Master Project: Simple Training Objective of Variational Autoencoder with Detailed Reconstructions

Student Name: Hong Khay Boon

Student ID: TP069358

Degree and Programme: MSc in Artificial Intelligence

Supervisor: Assoc. Prof. Dr. Sivakumar Vengusamy

2nd Marker: Dr. Vazeerudeen Hameed

Deadline: 21st July 2023

Abstract

Variational Autoencoders (VAE) are unsupervised model that learns the data distribution and attempt to reconstruct the original data by using compressed latent variables. By incorporating Variational Bayes methods, VAE can learn regularized posterior within the latent space and thus enable the interpolation between two existing datapoint. However, VAE often suffers from blurry reconstructions and thus is often inferior to Generative Adversarial Networks (GAN) in practice, but VAE is in general easier to configure and converge than GAN. In this report, we improve VAE model by creating a VAE architecture that requires no use of hyperparameter, a general architecture that works on most image datasets, while at the same time creating sharper reconstructions with low dimensional latent. Based on the small scale of this Master degree project, we do not dive into very deep networks and only implement models that are 5 – 8 layers deep, but our novelty lies in the use of thorough residual connections separately within the encoder and decoder of the VAE, thus effectively make use of information on every stage when computing latent variables.

Keywords: ***Deep Learning, Variational Bayes, Variational Autoencoder, VAE, Evidence Lower Bound, Unsupervised Learning, Computer Vision, Generative Artificial Intelligence.***

Table of Contents

Introduction

Recent advancements in deep learning have made itself known to more people, especially those who are fluent in technology but know little about machine learning. This can be seen more obviously with the increased popularity of the Chat-GPT application developed by OpenAI and the various text-to-image models such as DALL-E 2 and Stable Diffusion. Variational Autoencoder (VAE) and Generative Adversarial Network (GAN) are some of the basic building blocks within the architectures of ChatGPT and Stable Diffusion. They are also two popular deep learning architectures capable of generating images in an unsupervised manner. While GAN is trained on the objective of generating photorealistic images from gaussian noises, VAE is aimed for reconstructing images using a small bottleneck.

During the training process of GAN, a generator and a discriminator are trained simultaneously (Goodfellow et al., 2014). At first, a large image dataset is provided to the model, and the discriminator is supposed to give a verdict on whether an image is real (i.e., taken from the dataset) or fake (generated by the generator), this verdict is often a number from 0 to 1 representing the probability of the image being real. Then, a generator should generate an image from a vector of random noise, with the objective of creating image so realistic that it could fool the discriminator into thinking the generated image is real. The ideal training result should be the discriminator give a verdict of 0.5 to every input, and the generator could generate very realistic image comparable to the original dataset. In practice, however, this is hard to achieve as there is no fix convergence scenario to seek for, only the adversarial shift of the states of generator and discriminator. But if it is possible, such model will be powerful for many practical tasks. It is a major component in most of the Stable Diffusion models as well.

On the other hand, VAE (and general autoencoder) act more like a JPG/PNG compression algorithm (Kingma & Welling, 2019). It typically consists of an encoder that transforms an image to a low dimensional space, then a decoder to upscale such compressed data back to the same dimension as the original image. A main difference between VAE and autoencoders is that instead of generate a deterministic latent variable for each data, VAE generate the density of data. It is done by encode the data into a mean vector and variance vector, then a latent variable is sampled from a multivariate normal distribution. The reason behind this is to prevent the model to learn a lookup table that generalizes badly. By learning

a distribution for each data point, VAE hopes to utilize the overlap between similar datapoint and thus recreating plausible interpolated images. Theoretically, dimension reduction model like this is considered useful when the bandwidth of sending data between places to places is very low: For example, communication between a NASA space station on the moon to earth, a satellite image should be compressed to a very low dimension, send to earth, then use a decoder to decompressed into an image. However, in practice, VAE is more commonly used in the context of anomaly detection since it can catch on something abnormal going on within the image, while not necessarily reconstruct the entire image accurately. Considering text to image models, VAE is oftentimes optional, and could generate more vibrant and crisper images when incorporated into a Stable Diffusion model (Ma, 2023).

Originally, this master report is to study the architectures of text to image models. But due to the time limitation assigned to this project, we step back and turn to study VAE. This is because while GAN is more powerful, VAE on the other hand is easier to train and works generally well when the requirements on image sharpness is not high. By pursuing this direction, we aimed to create techniques that can generate sharper images, thus resolve this issue that is inferior to GAN.

The report will revolve around creating a VAE that require no hyperparameter, hence the use of Optimal Sigma objective will be introduced in the Background and Literature Review section and frequently mentioned afterward. On the other hand, while we will discuss about VAE with hyperparameters (Beta-VAE, InfoVAE), we will not utilize their structure in our model design. Following sections will be our Research Methodology, then the implementation of known models and our models, finally we report our experiment results and visualizations, together with a conclusion to this project.

Aim, Objectives, Problem Statements

The general aim for this master project is to make VAE easier to train, thus promoting the usage of VAE in current deep learning landscape. Although there are many types of autoencoders including denoising autoencoder and sparse autoencoder, our study will focus on VAE that can compress data well, hence we will not consider VAE architecture that has latent dimension similar to the original data dimension (such as denoising autoencoder), but we will consider latent dimension as little as 10% of original dimension to achieve compression.

Our objectives can be described as:

- Design a VAE training objective requiring no hyperparameter tuning.
- Little to no additional changes when using different datasets.
- Increase output sharpness by concatenating different level of information into the latent space.

Current VAE frameworks often suffer from blurry reconstructions, mainly caused by the mean square loss function (Isola et al., 2016) or the unbalanced optimization between reconstruction and latent losses, as talked about in beta-VAE (Higgins et al., 2017). VAE also tend to have posterior collapse issue, which means the encoder does not model the latent distribution where decoder do all the job on reconstructing images. Moreover, disentanglement is an important feature in the latent space, as in each dimension of the latent space are controlling a salient feature from the image, where different dimension should only have minimal correlation to each other.

To our knowledge, there is no model that utilize a fully skip connection within encoder and decoder separately. Dieng et al. (2019) created a VAE that employs skip connection only in the decoder of the VAE, thus creating an asymmetrical VAE. Moreover, the U-Net VAE hybrid architecture from Li et al. (2020) contains skip connection, but such skip connection can pass information across the bottleneck, which doesn't suit our VAE framework at all since we are looking to compress information, there should not be any information coming from encoder to decoder except of the bottleneck. On the other hand, a flow-based framework in most time used the same dimension for latent vector as original data as it is not intended to compress data, but to construct a complex multimodal distribution in the latent space. Su & Wu (2018) introduced a hybrid model that incorporate flow model into

the latent space of a VAE, that essentially compress the data into latent space, then use flow to model multimodal distribution on the latent space, which did achieve very good result.

Background and Literature Review

Variational Autoencoder (VAE) is an encoder-decoder architecture focusing on extracting important features from an image with the encoder, then additionally recover the original image by using the decoder. The decoder serves as a mean of illustrating the preservation of data from the encoder. Different from autoencoder, a VAE model will try to model the distribution of each datapoint differently as a distribution in the latent space. When no data point is given, the decoder of a VAE might generate a sample by choosing a latent variable from latent space. The way of choosing is to assume a distribution, which is commonly assume to be standard multivariate gaussian. This unconditional distribution is called "prior" and is written as $p(z) \sim N(z; 0, I)$. Let x be a datapoint, q be the encoder and ϕ the parameters of the encoder, then $q_\phi(x)$ represents the distribution of latent variables in the latent space, it is typically called the "posterior" of x, written as $p(z|x)$. When the posterior is assumed to be gaussian, it is normally modeled by $p(z|x) \sim N(z;\ \mu_\phi(x), \sigma^2_\phi(x))$ where $(\mu_\phi(x), \sigma_\phi(x)) = q_\phi(x)$. To make sure posterior of each datapoint has reasonable values (near the origin), VAE objective includes the proximity of posterior to the prior. For instance, Figure 1 shows the clusters of 10 digits from MNIST dataset once a VAE is trained on it, when not considering the labels, all latent variables are centered around the origin, adhering to the prior, but each individual label actually follows its own slightly shifted posterior mean. Nearer clusters also shown similar behavior, as digit 3 and digit 8 are near because their written shape is similar, but digit 5 and digit 9 are further from each other.

