Preprints.org

Article

Adversarial Variational Autoencoders to Extend and Improve Generative Model

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

Article

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; ng_phloc@yahoo.com
[2] College of Technological Innovation, Zayed University, Abu Dhabi, UAE; hassan.abdalla@zu.ac.ae (H.I.A.); aliaaa2004@yahoo.com (A.A.A.)

Abstract: Generative artificial intelligence (GenAI) has been developing with many incredible achievements like ChatGPT and Bard. Deep generative model (DGM) is a branch of GenAI, which is preeminent in generating raster data such as image and sound due to strong points of deep neural network (DNN) in inference and recognition. The built-in inference mechanism of DNN, which simulates and aims to synaptic plasticity of human neuron network, fosters generation ability of DGM which produces surprised results with support of statistical flexibility. Two popular approaches in DGM are Variational Autoencoders (VAE) and Generative Adversarial Network (GAN). Both VAE and GAN have their own strong points although they share and imply underline theory of statistics as well as incredible complex via hidden layers of DNN when DNN becomes effective encoding/decoding functions without concrete specifications. In this research, VAE and GAN is unified into a consistent and consolidated model called Adversarial Variational Autoencoders (AVA) in which VAE and GAN complement each other, for instance, VAE is a good data generator by encoding data via excellent ideology of Kullback-Leibler divergence and GAN is a significantly important method to assess reliability of data which is realistic or fake. In other words, AVA aims to improve accuracy of generative models, besides AVA extends function of simple generative models. In methodology this research focuses on 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 Autoencoders (VAE); Generative Adversarial Network (GAN)

1. Introduction

Variational Autoencoders (VAE) and Generative Adversarial Network (GAN) are two popular approaches for developing deep generative model with support of deep neural network (DNN) where high capacity of DNN contributes significantly to successes of GAN and VAE. There are some researches which combined VAE and GAN. Larsen et al. (Larsen, Sønderby, Larochelle, & Winther, 2016) proposed a traditional combination of VAE and GAN by considering decoder of VAE as generator of GAN (Larsen, Sønderby, Larochelle, & Winther, 2016, p. 1558). They constructed target optimization function as sum of likelihood function of VAE and target function of GAN (Larsen, Sønderby, Larochelle, & Winther, 2016, p. 1560). This research is similar to their research (Larsen, Sønderby, Larochelle, & Winther, 2016, p. 1561) except that the construction optimization functions in two researches are slightly different where the one in this research does not include target function of GAN according to traditional approach of GAN. However uncorrelated variables will be removed after gradients are determined. Moreover, because encoded data **z** is basically randomized in this research, it does not make a new random **z'** to be included into target function of GAN. This research also mentions skillful techniques of derivatives in backpropagation algorithm.

Mescheder et al. (Mescheder, Nowozin, & Geiger, 2017) transformed gain function of VAE including Kullback-Leibler divergence into gain function of GAN via a so-called real-valued

discrimination network (Mescheder, Nowozin, & Geiger, 2017, p. 2394) related to Nash equilibrium equation and sigmoid function and then, they trained the transformed VAE by stochastic gradient descent method. Actually, they estimated three parameters (Mescheder, Nowozin, & Geiger, 2017, p. 2395) like this research, but their method focused on mathematical transformation while this research focuses on skillful techniques in implementation. In other words, Mescheder et al. (Mescheder, Nowozin, & Geiger, 2017) tried to fuse VAE into GAN whereas this research combines them by mutual and balancing way but both of us try to make unification of VAE and GAN. Rosca et al. (Rosca, Lakshminarayanan, Warde-Farley, & Mohamed, 2017, p. 4) used a density ratio trick to convert Kullback-Leibler divergence of VAE into the mathematical form $\log(x / (1{-}x))$ which is similar to GAN target function $\log(x) + \log(1{-}x)$. Actually, they made a fusion of VAE and GAN like Mescheder et al. did. The essence of their methods is based on convergence of Nash equilibrium equation.

