acrt AI, Computer Science and Robotics Technology

IntechOpen Journals

RESEARCH PAPER

Adversarial Variational Autoencoders to Extend and Improve Generative Model

Loc Nguyen[1,*], Hassan I. Abdalla[2] and Ali A. Amer[2]

1 Loc Nguyen's Academic Network, Vietnam
2 College of Technological Innovation, Zayed University, Abu Dhabi, UAE
*Corresponding author. E-mail: ng_phloc@yahoo.com; URL: www.locnguyen.net

Citation
Loc Nguyen, Hassan I. Abdalla and Ali A. Amer (2024), Adversarial Variational Autoencoders to Extend and Improve Generative Model. *AI, Computer Science and Robotics Technology* 3(1), 1–19.

DOI
https://doi.org/10.5772/acrt.20240003

Copyright
© The Author(s) 2024.

This is an Open Access article distributed under the terms of the Creative Commons Attribution License (https://creativecommons.org/licenses/by/4.0/), which permits unrestricted reuse, distribution, and reproduction in any medium, provided the original work is properly cited.

Received: 25 January 2024
Accepted: 29 July 2024
Published: 28 August 2024

Abstract

Generative artificial intelligence (GenAI) has been advancing with many notable achievements like ChatGPT and Bard. The deep generative model (DGM) is a branch of GenAI, which is preeminent in generating raster data such as image and sound due to the strong role of deep neural networks (DNNs) in inference and recognition. The built-in inference mechanism of DNN, which simulates and aims at synaptic plasticity of the human neuron network, fosters the generation ability of DGM, which produces surprising results with the support of statistical flexibility. Two popular approaches in DGM are the variational autoencoder (VAE) and generative adversarial network (GAN). Both VAE and GAN have their own strong points although they share and imply the underlying theory of statistics as well as significant complex via hidden layers of DNN when DNN becomes effective encoding/decoding functions without concrete specifications. This research unifies VAE and GAN into a consistent and consolidated model called the adversarial variational autoencoder (AVA) in which the VAE and GAN complement each other; for instance, the VAE is a good data generator by encoding data via the excellent ideology of Kullback–Leibler divergence and the GAN is a significantly important method to assess the reliability of data as to whether it is real or fake. In other words, the AVA aims to improve the accuracy of generative models; besides, the AVA extends the function of simple generative models. In methodology, this research focuses on the combination of applied mathematical concepts and skillful

techniques of computer programming in order to implement and solve complicated problems as simply as possible.

Keywords: deep generative model (DGM), variational autoencoder (VAE), generative adversarial network (GAN)

1. *Introduction*

The variational autoencoder (VAE) and the generative adversarial network (GAN) are two popular approaches for developing a deep generative model (DGM) [1] with the support of a deep neural network (DNN). The high capacity of DNN contributes significantly to the success of GAN and VAE. Some works have combined the VAE and the GAN. Larsen *et al.* [2] proposed a traditional combination of VAE and GAN by considering a decoder of VAE as a generator of GAN [2, p. 1558]. They constructed the target optimization function as the sum of the likelihood function of VAE and the target function of GAN [2, p. 1560]. This research is similar to theirs [2, p. 1561] except that the construction optimization function is slightly different. The construction optimization function in this research does not include the target function of GAN according to the traditional approach of GAN. However, uncorrelated variables are removed after gradients are determined. Moreover, because the encoded data $\mathbf{z}$ is basically randomized, this work does not construct a new random $\mathbf{z}'$ to included in the target function of GAN. This study also mentions skillful techniques of derivatives in a backpropagation algorithm.

Mescheder *et al.* [3] transformed the gain function of VAE including Kullback–Leibler divergence into the gain function of GAN via a so-called real-valued discrimination network [3, p. 2394] related to the Nash equilibrium equation and the sigmoid function. Then they trained the transformed VAE by the stochastic gradient descent (SGD) method. They estimated three parameters [3, p. 2395] as in this research, but their method focused on mathematical transformation while this work focuses on skillful techniques in implementation. In other words, Mescheder *et al.* [3] tried to fuse VAE into GAN whereas this work combines them in a mutual and balancing manner; but both studies try to unify VAE and GAN. Rosca *et al.* [4, p. 4] used a density ratio trick to convert the Kullback–Leibler divergence of VAE into the mathematical form $\log(x/(1-x))$, which is similar to the GAN target function $\log(x) + \log(1-x)$. Actually, they carried out a fusion of VAE and GAN like Mescheder *et al.* did. The essence of their methods is based on the convergence of the Nash equilibrium equation.

Ahmad *et al.* [5] combined VAE and GAN separately as did previous experimental research. First, they trained VAE and swapped the encoder–decoder network to a decoder–encoder network so that the output of VAE is transformed into some useful information, which in turn becomes the input of GAN instead of random information that is the usual input [5, p. 6]. Miolane *et al.* [6] combined VAE and GAN by summing the target functions of VAE and GAN weighted with regular hyperparameters [6, p. 974]. Later, they first trained VAE and then sent the output of VAE to the input of GAN [6, p. 975]. Ding *et al.* [7] proposed an interesting research that applies VAE and GAN to credit card fraud detection. The main point of their research is that because small fraud data is not enough to train well-supervised learning models like classification and discriminant analysis in a better way, VAE is applied to generate pseudo training data so that GAN will be trained well based on such sufficiently large training data in order to obtain a better discrimination function for detecting credit card fraud. In their VAEGAN model [7, p. 83,682], online credit data (original data) is fed into the VAE encoder to train the VAE decoder as a generator. Then the generator is used to generate fake data so that such fake data and the real data are integrated into sufficiently large data, which is used to train a GAN discriminator. As a result, such a trained discriminator is applied to detect credit card fraud.

In general, both VAE and GAN have their own strong points. For instance, they not only take advantage of solid statistical theory as well as DNN but they also suffer from drawbacks. For example, VAE does not have a mechanism to distinguish fake data from real data and GAN does not handle explicitly probabilistic distribution of encoded data. It is better to utilize their strong points and alleviate their weak points. Therefore, this research focuses on incorporating GAN into VAE by skillful techniques related to both SGD and software engineering architecture, which are neither based on purely mathematical fusion nor on experimental tasks. In practice, many complex mathematical problems can be solved effectively by some skillful techniques of computer programming. Moreover, the proposed model called adversarial variational autoencoder (AVA) aims to extend functions of VAE and GAN as a general architecture for the generative model. For instance, AVA will provide an encoding function that GAN does not possess and a discrimination function that VAE needs to distinguish fake data from real data. The combination of VAE and GAN into AVA is strengthened by a regular and balance mechanism, which obviously is natural and like the fusion mechanism. In some cases, it is better than the fusion mechanism because both built-in VAE and GAN inside AVA can retain their own strong features. Therefore, the experiment in this work is not very significant regarding large data when only AVA, VAE, and GAN are compared within a small dataset, which aims to prove the proposed method mentioned in the next section.

