2018 IEEE International Conference on Systems, Man, and Cybernetics

Autoencoder and its various variants

Sufang ZHANG[1], Junhai ZHAI[2], Junfen CHEN[2], Qiang HE[3], Member, IEEE
[1]Hebei Branch of China Meteorological Administration Training Center, China Meteorological Administration, Baoding 071000, China
[2]Key Lab. of Machine Learning and Computational Intelligence, College of Mathematics and Information Science, Hebei University, Baoding, 071002, China
[3]School of Science, Beijing University of Civil Engineering and Architecture, Beijing, 100044, China
mczsf@126.com, mczjh@126.com, 109665826@qq.com, qianghe08@yeah.net

***Abstract*—The concept of autoencoder was originally proposed by LeCun in 1987, early works on autoencoder were used for dimensionality reduction or feature learning. Recently, with the popularity of deep learning research, autoencoder has been brought to the forefront of generative modeling. Many variants of autoencoder have been proposed by different researchers and have been successfully applied in many fields, such as computer vision, speech recognition and natural language processing. In this paper, we present a comprehensive survey on autoencoder and its various variants. Furthermore, we also present the lineage of the surveyed autoencoders. This paper can provide researchers engaged in related works with very valuable help.**

Keywords-autoencoder; decoder; deep learning; feature learning; generative model

I. INTRODUCTION

The concept of autoencoder (AE) was originally proposed by LeCun in his PhD thesis [1]. An autoencoder consists of two parts: an encoder and a decoder (see figure 1), which generally are implemented by neural networks. The encoder and decoder can be viewed as two functions $z = f(x)$ and $r = g(z)$, the $f(x)$ maps data point x from data space to feature space, while $g(z)$ produces a reconstruction of data point x by mapping z from feature space to data space. In modern autoencoders, the two functions $z = f(x)$ and $r = g(z)$ usually are stochastic functions $p_{encoder}(z \mid x)$ and $p_{decoder}(r \mid z)$, where r is the reconstruction of x. From the view of applications, it is important to note that one does not wish autoencoders to simply learn to copy of the input x. In other words, autoencoders are usually restricted in some ways that allow them to approximately learn the copy of the inputs.

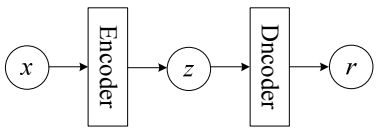

Figure 1. The structure of autoencoders

Traditionally, autoencoders were used for dimensionality reduction or feature extraction [2-4]. However, with the popularity of various deep learning models, especially, the models of generative adversarial networks [5], autoencoder has been brought to the forefront of generative modeling, many extended autoencoder models have been proposed by different researchers. The extended autoencoder models can be roughly classified into three categories: (1) Extended models based on instantiation of encoder function and decoder function. (2) Extended models based on regularization technique. (3) Extended models based on variational inference.

As far as the authors know, there is no survey on autoencoder and its variants reported in literature, accordingly in this paper, we present a comprehensive survey on autoencoder and its various variants. Furthermore, we also present the lineage of the surveyed autoencoders. This paper can provide researchers engaged in related works with very valuable help.

This paper is organized as follows. The preliminaries of autoencoder are given in Section 1. The extended models based on instantiation of encoder function and decoder function are presented Section 3. The extended models based on regularization technique are presented in Section 4. The extended models based on variational inference are given in Section 5. The trends and potential hot research topics on autoencoders are given in Section 6.

II. THE PRELIMINARIES OF AUTOENCODER

In this section, we briefly introduce the preliminaries of autoencoder. Given a training set $S = \{x_i \mid x_i \in R^d\}$, $1 \le i \le n$, the autoencoder illustrated in figure can be modeled by equation (1).

$$\begin{cases} z = f(w_e, b_e; x) \\ r = g(w_d, b_d; z) \end{cases} \tag{1}$$

where $f(\cdot)$ and $g(\cdot)$ are encoder and decoder functions which usually are implemented by neural networks. The w_e and b_e are parameters of encoder, and the w_d and b_d are parameters of decoder. If $f(\cdot)$ and $g(\cdot)$ are neural networks, then w_i and b_i are the weight matrices and the bias vectors with respect to encoder neural network and decoder neural network (i=e, d).