Ahmad et al. (Ahmad, Sun, You, Palade, & Mao, 2022) combined VAE and GAN separately as featured experimental research. Firstly, they trained VAE and swapped encoder-decoder network to decoder-encoder network so that output of VAE becomes some useful information which in turn becomes input of GAN instead that GAN uses random information as input as usual (Ahmad, Sun, You, Palade, & Mao, 2022, p. 6). Miolane et al. (Miolane, Poitevin, & Li, 2020) combined VAE and GAN by summing target functions of VAE and GAN weighted with regular hyperparameters (Miolane, Poitevin, & Li, 2020, p. 974). Later, they first trained VAE and then sent output of VAE to input of GAN (Miolane, Poitevin, & Li, 2020, p. 975).

In general, both VAE and GAN have their own strong points, for instance, they take advantages of solid statistical theory as well as incredible DNN but they are also stuck in drawbacks, for instance, VAE does not have mechanism to distinguish fake data from realistic data and GAN does not concern explicitly probabilistic distribution of encoded data. It is better to bring up their strong points and alleviate their weak points. Therefore, this research focuses on incorporating GAN into VAE by skillful techniques related to both stochastic gradient descent and software engineering architecture, which neither focuses on purely mathematical fusion nor focuses 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 Autoencoders (AVA) aims to extend functions of VAE and GAN as a general architecture for generative model. For instance, AVA will provide encoding function that GAN does not concern and provide discrimination function that VAE needs to distinguish fake data from realistic data. The corporation of VAE and GAN in AVA is strengthened by regular and balance mechanism, which obviously, is natural and like fusion mechanism. In some cases, it is better than fusion mechanism because both built-in VAE and GAN inside AVA can uphold their own strong features. Therefore, experiment in this research is not too serious with large data when AVA and VAE are only compared within 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 which incorporates Generative Adversarial Network (GAN) into Variational Autoencoders (VAE) for extending and improving deep generative model because GAN does not concern how to code original data and VAE lacks mechanisms to assess quality of generated data with note that data coding is necessary to some essential applications such as image impression and recognition whereas auditing quality can improve accuracy of generated data. As a convention, let vector variable $\boldsymbol{x} = (x_1, x_2, \ldots, x_m)^T$ and vector variable $\mathbf{z} = (z_1, z_2, \ldots, z_n)^T$ be original data and encoded data whose dimensions are m and n ($m > n$), respectively. A generative model is represented by a function $f(\boldsymbol{x} \mid \Theta) = \mathbf{z}$, $f(\boldsymbol{x} \mid \Theta) \approx \mathbf{z}$, or $f(\boldsymbol{x} \mid \Theta) \rightarrow \mathbf{z}$ where $f(\boldsymbol{x} \mid \Theta)$ is implemented by a deep neural network (DNN) whose weights are Θ, which converts the original data $\boldsymbol{x}$ to the encoded data $\mathbf{z}$ and is called encoder in VAE. A decoder in VAE which converts expectedly the encoded data $\mathbf{z}$ back to the original data $\boldsymbol{x}$ is represented by a function $g(\mathbf{z} \mid \Phi) = \boldsymbol{x}'$ where $g(\mathbf{z} \mid \Phi)$ is also implemented by a DNN whose weights are Φ with 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 (Kingma & Welling, 2022) is to minimize the following loss function for estimating the encoded parameter Θ and the decoded parameter Φ.