2. Methodology

This research proposes a method as well as a generative model that incorporates GAN into VAE for extending and improving the DGM because GAN does not deal with the coding of original data and VAE lacks mechanisms to assess the quality of generated data. Note that data coding is necessary for some essential applications such as image compression and recognition whereas auditing quality can improve the accuracy of generated data. As convention, let vector variables $\boldsymbol{x} = (x_1, x_2, \dots, x_m)^T$ and $\mathbf{z} = (z_1, z_2, \dots, z_n)^T$ be the original data and encoded data whose dimensions are m and n $(m > n)$, respectively. A generative model is represented by a function $f(\boldsymbol{x}|\Theta) = \mathbf{z}, f(\boldsymbol{x}|\Theta) \approx \mathbf{z}$, or $f(\boldsymbol{x}|\Theta) \to \mathbf{z}$, where $f(\boldsymbol{x}|\Theta)$ is implemented by a DNN whose weights are Θ, which converts the original data $\boldsymbol{x}$ to the encoded data $\mathbf{z}$ and is called an encoder in VAE. A decoder in VAE that converts expectedly the encoded data $\mathbf{z}$ back to the original data $\boldsymbol{x}$ is represented by a function $g(\mathbf{z}|\Phi) = \boldsymbol{x}'$, where $g(\mathbf{z}|\Phi)$ is also implemented by a DNN whose weights are Φ with the expectation that the decoded data $\boldsymbol{x}'$ is approximated to the original data $\boldsymbol{x}$ as $\boldsymbol{x}' \approx \boldsymbol{x}$. The essence of VAE developed by Kingma and Welling [8] is to minimize the following loss function for estimating the encoded parameter Θ and the decoded parameter Φ:

$$l_{\text{VAE}}(\Theta, \Phi) = \tfrac{1}{2}\|\boldsymbol{x} - \boldsymbol{x}'\|^2 + \text{KL}(\mu(\boldsymbol{x}), \Sigma(\boldsymbol{x})|N(\mathbf{0}, I)) \tag{1}$$

such that

$$\Theta^* = \underset{\Phi}{\operatorname{argmin}}\, \text{KL}(\mu(\boldsymbol{x}), \Sigma(\boldsymbol{x})|N(0, I))$$
$$\Phi^* = \underset{\Theta}{\operatorname{argmin}}\, \tfrac{1}{2}\|\boldsymbol{x} - \boldsymbol{x}'\|^2.$$

Note that $\|\boldsymbol{x} - \boldsymbol{x}'\|$ is the Euclidean distance between $\boldsymbol{x}$ and $\boldsymbol{x}'$ whereas KL$(\mu(\boldsymbol{x}), \Sigma(\boldsymbol{x})|N(\mathbf{0}, I))$, is the Kullback–Leibler divergence between the Gaussian distribution of $\boldsymbol{x}$ whose mean vector and covariance matrix are $\mu(\boldsymbol{x})$ and $\Sigma(\boldsymbol{x})$, respectively. The standard Gaussian distribution $N(\mathbf{0}, I)$ has the mean vector and covariance matrix $\mathbf{0}$ and identity matrix I, respectively.

The GAN developed by Goodfellow *et al.* [9] does not act on the encoder $f(\boldsymbol{x}|\Theta) = \mathbf{z}$ but it focuses on optimizing the decoder $g(\mathbf{z}|\Phi) = \boldsymbol{x}'$ by introducing a so-called discriminator, which is a discrimination function $d(\boldsymbol{x}|\Psi): \boldsymbol{x} \to [0, 1]$ from the considered data $\boldsymbol{x}$ or $\boldsymbol{x}'$ to range [0,1] in which $d(\boldsymbol{x}|\Psi)$ can distinguish fake data from real data. In other words, the larger the result the discriminator $d(\boldsymbol{x}'|\Psi)$ derives, the more real the generated data $\boldsymbol{x}'$. Obviously, $d(\boldsymbol{x}|\Psi)$ is implemented by a DNN whose weights are Ψ noting that this DNN has only one output neuron denoted by d_0. The essence of GAN is to optimize mutually the following target function for estimating the decoder parameter Φ and the discriminator parameter Ψ [9, p. 3]:

$$b_{\text{GAN}}(\Phi, \Psi) = \log(d(\boldsymbol{x}|\Psi)) + \log(1 - d(g(\boldsymbol{z}|\Phi)|\Psi)) \tag{2}$$

such that Φ and Ψ are optimized mutually as follows:

$$\begin{aligned}\Phi^* &= \underset{\Phi}{\operatorname{argmin}}\, b_{\text{GAN}}(\Phi, \Psi^*)\\ \Psi^* &= \underset{\Psi}{\operatorname{argmax}}\, b_{\text{GAN}}(\Phi^*, \Psi).\end{aligned}$$

The proposed generative model in this research is called adversarial variational autoencoder because it combines VAE and GAN by the fusing mechanism in which the loss function and the balance function are optimized parallelly. The AVA loss function implies loss information in encoder $f(\boldsymbol{x}|\Theta)$, decoder $g(\mathbf{z}|\Phi)$, and discriminator $d(\boldsymbol{x}|\Psi)$ as follows:

$$l_{\text{AVA}}(\Theta, \Phi, \Psi) = \tfrac{1}{2}\|\boldsymbol{x} - \boldsymbol{x}'\|^2 + \text{KL}(\mu(\boldsymbol{x}), \Sigma(\boldsymbol{x})|N(0, I)) + \log(1 - d(g(\boldsymbol{z}|\Phi)|\Psi)). \quad (3)$$

The balance function of AVA is to supervise the decoding mechanism, which is the GAN target function as follows:

$$b_{\text{AVA}}(\Phi, \Psi) = b_{\text{GAN}}(\Phi, \Psi) = \log(d(\boldsymbol{x}|\Psi)) + \log(1 - d(g(\boldsymbol{z}|\Phi)|\Psi)). \quad (4)$$

The key point of AVA is that the discriminator function occurs in both the loss function and the balance function via the expression $\log(1 - d(g(\mathbf{z}|\Phi)|\Psi))$, which means that the capacity of how to distinguish fake data from real data by the discriminator function affects the decoder DNN. As a result, the three parameters Θ, Φ, and Ψ are optimized mutually according to both the loss function and the balance function as follows:

$$\begin{aligned}\Theta^* &= \underset{\Theta}{\operatorname{argmin}}\, l_{\text{AVA}}(\Theta, \Phi^*, \Psi^*)\\ \Phi^* &= \underset{\Phi}{\operatorname{argmin}}\, l_{\text{AVA}}(\Theta^*, \Phi, \Psi^*)\\ \Psi^* &= \underset{\Psi}{\operatorname{argmax}}\, b_{\text{AVA}}(\Phi^*, \Psi).\end{aligned}$$

Because the encoder parameter Θ is independent of both the decoder parameter Φ and the discriminator parameter Ψ, its estimate is specified as follows:

$$\Theta^* = \underset{\Theta}{\operatorname{argmin}}(\text{KL}(\mu(\boldsymbol{x}), \Sigma(\boldsymbol{x})|N(0, I))).$$

Because the decoder parameter Φ is independent of the encoder parameter Θ, its estimate is specified as follows:

$$\Phi^* = \underset{\Phi}{\operatorname{argmin}}\left(\tfrac{1}{2}\|\boldsymbol{x} - \boldsymbol{x}'\|^2 + \log(1 - d(g(\boldsymbol{z}|\Phi)|\Psi^*))\right).$$

Note that the Euclidean distance $\|\boldsymbol{x} - \boldsymbol{x}'\|$ is only dependent on Φ. Because the discriminator tries to increase the credible degree of real data and decrease the credible degree of fake data, its parameter Ψ has the following estimate:

$$\Psi^* = \underset{\Psi}{\operatorname{argmax}}(\log(d(\boldsymbol{x}|\Psi)) + \log(1 - d(g(\boldsymbol{z}|\Phi^*)|\Psi))).$$

