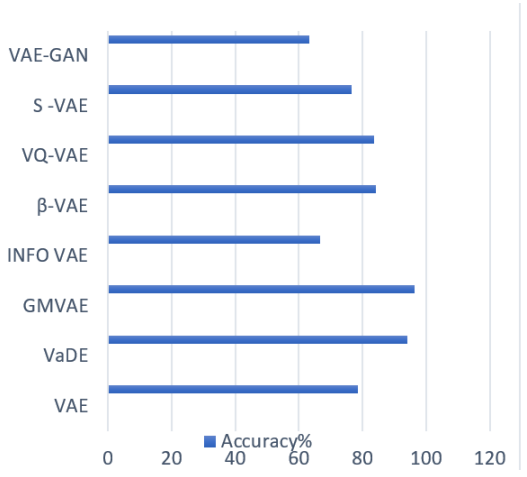

FIGURE 19. Comparison of classification accuracy on MNIST with GAN and its variants.

VaDE takes highest time to complete experiment i.e. 65 seconds. However, VAE takes 2 seconds to complete each epoch which is the shortest time. The computation time depends upon the system specifications as well. If you have faster GPU, then it helps to decrease the time. If we look at the loss values in Table 2, then there is high fluctuation among them. Loss values do not tell much about VAE's performance as compared to other deep learning models. We are not sure when to stop training in a VAE. In VAE, the KL divergence loss and the reconstruction loss compete with each other, and improvement in one term means more loss in the other. Depending upon the diversity of the dataset, both loss terms of KL divergence and reconstruction start to converge at some point after certain number of epochs. When there is no further decrease in loss of KL divergence and reconstruction terms, it indicates training is almost complete.

The convergence in loss terms of KL divergence and reconstruction shows that the model has learned well enough and it cannot be improved further. However, the loss value may bounce around a bit and this number is not very informative. If the model is not reaching convergence, we may need to change the learning rate or other hyper-parameters. Till now, there is no proper evaluation metric for VAEs. Sometimes good qualitative results may have less accuracy. Also, there are different loss functions used in VAEs making it difficult to perform fair comparison among them.

Therefore, evaluation metric and training procedure should be chosen according to desired application or task, e.g. a variety of disentanglement evaluation protocols have been proposed leveraging the statistical relations between the learned representation and the ground-truth factor of variations [95]. Also, a VAE model's good performance in one domain does not necessarily mean good performance in another domain.

V. COMPARATIVE EVALUATION OF VAES

We have introduced the most significant problems present in the original VAE design, which are 1) blurry outputs, 2) latent representations are not interpretable, 3) Gaussian distribution as priori has limitations because the representations that are learned can only be unimodal and do not allow for more complex features, 4) the Gaussian definition is based on the L_2-norm that suffers from the curse of dimensionality. We have surveyed significant VAE-variants that remedy these

TABLE 3. Description of different Models.

Models	Author & Year & Ref	Description	Variation Type
VaDE	Jiang et al.2017 [53]	Imposes a GMM prior on VAEs for better clustering	Prior-variant
GMVAE	Dilokthanakul et al.2017 [57]		Prior-variant
INFO VAE	Zhao et al.2016 [33]	Combines VAEs with representation learning, regularizes the ELBO by MMD	Regularizing posterior-variant
β-VAE	Higgins et al.2017 [36]	Combines VAEs with representation learning, regularizes the ELBO by selecting the appropriate hyperparameter β	Regularizing posterior-variant
VQ-VAE	van den Oord et al.2017 [45]	Combines VAEs with discrete latent representation, imposes vector quantization algorithm on latent space	Prior-variant
S -VAE	Davidson et al.2018 [94]	Combines VAEs with spherical latent representation, replaces Gaussian distribution prior with von Mises-Fisher (vMF)	Prior-variant
VAE-GAN	Larsen et al.2015 [58]	Combines VAEs with GANs, trains both a VAE and a GAN simultaneously	Architecture-variant
f-VAEGAN-D2	Xian et al.2019 [87]	Combines VAEs with GANs and any-shot learning, trains a VAE along with a GAN adding a non-conditional discriminator	Architecture-variant
CVAE	Sohn et al.2015 [96]	Combines VAEs with output representation learning and output prediction, maximizes the conditional log-likelihood on Gaussian latent variables	Architecture-variant
WAE	Tolstikhin et al.2017 [97]	Improves VAE with high quality sampling, minimizes the optimal transport cost	Regularizing posterior -variant
Zero-VAE-GAN	Gao et al.2020 [88]	Combines VAEs with GANs and zero-shot learning, trains a VAE along with a GAN and an adversarial categorizer, labels unseen data through two self-training pseudo-labeling strategies	Architecture-variant
S3VAE	Zhu et al.2020 [98]	Combines VAEs with disentangled time-invariant and time-varying representations of sequential data (e.g., video and audio) under self-supervision	Architecture-variant + Prior-variant

problems through three design considerations. We compare the performance between the variations of the VAE.