2577-1655/18/$31.00 ©2018 IEEE
DOI 10.1109/SMC.2018.00080

Training an autoencodor is to optimize (i.e. minimize) a predefined loss function:

$$J(\theta)=\frac{1}{n}\sum_{i=1}^{n}\left\|x_i-r_i\right\|_2^2 \tag{2}$$

where $\theta=(w_e,b_e;w_d,b_d)$. Gradient descent algorithm or stochastic gradient descent algorithm can be used for solving the above optimization problem (2).

III. EXTENDED MODELS BASED ON INSTANTIATION OF ENCODER FUNCTION AND DECODER FUNCTION

The popular extended models based on instantiation of encoder functions and decoder functions include convolutional autoencoder (CAE) [6] and extreme learning machine autoencoder (ELMAE) [7], etc.

A. Convolutional Autoencoder (CAE)

The CAE is extended from AE by instantiating encoder function and decoder function with convolutional neural networks (CNN) [8-10]. The basic building blocks of CNN are convolutional layers and pooling layer, a convolutional layer consists of multiple convolutional nodes whose inputs are 2-dimensional feature maps, the learning parameters are the elements of filter matrixes. The structure of a convolutional node can be illustrated by figure 2.

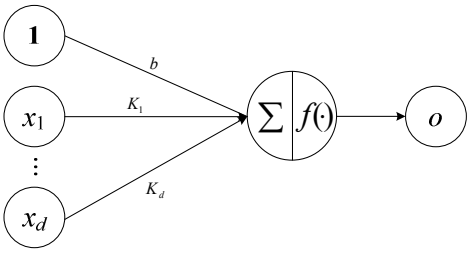

Figure 2. The structure of a convolutional node

In figure 2, the **1** is a matrix whose all elements are 1, the b is bias which is a scalar, the x_i (i=1, 2, …, d) are d input 2-dimensional feature maps which are d matrixes, K_i (i=1, 2, …, d) are d filter matrixes which are also d matrixes, but they are usually small matrixes, such as 3×3 or 5×5 matrixes, the $\sum$ is sum operator, the f is an activation function, the o is the output 2-dimensional feature map which is a matrix. The node in figure 2 can be modeled by the following equation (3).

$$o=f\left(\sum_{i=1}^{d}K_i*x_i+b\times\mathbf{1}\right) \tag{3}$$

where the symbol "*" is convolutional operator.

A pooling layer is also called sampling layer, the nodes of a pooling layer are obtained by sampling the corresponding nodes of convolutional layer. Consequently, if the number of nodes of convolutional layer is m, then the number of nodes of pooling layer is also m.

Because a CNN is a feedforward neural network, the CAE can be trained with gradient descent algorithm or stochastic gradient descent algorithm.

B. Extreme Learning Machine Autoencoder (ELMAE)

The ELMAE is extended from AE by instantiating encoder function and decoder function with ELM network [11-13]. The ELM network is a single hidden layer feedforward neural networks, its weights of input layer and hidden nodes biases are randomly generated, while its output weights are determined analytically. The structure of a ELMAE can be illustrated by figure 3.