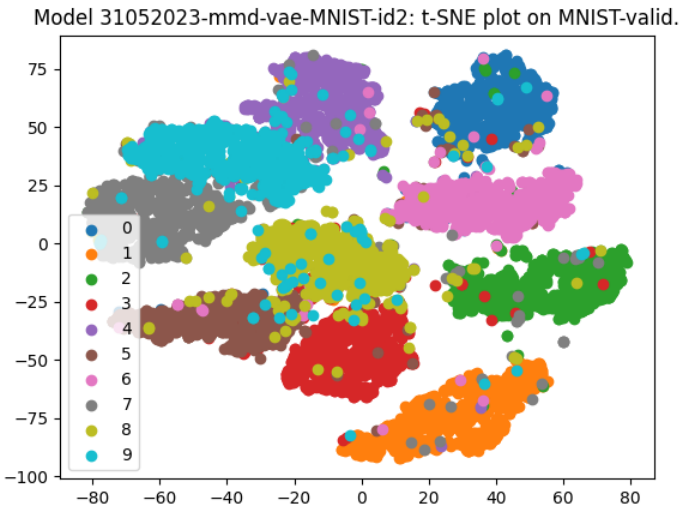

Figure 1 Sampling from posteriors of 10,000 MNIST images (validation dataset)

VAEs are always being compared with Generative Adversarial Networks (GAN) since they share some similarities, but GAN's generator and discriminator are not always working sequentially as in VAE's encoder-decoder pair, nor it can encoder an image to a compressed form. In VAE, one has the benefit of decode the latent vector back to original image, so it is acting mainly as a data compression model.

VAE Objective

The VAE training objective consists of two parts: Reconstruction loss and Latent loss. However, the original objective does not have this form. Let p_θ be the decoder of VAE with θ as its parameters. To optimize the model, one has to maximize the marginal likelihood $p_\theta(x)$ of every data point x, which is given by $p_\theta(x) = \int p_\theta(x,z)dz$ where the integral is over all latent vectors (Kingma & Welling, 2019). Since this integration is intractable as it needs to evaluate $p_\theta(x,z)$ uncountably many times, one has to use an approximation to serve as a lower bound for the marginal likelihood. At this stage, we have already defined $q_\phi(z|x)$ as posterior distribution of x, considering it to be a posterior estimated by VAE's encoder, the "true posterior" of the model is defined as $p_\theta(z|x)$, with the Bayesian relation $p_\theta(z|x) = \frac{p_\theta(x|z)p(z)}{p_\theta(x)}$. The ideal situation would be the "approximate" posterior $q_\phi(z|x)$ match the true posterior $p_\theta(z|x)$ exactly. This motivates the tractable Evidence Lower Bound (ELBO) enabling training VAE in an end-to-end manner.

ELBO, as the name suggests, is a lower bound to the intractable marginal log-likelihood: $\log p_\theta(x)$. It can be expressed as

$$\log p_\theta(x) = E_{q_\phi(z|x)}\left[\log\left[\frac{p_\theta(x,z)}{q_\phi(z|x)}\right]\right] + D_{KL}\left(q_\phi(z|x)||p_\theta(z|x)\right).$$

The first term is tractable in a VAE and the second term is the KL divergence of approximate posterior to true posterior. Since KL divergence is nonnegative, we have the following inequality for ELBO ($L_{\theta,\phi}(x)$):

$$L_{\theta,\phi}(x) = E_{q_\phi(z|x)}\left[\log\left[\frac{p_\theta(x,z)}{q_\phi(z|x)}\right]\right] \leq \log p_\theta(x).$$

Therefore, optimizing the ELBO can serve similar purpose as optimizing the true objective, and the gap between them is the divergence between the approximate and true posterior.

Common assumptions in VAE

The prior and posterior of a VAE are normally assumed to be gaussian distribution because it is easy to model. The prior $p(z)$ is assumed to be a standard gaussian distribution $N(z; 0, I)$, and the posterior $q_\phi(z|x)$ is a gaussian distribution where its mean and variance μ, σ^2 are functions of x. More specifically, the encoder of VAE will take an input x and output the mean and variance of the approximate posterior.

The data distribution conditioned on latent variable $p_\theta(x|z)$ is assumed to be a gaussian as well, but normally with unit variance (Higgins et al., 2017; Kingma & Welling, 2019). This is because unlike latent variable, the final reconstructed data $\hat{x}$ will be just the mean of the distribution $p_\theta(x|z)$. If we take a look at the ELBO with these assumptions in mind, it will look like this:

$$-L_{\phi,\theta}(x) = \frac{1}{2}||x - \hat{x}||^2 + \sum\left[\log \sigma_\phi(x) + \frac{1}{2}(\mu_\phi(x)^2 + \sigma_\phi(x)^2 - 1)\right]$$

The first L2 loss is simply the reconstruction loss of VAE, the second term is the latent loss: KL divergence of the distribution $N(z;\ \mu_\phi(x), \sigma_\phi(x)^2)$ and the standard gaussian distribution, in its closed form. However, this assumption is not even close to normal dataset. Images are typically with pixel values in the range of 0 to 1, so unit variance will make the data distribution too volatile. For a more explicit comparison, ImageNet has variances of [0.0524, 0.0502, 0.0506] for each color channel (Deng et al., 2009). This assumption often leads to unbalanced roles of reconstruction and latent losses in a VAE objective. Therefore, Higgins et al. (2017) suggested multiplying a constant with latent loss to balance the objective, thus created the beta-VAE model. This method serves as a mean of solving dimensional imbalance of data and latent variable, as data are usually of higher dimension, poor tuning of beta can lead to the model focusing too much on optimizing reconstruction loss. Burgess et al. (2018) further improves beta-VAE by adding a controlling term in latent loss, so to prevent posterior collapse and ensure disentanglement at the same time. However, these settings require manually choosing hyperparameters, which in practice will need more resource for hyperparameter tuning in order to find a suitable model.

In below subsections, we layout most existing VAE architectures, and we will append (Y) or (N) after the title to indicate whether we incorporate their technique into our model (yes or no). We hope the reader can keep in mind the sections with (Y) to better understand our model design.

InfoVAE – Maximizing Information between data and latent (N)

With regards to beta-VAE, Zhao et al. (2018) published a variant of VAE – InfoVAE - trained with maximizing mutual information of the data and latent variables. The InfoVAE model does not alter the VAE architecture but only modifies the objective. To alleviate the posterior collapse problem, they introduce mutual information $I_q(x; z)$ into the original ELBO:

$$L_{InfoVAE} = -E_{q(z)}\left[D_{KL}\left(q_\phi(x|z)||p_\theta(x|z)\right)\right] - \lambda D_{KL}\left(q_\phi(z)||p(z)\right) + \alpha I_q(x; z).$$

Note that the model reward mutual information between x and z, or in other words, make them become dependent on each other. This equation can be rewritten into a more familiar form:

$$L_{InfoVAE} \equiv E[\log p_\theta(x|z)] - (1-\alpha)D_{KL}\left(q_\phi(z|x)||p(z)\right) - (\alpha+\lambda-1)D_{KL}\left(q_\phi(z)||p(z)\right).$$

The first two terms are the negative version of reconstruction loss and latent loss (without the $1-\alpha$ factor). The last term is not easy to compute and often times require Monte-Carlo estimation. When $\alpha = 0, \lambda = 1$, it falls back to vanilla VAE objective. Moreover, when $\alpha + \lambda - 1 = 0$, there is one free parameter left and it becomes a beta-VAE with $\beta = 1 - \alpha$. It can be proven the D_{KL} divergence of the last term can be switched to other divergence function while still preserving the training objective, one of which is the Maximum-Mean Discrepancy (MMD) function. MMD is more commonly used when training an InfoVAE because its Monte-Carlo estimation is relatively easier. In practice, we found that using $\alpha = 1, \lambda = 500$ produce good results.