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

Such that:

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

$$\Phi^* = \underset{\Theta}{\operatorname{argmin}}\,\frac{1}{2}\|x - x'\|^2$$

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

GAN developed by Goodfellow et al. (Goodfellow, et al., 2014) does not concern the encoder $f(x \mid \Theta) = z$ but it focuses on optimizing the decoder $g(z \mid \Phi) = x'$ by introducing a so-called discriminator which is a discrimination function $d(x \mid \Psi)$: $x \to [0, 1]$ from concerned data x or x' to range $[0, 1]$ in which $d(x \mid \Psi)$ can distinguish fake data from real data. In other words, the larger result the discriminator $d(x' \mid \Psi)$ derives, the more realistic the generated data x' is. Obviously, $d(x \mid \Psi)$ is implemented by a DNN whose weights are Ψ with note that this DNN has only one output neuron denoted 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 (Goodfellow, et al., 2014, p. 3).

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

Such that Φ and Ψ are optimized mutually as follows:

$$\Phi^* = \underset{\Phi}{\operatorname{argmin}}\,b_{\text{GAN}}(\Phi, \Psi^*)$$

$$\Psi^* = \underset{\Psi}{\operatorname{argmax}}\,b_{\text{GAN}}(\Phi^*, \Psi)$$

The proposed generative model in this research is called Adversarial Variational Autoencoders (AVA) because it combines VAE and GAN by fusing mechanism in which loss function and balance function are optimized parallelly. The AVA loss function implies loss information in encoder $f(x \mid \Theta)$, decoder $g(z \mid \Phi)$, discriminator $d(x \mid \Psi)$ as follows:

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

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

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

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

$$\Theta^* = \underset{\Theta}{\operatorname{argmin}}\,l_{\text{AVA}}(\Theta, \Phi^*, \Psi^*)$$

$$\Phi^* = \underset{\Phi}{\operatorname{argmin}}\,l_{\text{AVA}}(\Theta^*, \Phi, \Psi^*)$$

$$\Psi^* = \underset{\Psi}{\operatorname{argmax}}\,b_{\text{AVA}}(\Phi^*, \Psi)$$

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

$$\Theta^* = \underset{\Phi}{\mathrm{argmin}} \left(\mathrm{KL}\big(\mu(\boldsymbol{x}), \Sigma(\boldsymbol{x}) \big| N(\boldsymbol{0}, I)\big) \right)$$

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

$$\Phi^* = \underset{\Theta}{\mathrm{argmin}} \left(\frac{1}{2} \|\boldsymbol{x} - \boldsymbol{x}'\|^2 + \log\big(1 - d(g(\boldsymbol{z}|\Phi^*)|\Psi^*)\big) \right)$$

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

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

By applying stochastic gradient descent (SDG) algorithm into backpropagation algorithm, these estimates are determined based on gradients of loss function and balance function as follows:

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

Where γ $(0 < \gamma \le 1)$ is 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 (Kingma & Welling, 2022, p. 5), (Doersch, 2016, p. 9), (Nguyen, 2015, p. 43):

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

The decoder gradient regarding Φ is:

$$\nabla_\Phi \left(\frac{1}{2} \|\boldsymbol{x} - \boldsymbol{x}'\|^2 + \log\big(1 - d(g(\boldsymbol{z}|\Phi^*)|\Psi^*)\big) \right) = -\left(\|\boldsymbol{x} - \boldsymbol{x}'\| + \frac{a_d'\big(d(\boldsymbol{x}'|\Psi^*)\big)}{1 - d(\boldsymbol{x}'|\Psi^*)} \right) a_g'(\boldsymbol{x}')$$

Where,

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

The discriminator gradient regarding Ψ is:

$$\nabla_\Psi\big(\log\big(d(\boldsymbol{x}|\Psi)\big) + \log\big(1 - d(\boldsymbol{x}'|\Psi)\big) \big) = \frac{a_d'\big(d(\boldsymbol{x}|\Psi)\big)}{d(\boldsymbol{x}|\Psi)} - \frac{a_d'\big(d(\boldsymbol{x}'|\Psi)\big)}{1 - d(\boldsymbol{x}'|\Psi)}$$