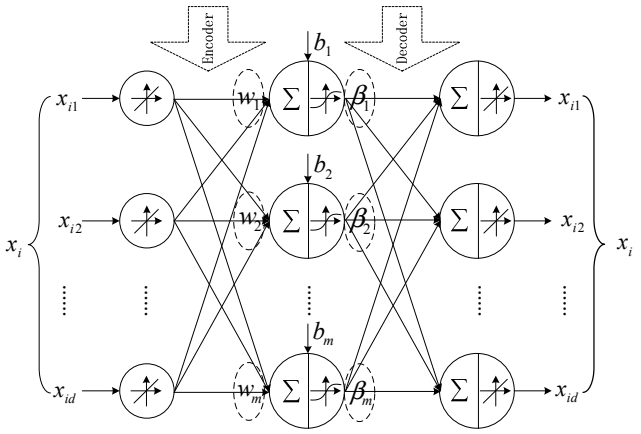

Figure 3. The structure of a ELMAE

The ELMAE illustrated in figure 3 can be modeled by $o=H\beta$, the H is the output weight matrix of hidden layer, and $H=g(WX+b)$, the g is the activation function of the ELM network, the W is the weight matrix of input layer, the elements of W are randomly assigned, the X is the data matrix, β is the weight matrix of output layer, the elements of β are determined analytically by solving the following optimization problem.

$$\min_{\beta}\left\|H\beta-Y\right\| \tag{4}$$

The smallest norm least-squares solution of (4) can be easily obtained by

$$\hat{\beta}=H^{+}Y \tag{5}$$

In (5), $H^{+}=\left(HH^{T}\right)^{-1}H$.

The ELMAE can be easily trained by ELM algorithm [11].

IV. EXTENDED MODELS BASED ON REGULARIZATION TECHNIQUE

This kind of extension is implemented by introducing regularization constraints into loss function, the constraints are usually imposed on the hidden variables z also named latent variables. The loss function (2) becomes the following one with a regularization term $R(z)$.

$$J(\theta)=\frac{1}{n}\sum_{i=1}^{n}\left\|x_i-r_i\right\|_2^2+R(z) \tag{6}$$

The representatives models of this kind extension include sparse autoencoder [4], contractive autoencoder [14, 15], information theoretic-learning autoencoder [16], etc.

A. The Sparse Autoencoder

The sparse autoencoder is extended from AE by imposing sparse regularization constraints on the hidden variable *z*, the commonly used sparse constraints include the following two ones.

(1) KL divergence sparse constraint

Given a training set $S = \{x_i \mid x_i \in R^d\}$, the encoder function transforms training instances from data space to feature space, in other words, the encoder function transforms *S* to Z, $Z = \{z_i \mid z_i \in R^m\}$, $1 \le i \le n$, n is the number of samples, and m is the dimension of feature space.

Let $\bar{z} = \frac{1}{n}\sum_{i=1}^{n} z_i$ and ρ is the sparse parameter also called response probability, which is usually a small positive real number, such as $\rho = 0.05$. The KL divergence sparse regularization item is given by the following equation (7).

$$\begin{aligned} R(z) &= \mathrm{KL}(\rho \,\|\, \bar{z}(j)) \\ &= \rho \times \log\left(\frac{\rho}{\bar{z}(j)}\right) + (1-\rho)\log\left(\frac{1-\rho}{1-\bar{z}(j)}\right) \end{aligned} \tag{7}$$

Where $\bar{z}(j)$ is the *j*th element of $\bar{z}$, $1 \le j \le m$.

(2) L_1 norm sparse constraint

L_1 norm sparse constraint is simple, which is given by the following equation (8).

$$R(z) = \frac{1}{n}\sum_{i=1}^{n} z_i = \frac{1}{n}\sum_{i=1}^{n}\sum_{j=1}^{m} |\, z_i(j)\,| \tag{8}$$

Where $z_i(j)$ is the *j*th element of z_i, $1 \le i \le n; 1 \le j \le m$.

B. Contractive Autoencoder

The contractive autoencoder is extended from AE by imposing contractive regularization constraints on the hidden variable *z*, encouraging the derivatives of $z = f(\cdot)$ to be small as possible [4]. Usually, the regularization item is given by the following equation (9).

$$R(z) = \lambda \left\| \frac{\partial f(x)}{\partial x} \right\|_{\mathrm{F}} \tag{9}$$

Where $\|\cdot\|_{\mathrm{F}}$ is the Frobenius norm (sum of squared elements) [14, 15].