PixelVAE, an autoregressive model (N)

While generating images using deconvolutional layers are standard practices in implementing a VAE, autoregressive generation is another way of creating images pixel by pixel. Oord, Kalchbrenner, & Kavukcuoglu (2016) proposed PixelRNN and PixelCNN to model natural images by assuming left-to-right then top-down pixel ordering in an image. Such models can look at an occluded or an unfinished canvas, then proceed to generate new pixels below existing ones to complete the image (Figure 2). PixelRNN while powerful, it would require generating pixel conditioned on unbounded number of previous pixels, thus PixelCNN is proposed along with PixelRNN with a large but unbounded convolutional

kernel. To ensure pixel generation does not conditioned on future pixels, PixelCNN uses masked convolution. For instance, a typical 3 by 3 convolution uses a weight of same shape consisting of 9 numbers, while the center is the objective, the right, bottom-left, bottom and bottom-right pixels are considered future pixels, so mask convolution will nullify the weights from the future 4 pixels and only uses the previous 4 pixels, and the same logic goes for a general 5 by 5 or 7 by 7 kernels.

Figure 2 PixelRNN generating bottom half of images (taken from Oord, Kalchbrenner, & Kavukcuoglu (2016))

While PixelRNN and PixelCNN are powerful image generators, they are computationally expensive in both training and inference stages. Oord, Kalchbrenner, Vinyals, et al. (2016) proposed image generation by PixelCNN conditioned on image labels or other form of vectors in lesser magnitude. More specifically, a decoder modelled by PixelCNN architecture can generate images pixel-by-pixel by only using a one-hot encoding vector used in a multi-label image dataset (For example, CIFAR10 dataset). The authors stated that such model can be constructed starting from a standard convolutional autoencoder, then replace the decoder with PixelCNN architecture before training with the autoencoder objective. Since PixelCNN is a model that models pixel values using conditional distribution, a single latent variable can generate different images, thus showing diversity and capability of the model.

The above two models are both image generation models, but they are not related to VAE because the original PixelRNN/PixelCNN do not use a latent variable, and the conditional PixelCNN use a fixed latent variable. Gulrajani et al. (2016) proposed PixelVAE that is an improvement from previous two models: PixelVAE can be trained in an unsupervised manner because it does not need an explicit label for each image, it can also learn the overlaps between different objects so that interpolation between different classes of images is viable. Comparing to a vanilla VAE that assumes independent relation between

reconstructed pixels, PixelVAE assumes hierarchical relation between pixels, thus possibly make the reconstructed images more coherent. Recent advancement has been made by Sadeghi et al. (2019) with the proposed PixelVAE++ model. It uses a three-layer hierarchical latent layers and share parameters of part of decoder to encoder. It has achieved state-of-the-art performance with the model capable of capturing local and global details.

Tackling Posterior Collapse by Encoder Aggressive Training (Y)

Posterior collapse is often a major problem discouraging people from choosing to use VAE. He et al. (2019) argued that posterior collapse in most time happened from uneven training for the encoder and decoder, which can be quantified by evaluating two posteriors: At the end of training of each epoch, the authors observe the means of approximate posterior $q_\phi(z|x)$ and true posterior $p_\theta(z|x)$. They then defined two types of collapse: Model collapse when mean of $p_\theta(z|x) \approx 0$, and Inference collapse when mean of $q_\phi(z|x) \approx 0$. Upon experiments, the authors found that posterior collapse is caused by approximated posterior lags behind true posterior during initial training stage. Below shows training result from a VAE with latent dimension 1 (Figure 3).

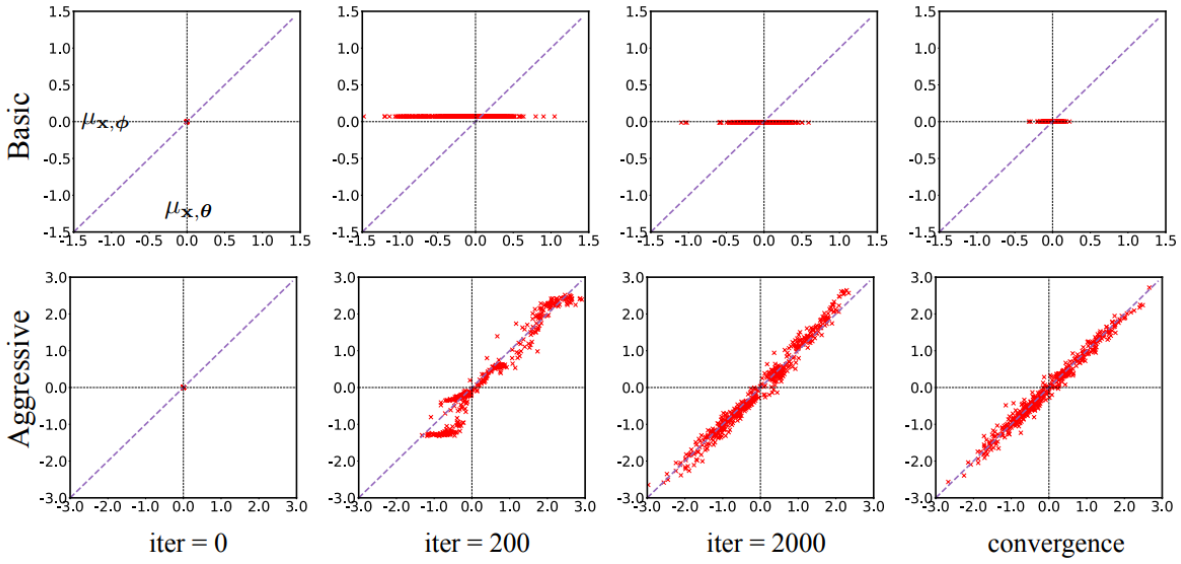

Figure 3 Basic training vs Aggressive training on Encoder (He et al., 2019)

To remedy this situation, He et al. (2019) proposed splitting the training of encoder and decoder, while **aggressively train** the encoder. In practice, the optimal training regime is to repeatedly train encoder for 5 times (over the training set) and then train the decoder for 1 time, all these considered one epoch, then repeat. There is a concern on whether this will significantly increase training time, but as long as the model's approximate posterior and true

posterior matched, the training can revert to the standard process after a few epochs. To measure this, one can calculate the mutual information (Zhao et al., 2018) between the data and latent vectors: Mutual information should keep increasing during initial stage of training, once it stops increasing the training can revert to standard procedure. Employing this training strategy, He et al. (2019) obtain non-collapse latent as shown in the "Aggressive" part of Figure 3. This concept has proven to be effective when training on text and image datasets.

VAE with Discrete Latent (N)

Instead of meddling with the usual settings of VAE, Oord et al. (2017) proposed a new framework for VAE which is called VQ-VAE. This version of VAE uses a discrete latent space to encode images as they believe natural images often contains information that are block-like and discrete, often times easier to be encoded in a non-continuous manner. Moreover, they proposed a new method for training VQ-VAE end to end so that the gradient can be backpropagated through the discrete bottleneck. Their experiments showed state-of-the-art quality when compare to other high-capability VAEs including PixelCNN.

A latter iteration of VQ-VAE improved upon itself with reconstructed images of higher coherence and fidelity (Razavi et al., 2019). Both versions use discrete latent vectors as a mean to resolve posterior collapse that are commonly observed in a standard VAE (i.e., with continuous latent), and they autoregressively create latent variables, unlike PixelCNN that generates similarly, but in the pixel space.

Optimal Sigma VAE - Training without hyperparameter (Y)

A hyperparameter-free method of training a VAE can be realized by revising the distribution of the reconstructed data, namely $p_\theta(x|z)$. We have seen ImageNet normally uses a variance of 0.05 for their datasets, which can be generally true for any other dataset as ImageNet is a big image dataset itself. We found the implementation of Optimal Sigma VAE (Rybkin et al., 2021) matches our objective: They assume a **non-learnable global reconstruction variance** σ_θ^2 on the conditional data distribution $p_\theta(x|z) \sim N(x;\ \mu_\theta(z), \sigma_\theta^2)$, in the sense that while σ_θ^2 is non-constant, it is calculated directly from the data minibatch, hence non-learnable. Its ELBO can be expressed as

$$-L_{\phi,\theta,optimal}(x) \equiv D \ln \sigma_\theta + \frac{1}{2\sigma_\theta^2}||x - \hat{x}||^2 + \sum\left[\log \sigma_\phi(x) + \frac{1}{2}(\mu_\phi(x)^2 + \sigma_\phi(x)^2 - 1)\right],$$

In the above expression, D is the dimension of the image. For example, Celebrity Attribute dataset consists of RGB images of dimension 64 by 64, hence $D_{CelebA} = 3 * 64 *$