A. ARCHITECTURE-VARIANTS

Some of the VAE that improve upon the traditional architecture are VAE-GAN and f-VAEGAN-D2. VAE-GAN combines VAE and GAN architectures and can generate high quality images and classifications. f-VAEGAN-D2 also combines conditional VAE and GAN architectures and can generate enough CNN visual features for small sample learning tasks. However, they both have complex network structures and high-computational complexity and instability on longer training problems

B. REGULARIZE POSTERIOR-VARIANTS

Regularization of the posterior or the loss function can help in obtaining better disentangled representations. INFO VAE improves representation learning by regularizing the ELBO by MMD. β-VAE, a special case of INFO VAE, regularizes the ELBO through hyperparameter β. They both combine VAEs with feature learning and are capable of learning useful latent disentangled representations automatically. They are scalable, stable to train, and are easy to implement. However, both suffer from the blurred image generation problem. Moreover, at this point there is no clear relationship between disentangled representations and downstream tasks such as classification [95]. β-VAE models cannot effectively trade-off weighing of X and Z and information preferences and is also encounters under-fitting or ignoring a subset of the latent variables.

C. PRIOR-VARIANT

One way to improve any Bayesian model is to change the prior distribution based on the data. VaDE and GMVAE adds clustering through imposition of a GMM priori on VAEs. The number of clusters can be set to the number of classes in each dataset. However, they both have problems of

TABLE 4. Pros and Cons between different Models.

Algorithms	Pros	Cons
Original VAE	• Generates samples in an unsupervised manner • Capable of learning smooth latent representations of the input data • Able to deal with probability distribution • Decent theoretical guarantee	• Latent representation is meaningless • Blurry outputs that generate. • The Gaussian distribution as priori has many limitations because the representations that be learned can only be unimodal and do not allow for more complex features • The Gaussian definition is based on the L_2-norm that suffers from the curse of dimensionality
VaDE	• Gives VAE ability for classification with unsupervised clustering • unsupervised clustering yields better representations • The number of clusters can be set to the number of classes in each dataset ensuring each class is properly represented	• High-computational complexity • Low generated image quality • Fidelity of reconstruction can suffer if number of clusters K does not match actual classes in data
GMVAE		• High-computational complexity • Low generated image quality • GMVAE is slightly more complicated than VaDE, and it is computationally expensive because of the need to sample an additional latent space (w)
InfoVAE	• Combines VAEs with feature learning • Capable of learning useful disentangled representations automatically • Scalable and stable to train • Easy to implement	• Blurred image generation • No clear relationship between disentangled representations and downstream tasks such as classification
β-VAE		• Blurred image generation • β-VAE models cannot effectively trade-off weighing of X and Z and information preferences • Under-fitting or ignoring the latent variables • No clear relationship between disentangled representations and downstream tasks such as classification
VQ-VAE	• Combines VAEs with discrete representations learning • Capable of learning useful and discrete representations • Abstracts away noise and details	• Instability on challenging datasets (i.e. High entropy) • Complex sampling procedure
S -VAE	• Combines VAEs with spherical representations learning on directional data • Capable of learning useful and spherical representations	• The available spherical surface area is limited • Collapses in higher dimensions
VAE-GAN	• Combines VAEs and GANs • Capable of generating high-quality image • Can be utilized in classification tasks	• Complex network structure • High computational complexity • Instability on longer training
f-VAEGAN-D2	• Combines VAEs and GANs • Generates significant CNN visual features • Performs any-shot learning	
Zero-VAE-GAN	• Combines VAEs and GANs • Generates significant CNN visual features • Labels unseen CNN visual features • Performs zero-shot learning	
S3VAE	• Combines VAEs with LSTM • Capable of learning useful and disentangled time-invariant and time-varying representations for sequential data.	
CVAE	• Combines VAEs and output prediction • Scalable and efficient in inference and learning • Can be used effectively for data augmentation to improve classification accuracy	• High-computational complexity • Requires testing on more complex image datasets
WAE	• Improves VAEs with a better reconstruction • Images sampled from latent space are of better quality • Stability of training	• Theoretical analysis of the dual formulations for WAE-GAN and WAE-MMD has not been documented • Criteria for matching the encoded distribution to the prior distribution has not been formalized

high-computational complexity and their generated image quality is low. Moreover, VaDE has no specified stability for differ ent settings of the number of clusters, K. GMVAE is slightly more complicated than VaDE, and it is computationally expensive because of the need to sample an additional latent space (w).

VQ-VAE proposes combining VAEs with discrete latent representation by imposing a vector quantization algorithm on the latent space. It can learn useful and discrete representations automatically as well as abstract away noise and details. However, VQ-VAE has the problem of a complex sampling procedure and is unstable in challenging datasets (i.e., high entropy). S-VAE utilizes spherical latent representation by replacing Gaussian distribution priori in the classical VAE with von Mises-Fisher (vMF) distribution. It can utilize the hyperspherical space to separate clusters for each class without forcing its mean to be close to the center.

Among the different approaches surveyed in this work, it was shown that variations of the VAEs can improve the generated image quality and their diversity. It has been indicated in [99] that the capacity and performance of VAEs are related to the network size and batch, which follows that a well-designed architecture is critical for good VAE performance. However, modifications to the architecture alone do not fully improve generation of data. Redesign of the loss function including regularization and normalization can help improve effective reconstruction for VAEs. In addition, replacing the Gaussian prior can improve the VAE model to learn appropriate latent representations.