C. Information Theoretic-Learning Autoencoder

Information Theoretic-Learning (ITL) is a cross field of machine learning and information theory, its objectives are to compute probability directly from samples using Parzen window density estimation method and Renyi's entropy.

Given sample set $S = \{x_i \mid x_i \in R^d\}$, it is supposed that the sample x_i is drawn from a distribution *p*, Parzen window method us the following equation (10) to estimate the density *p*.

$$\tilde{p}(x) = \frac{1}{n}\sum_{i=1}^{n} G_\sigma(x - x_i) \tag{10}$$

Where $G_\sigma(\cdot)$ is a Gaussian kernel function with size parameter σ.

Renyi's quadratic entropy for probability density *p* is given by the following equation (11).

$$\begin{aligned} H_2(x) &= -\log \int p^2(x)dx \\ &= -\log \int \left(\frac{1}{n}\sum_{i=1}^{n} G_\sigma(x - x_i)\right)^2 dx \\ &= -\log \int \frac{1}{n^2}\left(\sum_{i=1}^{n}\sum_{j=1}^{n} G_{\sigma\sqrt{2}}(x_j - x_i)\right) \end{aligned} \tag{11}$$

Where $G_\sigma(x, y) = \frac{1}{\sqrt{2\pi}\sigma}\exp\left(\frac{\|x-y\|^2}{2\sigma^2}\right)$ is the Gaussian kernel. The Renyi's cross-entropy can be estimated by the following equation (12).

$$\tilde{H}(x, y) = -\log \frac{1}{n_x n_y}\sum_{i=1}^{n_x}\sum_{j=1}^{n_y} G_{\sqrt{2}\sigma}(x_i - y_j) \tag{12}$$

Where n_x and n_x are the size of sample *x* and *y*.

Information theoretic-learning autoencoder is extended from AE by introducing regularization constraint $R(z)$ into loss function of AE. $R(z)$ is defined by (12), in [16], the authors also define $R(z)$ with other information theory measures.

V. Extended Models Based on Variational Inference

This kind of extension is implemented by introducing variational inference into AE, it is a hot research topic recently in the field of autoencoder. The pioneering work of this kind extension is the so called variational autoencoder (VAE) proposed by Kingma et al. [17]. The advances and trends of variational inference can be found in two excellent survey papers [18] and [19]. Another remarkable work of this kind extension is adversarial autoencoder (AAE) which introduce the training mechanism of generative adversarial networks (GANs) into AE [20, 21]. The advances and trends of GANs can be found in [22] and [23]. GAN is a very hot research topic recently in the field of deep learning [4, 8, 24].

A. Variational AutoEncoder (VAE)

VAE is extended from AE by introducing variational inference, its goal is to approximate $p_\theta(x)$ by introducing a variational lower bound. Given the latent variable z, VAE attempts to find the model parameter θ by maximum likelihood method. Due to the latent variable z affecting data x, maximum a posteriori with a prior knowledge of z must be considered instead of maximum likelihood [23]. Specifically, VAE estimates posteriori probability $p(z \mid x)$ with an assumption of a prior knowledge $p(z)$ being a normal Gaussian distribution and drives the approximating model $Q_\phi(z \mid x)$ to approximate real posteriori probability $p(z \mid x)$. A variational lower bound of the marginal log-likelihood $\log p_\theta(x)$ can be derived as follows:

$$\begin{aligned}
\log p_\theta(x) &= \int_z Q_\phi(z \mid x) \log p_\theta(x) dz \\
&= \int_z Q_\phi(z \mid x) \log \left(\frac{p_\theta(x,z)}{p_\theta(z \mid x)} \frac{Q_\phi(z \mid x)}{Q_\phi(z \mid x)} \right) dz \\
&= \int_z Q_\phi(z \mid x) \left(\log \left(\frac{Q_\phi(z \mid x)}{p_\theta(z \mid x)} \right) + \log \left(\frac{p_\theta(x,z)}{Q_\phi(z \mid x)} \right) \right) dz \\
&= \mathrm{KL}\left(Q_\phi(z \mid x) \,\|\, p_\theta(z \mid x)\right) + \mathrm{E}_{Q_\phi(z|x)} \left[\frac{\log p_\theta(x,z)}{\log Q_\phi(z \mid x)} \right]
\end{aligned}$$