By applying the SGD algorithm to the backpropagation algorithm, these estimates are determined based on gradients of the loss function and the balance function as follows:

$$\begin{aligned}
\Theta &= \Theta - \gamma\nabla_\Theta(\mathrm{KL}(\mu(\boldsymbol{x}), \Sigma(\boldsymbol{x})|N(0, I)))\\
\Phi &= \Phi - \gamma\nabla_\Phi\left(\tfrac{1}{2}\|\boldsymbol{x} - \boldsymbol{x}'\|^2 + \log(1 - d(g(\boldsymbol{z}|\Phi)|\Psi^*))\right)\\
\Psi &= \Psi + \gamma\nabla_\Psi(\log(d(\boldsymbol{x}|\Psi)) + \log(1 - d(g(\boldsymbol{z}|\Phi^*)|\Psi))),
\end{aligned}$$

where γ $(0 < \gamma \le 1)$ is the learning rate. Let $a_f(.)$, $a_g(.)$, and $a_d(.)$ be activation functions of encoder DNN, decoder DNN, and discriminator DNN, respectively, and so, let $a'_f(.)$, $a'_g(.)$, and $a'_d(.)$ be derivatives of these activation functions, respectively. The encoder gradient regarding Θ is ([8, p. 5], [10, p. 9], [11, p. 43])

$$\nabla_\Theta(\mathrm{KL}(\mu(\boldsymbol{x}), \Sigma(\boldsymbol{x})|N(0, I))) = \left(\mu(\boldsymbol{x}) - \tfrac{1}{2}(\Sigma(\boldsymbol{x}))^{-1} + \tfrac{1}{2}I\right) a'_f(\boldsymbol{x}).$$

The decoder gradient regarding Φ is

$$\nabla_\Phi\left(\tfrac{1}{2}\|\boldsymbol{x} - \boldsymbol{x}'\|^2 + \log(1 - d(g(\boldsymbol{z}|\Phi)|\Psi^*))\right) = -\left((\boldsymbol{x} - \boldsymbol{x}') + \frac{a'_d(d(\boldsymbol{x}'|\Psi^*))}{1 - d(\boldsymbol{x}'|\Psi^*)}\right) a'_g(\boldsymbol{x}'),$$

where

$$g(\boldsymbol{z}|\Phi) \cong g(\boldsymbol{z}|\Phi^*) = \boldsymbol{x}'.$$

The discriminator gradient regarding Ψ is

$$\nabla_\Psi(\log(d(\boldsymbol{x}|\Psi)) + \log(1 - d(\boldsymbol{x}'|\Psi))) = \frac{a'_d(d(\boldsymbol{x}|\Psi))}{d(\boldsymbol{x}|\Psi)} - \frac{a'_d(d(\boldsymbol{x}'|\Psi))}{1 - d(\boldsymbol{x}'|\Psi)}.$$

As a result, the SGD algorithm incorporated into the backpropagation algorithm for solving AVA is totally determined as follows:

$$\Theta = \Theta - \gamma\left(\mu(\boldsymbol{x}) - \tfrac{1}{2}(\Sigma(\boldsymbol{x}))^{-1} + \tfrac{1}{2}I\right) a'_f(\boldsymbol{x}) \tag{5}$$

$$\Phi[i] = \Phi[i] + \gamma\left((\boldsymbol{x}[i] - \boldsymbol{x}'[i]) + \frac{a'_d(d(\boldsymbol{x}'|\Psi^*))}{1 - d(\boldsymbol{x}'|\Psi^*)}\right) a'_g(\boldsymbol{x}'[i]) \tag{6}$$

$$\Psi = \Psi + \gamma\left(\frac{a'_d(d(\boldsymbol{x}|\Psi))}{d(\boldsymbol{x}|\Psi)} - \frac{a'_d(d(\boldsymbol{x}'|\Psi))}{1 - d(\boldsymbol{x}'|\Psi)}\right), \tag{7}$$

where notation $[i]$ denotes the ith element in the vector. Note the derivatives $a'_f(.)$, $a'_g(.)$, and $a'_d(.)$ because they are helpful techniques to consolidate AVA. The reason for two different occurrences of derivatives $a'_d(d(\boldsymbol{x}'|\Psi^*))$ and $a'_g(\boldsymbol{x}')$ in the decoder gradient regarding Φ is nontrivial because the unique output neuron of the discriminator DNN is considered the effect of the output layer of all output neurons in the decoder DNN.

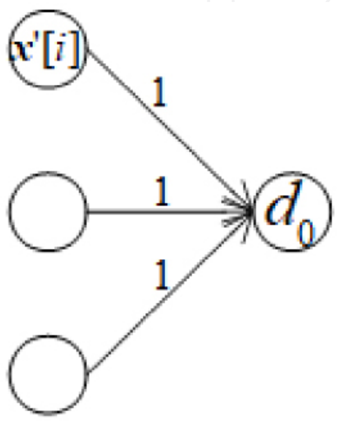

Figure 1. Causality–effect relationship between decoder DNN and discriminator DNN.

When weights are assumed to be 1, the error of the causal decoder neuron is the error of the discriminator neuron multiplied by the derivative at the decoder neuron. Moreover, the error of the discriminator neuron, in turn, is the product of its minus bias $-d'(.)$ and its derivative $a'_d(.)$, where $d'(.)$ is the derivative of the discriminator, shown in Figure 1.

$$\text{error}(\boldsymbol{x}'[i]) = 1 * \text{error}(d_0)a'_g(\boldsymbol{x}'[i])$$
$$\text{error}(d_0) = -d'(d_0)a'_d(d_0).$$

It is necessary to describe AVA architecture because skillful techniques cannot be applied to AVA without clear and solid architecture. The key point to incorporate GAN into VAE is that the error $\frac{a'_d(d(\boldsymbol{x}'|\Psi^*))}{1-d(\boldsymbol{x}'|\Psi^*)}$ of generated data is included in both the decoder and the discriminator besides the decoded data $\boldsymbol{x}'$, which is the output of the decoder DNN and which becomes the input of the discriminator DNN:

$$\Phi[i] = \Phi[i] + \gamma\left((\boldsymbol{x}[i] - \boldsymbol{x}'[i]) + \frac{a'_d(d(\boldsymbol{x}'|\Psi^*))}{1 - d(\boldsymbol{x}'|\Psi^*)}\right)a'_g(\boldsymbol{x}'[i])$$
$$\Psi = \Psi + \gamma\left(\frac{a'_d(d(\boldsymbol{x}|\Psi))}{d(\boldsymbol{x}|\Psi)} - \frac{a'_d(d(\boldsymbol{x}'|\Psi))}{1 - d(\boldsymbol{x}'|\Psi)}\right).$$

Figure 2 shows the AVA architecture.

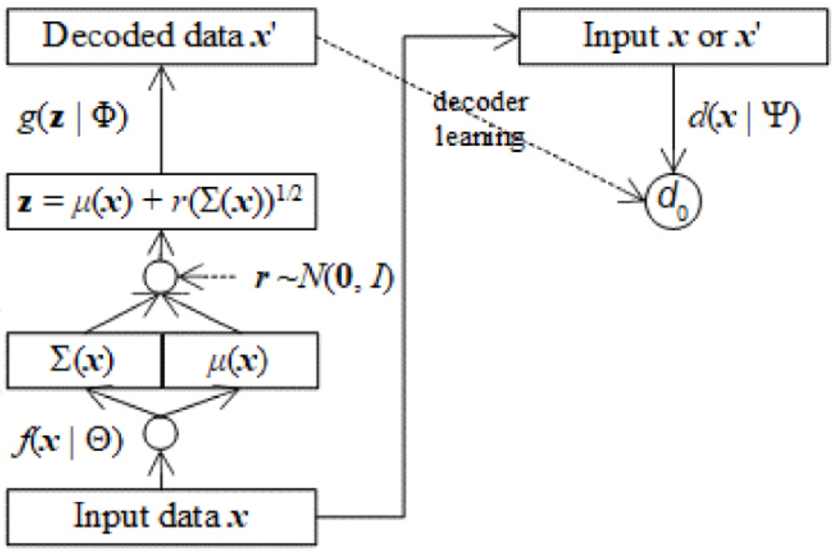

Figure 2. AVA architecture.