There are other types of VAEs that have been introduced but are not frequently used in applications. For example, Conditional VAE (CVAE) [96], which is similar to Conditional GAN, where a control vector "c" is used as an input with the data "x", as well as the latent variables "z", to be a part of the VAE structure. In most applications of this type of VAE, the label data is used as this control variable. Moreover, other types of VAE are introduced where the KL distribution similarity measurement metric is not used, and other metrics are utilized. As an example of these types, the Wass Wasserstein VAE (WAE) [97], where Wasserstein distance is used instead of the KL term in the loss function to measure the similarity between the model distribution and the target distribution.S3VAE [98] learns disentangled time-invariant and time-varying representations for sequential data (e.g., videos and audios) under self-supervision. This makes it possible for sequential data generation, high-resolution video generation, video prediction and image-to-video generation.

There is no single VAE design that can be claimed to be the best. The choice of a specific VAE type depends on the application. For instance, if an application requires the sampling of different classes in the latent space, there is a need of clustering. VaDE, GMVAE, S-VAE can be good choices here. S-VAE can do a better job on directional data compared to the other two. If an application requires production of enough high-quality images (requiring generation of images which are very diverse), VAE-GAN can be good choices here. If an application requires production of enough CNN visual images for few-shot/zero-shot learning (requiring generation of images which are very diverse), f-VAEGAN-D2/ Zero-VAE-GAN can be a good choice. If there is a need to learn useful latent disentangled representations automatically, in order to create more attributes of the image to further improve the classification problems, INFO VAE and β-VAE can be good choices.

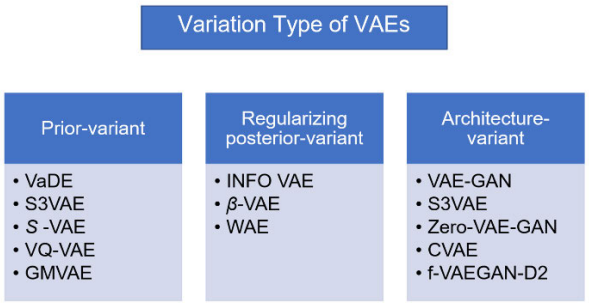

FIGURE 20. Variation Type of VAEs.

Table 3 summarizes the different variants along with their description and variation type. Table 4 lists the pros and cons of different VAE designs discussed in this paper. Figure 20. shows variation type of VAEs.

VI. FUTURE OPPORTUNITIES AND CONCLUSIONS

VAEs and its variations have played a very important role in unsupervised data generation (especially in image generation), deep clustering and representation learning.

The improvement in the variations of the VAEs can be summarized into three perspectives:

1) Architecture: By enhancing the network design with other architectures, VAEs can improve the image quality, e.g., VAEs combined with GANs can decrease the blurred effect of the image.

2) Posterior distribution: Regularization of the posterior distribution can be used to boost the disentangled features, e.g., β-VAE and Info-VAE provide disentanglement and hierarchical organization of features.

3) Structured prior distribution: VAE variants can also introduce a structured prior distribution such as imposing a GMM priori (GMVAE, VaDE), and Vector Quantization (VQ) on VAEs (VQ-VAE). These accomplish better clustering and representation of data.

Based on the analyses and the comparative evaluation provided in this paper, we believe that understanding the VAE model from the perspective of variational optimization and information theory will be important research trends in the near future. We summarize some of the potential areas of research in the VAE field as:

1) Enhancing the VAE model by improving the variational optimization of the latent variable space, thereby avoiding or minimizing the limitations of the existing methods. This can make learning more meaningful by providing valuable

information in the variational inference process. Many research opportunities have not been explored in the intersection of these methods, i.e., integrating regularization-based methods while bringing in structured priors.

2) Separation of information: By weakening the dependencies between non-associative features, disentanglement and generation capabilities of VAEs will be greatly improved. By calculating the mutual information index between the various influence factors in VAEs, the model can potentially discriminate influencing factors from non-descriptive ones.

3) Disentanglement learning: It is unclear whether the solution of disentangled representations is useful for downstream tasks. Therefore, future research on disentangled representations learning should consider the role of inductive biases supervision.

4) Posterior collapse: Posterior collapse in VAEs arises when the variational posterior distribution closely matches the prior for a subset of latent variables [100]. Conventional wisdom largely assigns blame for this phenomenon on the undue influence of KL-divergence regularization. Although there is now a vast literature on the various potential causes of posterior collapse, there remains ambiguity as to exactly what is this phenomena [101]. Therefore, more significant progress towards understanding the causes of posterior collapse is needed.

These proposed enhancements will improve the ability to generate meaningful artificial data. This data can be used for representation learning or to improve the classification in deep networks where currently there is not enough training data, or a particular class is underrepresented.

We have provided a comprehensive insight, and a comparative evaluation summary of the variations in VAEs so that researchers can grasp the fundamental theory as well as the intuition behind the variations on VAEs. Further, we have provided reference implementations for the different VAE variations on github. We hope that this will be useful in improving the state of the art leading to research breakthroughs in related fields.