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

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

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

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

Where notation $[i]$ denotes the ith element in vector. Please pay attention to the derivatives $a_f{}'(.)$, $a_g{}'(.)$, and $a_d{}'(.)$ because they are helpful techniques to consolidate AVA. The reason of two different occurrences of derivatives $a_d{}'(d(\boldsymbol{x}' \mid \Psi^*))$ and $a_g{}'(\boldsymbol{x}')$ in decoder gradient regarding Φ is nontrivial

because the unique output neuron of discriminator DNN is considered as effect of the output layer of all output neurons in decoder DNN.
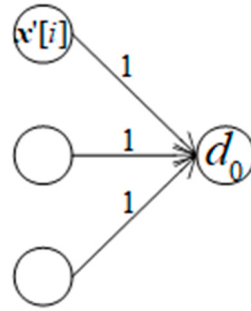

Figure 1. Causality effect relationship between decoder DNN and discriminator DNN.

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

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

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

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

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

Figure 2 shows the AVA architecture.
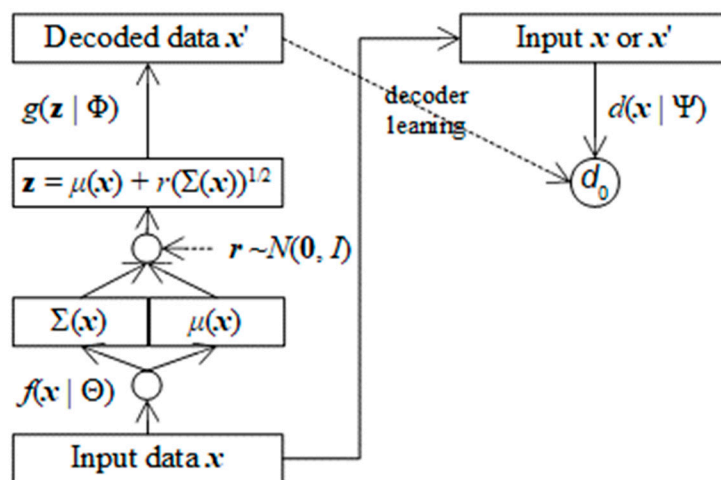

Figure 2. AVA architecture.

AVA architecture follows an important aspect of VAE where the encoder $f(\boldsymbol{x} \mid \Theta)$ does not produce directly decoded data $\mathbf{z}$ as $f(\boldsymbol{x} \mid \Theta) = \mathbf{z}$. It actually produces mean vector $\mu(\boldsymbol{x})$ and covariance

matrix $\Sigma(x)$ belonging to x instead. In this research, $\mu(x)$ and $\Sigma(x)$ are flattened into an array of neurons output layer of the encoder $f(x \mid \Theta)$.

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

The actual decoded data z is calculated randomly from $\mu(x)$ and $\Sigma(x)$ along with a random vector r.

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

Where r follows standard Gaussian distribution with mean vector 0 and identity covariance matrix I and each element of $(\Sigma(x))^{1/2}$ is squared root of the corresponding element of $\Sigma(x)$. This is an excellent invention in traditional literature which made the calculation of Kullback-Leibler divergence much easier without loss of information.

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

$$b_{\text{AVA}}(\Phi, \Psi) = b_{\text{GAN}}(\Phi, \Psi) - \frac{1}{2}\|x - x'\|^2$$

$$= \log\left(d(x|\Psi)\right) + \log\left(1 - d(g(z|\Phi)|\Psi)\right) - \frac{1}{2}\|x - x'\|^2 \tag{9}$$

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

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

In a reverse causality effect relationship in which the unique output neuron of discriminator DNN is cause of all output neurons of decoder DNN as shown in Figure 3.
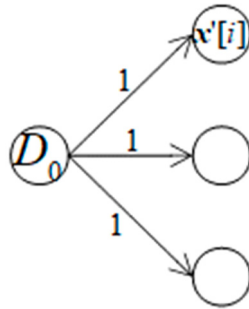

Figure 3. Reverse causality effect relationship between discriminator DNN and decoder DNN.