$64 = 12288$. It can be seen that the global variance σ_θ^2 only affects the reconstruction loss, as the latent loss only come from the encoder. This formula is a generalization of the standard objective with unit variance, because they agree when $\sigma_\theta^2 = 1$. In most of the cases, σ_θ^2 might takes the value within $0.03 \sim 0.10$ depending on the dataset, which makes the coefficient of reconstruction loss $\frac{1}{2\sigma_\theta^2}$ large, roughly within $5 \sim 17$. The heavy focus on reconstruction loss thus makes VAE optimize it more, and put a minimal focus on latent loss. This in general will improve reconstruction quality and decrease posterior collapse because latent loss will be larger. Looking back, beta-VAE are essentially doing the same, but it uses manual/heuristic way to determine the number beta. If we compare the two objectives, it is not hard to notice that when $\beta = 2\sigma_\theta^2$,

$$L_{\phi,\theta}(x) = 2\sigma_\theta^2 L_{\phi,\theta,optimal}(x),$$

This also shows that one does not need to tune beta, there is a more rigid way to choosing it. Rybkin et al. (2021) managed to use the relation $\sigma_\theta^2 = \frac{\beta}{2}$ to show the equivalence between beta-VAE and Optimal Sigma VAE. Since the ELBO for beta-VAE has smaller scale comparing to Optimal Sigma VAE, one has to use larger learning rate when training a beta-VAE to strike back the balance for gradient backpropagation, that is about the main difference when training these two versions of VAE.

Research Methodology

This project will be mostly experimental as we were studying different existing VAE architectures on preset datasets. After examine their effectiveness, we will combine useful mechanisms and create new architectures. While we were not obtaining state-of-the-art performances since our model is not as deep as recently proposed models, we aimed to uncover new paths on training VAE so the model can generate acceptable reconstructions while being easy to train. We were conducting our experiments with the following objectives in mind: Using a training regime without unnecessary hyperparameters settings, capturing local and global details, and high compression rate.

Data and Models

Based on the limitation of time during this project, we choose to not collect image dataset ourselves, but use publicly available datasets. These datasets are MNIST Handwritten Digits (Deng, 2012), Fashion MNIST (Xiao et al., 2017), CIFAR10 and CIFAR100 (Krizhevsky, 2009), and lastly Celebrity Face Attributes (aka CelebA dataset) (Liu et al., 2015). MNIST digits dataset is a classic and simple dataset for benchmarking machine learning model. As current image classifiers often achieving more than 99% accuracy on MNIST digits dataset, Fashion MNIST dataset serves as a harder alternative: It has the same image format as MNIST digits, but contains more complicated features. To further scrutinize current VAE models, we choose CIFAR10 and CIFAR100 datasets with complexity higher than both MNIST datasets by several magnitude. Finally, we choose CelebA dataset to test for additional VAE features such as latent interpolation and face feature latent translations.

Our choices of model in this project originally include Autoencoder and (vanilla) VAE, but at the end we figured they are too simplistic when deploying on difficult datasets, hence we only preserved the metrics of Autoencoder for benchmarking the results of other more advanced models. Existing models implemented in our project include Autoencoder, beta-VAE (Burgess et al., 2018; Higgins et al., 2017), MMD InfoVAE (Zhao et al., 2018), Optimal Sigma VAE (Rybkin et al., 2021), Lagging-encoder VAE (He et al., 2019). We revised these architectures and came out with our own version of VAE, named ResLag VAE, which stands for Residual Lagging-encoder VAE with Optimal Sigma Objective. The first version of ResLag VAE has some intrinsic redundancy, which lead us to propose a lightweight version of ResLag VAE, reducing model size by almost 50% while offering similar performance.

We create VAEs with 3, 5 or 8 blocks of convolutional layers in each of encoder and decoder for all variants of VAE, depending on the dataset complexity. This is to ensure the performances from different VAE architecture are comparable, where most differences should come from the novelty of tweaking model structure or training objective instead of sheer model depth.

Inspiration and Training Procedure

Our training procedure for all of our selected VAE architectures are different from each other based on related literatures. It is worth noting that across multiple VAE variants from our experiments, most of them share similar sequential structure, while only our own ResLag-VAE has residual connections. On the other hand, most differences come from their training objectives. Beta-VAE required choosing a weighting parameter for the latent loss (Burgess et al., 2018; Higgins et al., 2017), MMD InfoVAE required calculating an additional mutual information and setting two hyperparameters (Zhao et al., 2018), Optimal Sigma VAE calculate data variance analytically and use it in place of the beta in beta-VAE (Rybkin et al., 2021), Lagging-encoder VAE trains encoder significantly more than decoder while keeping the standard VAE objective.

Through our experiments with the above-mentioned VAE variants, we found the approach of MMD InfoVAE require more experiences as it uses two hyperparameters, together with the difficulty of calculating mutual information on large dataset such as CelebA dataset, we respectfully discard the benefit and training objective learned from InfoVAE. Regardless of this observation, we manage to discover great benefit from the simplicity of Optimal Sigma VAE and the improved performance from lagging training of Lagging-encoder VAE. More importantly, the derivation of Optimal Sigma VAE's objective taught us a lesson about how an analytically calculated constant (data variance) can outperform heuristic hyperparameter setting (beta in beta-VAE, for example), automating the selection of such hyperparameter.

Our ResLag-VAE model consists of residual connections within its encoder and decoder, with its objective as a hybrid form of evaluating data variance analytically (Optimal Sigma VAE) and aggressively training its encoder (Lagging-encoder VAE). We claim that our novelty on this creation comes from the proposed residual connections. First, Maaløe et al. (2019) proposed BIVA, a VAE that performs bi-directional inference within the encoder. Their model architecture is an improvement from the Ladder-VAE (Sønderby et al., 2016)

and consists of 15 or 20 layers of ResNet blocks depending on the choice of dataset, considerably much deeper than our implementations. While their architecture utilizes residual connections, it is slightly different than our implementation. Moreover, Dieng et al. (2019) proposed generative skip connection in VAE, but only implement such residual connections in decoder.

More of our parameter choices will be detailed in the implementation section.

Evaluation

On each architecture, we evaluate the trained model by looking at their reconstruction loss, latent loss, and other loss in some specific architecture. We also calculate total loss which might be a linear combination of previous said loss in some way, depending on the objective of that architecture. We always choose to evaluate the model with the best validation total loss.

After selecting the best model on each dataset, we perform some visual analysis on them, that includes direct reconstruction, prior sampling for examining the quality of latent space, latent interpolation of 2 or 4 images. For simpler datasets such as both MNIST datasets, we also plot a t-SNE graphs to observe the clustering of different classes.

Implementation

We provide details of our model in this section, including model architecture, choice of datasets, optimizer and learning rate.

We build our ResLag-VAE with different depths for different datasets, here we show our implementation on the CelebA dataset (Figure 4). The code for implementations on other datasets have been published in our GitHub repository.

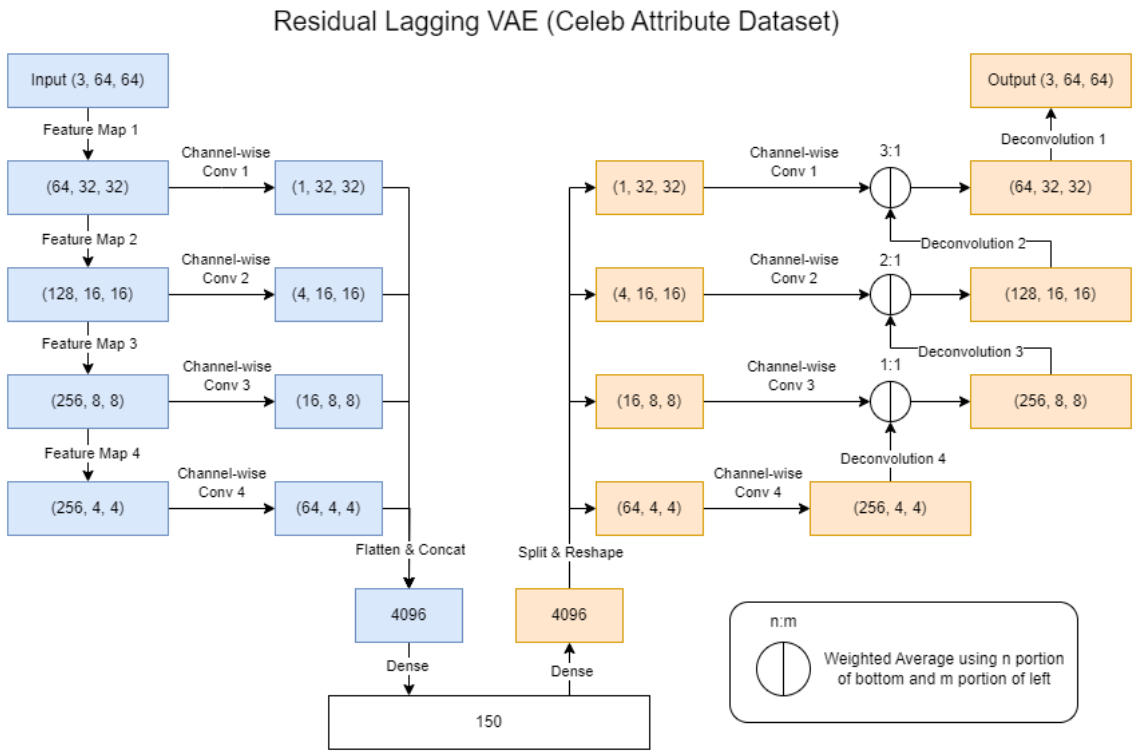

Figure 4 Model Architecture of VAE trained on CelebA dataset. Blue color is the encoder, and light orange color is the decoder.