REFERENCES

[1] M. I. Jordan and T. M. Mitchell, "Machine learning: Trends, perspectives, and prospects," *Science*, vol. 349, no. 6245, pp. 255–260, Jul. 2015.
[2] D. P Kingma and M. Welling, "Auto-encoding variational Bayes," 2013, *arXiv:1312.6114*. [Online]. Available: http://arxiv.org/abs/1312.6114
[3] I. Goodfellow, J. Pouget-Abadie, M. Mirza, B. Xu, D. Warde-Farley, S. Ozair, A. Courville, and Y. Bengio, "Generative adversarial nets," in *Proc. Adv. Neural Inf. Process. Syst.*, 2014, pp. 2672–2680.
[4] C. Doersch, "Tutorial on variational autoencoders," 2016, *arXiv:1606.05908*. [Online]. Available: http://arxiv.org/abs/1606.05908
[5] I. Goodfellow, "NIPS 2016 tutorial: Generative adversarial networks," 2017, *arXiv:1701.00160*. [Online]. Available: http://arxiv.org/abs/1701.00160
[6] L. Mi, M. Shen, and J. Zhang, "A probe towards understanding GAN and VAE models," 2018, *arXiv:1812.05676*. [Online]. Available: http://arxiv.org/abs/1812.05676
[7] Y. Bengio, A. Courville, and P. Vincent, "Representation learning: A review and new perspectives," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 35, no. 8, pp. 1798–1828, Aug. 2013.
[8] J. Walker, C. Doersch, A. Gupta, and M. Hebert, "An uncertain future: Forecasting from static images using variational autoencoders," in *Proc. Eur. Conf. Comput. Vis.* Cham, Switzerland: Springer, 2016, pp. 835–851.
[9] A. Mishra, S. K. Reddy, A. Mittal, and H. A. Murthy, "A generative model for zero shot learning using conditional variational autoencoders," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. Workshops (CVPRW)*, Jun. 2018, pp. 2188–2196.
[10] E. Schonfeld, S. Ebrahimi, S. Sinha, T. Darrell, and Z. Akata, "Generalized zero- and few-shot learning via aligned variational autoencoders," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2019, pp. 8247–8255.
[11] W. Wang, Y. Pu, V. K. Verma, K. Fan, Y. Zhang, C. Chen, P. Rai, and L. Carin, "Zero-shot learning via class-conditioned deep generative models," in *Proc. 32nd AAAI Conf. Artif. Intell.*, 2018, pp. 1–8.
[12] C. K. Sønderby, J. Caballero, L. Theis, W. Shi, and F. Huszár, "Amortised MAP inference for image super-resolution," 2016, *arXiv:1610.04490*. [Online]. Available: http://arxiv.org/abs/1610.04490
[13] J. Bruna, P. Sprechmann, and Y. LeCun, "Super-resolution with deep convolutional sufficient statistics," 2015, *arXiv:1511.05666*. [Online]. Available: http://arxiv.org/abs/1511.05666
[14] R. A. Yeh, C. Chen, T. Y. Lim, A. G. Schwing, M. Hasegawa-Johnson, and M. N. Do, "Semantic image inpainting with deep generative models," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, Jul. 2017, pp. 5485–5493.
[15] J. Xu and Y. W. Teh, "Controllable semantic image inpainting," 2018, *arXiv:1806.05953*. [Online]. Available: http://arxiv.org/abs/1806.05953
[16] A. Makhzani, J. Shlens, N. Jaitly, I. Goodfellow, and B. Frey, "Adversarial autoencoders," 2015, *arXiv:1511.05644*. [Online]. Available: http://arxiv.org/abs/1511.05644
[17] Y. Zhang, E. Zhang, and W. Chen, "Deep neural network for halftone image classification based on sparse auto-encoder," *Eng. Appl. Artif. Intell.*, vol. 50, pp. 245–255, Apr. 2016.
[18] C. Xing, L. Ma, and X. Yang, "Stacked denoise autoencoder based feature extraction and classification for hyperspectral images," *J. Sensors*, vol. 2016, pp. 1–10, Nov. 2016.
[19] I. Goodfellow, Y. Bengio, and A. Courville, *Deep Learning*. Cambridge, MA, USA: MIT Press, 2016.
[20] J. Gao, H. Qi, X. Xia, and J.-Y. Nie, "Linear discriminant model for information retrieval," in *Proc. 28th Annu. Int. ACM SIGIR Conf. Res. Develop. Inf. Retr. (SIGIR)*, 2005, pp. 290–297.
[21] A. Radford, L. Metz, and S. Chintala, "Unsupervised representation learning with deep convolutional generative adversarial networks," 2015, *arXiv:1511.06434*. [Online]. Available: http://arxiv.org/abs/1511.06434
[22] G. E. Hinton, "Reducing the dimensionality of data with neural networks," *Science*, vol. 313, no. 5786, pp. 504–507, Jul. 2006.
[23] R. Salakhutdinov and G. E. Hinton, "Deep Boltzmann machines," in *Proc. Conf. Artif. Intell. Statist.*, 2009, pp. 448–455.
[24] A. Voulodimos, N. Doulamis, A. Doulamis, and E. Protopapadakis, "Deep learning for computer vision: A brief review," *Comput. Intell. Neurosci.*, vol. 2018, pp. 1–13, Feb. 2018.
[25] D. J. Rezende and S. Mohamed, "Variational inference with normalizing flows," 2015, *arXiv:1505.05770*. [Online]. Available: http://arxiv.org/abs/1505.05770
[26] D. J. Rezende, S. Mohamed, and D. Wierstra, "Stochastic backpropagation and approximate inference in deep generative models," 2014, *arXiv:1401.4082*. [Online]. Available: http://arxiv.org/abs/1401.4082
[27] P. Langley, "Selection of relevant features in machine learning," in *Proc. AAAI Fall Symp. Relevance*, vol. 184, 1994, pp. 245–271.
[28] C. R. Turner, A. Fuggetta, L. Lavazza, and A. L. Wolf, "A conceptual basis for feature engineering," *J. Syst. Softw.*, vol. 49, no. 1, pp. 3–15, Dec. 1999.
[29] W. Chan, N. Jaitly, Q. Le, and O. Vinyals, "Listen, attend and spell: A neural network for large vocabulary conversational speech recognition," in *Proc. IEEE Int. Conf. Acoust., Speech Signal Process. (ICASSP)*, Mar. 2016, pp. 4960–4964.
[30] X. Dong, D. Thanou, P. Frossard, and P. Vandergheynst, "Learning Laplacian matrix in smooth graph signal representations," *IEEE Trans. Signal Process.*, vol. 64, no. 23, pp. 6160–6173, Dec. 2016.
[31] Y. Aytar, C. Vondrick, and A. Torralba, "SoundNet: Learning sound representations from unlabeled video," in *Proc. Adv. Neural Inf. Process. Syst.*, 2016, pp. 892–900.
[32] T. Young, D. Hazarika, S. Poria, and E. Cambria, "Recent trends in deep learning based natural language processing [review article]," *IEEE Comput. Intell. Mag.*, vol. 13, no. 3, pp. 55–75, Aug. 2018.
[33] S. Zhao, J. Song, and S. Ermon, "InfoVAE: Information maximizing variational autoencoders," 2017, *arXiv:1706.02262*. [Online]. Available: http://arxiv.org/abs/1706.02262