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

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

$$\text{bias}[i] = x[i] - x'[i]$$

Because the balance function $b_{\text{AVA}}(\Phi, \Psi)$ aims to improve the decoder $g(z \mid \Phi)$, it is possible to improve the encoder $f(x \mid \Theta)$ by similar technique with note that output of encoder is mean vector $\mu(x)$ and covariance matrix $\Sigma(x)$. In this research, another balance function $B_{\text{AVA}}(\Theta, \Lambda)$ is proposed to assess reliability of the mean vector $\mu(x)$ because $\mu(x)$ is most important to randomize z and $\mu(x)$ is linear. Let $D(\mu(x) \mid \Lambda)$ be discrimination function for encoder DNN from $\mu(x)$ to range [0, 1] in which $D(\mu(x) \mid \Lambda)$ can distinguish fake mean $\mu(x)$ from real mean $\mu(x')$. Obviously, $D(\mu(x) \mid \Lambda)$ is implemented by

a so-called encoding discriminator DNN whose weights are Λ with note that this DNN has only one output neuron denoted D_0. The balance function $B_{AVA}(\Theta, \Lambda)$ is specified as follows:

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

Note,

$$g(z|\Phi) = x'$$

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

$$l_{AVA}(\Theta, \Phi, \Psi, \Lambda)$$
$$= \frac{1}{2}\|x - x'\|^2 + KL\big(\mu(x), \Sigma(x)|N(0, I)\big) + \log\big(1 - d(x'|\Psi)\big) \tag{12}$$
$$+ \log\big(1 - D(\mu(x')|\Lambda)\big)$$

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

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

Where $a_D(.)$ and $a'_D(.)$ are activation function of the discriminator $D(\mu(x) \mid \Lambda)$ and its derivative, respectively.

The encoder parameter Θ is consisted of two separated parts Θ_μ and Θ_Σ because the output of encoder $f(x \mid \Theta)$ is consisted of mean vector $\mu(x)$ and covariance matrix $\Sigma(x)$.

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

Where,

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

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

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

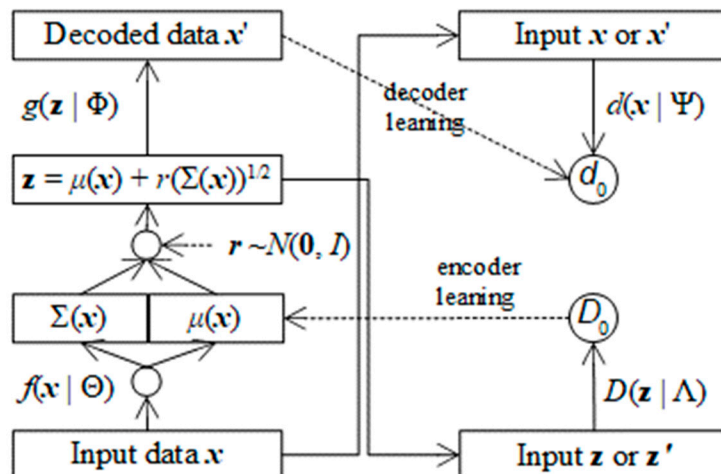Figure 4 shows AVA architecture with support of assessing encoder.

Figure 4. AVA architecture with support of encoder assessing.

Similarly, the balance function $B_{\text{AVA}}(\Phi, \Lambda)$ can lean forward encoding task for improving accuracy of encoder $f(\boldsymbol{x} \mid \Theta)$ by concerning the error of original mean $\mu(\boldsymbol{x})$ and decoded data mean $\mu(\boldsymbol{x}')$ as follows:

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

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

$$\Lambda = \Lambda + \gamma \left(\frac{a_D'\big(D(\mu(\boldsymbol{x})|\Lambda)\big)}{D(\mu(\boldsymbol{x})|\Lambda)} - \frac{a_D'\big(D(\mu(\boldsymbol{x}')|\Lambda)\big)}{1 - D(\mu(\boldsymbol{x}')|\Lambda)} \right.$$

$$\left. + a_D'(D_0) \sum_i (\mu(\boldsymbol{x})[i] - \mu(\boldsymbol{x}')[i]) a_g'(\mu(\boldsymbol{x}')[i]) \right) \qquad (16)$$

These variants of AVA are summarized, and their tests are described in the next section.