Since $\mathrm{KL}\left(Q_\phi(z \mid x) \,\|\, p_\theta(z \mid x)\right) \geq 0$, variational lower bound $L(\theta, \phi; x)$ can be formulated as

$$\begin{aligned}
\log p_\theta(x) &\geq E_{Q_\phi(z|x)}[\log p_\theta(x,z) - \log Q_\phi(z \mid x)] \\
&= E_{Q_\phi(z|x)}[\log p_\theta(z) - \log p_\theta(x \mid z) - \log Q_\phi(z \mid x)] \\
&= -KL(Q_\phi(z \mid x) \,\|\, p_\theta(z)) + E_{Q_\phi(z|x)}[\log p_\theta(z \mid x)] \\
&= L(\theta, \phi; x)
\end{aligned}$$

Intuitively, $Q_\phi(z \mid x)$ is an encoder which generates the latent variable z given sample x; while $p_\theta(x \mid z)$ is a decoder which generates samples x given latent variable z.

B. Adversarial AutoEncoder (AAE)

The AAE combines AE and GAN (Generative Adversarial Networks) [21], different from VAE, AAE uses GAN to perform variational inference. GAN is a two-player game model (see figure 4). In figure 4, the generator denoted by G in short takes a random noise z as input and produce a sample $G(z)$, while the discriminator denoted by D in short identifies whether a sample comes from the true data distribution $p(x)$ or the generator. The models G and D needs simultaneously training, the training goal is that the model G captures the data distribution as precise as possible, and the model D correctly estimates the probability which indicates the possibility of a sample comes from true training data rather than comes from imitated data generated by G. The GANs can be modeled by the following minimax optimization problem:

$$\min_G \max_D V(G,D) = E_{x \sim p_{data}(x)}[\log D(x)] + E_{z \sim p(z)}[\log(1 - D(G(z)))]$$

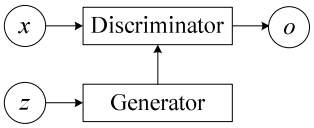

Figure 4. The structure of GANs

Actually, VAEs and GANs are all generative models, and both models are based on maximum likelihood, the difference between VAEs and GANs can be characterized by the following figure 5.

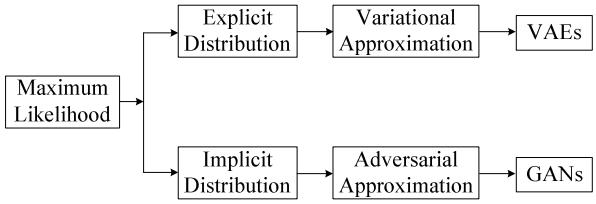

Figure 5. The difference between VAEs and GANs

The structure of an AAE is given in figure 6. The top row is a standard autoencoder, while the bottom row diagrams another model (part of GANs) trained to discriminatively predict whether a sample comes from the hidden code of the autoencoder or from a data distribution. From the view point of GANs, the autoencoder in figure 6 act as the role of generator.