[34] A. Gretton, K. M. Borgwardt, M. J. Rasch, B. Schölkopf, and A. Smola, "A kernel two-sample test," *J. Mach. Learn. Res.*, vol. 13, pp. 723–773, Mar. 2012.

[35] M. Tschannen, O. Bachem, and M. Lucic, "Recent advances in autoencoder-based representation learning," 2018, *arXiv:1812.05069*. [Online]. Available: http://arxiv.org/abs/1812.05069

[36] I. Higgins, L. Matthey, A. Pal, C. Burgess, X. Glorot, M. Botvinick, S. Mohamed, and A. Lerchner, "Beta-VAE: Learning basic visual concepts with a constrained variational framework," in *Proc. ICLR*, 2017, vol. 2, no. 5, p. 6.

[37] T. Q. Chen, X. Li, R. B. Grosse, and D. K. Duvenaud, "Isolating sources of disentanglement in variational autoencoders," in *Proc. Adv. Neural Inf. Process. Syst.*, 2018, pp. 2610–2620.

[38] Z. Li, Y. Tang, and Y. He, "Unsupervised disentangled representation learning with analogical relations," 2018, *arXiv:1804.09502*. [Online]. Available: http://arxiv.org/abs/1804.09502

[39] P. Vincent, H. Larochelle, I. Lajoie, Y. Bengio, and P.-A. Manzagol, "Stacked denoising autoencoders: Learning useful representations in a deep network with a local denoising criterion," *J. Mach. Learn. Res.*, vol. 11, no. 12, pp. 3371–3408, Dec. 2010.

[40] X. Chen, Y. Duan, R. Houthooft, J. Schulman, I. Sutskever, and P. Abbeel, "InfoGAN: Interpretable representation learning by information maximizing generative adversarial nets," in *Proc. Adv. Neural Inf. Process. Syst.*, 2016, pp. 2172–2180.

[41] E. Denton, S. Gross, and R. Fergus, "Semi-supervised learning with context-conditional generative adversarial networks," 2016, *arXiv:1611.06430*. [Online]. Available: http://arxiv.org/abs/1611.06430

[42] A. Mnih and K. Gregor, "Neural variational inference and learning in belief networks," 2014, *arXiv:1402.0030*. [Online]. Available: https://arxiv.org/abs/1402.0030

[43] A. Courville, J. Bergstra, and Y. Bengio, "A spike and slab restricted Boltzmann machine," in *Proc. 14th Int. Conf. Artif. Intell. Statist.*, 2011, pp. 233–241.