The AVA architecture follows an important aspect of VAE where the encoder $f(\boldsymbol{x}|\Theta)$ does not produce directly decoded data **z** as $f(\boldsymbol{x}|\Theta) = \mathbf{z}$. It actually produces the mean vector $\mu(\boldsymbol{x})$ and the covariance matrix $\Sigma(\boldsymbol{x})$ belonging to $\boldsymbol{x}$ instead. In this research, $\mu(\boldsymbol{x})$ and $\Sigma(\boldsymbol{x})$ are flattened into an array of neurons' output layer of the encoder $f(\boldsymbol{x}|\Theta)$:

$$f(\boldsymbol{x}|\Theta) = \begin{pmatrix} \mu(\boldsymbol{x}) \\ \Sigma(\boldsymbol{x}) \end{pmatrix} \rightarrow \boldsymbol{z}.$$

The actual decoded data **z** is calculated randomly from $\mu(\boldsymbol{x})$ and $\Sigma(\boldsymbol{x})$ along with a random vector $\boldsymbol{r}$:

$$\boldsymbol{z} = \mu(\boldsymbol{x}) + (\Sigma(\boldsymbol{x}))^{\frac{1}{2}}\boldsymbol{r}, \tag{8}$$

where $\boldsymbol{r}$ follows the standard Gaussian distribution with mean vector $\mathbf{0}$ and identity covariance matrix I, and each element of $(\Sigma(\boldsymbol{x}))^{1/2}$ is the square root of the corresponding element of $\Sigma(\boldsymbol{x})$. This is an excellent finding in the traditional literature that made the calculation of Kullback–Leibler divergence much easier without loss of information.

The balance function $b_{\text{AVA}}(\Phi, \Psi)$ aims to balance the decoding task and the discrimination task without partiality, but it can lean forward the decoding task for improving the accuracy of the decoder by including the error of the original data $\boldsymbol{x}$ and the decoded data $\boldsymbol{x}'$ into the balance function as follows:

$$\begin{aligned} b_{\text{AVA}}(\Phi,\Psi) &= b_{\text{GAN}}(\Phi,\Psi) - \tfrac{1}{2}\|\boldsymbol{x}-\boldsymbol{x}'\|^2 \\ &= \log(d(\boldsymbol{x}|\Psi)) + \log(1 - d(g(\boldsymbol{z}|\Phi)|\Psi)) - \tfrac{1}{2}\|\boldsymbol{x}-\boldsymbol{x}'\|^2. \end{aligned} \tag{9}$$

As a result, the estimate of the discriminator parameter Ψ is

$$\Psi = \Psi + \gamma\left(\frac{a_d'(d(\boldsymbol{x}|\Psi))}{d(\boldsymbol{x}|\Psi)} - \frac{a_d'(d(\boldsymbol{x}'|\Psi))}{1 - d(\boldsymbol{x}'|\Psi)} + a_d'(d_0)\sum_i(\boldsymbol{x}[i] - \boldsymbol{x}'[i])a_g'(\boldsymbol{x}'[i])\right), \quad (10)$$

where $d_0 = d(\boldsymbol{x}'|\Psi)$ as usual. In a reverse causality–effect relationship, the unique output neuron of discriminator DNN is the cause of all output neurons of decoder DNN as shown in Figure 3.

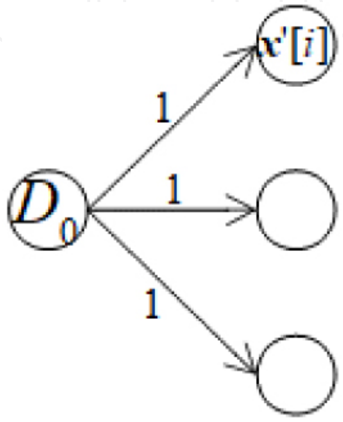

Figure 3. Reverse causality–effect relationship between discriminator DNN and decoder DNN.

Suppose the bias of each decoder output neuron is bias$[i]$ and the error of the discriminator output neuron, error$[i]$, is the sum of weighted biases, which is in turn multiplied with the derivative at the discriminator output neuron noting that every weighted bias is also multiplied with the derivative at every decoder output neuron. Suppose all weights are 1; we have

$$\text{error}[i] = a_d'(d_0)\sum_i \text{bias}[i]a_g'(\boldsymbol{x}'[i])$$
$$\text{bias}[i] = \boldsymbol{x}[i] - \boldsymbol{x}'[i].$$

Because the balance function $b_{\text{AVA}}(\Phi, \Psi)$ aims to improve the decoder $g(\mathbf{z}|\Phi)$, it is possible to improve the encoder $f(\boldsymbol{x}|\Theta)$ by a similar technique noting that the output of the encoder is the mean vector $\mu(\boldsymbol{x})$ and the covariance matrix $\Sigma(\boldsymbol{x})$. This research proposes another balance function $B_{\text{AVA}}(\Theta, \Lambda)$ to assess the reliability of the mean vector $\mu(\boldsymbol{x})$ because $\mu(\boldsymbol{x})$ is very important to randomize $\mathbf{z}$ and $\mu(\boldsymbol{x})$ is linear. Let $D(\mu(\boldsymbol{x})|\Lambda)$ be the discrimination function for encoder DNN from $\mu(\boldsymbol{x})$ to range [0,1] in which $D(\mu(\boldsymbol{x})|\Lambda)$ can distinguish fake mean $\mu(\boldsymbol{x}')$ from real mean $\mu(\boldsymbol{x})$. Obviously, $D(\mu(\boldsymbol{x})|\Lambda)$ is implemented by a so-called encoding discriminator DNN whose weights are Λ noting that this DNN has only one output neuron denoted by D_0. The balance function $B_{\text{AVA}}(\Theta, \Lambda)$ is specified as follows:

$$B_{\text{AVA}}(\Theta, \Lambda) = \log(D(\mu(\boldsymbol{x})|\Lambda)) + \log(1 - D(\mu(\boldsymbol{x}')|\Lambda)). \tag{11}$$

Note that

$$g(\boldsymbol{z}|\Phi) = \boldsymbol{x}'.$$

The AVA loss function is modified with regard to the balance function $B_{\text{AVA}}(\Theta, \Lambda)$ as follows:

$$\begin{aligned} l_{\text{AVA}}(\Theta, \Phi, \Psi, \Lambda) \quad = \quad & \tfrac{1}{2}\|\boldsymbol{x} - \boldsymbol{x}'\|^2 + \text{KL}(\mu(\boldsymbol{x}), \Sigma(\boldsymbol{x})|N(0, I)) + \log(1 - d(\boldsymbol{x}'|\Psi)) \\ & + \log(1 - D(\mu(\boldsymbol{x}')|\Lambda)). \end{aligned} \tag{12}$$

By following a similar way of applying the SGD algorithm, it is easy to estimate the encoding discriminator parameter Λ as follows:

$$\Lambda = \Lambda + \gamma\left(\frac{a'_D(D(\mu(\boldsymbol{x})|\Lambda))}{D(\mu(\boldsymbol{x})|\Lambda)} - \frac{a'_D(D(\mu(\boldsymbol{x}')|\Lambda))}{1 - D(\mu(\boldsymbol{x}')|\Lambda)}\right), \tag{13}$$

where $a_D(.)$ and $a'_D(.)$ are activation functions of the discriminator $D(\mu(\boldsymbol{x})|\Lambda)$ and its derivative, respectively.