3. Experimental Results and Discussions

In this experiment, AVA is tested with VAE and GAN but there are 5 versions of AVA such as AVA1, AVA2, AVA3, AVA4, and AVA5. Recall that AVA1 is normal version of AVA whose parameters are listed as follows:

$$\Theta = \Theta - \gamma \left(\mu(\boldsymbol{x}) - \frac{1}{2}\big(\Sigma(\boldsymbol{x})\big)^{-1} + \frac{1}{2}I \right) a_f'(\boldsymbol{x})$$

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

$$\Psi = \Psi + \gamma \left(\frac{a_d'\big(d(\boldsymbol{x}|\Psi)\big)}{d(\boldsymbol{x}|\Psi)} - \frac{a_d'\big(d(\boldsymbol{x}'|\Psi)\big)}{1 - d(\boldsymbol{x}'|\Psi)} \right)$$

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

$$\Theta = \Theta - \gamma \left(\mu(\boldsymbol{x}) - \frac{1}{2}\big(\Sigma(\boldsymbol{x})\big)^{-1} + \frac{1}{2}I \right) a_f'(\boldsymbol{x})$$

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

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

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

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

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

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

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

AVA4 is a variant of AVA3 along with leaning forward improving accuracy of encoder $f(x \mid \Theta)$ like AVA2. Its parameters are listed as follows:

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

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

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

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

AVA5 is the last one which supports all functions such as decoder supervising, leaning decoder, encoder supervising, and leaning encoder.

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

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

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

$$\Lambda = \Lambda + \gamma\left(\frac{a'_D\big(D(\mu(x)|\Lambda)\big)}{D(\mu(x)|\Lambda)} - \frac{a'_D\big(D(\mu(x')|\Lambda)\big)}{1 - D(\mu(x')|\Lambda)} + a'_d(d_0)\sum_i (x[i] - x'[i]) a'_g(x'[i])\right.$$
$$\left. + a'_D(D_0)\sum_i (\mu(x)[i] - \mu(x')[i]) a'_g(\mu(x')[i])\right)$$

The experiment is performed on a laptop with CPU AMD64 4 processors, 4GB RAM, Windows 10, and Java 15 given dataset is a set of thirty-six 100x64 images. It is necessary to define how good deep generative models (DGMs) such as VAE, GAN, and AVA are. Let imageGen be the best image generated by a deep generative model (DGM), which is compared with the ith image denoted

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

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

Because the product $d_{ij}(1-d_{ij})$ is second-order function, its maximizer exists and so, the generated image whose product $d_{ij}(1-d_{ij})$ is larger is the better one when its balance is more stable. As a result, let balance metric (BM) be the metric to assess 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 is, the better the generated image is, the better the balance of similarity and diversity is. The overall BM of a DGM is average BM[i] over N=10 test images as follows:

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

Where,

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

Recall that the larger the BM is, the better the DGM is. However, if the similarity quality is concern, the DGM will be better when its BM is smaller because a small BM implies good similarity in this test with note that such small BM implies small distance or small diversity. Therefore, the DGM whose BM is largest or smallest is preeminent. If the DGM whose BM is largest, it is best in balance of similarity and diversity. If the DGM whose BM is smallest, it is best in similarity. Both maximum and minimum of BM, which indicates both balance quality and similarity quality, respectively, are concerned in this test but balance quality with large 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 stochastic gradient descent (SGD) algorithm is affected by learning rate and the accuracy of AVA varies a little bit within a learning rate because of randomizing encoded data **z** in VAE algorithm. Table 1 shows 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 BM values of AVAs, VAE, and GAN with 9 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 shows BM means, BM maxima, BM minima, and BM standard deviations of AVAs, VAE, and GAN.