[44] A. Mnih and D. J. Rezende, "Variational inference for Monte Carlo objectives," 2016, *arXiv:1602.06725*. [Online]. Available: http://arxiv.org/abs/1602.06725

[45] A. van den Oord, O. Vinyals, and K. Kavukcuoglu, "Neural discrete representation learning," in *Proc. Adv. Neural Inf. Process. Syst.*, 2017, pp. 6306–6315.

[46] J. Chorowski, R. J. Weiss, S. Bengio, and A. van den Oord, "Unsupervised speech representation learning using WaveNet autoencoders," *IEEE/ACM Trans. Audio, Speech, Language Process.*, vol. 27, no. 12, pp. 2041–2053, Dec. 2019.

[47] S. Dieleman, A. van den Oord, and K. Simonyan, "The challenge of realistic music generation: Modelling raw audio at scale," in *Proc. Adv. Neural Inf. Process. Syst.*, 2018, pp. 7989–7999.

[48] M. Ester, H.-P. Kriegel, J. Sander, and X. Xu, "A density-based algorithm for discovering clusters in large spatial databases with noise," in *Proc. KDD*, 1996, vol. 96, no. 34, pp. 226–231.

[49] D. Arthur and S. Vassilvitskii, "K-means++: The advantages of careful seeding," in *Proc. ACM-SIAM Symp. Discr. Algorithms*, Jan. 2007, pp. 1027–1035.

[50] A. Likas, N. Vlassis, and J. J. Verbeek, "The global k-means clustering algorithm," *Pattern Recognit.*, vol. 36, no. 2, pp. 451–461, Feb. 2003.

[51] A. ElSayed, A. Mahmood, and T. Sobh, "Unsupervised sub-graph selection and its application in face recognition techniques," in *Proc. Int. Conf. Image Anal. Recognit.* Cham, Switzerland: Springer, 2015, pp. 247–256.

[52] J. Xie, R. Girshick, and A. Farhadi, "Unsupervised deep embedding for clustering analysis," in *Proc. Int. Conf. Mach. Learn.*, 2015, pp. 478–487.

[53] Z. Jiang, Y. Zheng, H. Tan, B. Tang, and H. Zhou, "Variational deep embedding: An unsupervised and generative approach to clustering," 2016, *arXiv:1611.05148*. [Online]. Available: https://arxiv.org/abs/1611.05148

[54] D. A. Reynolds, "Gaussian mixture models," in *Encyclopedia of Biometrics*. Springer, 2015, pp. 827–832.

[55] F. Riaz, S. Rehman, M. Ajmal, R. Hafiz, A. Hassan, N. R. Aljohani, R. Nawaz, R. Young, and M. Coimbra, "Gaussian mixture model based probabilistic modeling of images for medical image segmentation," *IEEE Access*, vol. 8, pp. 16846–16856, 2020.

[56] E. Min, X. Guo, Q. Liu, G. Zhang, J. Cui, and J. Long, "A survey of clustering with deep learning: From the perspective of network architecture," *IEEE Access*, vol. 6, pp. 39501–39514, 2018, doi: 10.1109/ACCESS.2018.2855437.

[57] N. Dilokthanakul, P. A. M. Mediano, M. Garnelo, M. C. H. Lee, H. Salimbeni, K. Arulkumaran, and M. Shanahan, "Deep unsupervised clustering with Gaussian mixture variational autoencoders," 2016, *arXiv:1611.02648*. [Online]. Available: https://arxiv.org/abs/1611.02648

[58] A. B. L. Larsen, S. K. Sønderby, H. Larochelle, and O. Winther, "Autoencoding beyond pixels using a learned similarity metric," in *Proc. Int. Conf. Mach. Learn.*, 2015, pp. 1558–1566.

[59] M. Rosca, B. Lakshminarayanan, D. Warde-Farley, and S. Mohamed, "Variational approaches for auto-encoding generative adversarial networks," 2017, *arXiv:1706.04987*. [Online]. Available: https://arxiv.org/abs/1706.04987

[60] R. L. Whitwell, A. D. Milner, C. Cavina-Pratesi, C. M. Byrne, and M. A. Goodale, "DF's visual brain in action: The role of tactile cues," *Neuropsychologia*, vol. 55, pp. 41–50, Mar. 2014, doi: 10.1016/j.neuropsychologia.2013.11.019.

[61] L. Jing and Y. Tian, "Self-supervised visual feature learning with deep neural networks: A survey," *IEEE Trans. Pattern Anal. Mach. Intell.*, early access, May 4, 2020, doi: 10.1109/TPAMI.2020.2992393.

[62] D. Pathak, R. Girshick, P. Dollár, T. Darrell, and B. Hariharan, "Learning features by watching objects move," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jul. 2017, pp. 2701–2710.

[63] M. Caron, P. Bojanowski, A. Joulin, and M. Douze, "Deep clustering for unsupervised learning of visual features," in *Proc. Eur. Conf. Comput. Vis.*, 2018, pp. 132–149.

[64] J. Donahue, P. Krähenbühl, and T. Darrell, "Adversarial feature learning," 2016, *arXiv:1605.09782*. [Online]. Available: http://arxiv.org/abs/1605.09782

[65] G. Lin, A. Milan, C. Shen, and I. Reid, "RefineNet: Multi-path refinement networks for high-resolution semantic segmentation," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jul. 2017, pp. 1925–1934.