The encoder parameter Θ consists of two separate parts Θ_μ and Θ_Σ because the output of encoder $f(\boldsymbol{x}|\Theta)$ consists of mean vector $\mu(\boldsymbol{x})$ and covariance matrix $\Sigma(\boldsymbol{x})$:

$$\Theta = \begin{pmatrix} \Theta_\mu \\ \Theta_\Sigma \end{pmatrix},$$

where

$$\begin{aligned} \Theta_\mu &= \Theta_\mu - \gamma\mu(\boldsymbol{x})a'_f(\boldsymbol{x}) \\ \Theta_\Sigma &= \Theta_\Sigma - \gamma\left(-\tfrac{1}{2}(\Sigma(\boldsymbol{x}))^{-1} + \tfrac{1}{2}I\right)a'_f(\boldsymbol{x}). \end{aligned}$$

When the balance function $B_{\text{AVA}}(\Theta, \Lambda)$ is included in the AVA loss function, the part Θ_μ is recalculated whereas the part Θ_Σ is kept intact as follows:

$$\Theta_\mu[i] = \Theta_\mu[i] - \gamma\left(\mu(\boldsymbol{x})[i] - \frac{a'_D(D(\boldsymbol{x}'|\Lambda))}{1 - D(\boldsymbol{x}'|\Lambda)}\right)a'_f(\boldsymbol{x}[i]). \tag{14}$$

Figure 4 shows the AVA architecture with the support of the assessing encoder.

Similarly, the balance function $B_{\text{AVA}}(\Phi, \Lambda)$ can lean forward the encoding task for improving the accuracy of encoder $f(\boldsymbol{x}|\Theta)$ by considering the error of original

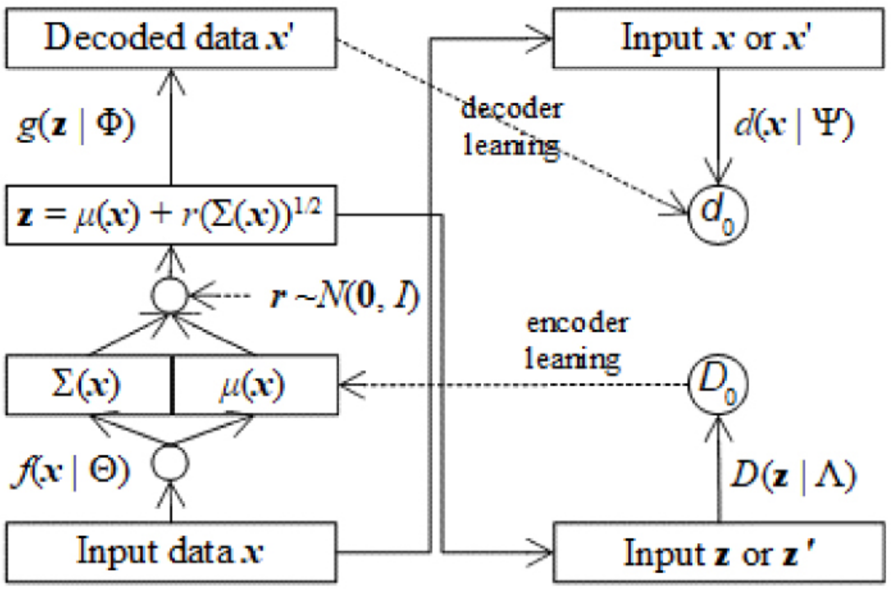

Figure 4. AVA architecture with support of assessing encoder.

mean $\mu(\boldsymbol{x})$ and decoded data mean $\mu(\boldsymbol{x}')$ as follows:

$$B_{\text{AVA}}(\Phi, \Lambda) = \log(D(\mu(\boldsymbol{x})|\Lambda)) + \log(1 - D(\mu(\boldsymbol{x}')|\Lambda)) - \frac{1}{2}\|\mu(\boldsymbol{x}) - \mu(\boldsymbol{x}')\|^2. \tag{15}$$

Without repeating explanations, the estimate of discriminator parameter Λ is modified as follows:

$$\begin{aligned}\Lambda &= \Lambda + \gamma\left(\frac{a_D'(D(\mu(\boldsymbol{x})|\Lambda))}{D(\mu(\boldsymbol{x})|\Lambda)} - \frac{a_D'(D(\mu(\boldsymbol{x}')|\Lambda))}{1 - D(\mu(\boldsymbol{x}')|\Lambda)}\right.\\ &\quad \left. + a_D'(D_0)\sum_i (\mu(\boldsymbol{x})[i] - \mu(\boldsymbol{x}')[i])a_g'(\mu(\boldsymbol{x}')[i])\right),\end{aligned} \tag{16}$$

where $D_0 = D(\boldsymbol{x}'|\Lambda)$ as usual. These variants of AVA are summarized, and their tests are described in the next section. Moreover, the ideology of fusing VAE and GAN like AVA does is not new when reviewing the research by Larsen *et al.* [2] in which their unification mechanism is like AVA. The contribution of this research is to propose a solid architecture of a generative model based on two powerful models VAE and GAN, which aims at flexibility with plentiful functions including encoder, decoder, and leaning mechanism that allows developers to customize AVA according to their individual purposes. The generative AI application supporting AVA is available at https://github.com/ngphloc/ai/tree/main/3_implementation, which requires Java 15.

3. Experimental results and discussion

In this experiment, AVA is tested with VAE and GAN; but there are five versions of AVA such as AVA1, AVA2, AVA3, AVA4, and AVA5. Recall that AVA1 is the normal version of AVA whose parameters are listed as follows:

$$\Theta = \Theta - \gamma\left(\mu(\boldsymbol{x}) - \tfrac{1}{2}(\Sigma(\boldsymbol{x}))^{-1} + \tfrac{1}{2}I\right)a_f'(\boldsymbol{x})$$
$$\Phi[i] = \Phi[i] + \gamma\left((\boldsymbol{x}[i] - \boldsymbol{x}'[i]) + \frac{a_d'(d(\boldsymbol{x}'|\Psi^*))}{1 - d(\boldsymbol{x}'|\Psi^*)}\right)a_g'(\boldsymbol{x}'[i])$$
$$\Psi = \Psi + \gamma\left(\frac{a_d'(d(\boldsymbol{x}|\Psi))}{d(\boldsymbol{x}|\Psi)} - \frac{a_d'(d(\boldsymbol{x}'|\Psi))}{1 - d(\boldsymbol{x}'|\Psi)}\right).$$

AVA2 leans forward, improving the accuracy of decoder DNN by modifying discriminator parameter Ψ as follows:

$$\Theta = \Theta - \gamma\left(\mu(\boldsymbol{x}) - \tfrac{1}{2}(\Sigma(\boldsymbol{x}))^{-1} + \tfrac{1}{2}I\right)a_f'(\boldsymbol{x})$$
$$\Phi[i] = \Phi[i] + \gamma\left((\boldsymbol{x}[i] - \boldsymbol{x}'[i]) + \frac{a_d'(d(\boldsymbol{x}'|\Psi^*))}{1 - d(\boldsymbol{x}'|\Psi^*)}\right)a_g'(\boldsymbol{x}'[i])$$
$$\Psi = \Psi + \gamma\left(\frac{a_d'(d(\boldsymbol{x}|\Psi))}{d(\boldsymbol{x}|\Psi)} - \frac{a_d'(d(\boldsymbol{x}'|\Psi))}{1 - d(\boldsymbol{x}'|\Psi)} + a_d'(d_0)\sum_i(\boldsymbol{x}[i] - \boldsymbol{x}'[i])a_g'(\boldsymbol{x}'[i])\right).$$