In Figure 4, light blue and orange cells indicates the encoder and decoder parts. Within the encoder, we use a simple convolutional block to create deep feature maps, specifically 4 of it for this dataset. Then, we perform channel-wise convolution (i.e., convolution with kernels of shape 1 by 1) to get four blocks of features where each of them is of 1024 dimension. We simply concatenate these to obtain a 4096-dimensional vector before using a fully connected layer mapping it to a customized latent space. Here we choose 150-dimensional latent space. To decode, we mostly mirror the operation in encoder. Splitting the 4096-dimensional vector into 4 equal parts, we proceed to create 4 different tensors (similar

to a feature map) from it by using deconvolutional layers. Then, from the tensor with lowest dimension (64, 4, 4) we use channel-wise convolution to get a (256, 4, 4) tensor, then upscale it using a scale of 2 to obtain a tensor of shape (256, 8, 8). We take the (16, 8, 8) tensor and perform channel-wise convolution on it to obtain (256, 8, 8) tensor. Using the two (256, 8, 8) tensors, averaging them give us a new (256, 8, 8) tensor. We repeat the same process 2 more times, each time with a tensor from different layer, while applying a 2 to 1 or 3 to 1 weighted average to eventually obtain a tensor of shape (64, 32, 32). Finally, we upscale it one more time to get the reconstructed image of shape (3, 64, 64).

Above actually shows the architecture of lightweight version of ResLag-VAE. In the original version, instead of getting a 4096-dimensional vector after concatenating in the encoder, we would get a 16384-dimensional vector at it. Note that the original image is of dimension 3*64*64=12288, less than that of concatenated vector. We figured this would put too much pressure on the last linear layer in the encoder as the previous part has more dimension than original image, thus propose a version with reduced concatenated vector. While we build such model concatenating 4 different layers of feature maps, we only concatenate 3 layers of feature maps on the CIFAR10 and CIFAR100 datasets, and we discard residual connection altogether on the MNIST and FashionMNIST datasets as they both are simple datasets.

During training our ResLag-VAE, we use automated calculation of reconstruction variance, also used in calibrated decoder from Rybkin et al. (2021). The authors of calibrated decoder argued that the assumption of distribution of reconstruction having unit variance (i.e., $p_\theta(x|z) \sim N(x; p_\theta(z), I)$ is not realistic, while going for a lower practical variance ($\sim (0.229, 0.224, 0.225)^2$ for RGB channels) as evident from ImageNet dataset is more suitable in theory(Deng et al., 2009). Under this assumption, the negative ELBO with a non-unit reconstruction variance becomes

$$-L_{ResLag}(x; \phi, \theta) = \frac{1}{2} D \ln 2\pi + D \ln \sigma_\theta + \frac{1}{2\sigma_\theta^2} ||x - \tilde{x}||_2^2 + D_{KL}\left(q_\phi(z|x)||N(z; 0, I)\right).$$

In the equation above, D stands for the dimension of data, σ_θ^2 stands for the reconstruction variance that can be either learned, or provided as a constant, that are either globally, different channel-wise or different pixel-wise; In the setting of Rybkin et al. (2021), the authors did not learn the value of σ_θ^2, but they use the minibatch during training to calculate the variance, and then using the moving average of training dataset during

evaluation on validation dataset. During our own experiments, we notice the reconstruction variances normally lies around the range of 0.05~0.10, agreeing with the variance from ImageNet dataset. Note that this is equivalent of using a beta of value 0.1 ~ 0.2 in Beta-VAE.

```
8159.7s    2      ----Carrying 'sigma2'
8159.7s    3                    Mean 0.07131653706063074
8159.7s    4                    Std 0.0072859361799333736
8159.7s    5                    Minimum 0.05148473009467125
8159.7s    6                    Maximum 0.09896250814199448
```

Figure 5 Reconstruction variance of CIFAR10 dataset during training in a Kaggle Notebook. We output the variances after each epoch, which is a running list of variance for each minibatch. We print out the mean, standard deviation, min and max of the sigma^2 values to better understand the data.

To address posterior collapse in VAE, we integrate lagging encoder into our training process (He et al., 2019). Using two Adam optimizers with learning rate 0.002 separately for encoder and decoder, we define **aggressive training of ratio n** for an epoch as optimize encoder n times then optimize decoder once. On MNIST and FashionMNIST datasets, we trained them for a total of 40 epochs, where only the first 5 epochs are using aggressive training of ratio 3. For both CIFAR datasets, we trained them for 40 epochs as well, where the first 20 epochs are using aggressive training of ratio 5. Training on CelebA dataset is significantly more resource intensive, we only train them for 20 epochs, where only the first 10 epochs are using aggressive training of ratio 5.

Result and Discussion

Metrics

Combining the techniques we discussed in the previous section we have created ResLag-VAE, a hybrid model combining benefits from several existing models. We built ResLag-VAE for the five datasets, but we do not establish residual connection for both MNIST datasets because they are simpler. All the training results and evaluation are provided below (Table 1).

Experiments

Architecture/Dataset	MNIST (LD5)	FashionMNIST (LD10)	CIFAR10 (LD100)	CIFAR100 (LD100)	CelebA (LD150)
AutoEncoder	Rec: 15.9744	Rec: 9.2723	Rec: 16.5963	Rec: 16.7859	
beta-VAE	Rec: 19.2091 Lat: 1.9357 Total: 26.9521 beta=4	Rec: 13.0418 Lat: 1.1286 Total: 17.5560 beta=4	Rec: 18.0926 Lat: 1.7194 Total: 23.2508 beta=3	Rec: 18.2742 Lat: 1.7193 Total: 23.4322 beta=3	Rec: 109.6476 Lat: 1.968332 Total: 117.5209 beta=4
MMD InfoVAE	Rec: 16.5469 Lat: 7.7451 MMD: 0.031436 Total: 32.2647 alpha=1, lambda=500	Rec: 8.2899 Lat: 7.2997 MMD: 0.0101 Total: 13.3387 alpha=1, lambda=500	Rec: 20.3646 Lat: 11.6778 MMD: 0.001186 Total: 20.9576 alpha=1, lambda=500	Rec: 22.4772 Lat: 11.6155 MMD: 0.001248 Total: 23.1012 alpha=1, lambda=500	Rec: 91.9245 Lat: 13.0922 MMD: 0.001801 Total: 92.8250 alpha=1, lambda=500
Optimal Sigma VAE	Rec: 16.718534 Lat: 14.588510 **Total: -817.6128**	Rec: 12.1918 Lat: 14.1663 Total: -751.9987	Rec: 22.0229 Lat: 101.4148 Total: -3961.8050	Rec: 23.6347 Lat: 92.8076 **Total: -3786.0454**	Rec: 81.95391 Lat: 193.7795 Total: -14161.9359
Lagging VAE with Optimal Objective (No residual)	Rec: 17.2930 Lat: 14.3803 Total: -813.4669	Rec: 11.220689 Lat: 15.73822 **Total: -756.443231**			
Residual Lagging VAE with Optimal Objective (Our)			Rec: 22.7998 Lat: 100.2814 Total: -3964.9472	Rec: 23.7575 Lat: 95.1944 Total: -3778.4394	Rec: 84.4005 Lat: 197.3168 **Total: -14181.7860**
Light-weight Residual Lagging VAE with Optimal Objective (Our)			Rec: 21.8595 Lat: 101:2951 **Total: -3971.4712**	Rec: 23.7379 Lat: 91.6886 Total: -3782.1070	Rec: 85.1928 Lat: 194.1816 Total: -14180.4211

Glossary

* N-dimensional Latent Space (LD[N])
* Reconstruction Loss (Rec)
* Latent Loss (Lat)
* Maximum-Mean Discrepancy Loss (MMD)
* ELBO, depends on objective design (Total)

Hyperparameters

Table 1 Training results breakdown as reconstruction, latent losses, optionally having MMD loss. All losses are evaluated on validation dataset. The last four models have negative ELBO because they are using optimal sigma objective, which includes the logarithm of reconstruction variance.