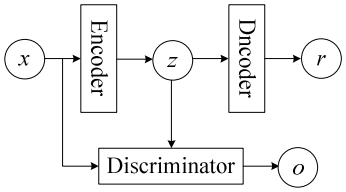

Figure 5. The structure of AAE

Regarding to the training and applications of AAE, the interested readers may refer to reference [20]. Based on AAE, Mescheder et al. [25] proposed adversarial variational Bayes model, which unify variational autoencoders and generative adversarial networks. Pu et al. [26] proposed adversarial symmetric variational autoencoder. Creswell and Bharath [27] proposed denoising adversarial autoencoders. Follow this technical route, Tolstikhin et al. [28] combine VAE and Wasserstein GAN and proposed Wasserstein autoencoders. Burda et al. [29] proposed importance weighted autoencoders, which have same architecture with VAE, but use a strictly tighter log-likelihood lower bound derived from importance weighting. Khoshaman et al. [30] proposed quantum variational autoencoder whose latent generative process is implemented as a quantum Boltzmann machine. Based on the observation that discrete data can be represented as a parse tree from a context-free grammar, Kusner et al. [31] extended VAE from continuous data to discrete data, and proposed grammar variational autoencoder. Based on the Stein's operator, Pu et al. [32] proposed stein variational autoencoder. Based on replacing instance-specic local inference with a global inference

network, Cremer et al. [33] proposed amortized variational autoencoders. From these works, we can find that in recent years, improved AAEs or extended AAEs are hot research topic in the field of deep learning.

VI. Conclusion

This paper presents a comprehensive survey on autoencoder and its various variants. We mainly focus on 7 variants: convolutional autoencoder, extreme learning machine autoencoder, sparse autoencoder, contractive autoencoder, information theoretic-learning autoencoder, variational autoencoder, adversarial autoencoder. We analyzed their characteristics and training. From this paper, potential authors can find the lineage of the surveyed autoencoders. The authors think that future trends or hot research topics of autoencoders include the following three aspects:

(1) Autoencoders based on multiple-player (more than two) GANs, and their adversarial inference and training;

(2) Generating samples with autoencoder or its variants for imbalanced learning.

(3) The extensions of autoencoder or its variants in big data scenarios.

Acknowledgment

This research is supported by the natural science foundation of Hebei Province (F2017201026, F2016201161), by the National Natural Science Foundation of China under Grants 61473111, by the natural science foundation of Hebei University (799207217071), and Research Foundation of Beijing University of Civil Engineering and Architecture (KYJJ2017017).

References

[1] Y. LeCun. Connexionist learning models. PhD thesis, Universite Pierre et Marie Curie (Paris), 1987.

[2] G. E. Hinton, R. S. Zemel. Autoencoders, minimum description length, and Helmholtz free energy. Advances in Neural Information Processing Systems 6. J. D. Cowan, G. Tesauro and J. Alspector (Eds.), Morgan Kaufmann: San Mateo, CA.

[3] H. Bourlard, Y. Kamp. Auto-association by multiplayer perceptrons and singular value decomposition. Biological Cybernetics, 1988, 59:291-294.

[4] I. Goodfellow, Y. Bengio, A. Courville. Deep Learning. MIT Press, 2016.

[5] I. Goodfellow, J. Pouget-Abadie, M. Mirza, et al. Generative adversarial nets. Advances in Neural Information Processing Systems, 2014, vol. 1, pp. 2672-2680.

[6] J. Masci, U. Meier, D. Cireşan, et al. Stacked convolutional auto-encoders for hierarchical feature extraction. International Conference on Artificial Neural Networks. Springer-Verlag, 2011:52-59.

[7] L. L. C. Kasun, H. Zhou, G. B. Huang, et al. Representational Learning with Extreme Learning Machine for Big Data. IEEE Intelligent Systems, 2013, 28(6):31-34.

[8] Y. Lecun, Y. Bengio, G. E. Hinton. Deep Learning. Nature, 2015,521:436-444.

[9] Y. Lecun, L. Bottou, Y. Bengio, et al. Gradient-based learning applied to document recognition. Proceedings of the IEEE, 1998, 86(11):2278-2324.

[10] A. Krizhevsky, I. Sutskever, G. E. Hinton G E. ImageNet classification with deep convolutional neural networks. International Conference on Neural Information Processing Systems. Curran Associates Inc. 2012:1097-1105.

[11] G. B. Huang, Q. Y. Zhu, C. K. Siew. Extreme learning machine: Theory and applications, Neurocomputing, 2006, 70:489-501.