AVA3 supports the balance function $B_{\text{AVA}}(\Theta, \Lambda)$ for assessing the reliability of encoder $f(\boldsymbol{x}|\Theta)$. Its parameters are listed as follows:

$$\Theta_\mu[i] = \Theta_\mu[i] - \gamma\left(\mu(\boldsymbol{x})[i] - \frac{a_D'(D(\boldsymbol{x}'|\Lambda))}{1 - D(\boldsymbol{x}'|\Lambda)}\right)a_f'(\boldsymbol{x}[i])$$
$$\Theta_\Sigma = \Theta_\Sigma - \gamma\left(-\tfrac{1}{2}(\Sigma(\boldsymbol{x}))^{-1} + \tfrac{1}{2}I\right)a_f'(\boldsymbol{x})$$
$$\Phi = \Phi + \gamma(\boldsymbol{x} - \boldsymbol{x}')a_g'(\boldsymbol{x}')$$
$$\Lambda = \Lambda + \gamma\left(\frac{a_D'(D(\mu(\boldsymbol{x})|\Lambda))}{D(\mu(\boldsymbol{x})|\Lambda)} - \frac{a_D'(D(\mu(\boldsymbol{x}')|\Lambda))}{1 - D(\mu(\boldsymbol{x}')|\Lambda)}\right).$$

AVA4 is a variant of AVA3 along with the leaning forward mechanism, improving the accuracy of encoder $f(\boldsymbol{x}|\Theta)$ like AVA2. Its parameters are listed as follows:

$$\Theta_\mu[i] = \Theta_\mu[i] - \gamma\left(\mu(\boldsymbol{x})[i] - \frac{a_D'(D(\boldsymbol{x}'|\Lambda))}{1 - D(\boldsymbol{x}'|\Lambda)}\right)a_f'(\boldsymbol{x}[i])$$
$$\Theta_\Sigma = \Theta_\Sigma - \gamma\left(-\tfrac{1}{2}(\Sigma(\boldsymbol{x}))^{-1} + \tfrac{1}{2}I\right)a_f'(\boldsymbol{x})$$
$$\Phi = \Phi + \gamma(\boldsymbol{x} - \boldsymbol{x}')a_g'(\boldsymbol{x}')$$

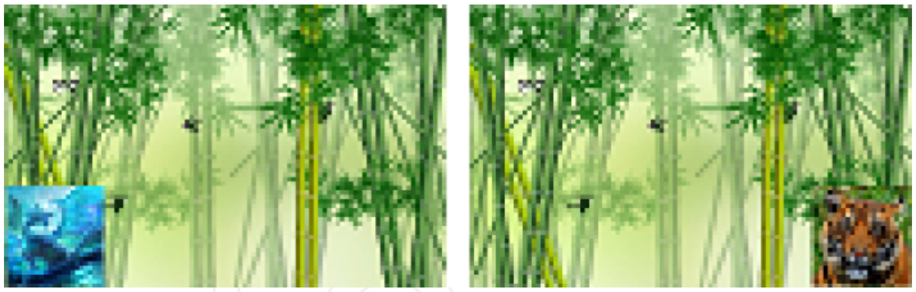

Figure 5. Images for DGM training and testing.

$$\Lambda = \Lambda + \gamma \left(\frac{a'_D(D(\mu(\boldsymbol{x})|\Lambda))}{D(\mu(\boldsymbol{x})|\Lambda)} - \frac{a'_D(D(\mu(\boldsymbol{x}')|\Lambda))}{1 - D(\mu(\boldsymbol{x}')|\Lambda)} + a'_D(D_0) \sum_i (\mu(\boldsymbol{x})[i] - \mu(\boldsymbol{x}')[i]) a'_g(\mu(\boldsymbol{x}')[i]) \right).$$

The last version AVA5 supports all functions such as decoder supervising, leaning decoder, encoder supervising, and leaning encoder:

$$\Theta_\mu[i] = \Theta_\mu[i] - \gamma \left(\mu(\boldsymbol{x})[i] - \frac{a'_D(D(\boldsymbol{x}'|\Lambda))}{1 - D(\boldsymbol{x}'|\Lambda)} \right) a'_f(\boldsymbol{x}[i])$$

$$\Theta_\Sigma = \Theta_\Sigma - \gamma \left(-\tfrac{1}{2}(\Sigma(\boldsymbol{x}))^{-1} + \tfrac{1}{2}I \right) a'_f(\boldsymbol{x})$$

$$\Phi[i] = \Phi[i] + \gamma \left((\boldsymbol{x}[i] - \boldsymbol{x}'[i]) + \frac{a'_d(d(\boldsymbol{x}'|\Psi^*))}{1 - d(\boldsymbol{x}'|\Psi^*)} \right) a'_g(\boldsymbol{x}'[i])$$

$$\Psi = \Psi + \gamma \left(\frac{a'_d(d(\boldsymbol{x}|\Psi))}{d(\boldsymbol{x}|\Psi)} - \frac{a'_d(d(\boldsymbol{x}'|\Psi))}{1 - d(\boldsymbol{x}'|\Psi)} + a'_d(d_0) \sum_i (\boldsymbol{x}[i] - \boldsymbol{x}'[i]) a'_g(\boldsymbol{x}'[i]) \right)$$

$$\Lambda = \Lambda + \gamma \left(\frac{a'_D(D(\mu(\boldsymbol{x})|\Lambda))}{D(\mu(\boldsymbol{x})|\Lambda)} - \frac{a'_D(D(\mu(\boldsymbol{x}')|\Lambda))}{1 - D(\mu(\boldsymbol{x}')|\Lambda)} + a'_D(D_0) \sum_i (\mu(\boldsymbol{x})[i] - \mu(\boldsymbol{x}')[i]) a'_g(\mu(\boldsymbol{x}')[i]) \right).$$

The experiment is performed on a laptop with CPU AMD64 4 sub-processor core, 4 GB RAM, Windows 10, and Java 15. The given dataset is a set of thirty-six 100 × 64 images available at https://github.com/ngphloc/ai/tree/main/3_implementation/datasets/orbit/base-100x64. The 36 images are animated images that imitate the movements of a dragon and a tiger in a bamboo jungle. Each image depicts the position of a dragon or a tiger; note that the background, which is the bamboo jungle, is not changed. For example, the following two images (Figure 5) depict two positions of a dragon and a tiger among 36 positions. For each tested image, DGMs are not retrained for fair testing because there is no splitting of the training set and the testing set.