We use a learning rate of 0.002 and Adam optimizer for all of our training, and on beta-VAE and MMD InfoVAE we use a learning rate scheduler that reduce learning rate in half when validation loss doesn't improve for 3 epochs, we also put early stopping when learning rate is smaller than 5e-5. As the last four models are using optimal sigma objective which includes logarithm of reconstruction variance into the equation, their total ELBO are mostly negative. Different datasets also use different latent dimension based on the complexity and achieving high compression rate as below:

- MNIST Digit: Latent dimension is 5,
- Fashion MNIST: Latent dimension is 10,
- CIFAR10 and CIFAR100: Latent dimensions are both 100,
- CelebA: Latent dimension is 150.

Among the last four models, we found that our residual variants can sometimes outperform optimal sigma VAE, while otherwise not deviate too much from its performance. Interestingly, models with best total loss do not necessarily have the best reconstruction loss. This means the model is not solely optimizing the reconstruction quality, but it balances with the latent loss (information retain) as well. Our visual results on the next subsection will shows parallel conclusion that the best reconstruction does not comes from a model with the best reconstruction loss, but instead comes from model with the best total loss.

Visual Results

We will focus on using reconstruction and total losses as a guide to evaluate models mentally. Total loss serves as an objective metric, while reconstruction loss tells us the degree of deviation from original image.

Reconstructions

Figure 6 shows the reconstructed images comparing to its original input. We can notice that on MNIST, autoencoder and lagging-VAE perform pretty well. On FashionMNIST, there is not much difference on different architectures, instead what we noticed is all architectures will get rid most of the texture details on the garments.

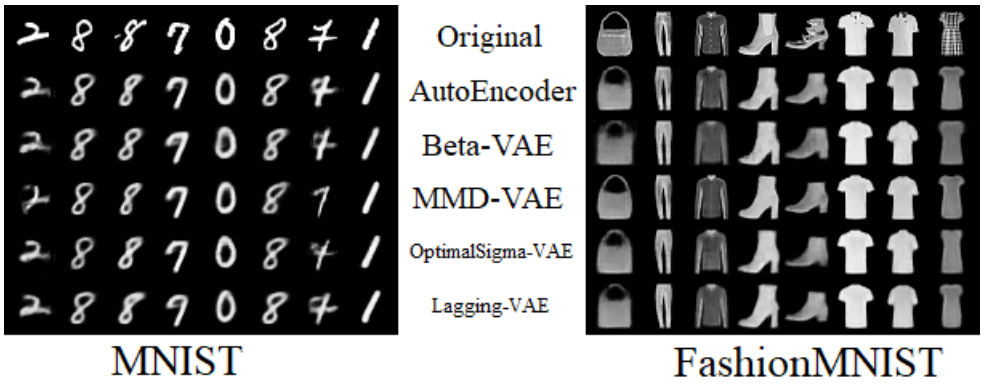

Figure 6 Comparing both MNIST datsets.

Next, Figure 7 shows the reconstructions result from different architectures for both CIFAR datasets. Stand as difficult datasets, our models couldn't capture their intrinsic properties. For the 4th image in CIFAR10 (orange cat) the model can only capture an orange region, but doesn't seem to acknowledge a cat anatomy, the cat's head become deformed from our lightweight ResLag-VAE. On the first image in CIFAR100 (clock on grid windows), all models could only get a rough shape of the clock, but wouldn't perceive grid-like structure from the windows. On its 7th image (lighting pole with sunset behind it), the lighting pole got

almost eroded from the orange background, render the model could not perceive the center object. Nevertheless, the overall ResLag-VAE generate on par results to previous counterparts, where it can be noticed that MMD-VAE sometimes generate blurrier results than others, although just by a little bit.

Figure 7 Comparisons on both CIFAR datasets.

At last, Figure 8 is the comparisons on CelebA dataset. Our models seem to capture better side details (hair colors and hair layers) than previous models. They also generate faces with more sophisticated and realistic lightning conditions, compared to slightly monotone reconstructions from previous models. Moreover, our models are also capable of captured wider range of face emotions, being more faithful to its original images. On other models, some reconstructions might be smiling but the original image is not. We figure the reason for these improvements is because CelebA datasets are originally centered nicely, hence model can focus on reconstructing center face and other non-face features separately. However, we found the quality of background reconstruction rather unsatisfying, probably due to our model's lesser complexity to current state-of-the-art models or our training pipeline could hinder the process of learning background.

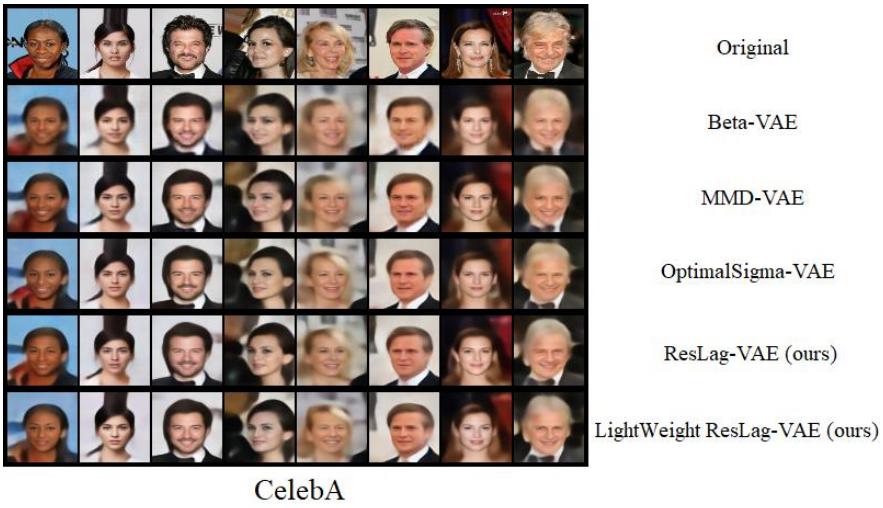

Figure 8 CelebA dataset comparison on reconstructions

Prior Sampling

Solely observing the quality of reconstruction might not be extensive enough for evaluating VAE models, it is equally important to evaluate when reconstruction is made from a latent variable that is randomly sampled from a prior distribution. In our case, we sample our latent variables from a unit gaussian distribution. This has two benefits: We can judge whether the VAE have made use of the full capacity of the prior, and secondly posterior collapse can be easily identified by prior sampling. In the visualizations below we sample 16 random latent variables and reconstruct images on every model architecture and every dataset.

In Figure 9, we have the results of prior sampling for both of the MNIST datasets. We notice similar behavior on both datasets. Autoencoder doesn't seem to learn the representation of original data, it fails to construct viable digits from the latent, and garments often do not have meaningful shapes. Beta-VAE and MMD-VAE do generate better reconstructions, where occasionally some black spots will appear on Fashion MNIST results. For Optimal Sigma VAE and Lagging-VAE, there seems to be nearly coherent digits and garments, although not as perfect, reconstructions. What interest us is the observation that although FashionMNIST datasets reconstructions seems the best from Lagging-VAE, but MMD-VAE is the model with the best validation loss. This shows the quality of reconstruction cannot be evaluated solely on reconstruction loss, but should be combined with other factors.