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

Note that VAE and GAN represent a pole of similarity quality and a pole of balance quality, respectively. From experimental results shown in Table 3, AVA5 is the best DGM because it gains highest BM mean (0.2096) which is also larger than BM mean (0.2089) of the pole GAN. It is easy to explain this result because AVA5 is the one which improves both decoding task and encoding task when it embeds both decoder discriminator and encoder discriminator as well as both leaning decoder and leaning encoder. Moreover, both AVA1 and AV2 are better than GAN because their BM means (0.2093, 0.2092) are larger than BM mean (0.2089) of GAN. If the similarity quality is concerned, AVA3 is the best DGM because it gains the lowest BM mean (0.1202) which is also larger than BM mean (0.1207) of the pole VAE. It is easy to explain this result because AVA3 is the one which improves encoding task when it embeds encoder discriminator. Moreover, AVA1, which is a fair AVA because it embeds decoder discriminator but it does not support leaning decoder, is better than the pole GAN whereas AVA3, which is a fair AVA because it embeds encoder discriminator but it does not support 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 leaning decoder and leaning encoder. Therefore, about BM mean which the most important metrics, all AVA variants are better than traditional DGMs such as VAE and GAN with regards to both similarity quality and balance quality.

Although BM mean is the most important metrics, it is necessary to check other metrics related to extreme values which are BM maximum and BM minimum where BM maximum implies best balance quality and BM minimum implies 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 decoder and encoder. AVA2 and AVA5 are better DGMs about extreme balance quality because their BM maxima (0.2329, 0.2326) are larger than BM maximum (0.2325) of GAN. Similarly, AVA3 and AVA4 are better DGMs about extreme similarity quality because their BM minima (0.0546, 0.0546) are smaller than BM minimum (0.0568) of VAE. Therefore, about BM extreme values, AVA variants are better than traditional DGMs such as VAE and GAN with regards to both similarity quality and balance quality.

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

Figure 5 depicts BM means, BM maxima, BM minima, and BM standard deviations of AVAs, VAE, and GAN by charts.
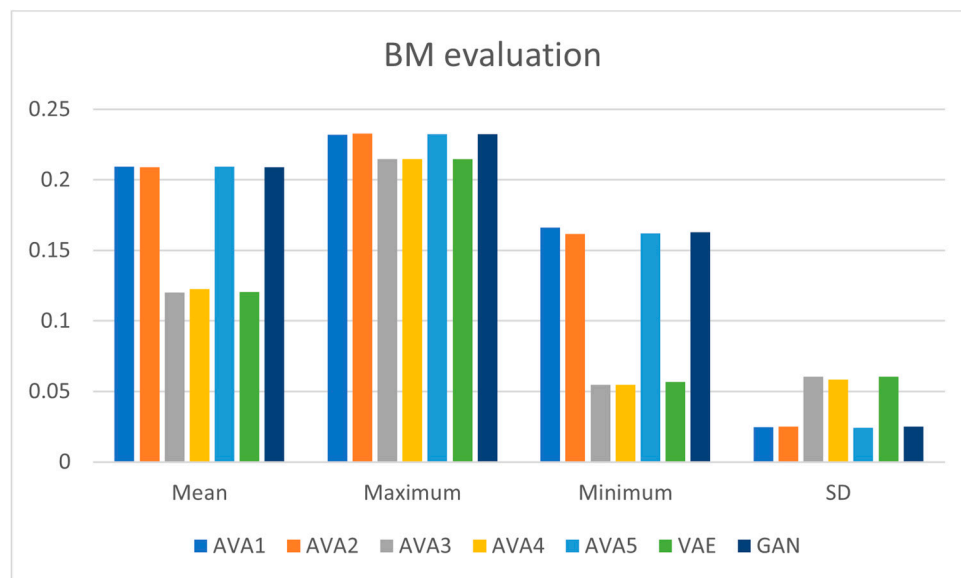

Figure 5. Evaluation of AVAs, VAE, and GAN.