[66] D. Pathak, P. Krahenbuhl, J. Donahue, T. Darrell, and A. A. Efros, "Context encoders: Feature learning by inpainting," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2016, pp. 2536–2544.

[67] Y. Xian, S. Choudhury, Y. He, B. Schiele, and Z. Akata, "Semantic projection network for zero- and few-label semantic segmentation," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2019, pp. 8248–8257.

[68] A. Krizhevsky, I. Sutskever, and G. E. Hinton, "ImageNet classification with deep convolutional neural networks," in *Proc. Adv. Neural Inf. Process. Syst. (NIPS)*, 2012, pp. 1097–1105.

[69] K. Simonyan and A. Zisserman, "Very deep convolutional networks for large-scale image recognition," 2014, *arXiv:1409.1556*. [Online]. Available: https://arxiv.org/abs/1409.1556

[70] C. Szegedy, W. Liu, Y. Jia, P. Sermanet, S. Reed, D. Anguelov, D. Erhan, V. Vanhoucke, and A. Rabinovich, "Going deeper with convolutions," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2015, pp. 1–9.

[71] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2016, pp. 770–778.

[72] G. Huang, Z. Liu, L. Van Der Maaten, and K. Q. Weinberger, "Densely connected convolutional networks," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jul. 2017, pp. 4700–4708.

[73] D. Tran, L. Bourdev, R. Fergus, L. Torresani, and M. Paluri, "Learning spatiotemporal features with 3D convolutional networks," in *Proc. IEEE Int. Conf. Comput. Vis. (ICCV)*, Dec. 2015, pp. 4489–4497.

[74] C. Ledig, L. Theis, F. Huszar, J. Caballero, A. Cunningham, A. Acosta, A. Aitken, A. Tejani, J. Totz, Z. Wang, and W. Shi, "Photo-realistic single image super-resolution using a generative adversarial network," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jul. 2017, pp. 4681–4690.

[75] O. Vinyals, A. Toshev, S. Bengio, and D. Erhan, "Show and tell: A neural image caption generator," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2015, pp. 3156–3164.

[76] E. Shelhamer, J. Long, and T. Darrell, "Fully convolutional networks for semantic segmentation," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 39, no. 4, pp. 640–651, Apr. 2017, doi: 10.1109/TPAMI.2016.2572683.

[77] R. Girshick, J. Donahue, T. Darrell, and J. Malik, "Rich feature hierarchies for accurate object detection and semantic segmentation," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, Jun. 2014, pp. 580–587.

[78] Z. Akata, F. Perronnin, Z. Harchaoui, and C. Schmid, "Label-embedding for attribute-based classification," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, Jun. 2013, pp. 819–826.