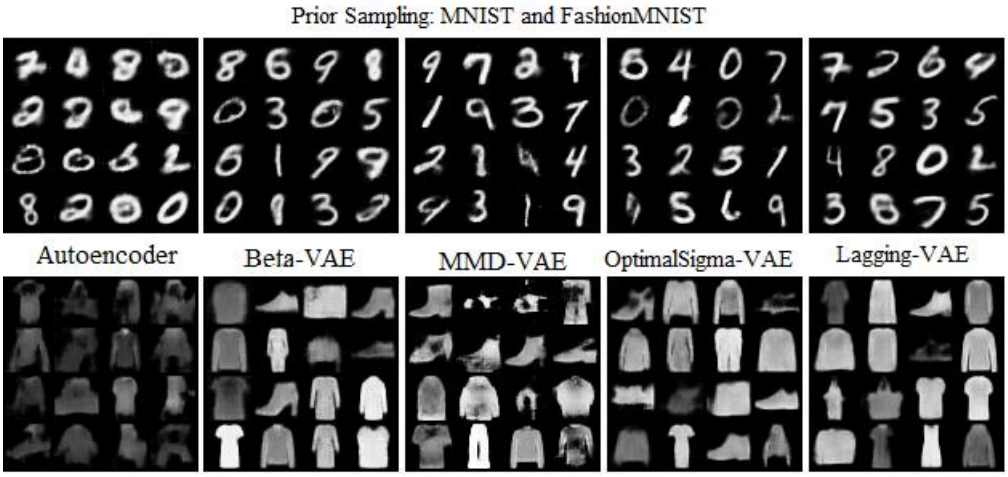

Figure 9 Examine prior reconstructions on models trained on both MNIST and FashionMNIST datasets.

On both CIFAR10 and CIFAR100 datasets (Figure 10, Figure 11), prior sampling generally does not construct coherent images, but there are still noticeable differences nonetheless. The most obvious one is that Autoencoder seems to suffered from some kind of "collapse", but we could not say it is posterior collapse because it is not variational. We postulate that our Autoencoders learned a lookup-table, so that only a very small subset (or finitely many) of latent vectors are corresponding to meaningful reconstructions, else other latent vectors are just collapsed to a unified image. On Beta-VAE and MMD-VAE models, we notice the model only generate noisy little details all over the place with black spots while having colorful pixels, we think the model just learned some random visuals from many images and blend them together. From the progression of Optimal Sigma VAE to our two ResLag-VAE Models, the major color scheme of each image starts to decrease to only one or two, which means the latent contains information about the general context of image, although they still cannot generate more realistic image.

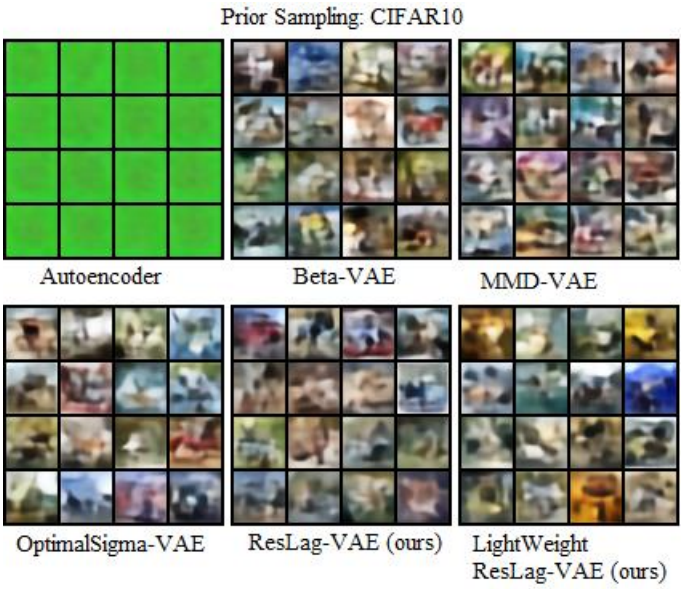

Figure 10 Prior Reconstruction for models trained on CIFAR10.

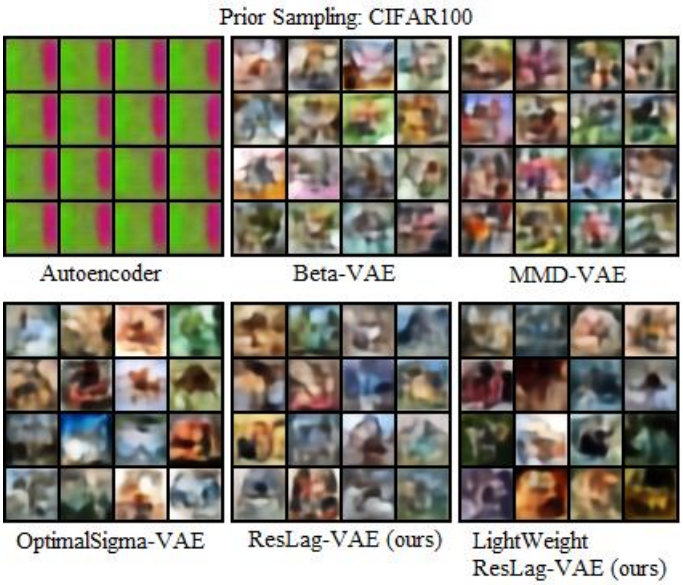

Figure 11 Prior Reconstruction for models trained on CIFAR100.

On the last dataset, we have experiments on CelebA (Figure 12). Beta-VAE and MMD-VAE only reconstructed face without meaningful background, which is quite similar to the case of CIFAR datasets: Background become mixture of colorful pixels. We however could notice a slightly better facial features on MMD-VAE than Beta-VAE. On Optimal Sigma VAE and our two ResLag-VAE models, the improvements is significant although the background is still random. We start to see diverse skin color and better lighting on face, different types of emotion (smile, no emotion, frown, visible teeth or not). Overall, we think the reason for our model to capture face details is not entirely depending on the power of all of the models, but because the CelebA faces are generally aligned at the center and has a

normalized face scale, hence the model can learn to always predict facial features on the center circle region. While it is true that the way of collecting data can influence experiment results, our two ResLag-VAE models did create better looking reconstructions comparing to other models.

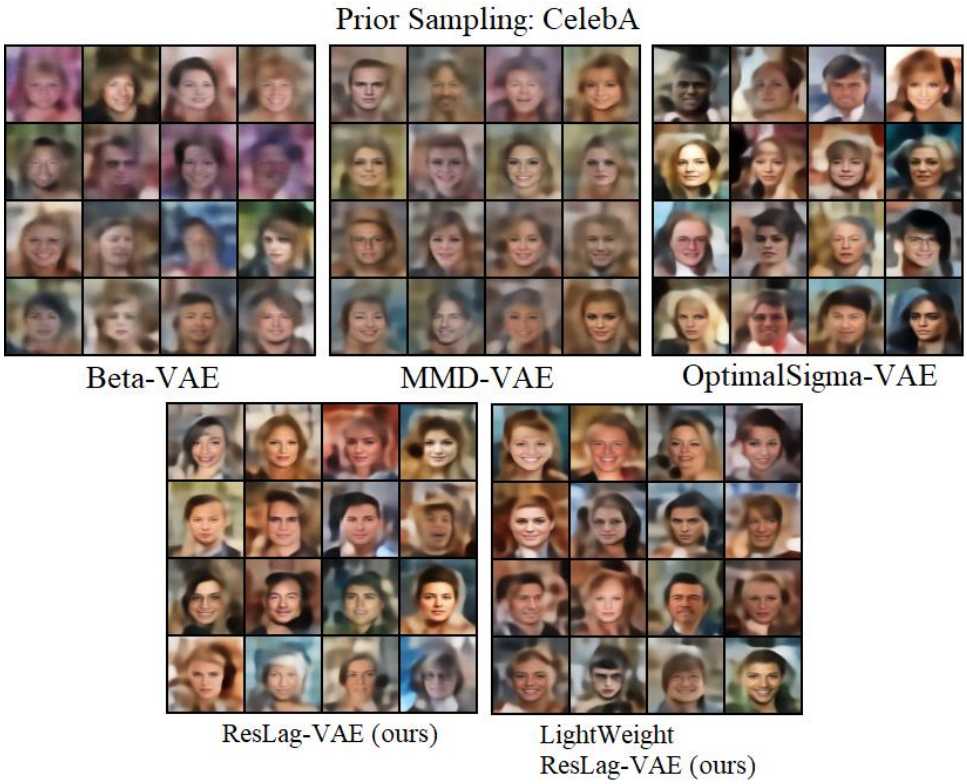

Figure 12 Prior Reconstruction for models trained on CelebA dataset.