It is necessary to define how efficient DGMs such as VAE, GAN, and AVA are. Let imageGen be the best image generated by a DGM, which is compared with the ith image denoted by images$[i]$ in the dataset. Then let d_{ij} be the pixel distance between imageGen and the ith image at the jth pixel as follows:

$$d_{ij} = \|\text{imageGen}[j] - \text{image}[i][j]\|.$$

Obviously, image$[i][j]$ (imageGen$[j]$) is the jth pixel of the ith image (the generated image). The notation $\|\cdot\|$ denotes the norm of a pixel. For example, the norm of the RGB pixel is $\sqrt{r^2 + g^2 + b^2}$, where r, g, and b are red, green, and blue colors of such pixels. Suppose all pixel values are normalized in the interval $[0,1]$. The quantity d_{ij} implies the difference between two images, and so it expresses the similarity quality of the generated image, which is as small as possible. The inverse $1 - d_{ij}$ expresses the diversity quality of the generated image, which is as large as possible. Therefore, the best image should balance the quantities d_{ij} and $1 - d_{ij}$ so that the product $d_{ij}(1 - d_{ij})$ becomes as larger as possible:

$$d_{ij}(1 - d_{ij}) \to \max.$$

Because the product $d_{ij}(1 - d_{ij})$ is a second-order function, its maximizer exists, and so the generated image whose product $d_{ij}(1 - d_{ij})$ is larger is the better image when its balance is more stable. As a result, let the balance metric (BM) be the metric to assess the quality of the generated image (the best image) with regard to the ith image, which is formulated as follows:

$$\text{BM}_i = \frac{1}{n_i} \sum_j d_{ij}(1 - d_{ij}),$$

where n_i is the number of pixels of the ith image. The larger the BM$_i$, the better the generated image and the better the balance of similarity and diversity. The overall BM of a DGM is the average BM$[i]$ over $N = 36$ test images as follows:

$$\text{BM} = \frac{1}{N} \sum_i \text{BM}_i = \frac{1}{N} \sum_i \frac{1}{n_i} \sum_j d_{ij}(1 - d_{ij}), \quad (17)$$

where

$$d_{ij} = \|\text{imageGen}[j] - \text{image}[i][j]\|.$$

Recall that the larger the BM, the better the DGM. However, regarding the similarity quality, the DGM will be better when its BM is smaller because a small BM implies good similarity in this test; note that such a small BM implies small distance or small diversity. Therefore, the DGM whose BM is largest or smallest is preeminent. The

DGM whose BM is the largest is the best in balance of similarity and diversity. The DGM whose BM is the smallest is the best in similarity. Both the maximum and minimum of BM, which indicate both balance quality and similarity quality, respectively, are considered in this test but the balance quality with a large BM is more important.

The four AVA variants (AVAs) as well as VAE and GAN are evaluated by BM with 19 learning rates (γ = 1, 0.9, … , 0.1, 0.09, … , 0.01) because the SGD algorithm is affected by the learning rate and the accuracy of AVA varies slightly within a learning rate because of randomizing encoded data **z** in the VAE algorithm. Table 1 shows the BM values of AVAs, VAE, and GAN with 10 learning rates: γ = 1, 0.9, 0.8, 0.7, 0.6, 0.5, 0.4, 0.3, 0.2, 0.1.

Table 1. BM regarding learning rates from 1 down to 0.1.

	AVA1	AVA2	AVA3	AVA4	AVA5	VAE	GAN
γ = 1.0	0.2298	0.2301	0.0642	0.0766	0.2301	0.0583	0.2298
γ = 0.9	0.2307	0.2294	0.0546	0.0594	0.2293	0.0681	0.2283
γ = 0.8	0.2309	0.2316	0.0596	0.0546	0.2301	0.0587	0.2311
γ = 0.7	0.2316	0.2305	0.0629	0.0631	0.2305	0.0665	0.2311
γ = 0.6	0.2309	0.2317	0.0555	0.0657	0.2318	0.0623	0.2315
γ = 0.5	0.2318	0.2319	0.0591	0.0598	0.2313	0.0610	0.2311
γ = 0.4	0.2322	0.2329	0.0629	0.0732	0.2322	0.0568	0.2312
γ = 0.3	0.2318	0.2321	0.0741	0.0655	0.2326	0.0651	0.2325
γ = 0.2	0.2300	0.2312	0.0740	0.0929	0.2302	0.0735	0.2315
γ = 0.1	0.2103	0.2105	0.1230	0.1217	0.2114	0.1238	0.2107

Table 2 shows the BM values of AVAs, VAE, and GAN with nine learning rates: γ = 0.09, 0.08, 0.07, 0.06, 0.05, 0.04, 0.03, 0.02, 0.01.

Table 2. BM regarding learning rates from 0.09 down to 0.01.

	AVA1	AVA2	AVA3	AVA4	AVA5	VAE	GAN
γ = 0.09	0.2038	0.2015	0.1319	0.1328	0.2026	0.1338	0.2031
γ = 0.08	0.1924	0.1938	0.1417	0.1446	0.1978	0.1435	0.1916
γ = 0.07	0.1842	0.1826	0.1566	0.1574	0.1834	0.1555	0.1818
γ = 0.06	0.1685	0.1772	0.1662	0.1659	0.1785	0.1676	0.1699
γ = 0.05	0.1664	0.1617	0.1792	0.1785	0.1621	0.1805	0.1628
γ = 0.04	0.1675	0.1655	0.1918	0.1906	0.1662	0.1924	0.1665
γ = 0.03	0.1845	0.1832	0.2017	0.2014	0.1855	0.2021	0.1857
γ = 0.02	0.2047	0.2032	0.2098	0.2098	0.2028	0.2099	0.2046
γ = 0.01	0.2147	0.2146	0.2147	0.2147	0.2146	0.2147	0.2148

Table 3. Evaluation of AVAs, VAE, and GAN.

	AVA1	AVA2	AVA3	AVA4	AVA5	VAE	GAN
Mean	0.2093	0.2092	0.1202	0.1225	0.2096	0.1207	0.2089
Maximum	0.2322	0.2329	0.2147	0.2147	0.2326	0.2147	0.2325
Minimum	0.1664	0.1617	0.0546	0.0546	0.1621	0.0568	0.1628
SD	0.0249	0.0251	0.0606	0.0586	0.0244	0.0606	0.0252

Table 3 shows BM means, BM maxima, BM minima, and BM standard deviations (SDs) of AVAs, VAE, and GAN.

Note that VAE and GAN represent a pole of similarity quality and a pole of balance quality, respectively. From the experimental results shown in Table 3, AVA5 is the best DGM because it gains the highest BM mean (0.2096), which is also larger than the BM mean (0.2089) of the pole GAN. It is easy to explain this result because AVA5 is the one that improves both the decoding task and the encoding task when it embeds both the decoder discriminator and the encoder discriminator as well as both the leaning decoder and the leaning encoder. Moreover, both AVA1 and AVA2 are better than GAN because their BM means (0.2093, 0.2092) are larger than the BM mean (0.2089) of GAN. If the similarity quality is considered, AVA3 is the best DGM because it gains the lowest BM mean (0.1202), which is also smaller than the BM mean (0.1207) of the pole VAE. It is easy to explain this result because AVA3 is the one that improves the encoding task when it embeds the encoder discriminator. Moreover, AVA1, which is a fair AVA because it embeds the decoder discriminator but does not support the leaning decoder, is better than the pole GAN whereas AVA3, which is a fair AVA because it embeds the encoder discriminator but does not support the leaning encoder, is better than the pole VAE. This result is important because the best AVA5 is not a fair one because it supports both the leaning decoder and the leaning encoder. Therefore, about the BM mean, which is the most important metric, all AVA variants are better than traditional DGMs such as VAE and GAN with regard to both similarity quality and balance quality.

Although the BM mean is the most important metric, it is necessary to check other metrics related to extreme values that are BM maximum and BM minimum, where BM maximum implies the best balance quality and BM minimum implies the best similarity quality. Note from experimental results shown in Table 3 that the decoder improvement with AVA1 and AVA2 aims to improve balance quality with high BM and the encoder improvement with AVA3 and AVA4 aims to improve similarity quality with low BM whereas AVA5 improves both the decoder and the encoder. AVA2 and AVA5 are better DGMs about the extreme balance quality because their BM maxima (0.2329, 0.2326) are larger than the BM maximum (0.2325) of GAN. Similarly, AVA3 and AVA4 are better DGMs about the extreme similarity quality because their

BM minima (0.0546, 0.0546) are smaller than the BM minimum (0.0568) of VAE. Therefore, about BM extreme values, AVA variants are better than traditional DGMs such as VAE and GAN with regard to both similarity quality and balance quality.