[79] E. Kodirov, T. Xiang, and S. Gong, "Semantic autoencoder for zero-shot learning," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jul. 2017, pp. 3174–3183.
[80] B. Romera-Paredes and P. H. S. Torr, "An embarrassingly simple approach to zero-shot learning," in *Proc. Int. Mach. Learn. Soc. (IMLS)*, vol. 3, Jan. 2015, pp. 2142–2151.
[81] L. Zhang, T. Xiang, and S. Gong, "Learning a deep embedding model for zero-shot learning," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jul. 2017, pp. 2021–2030.
[82] S.-C. Hsiao, D.-Y. Kao, Z.-Y. Liu, and R. Tso, "Malware image classification using one-shot learning with siamese networks," *Procedia Comput. Sci.*, vol. 159, pp. 1863–1871, Jan. 2019, doi: 10.1016/j.procs.2019.09.358.
[83] O. Vinyals, C. Blundell, T. Lillicrap, K. Kavukcuoglu, and D. Wierstra, "Matching networks for one shot learning," in *Proc. Adv. Neural Inf. Process. Syst.*, 2016, pp. 3630–3638.
[84] J. Wang and Y. Zhai, "Prototypical siamese networks for few-shot learning," in *Proc. IEEE 10th Int. Conf. Electron. Inf. Emergency Commun. (ICEIEC)*, Jul. 2020, pp. 178–181.
[85] S. Ravi and H. Larochelle, "Optimization as a model for few-shot learning," in *Proc. Int. Conf. Learn. Represent. (ICLR)*, Jan. 2019, pp. 1–11.
[86] H. Qi, M. Brown, and D. G. Lowe, "Low-shot learning with imprinted weights," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit.*, Jun. 2018, pp. 5822–5830.
[87] Y. Xian, S. Sharma, B. Schiele, and Z. Akata, "F-VAEGAN-D2: A feature generating framework for any-shot learning," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2019, pp. 10267–10276.
[88] R. Gao, X. Hou, J. Qin, J. Chen, L. Liu, F. Zhu, Z. Zhang, and L. Shao, "Zero-VAE-GAN: Generating unseen features for generalized and transductive zero-shot learning," *IEEE Trans. Image Process.*, vol. 29, pp. 3665–3680, 2020.
[89] T.-T. Wong, "Alternative prior assumptions for improving the performance of naïve Bayesian classifiers," *Data Mining Knowl. Discovery*, vol. 18, no. 2, pp. 183–213, Apr. 2009.
[90] C. Aytekin, X. Ni, F. Cricri, and E. Aksu, "Clustering and unsupervised anomaly detection with L2 normalized deep auto-encoder representations," in *Proc. Int. Joint Conf. Neural Netw. (IJCNN)*, Jul. 2018, pp. 1–6.
[91] K. V. Mardia, *Statistics of Directional Data*. New York, NY, USA: Academic, 2014.
[92] M. A. Hasnat, J. Bohné, J. Milgram, S. Gentric, and L. Chen, "von Mises-Fisher mixture model-based deep learning: Application to face verification," 2017, *arXiv:1706.04264*. [Online]. Available: https://arxiv.org/abs/1706.04264
[93] T. Diethe, "A note on the kullback-leibler divergence for the von Mises-Fisher distribution," 2015, *arXiv:1502.07104*. [Online]. Available: http://arxiv.org/abs/1502.07104
[94] T. R. Davidson, L. Falorsi, N. De Cao, T. Kipf, and J. M. Tomczak, "Hyperspherical variational auto-encoders," 2018, *arXiv:1804.00891*. [Online]. Available: https://arxiv.org/abs/1804.00891
[95] F. Locatello, S. Bauer, M. Lucic, G. Rätsch, S. Gelly, B. Schölkopf, and O. Bachem, "Challenging common assumptions in the unsupervised learning of disentangled representations," 2018, *arXiv:1811.12359*. [Online]. Available: http://arxiv.org/abs/1811.12359
[96] K. Sohn, H. Lee, and X. Yan, "Learning structured output representation using deep conditional generative models," in *Proc. Adv. Neural Inf. Process. Syst.*, 2015, pp. 3483–3491.
[97] S. Zhang, Y. Gao, Y. Jiao, J. Liu, Y. Wang, and C. Yang, "Wasserstein-wasserstein auto-encoders," 2019, *arXiv:1902.09323*. [Online]. Available: http://arxiv.org/abs/1902.09323
[98] Y. Zhu, M. R. Min, A. Kadav, and H. P. Graf, "S3 VAE: Self-supervised sequential VAE for representation disentanglement and data generation," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2020, pp. 6538–6547.
[99] J. M. Tomczak and M. Welling, "Improving variational auto-encoders using householder flow," 2016, *arXiv:1611.09630*. [Online]. Available: http://arxiv.org/abs/1611.09630
[100] J. Lucas, G. Tucker, R. B. Grosse, and M. Norouzi, "Don't blame the ELBO! A linear VAE perspective on posterior collapse," in *Proc. Adv. Neural Inf. Process. Syst.*, 2019, pp. 9408–9418.
[101] B. Dai, Z. Wang, and D. Wipf, "The usual suspects? Reassessing blame for VAE posterior collapse," 2019, *arXiv:1912.10702*. [Online]. Available: http://arxiv.org/abs/1912.10702

RUOQI WEI received the M.Sc. degree in computer science from the Department of Computer Science and Engineering, University of Bridgeport, Bridgeport, CT, USA, in 2016, where she is currently pursuing the Ph.D. degree. Her research interests include machine learning, transfer learning, few-shot learning, computer vision, and biomedical informatics.

CESAR GARCIA received the B.S. degree in general studies with a concentration in business and the M.B.A. degree from the Ernest C. Trefz School of Business, University of Bridgeport, Bridgeport, CT, USA, in 2002 and 2004, respectively, where he is currently pursuing the Ph.D. degree in computer science and engineering. He is also a Business Entrepreneur with over 100 employees.

AHMED EL-SAYED (Member, IEEE) received the B.Sc. degree in electrical engineering from the Department of Electrical Engineering, Alexandria University, Egypt, in 2003, the M.Sc. degree in engineering mathematics from the Department of Engineering Mathematics and Physics, Alexandria University, in 2006, and the M.Sc. and Ph.D. degrees in computer engineering from the Department of Computer Science and Engineering, University of Bridgeport, Bridgeport, CT, USA, in 2011 and 2016, respectively. He is currently a Postdoctoral Research Fellow with the University of Bridgeport. He published articles in the fields of robotics, soft computing, and computer vision. His research interests include robotics, AI, fuzzy systems, soft computing, machine learning, pattern recognition, and computer vision. He is a member of the Honor Society of Phi Kappa Phi and the Honor Society for Computing and Information Disciplines Upsilon Pi Epsilon (UPE).

VIYALETA PETERSON (Member, IEEE) received the B.A. degree in applied mathematics from the Department of Mathematics and Economics, University of Connecticut, Storrs, CT, USA, in 2013, and the M.Sc. degree in computer science from the Department of Computer Science and Engineering, University of Bridgeport, Bridgeport, CT, USA, in 2019. She is currently a Business Intelligence Developer with Indeed.com.

AUSIF MAHMOOD (Member, IEEE) is currently a Professor with the Department of Computer Science and Engineering. He is also the Director of the School of Engineering, University of Bridgeport. His research interests include computer vision, machine and deep learning, computer architecture, and parallel processing.