Conclusion

In the report of this master project, we introduced the concept of Autoencoder and how Variational Bayes can improve the reconstructions and general behavior on latent space. Our aim is to make training VAE easier and promote future researches on VAE. We listed three objectives to help achieve our goal: Design a VAE training method that requires no hyperparameter, little changes to architecture and objective when switching datasets, increasing output sharpness by utilizing different stages of information into the latent space while keeping latent dimension low.

Then, literature review was conducted to study the progress of VAE development from the past until state-of-the-art. While depending on the scale of this master project, we didn't utilize everything we learned from the literatures, our model still has ingenuity in its architecture design. Our model deployed the Optimal Sigma objective, which assumes the reconstructed images follows a Gaussian distribution with an analytically calculated variance instead of simply assuming unit variance, this leads to a slightly different ELBO that acts like a weighted sum of two losses. From visualizations, we observed that generally models with Optimal Sigma objective tends to generate better reconstructions than those not using it. We also show our models have the potentials to further increase the quality of reconstructions.

On top of that, we report our ELBO results as a breakdown list (including reconstruction loss, latent loss, MMD loss) instead of a single total value as in other works. We figured not only will this makes our work more transparent for critical evaluation, it also helps us identify the role of latent losses when training a VAE: We generally could expect better reconstruction when the latent loss is not too small, which can be achieved when using the Optimal Sigma objective as it more heavily penalizes reconstruction loss. Moreover, lower reconstruction loss by itself does not mean better reconstruction, as mean square error only calculate the mean of errors, but our human eyes perceive image differently. For example, an image of a street with a light pole and a street without a light pole seems very different from our eyes, but since light pole only occupy a small count of pixels in an image, the mean square error between these two images is relatively small.

In this project, we did not create very deep architectures as they need significantly more time and resource to study, hence we strongly suggest our strategies – residual, lagging encoder, optimal sigma – on a deeper VAE architecture to examine whether better

reconstructions can be made, and hopefully it can be on par with the reconstruction quality of GAN models so that we have an alternative to GAN that is easier to train.

Values and Ethics

This project aims to brings practical values to machine learning practitioners who are studying and improving VAE architectures. It tries to show that VAE training can be made simpler and more intuitive by not using much hyperparameter.

All of our datasets are sourced from the internet intended for research purposes.

References

Burgess, C. P., Higgins, I., Pal, A., Matthey, L., Watters, N., Desjardins, G., & Lerchner, A. (2018). Understanding disentangling in beta-VAE. *Arxiv*. https://doi.org/10.48550/arXiv.1804.03599

Deng, J., Dong, W., Socher, R., Li, L.-J., Li, K., & Fei-Fei, L. (2009). ImageNet: A large-scale hierarchical image database. *IEEE Conference on Computer Vision and Pattern Recognition*, 248–255. https://doi.org/10.1109/CVPR.2009.5206848

Deng, L. (2012). The mnist database of handwritten digit images for machine learning research. *IEEE Signal Processing Magazine*, *29*(6), 141–142.

Dieng, A. B., Kim, Y., Rush, A. M., & Blei, D. M. (2019). Avoiding Latent Variable Collapse With Generative Skip Models. *Arxiv*. https://doi.org/10.48550/arXiv.1807.04863

Goodfellow, I. J., Pouget-Abadie, J., Mirza, M., Xu, B., Warde-Farley, D., Ozair, S., Courville, A., & Bengio, Y. (2014). Generative Adversarial Nets. *Neural Information Processing Systems*. https://doi.org/10.48550/arXiv.1406.2661

Gulrajani, I., Kumar, K., Ahmed, F., Taiga, A. A., Visin, F., Vazquez, D., & Courville, A. (2016). PixelVAE: A Latent Variable Model for Natural Images. *ArXiv Preprint*. https://doi.org/10.48550/arXiv.1611.05013

He, J., Spokoyny, D., Neubig, G., & Berg-Kirkpatrick, T. (2019). Lagging Inference Networks and Posterior Collapse in Variational Autoencoders. *ArXiv Preprint*. https://doi.org/10.48550/arXiv.1901.05534

Higgins, I., Matthey, L., Pal, A., Burgess, C., Glorot, X., Botvinick, M., Mohamed, S., & Lerchner, A. (2017). Beta-VAE: Learning Basic Visual Concepts with a Constrained Variational Framework. *International Conference on Learning Representations 2017*. https://openreview.net/pdf?id=Sy2fzU9gl

Isola, P., Zhu, J.-Y., Zhou, T., & Efros, A. A. (2016). Image-to-Image Translation with Conditional Adversarial Networks. *ArXiv Preprint*. https://doi.org/10.48550/arXiv.1611.07004

Kingma, D. P., & Welling, M. (2019). An Introduction to Variational Autoencoders. *Foundations and Trends in Machine Learning*, 1–18. https://doi.org/10.1561/2200000056

Krizhevsky, A. (2009). *Learning multiple layers of features from tiny images*. https://www.cs.toronto.edu/~kriz/cifar.html

Li, K., Kong, L., & Zhang, Y. (2020). 3D U-Net Brain Tumor Segmentation Using VAE Skip Connection. *IEEE International Conference on Image, Vision and Computing*, 97–101. https://doi.org/10.1109/ICIVC50857.2020.9177441

Liu, Z., Luo, P., Wang, X., & Tang, X. (2015). Deep Learning Face Attributes in the Wild. *Proceedings of the IEEE International Conference on Computer Vision*. https://mmlab.ie.cuhk.edu.hk/projects/CelebA.html

Ma, Y. (2023, April 27). Stable Diffusion VAE Guide and Comparison. *Aituts*. https://aituts.com/vae/

Maaløe, L., Fraccaro, M., Liévin, V., & Winther, O. (2019). BIVA: A Very Deep Hierarchy of Latent Variables for Generative Modeling. *Advances in Neural Information Processing Systems 32*. https://doi.org/10.48550/arXiv.1902.02102

Oord, A. van den, Kalchbrenner, N., & Kavukcuoglu, K. (2016). Pixel Recurrent Neural Networks. *International Conference on Machine Learning, Proceedings of Machine Learning Research*, 1747–1756. https://doi.org/10.48550/arXiv.1601.06759

Oord, A. van den, Kalchbrenner, N., Vinyals, O., Espeholt, L., Graves, A., & Kavukcuoglu, K. (2016). Conditional Image Generation with PixelCNN Decoders. *Advances in Neural Information Processing Systems 29*. https://doi.org/10.48550/arXiv.1606.05328

Oord, A. van den, Vinyals, O., & Kavukcuoglu, K. (2017). Neural Discrete Representation Learning. *Advances in Neural Information Processing Systems 30*. https://doi.org/10.48550/arXiv.1711.00937

Razavi, A., Oord, A. van den, & Vinyals, O. (2019). Generating Diverse High-Fidelity Images with VQ-VAE-2. *Advances in Neural Information Processing Systems 32*. https://doi.org/10.48550/arXiv.1906.00446

Rybkin, O., Daniilidis, K., & Levine, S. (2021). Simple and Effective VAE Training with Calibrated Decoders. *International Conference on Machine Learning, Proceedings of Machine Learning Research*. https://doi.org/10.48550/arXiv.2006.13202

Sadeghi, H., Andriyash, E., Vinci, W., Buffoni, L., & Amin, M. H. (2019). PixelVAE++: Improved PixelVAE with Discrete Prior. *ArXiv Preprint*. https://doi.org/10.48550/arXiv.1908.09948

Sønderby, C. K., Raiko, T., Maaløe, L., Sønderby, S. K., & Winther, O. (2016). Ladder Variational Autoencoders. *Advances in Neural Information Processing Systems 29*. https://doi.org/10.48550/arXiv.1602.02282

Su, J., & Wu, G. (2018). f-VAEs: Improve VAEs with Conditional Flows. *Arxiv*. https://doi.org/10.48550/arXiv.1809.05861

Xiao, H., Rasul, K., & Vollgraf, R. (2017). Fashion-MNIST: a Novel Image Dataset for Benchmarking Machine Learning Algorithms. *ArXiv Preprint*. https://doi.org/10.48550/arXiv.1708.07747

Zhao, S., Song, J., & Ermon, S. (2018). InfoVAE: Information Maximizing Variational Autoencoders. *ArXiv Preprint*. https://doi.org/10.48550/arXiv.1706.02262