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

4. Conclusions

It is undoubtful that AVA is better than 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 reliability of data. We think that VAE and GAN are solid models in both theory and practice when their mathematical foundation cannot be changed or transformed but it is still possible to improve them by modifications or combinations as well as applying them into specific applications where their strong points are brought into play. In applications related to raster data like image, VAE has a drawback of consuming much memory because probabilistic distribution

represents entire image whereas some other deep generative models focus on representing product of many conditional probabilistic distributions for pixels. However, this pixel approach for modeling pixels by recurrent neural network does not consume less memory but it is significantly useful to fill in or recover smaller damaged areas in a bigger image. In the future trend, we try to apply the pixel approach into AVA, for instance, AVA processes a big image block by block and then, every block is modeled by conditional probability distribution with recurrent neural network as well as long short-term memory network.

References

Ahmad, B., Sun, J., You, Q., Palade, V., & Mao, Z. (2022, January 21). Brain Tumor Classifcation Using a Combination of Variational Autoencoders and Generative Adversarial Networks. *biomedicines, 10*(2), 1-19. doi:10.3390/biomedicines10020223

Doersch, C. (2016, January 3). Tutorial on Variational Autoencoders. *arXiv preprint*. Retrieved from https://arxiv.org/abs/1606.05908

Goodfellow, I., Pouget-Abadie, J., Mirza, M., Xu, B., Warde-Farley, D., Ozair, S., . . . Bengio, Y. (2014). Generative Adversarial Nets. In Z. Ghahramani, M. Welling, C. Cortes, N. Lawrence, & K. Weinberger (Ed.), *Advances in Neural Information Processing Systems 27 (NIPS 2014). 27*. Montreal: NeurIPS. Retrieved from https://proceedings.neurips.cc/paper_files/paper/2014/file/5ca3e9b122f61f8f06494c97b1afccf3-Paper.pdf

Kingma, D. P., & Welling, M. (2022, December 10). Auto-Encoding Variational Bayes. *arXiv Preprint*, 1-14. doi:10.48550/arXiv.1312.6114

Larsen, A. B., Sønderby, S. K., Larochelle, H., & Winther, O. (2016). Autoencoding beyond pixels using a learned similarity metric. *International conference on machine learning. 48*, pp. 1558-1566. New York: JMLR. Retrieved from http://proceedings.mlr.press/v48/larsen16.pdf

Mescheder, L., Nowozin, S., & Geiger, A. (2017). Adversarial Variational Bayes: Unifying Variational Autoencoders and Generative Adversarial Networks. *Proceedings of the 34 th International Conference on Machine. 70*, pp. 2391-2400. Sydney: PMLR. Retrieved from http://proceedings.mlr.press/v70/mescheder17a/mescheder17a.pdf

Miolane, N., Poitevin, F., & Li, Y.-T. (2020). Estimation of Orientation and Camera Parameters from Cryo-Electron Microscopy Images with Variational Autoencoders and Generative Adversarial. *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops* (pp. 970-971). New Orleans: IEEE. Retrieved 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

Nguyen, L. (2015). *Matrix Analysis and Calculus* (1st ed.). (C. Evans, Ed.) Hanoi, Vietnam: Lambert Academic Publishing. Retrieved March 3, 2014, from https://www.shuyuan.sg/store/gb/book/matrix-analysis-and-calculus/isbn/978-3-659-69400-4

Rosca, M., Lakshminarayanan, B., Warde-Farley, D., & Mohamed, S. (2017, October). Variational Approaches for Auto-Encoding Generative Adversarial Networks. *arXiv preprint*. Retrieved from https://arxiv.org/abs/1706.04987