[12] G. B. Huang, D. H. Wang, Y. Lan. Extreme learning machines: a survey. International Journal of Machine Learning and Cybernetics, 2011, 2(2):107-122.

[13] G. Huang, G. B. Huang, S. Song, K. You. Trends in extreme learning machines: A review. Neural Networks, 2015, 61:32-48.

[14] S. Rifai, P. Vincent, X. Muller, et al. Contractive auto-encoders: explicit invariance during feature extraction. Proceedings of the 28 th International Conference on Machine Learning, Bellevue, WA, USA, 2011, 524-611.

[15] S. Rifai, G. Mesnil, P. Vincent, et al. Higher order contractive auto-encoder. European Conference on Machine Learning and Knowledge Discovery in Databases. Springer-Verlag, 2011:645-660.

[16] E. Santana, M. Emigh, J. C. Principe. Information theoretic-learning auto-encoder. International Joint Conference on Neural Networks. IEEE, 2016:3296-3301.

[17] D. P. Kingma and M. Welling. Auto-encoding variational bayes. Proceedings of the 2nd International Conference on Learning Representations (ICLR), 2014.

[18] D. M. Blei，A. Kucukelbir，JD Mcauliffe. Variational inference: a review for statisticians. Journal of the American Statistical Association, 2017, 112(518):859-877.

[19] C. Zhang, J. Butepage, H. Kjellstrom, et al. Advances in variational inference. http://arxiv.org/abs/1711.05597, 2017.

[20] A. Makhzani, J. Shlens, N. Jaitly, et al. Adversarial autoencoders. International Conference on Learning Representations, http://arxiv.org/abs/1511.05644, 2016.

[21] I. Goodfellow, J. Pouget-Abadie, M. Mirza, B. Xu, et al. Generative adversarial nets. Advances in Neural Information Processing Systems, 2014, pp.2672-2680.

[22] A. Creswell, T. White, V. Dumoulin, et al. Generative adversarial networks: an overview. IEEE Signal Processing Magazine, 2018, 35(1):53-65.

[23] Y. J. Hong, U. Hwang, J. Yoo J, et al. How generative adversarial networks and its variants work: an overview of GAN. http://arxiv.org/abs/1711.05914v4, 2017.

[24] G. E. Hinton, R. R. Salakhutdinov. Reducing the Dimensionality of data with neural networks. Science, 2006, 313(5786):504-507.

[25] L. M. Mescheder, S. Nowozin, and A. Geiger. Adversarial variational bayes: Unifying variational autoencoders and generative adversarial networks. http://arxiv.org/abs/1701.04722, 2017.

[26] Y. Pu, W. Wang, R. Henao, et al. Adversarial Symmetric Variational Autoencoder. http://arxiv.org/abs/1711.04915v2, 2017.

[27] A. Creswell, A. A. Bharath. Denoising Adversarial Autoencoders. http://arxiv.org/abs/1703.01220V4, 2018.

[28] I. Tolstikhin, O. Bousquet, S. Gelly, et al. Wasserstein Auto-Encoders. http://arxiv.org/abs/1711.01558, 2017.

[29] Y. Burda, R. rosse, R. Salakhutdinov. Importance Weighted Autoencoders. http://arxiv.org/abs/1509.00519v4, 2016.

[30] A. Khoshaman, W. Vinci, B. Denis, et al. Quantum Variational Autoencoder. http://arxiv.org/abs/1802.05779, 2018.

[31] M. J. Kusner, B. Paige, J. M. Hernández-Lobato. Grammar variational autoencoder. Proceedings of the 34th International Conference on Machine Learning, Sydney, Australia, PMLR 70, 2017.

[32] Y. C. Pu, Z. Gan, R. Henao, et al. VAE Learning via Stein Variational Gradient Descent. http://arxiv.org/abs/1704.05155.

[33] L. Cremer, X. Li, D. Duvenaud. Inference suboptimality in variational autoencoders. http://arxiv.org/abs/1801.03558, 2018.