Because the two poles VAE and GAN are stabler than AVAs in theory as each AVA includes functions from VAE and GAN so that each AVA is more complicated than VAE and GAN, it is necessary to check the SD of BM, which reflects the stability of DGMs. The smaller the SD, the stabler the DGM. AVA1 and AVA2 are stabler than GAN when their SDs (0.0249, 0.0251) are smaller than the SD (0.0252) of GAN. AVA3 and AVA4 are slightly stabler than VAE when their SDs (0.0606, 0.0586) are smaller than or equal to the SD (0.0606) of VAE. Moreover, AVA5 is the best one about the stability quality when its SD (0.0244) is the smallest. Therefore, AVA variants are stabler than traditional DGMs such as VAE and GAN.

Figure 6 depicts BM means, BM maxima, BM minima, and BM standard deviations of AVAs, VAE, and GAN by charts.

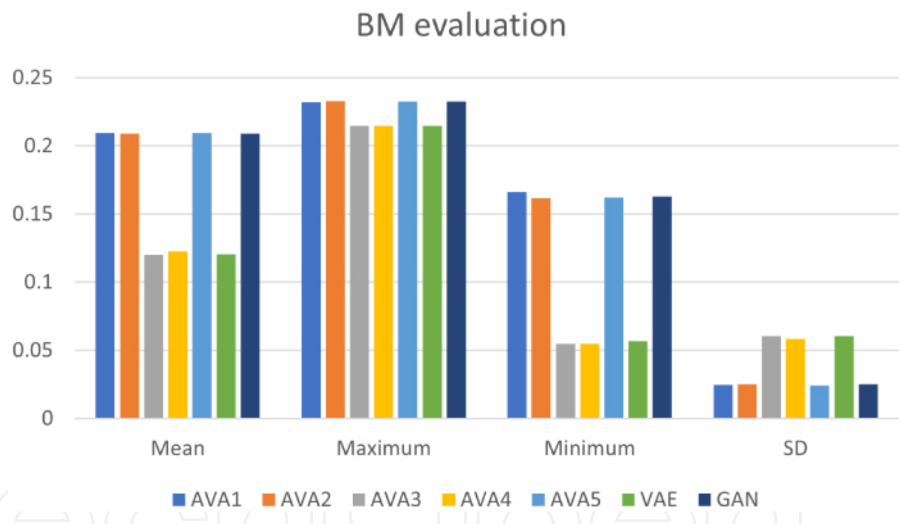

Figure 6. Evaluation of AVAs, VAE, and GAN.

It is concluded that the combination of GAN and VAE, which produces AVA in this research, results in better encoding and decoding performance of the DGM when metrics such as BM means, BM maxima, BM minima, and BM standard deviations of AVAs are better with regard to contexts of balance quality and similarity quality. Moreover, AVA5, which is full of functions including the decoder discriminator, decoder leaning, encoder discrimination, and encoder leaning, produces the best results with the highest balance quality given the largest BM mean (0.2096) and the highest stability given the smallest SD (0.0244).

4. Conclusions

It is certain that AVA is better than the traditional VAE and GAN due to the support of Kullback–Leibler divergence that establishes the encoder as well as the built-in discriminator function of GAN that assesses the reliability of data. It is possible to think that VAE and GAN are solid models in both theory and practice when their mathematical foundation cannot be changed or transformed. However, it is still possible to improve them by modifications or combinations as well as applying them to specific tools where their strong points are brought into play. In applications related to raster data like images, VAE has a drawback of consuming much memory because probabilistic distribution represents the entire image whereas some other DGMs focus on representing the product of many conditional probabilistic distributions for pixels. Although this approach for modeling pixels by the recurrent neural network may consume less memory, it is significantly useful to fill in or recover smaller damaged areas in a bigger image. In the future, we will try to apply the pixel approach to AVA; for instance, AVA processes a big image block by block and then every block is modeled by a conditional probability distribution with a recurrent neural network as well as a long short-term memory network.

Conflict of interest

The authors declare no conflict of interest.

References

1 Ruthotto L, Haber E. An Introduction to Deep Generative Modeling [Internet]. arXiv; 2021. Available from: https://arxiv.org/10.48550/arXiv.2103.05180.

2 Larsen AB, Sønderby SK, Larochelle H, Winther O. Autoencoding beyond pixels using a learned similarity metric [Internet]. In: *International Conference on Machine Learning*, vol. 48, New York: JMLR; 2016. p. 1558–1566. Available from: http://proceedings.mlr.press/v48/larsen16.pdf.

3 Mescheder L, Nowozin S, Geiger A. Adversarial variational Bayes: unifying variational autoencoders and generative adversarial networks [Internet]. In: *Proceedings of the 34th International Conference on Machine*, vol. 70, Sydney: PMLR; 2017. p. 2391–2400. Available from: http://proceedings.mlr.press/v70/mescheder17a/mescheder17a.pdf.

4 Rosca M, Lakshminarayanan B, Warde-Farley D, Mohamed S. Variational Approaches for Auto-encoding Generative Adversarial Networks [Internet]. arXiv; 2017. Available from: https://arxiv.org/abs/1706.04987.

5 Ahmad B, Sun J, You Q, Palade V, Mao Z. Brain tumor classification using a combination of variational autoencoders and generative adversarial networks. *Biomedicines*. 2022;**10**(2):1–19. doi:10.3390/biomedicines10020223.

6 Miolane N, Poitevin F, Li Y-T. Estimation of orientation and camera parameters from cryo-electron microscopy images with variational autoencoders and generative adversarial [Internet]. In: *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops*. New Orleans: IEEE; 2020.

p. 970–971. Available from: http://openaccess.thecvf.com/content_CVPRW_2020/papers/w57/Miolane_Estimation_of_Orientation_and_Camera_Parameters_From_Cryo-Electron_Microscopy_Images_CVPRW_2020_paper.pdf.

7 Ding Y, Kang W, Feng J, Peng B, Yang A. Credit card fraud detection based on improved variational autoencoder generative adversarial network. In: Abbott D, editor. *IEEE Access*, vol. 11, 2023. p. 83680–83691. doi:10.1109/ACCESS.2023.3302339.

8 Kingma DP, Welling M. Auto-encoding Variational Bayes [Internet]. arXiv; 2022. 1–14. Available form: https://arxiv.org/10.48550/arXiv.1312.6114.

9 Goodfellow I, Pouget-Abadie J, Mirza M, Xu B, Warde-Farley D, Ozair S, et al. Generative adversarial nets [Internet]. In: Ghahramani Z, Welling M, Cortes C, Lawrence N, Weinberger K, editors. *Advances in Neural Information Processing Systems 27 (NIPS 2014)*, vol. 27, Montreal: NeurIPS; 2014 Available from: https://proceedings.neurips.cc/paper_files/paper/2014/file/5ca3e9b122f61f8f06494c97b1afccf3-Paper.pdf.

10 Doersch C. Tutorial on Variational Autoencoders [Internet]. arXiv; 2016. Available form: https://arxiv.org/abs/1606.05908.

11 Nguyen L. In: Evans C, editor. *Matrix analysis and calculus [Internet]*. 1st ed. Hanoi, Vietnam: Lambert Academic Publishing; 2015 [cited 2014 Mar 3]. Available from: https://www.shuyuan.sg/store/gb/book/matrix-analysis-and-calculus/isbn/978-3-659-69400-4.

IntechOpen