Auto-Encoding Variational Bayes

Andreea Andriciuc
Computer Science Department, University of Bucharest, Romania

Abstract

This paper is a replication study of the seminal paper [5], implementing and analyzing the Variational Auto-Encoder (VAE) architecture using modern deep learning frameworks, focusing on the MNIST and dataset experiments. It compares the original ELBO optimization with the wake-sleep algorithm, and evaluates the results using both quantitative metrics and qualitative analysis of the learned latent space. The findings confirm the original paper's results regarding the effectiveness of the reparameterization trick and the VAE's ability to learn meaningful latent representations.

1 Introduction

Deep latent variable models are powerful tools for learning complex probability distributions and generating new data. The Variational Auto-Encoder (VAE), introduced by Kingma and Welling in 2013, represented a breakthrough in training such models by combining variational inference with deep learning. The key innovation was the reparameterization trick, which enabled efficient gradient-based training of the variational parameters.

Prior to VAEs, training deep latent variable models was challenging due to the difficulty of computing gradients through random sampling operations. Earlier approaches like the wake-sleep algorithm [3] required complex training procedures and often suffered from unstable optimization. VAEs solved this by introducing a differentiable sampling process through the reparameterization trick.

In this paper, the objective is to replicate the original VAE experiments on the MNIST dataset, implementing both the ELBO optimization and wake-sleep algorithm for comparison. Modern deep learning frameworks and optimizers (specifically PyTorch and Adam) are used to verify the reproducibility of the original results in a contemporary context.

2 Technical description of the problem

Consider a continuous random variable z that generates a dataset $X = \{x_1, x_2, ..., x_N\}$ with N independent and identically distributed samples. z comes from a prior distribution denoted $p_{\theta^*}(z)$, while datapoints from X are obtained from a conditional distribution denoted $p_{\theta^*}(x|z)$. We are interested in:

- learning the parameters θ of p, as close to the true θ^*,
- approximate posterior inference over z,
- approximate marginal inference over x.

We know that:

$$p_\theta(x) = \int p_\theta(z) p_\theta(x|z) dz \tag{1}$$

However, this integral is intractable for most interesting models due to its high dimensionality and the complex dependencies between x and z introduced by neural networks. To address this, we introduce an approximate posterior $q_\phi(z|x)$, known as the recognition model or probabilistic encoder, with parameters ϕ. This distribution is chosen to approximate the true posterior $p_\theta(z|x)$. The log marginal likelihood can be rewritten as:

$$\log p_\theta(x) = \log \int q_\phi(z|x) \frac{p_\theta(z) p_\theta(x|z)}{q_\phi(z|x)} dz \tag{2}$$
$$= \log \mathbb{E}_{q_\phi(z|x)} \left[\frac{p_\theta(z) p_\theta(x|z)}{q_\phi(z|x)} \right] \tag{3}$$
$$\geq \mathbb{E}_{q_\phi(z|x)} \left[\log \frac{p_\theta(z) p_\theta(x|z)}{q_\phi(z|x)} \right] \tag{4}$$
$$= \mathbb{E}_{q_\phi(z|x)}[\log p_\theta(x|z)] - KL(q_\phi(z|x)||p_\theta(z)) \tag{5}$$
$$= \mathcal{L}(\theta, \phi; x) \tag{6}$$

In the end, we have:

$$\mathcal{L}(\theta, \phi; x) = -KL(q_\phi(z|x)||p_\theta(z)) + \mathbb{E}_{q_\phi(z|x)}[\log p_\theta(x|z)], \tag{7}$$

where the quantity to be optimized is in the left term (variational lower bound) and in the right term we have the KL-divergence of the approximate from the true posterior and the expected reconstruction error [2].

3 Possible solution

3.1 Reparameterization Trick

To optimize the ELBO using gradient-based methods, we need to be able to backpropagate through the expectation term $\mathbb{E}_{q_\phi(z|x)}[\log p_\theta(x|z)]$, which can be approximated using Monte Carlo sampling:

$$\mathbb{E}_{q_\phi(z|x)}[\log p_\theta(x|z)] \approx \frac{1}{L} \sum_{l=1}^{L} \log p_\theta(x|z^{(l)}), \tag{8}$$
$$z^{(l)} \sim q_\phi(z|x) \tag{9}$$

However, sampling from $q_\phi(z|x)$ introduces a random node in our computational graph that prevents gradient flow. To address this, we reparameterize the random variable z using a differentiable transformation $g_\phi(\epsilon, x)$ of an auxiliary noise variable ϵ:

$$z = g_\phi(\epsilon, x), \quad \epsilon \sim p(\epsilon) \tag{10}$$

For the common case where $q_\phi(z|x)$ is a Gaussian distribution with mean $\mu_\phi(x)$ and variance $\sigma^2_\phi(x)$, this becomes:

$$z = \mu_\phi(x) + \sigma_\phi(x) \odot \epsilon, \quad \epsilon \sim \mathcal{N}(0, I) \tag{11}$$

where $\odot$ denotes element-wise multiplication.

This gives us the SGVB (Stochastic Gradient Variational Bayes) estimator:

$$\tilde{\mathcal{L}}^A(\theta, \phi; x) = \frac{1}{L} \sum_{l=1}^{L} \log p_\theta(x|z^{(l)}) - KL(q_\phi(z|x)||p_\theta(z)) \tag{12}$$

For a dataset with N datapoints, the full objective becomes:

$$\mathcal{L}(\theta, \phi; X) = \sum_{i=1}^{N} \mathcal{L}(\theta, \phi; x^{(i)}) \approx \sum_{i=1}^{N} \tilde{\mathcal{L}}^A(\theta, \phi; x^{(i)}) \tag{13}$$

3.2 Optimization with Adam

While the original paper used SGVB, we can optimize the same objective using the Adam optimizer[4] . The variational bound remains the same:

$$\tilde{\mathcal{L}}^A(\theta, \phi; x) = \frac{1}{L} \sum_{l=1}^{L} \log p_\theta(x|z^{(l)}) - KL(q_\phi(z|x)||p_\theta(z)) \tag{14}$$

3.3 Variational Auto-Encoders

The algorithm presented in the paper is summed up in Algorithm 1

It is worth mentioning that VAE and AEVB are two different concepts: AEVB is the algorithm that can be used to perform approximate variational inference, while the VAE is a probabilistic auto-encoder where both the encoder and the decoder are neural networks [1].

4 Experiments and Results

We compared, as per the paper, the AEVB algorithm (1) and the Wake-Sleep algorithm (2) [1].

[1]Code and dataset available on GitHub: `https://github.com/AndriciucAndreeaCristina/anomaly_detection/blob/main/project/vae_mnist.ipynb`

Algorithm 1: Auto-Encoding Variational Bayes (AEVB) Algorithm

```
Data: Training dataset $X$, encoder $q_\phi(z|x)$, decoder $p_\theta(x|z)$
Result: Trained encoder and decoder
1  Initialize: Parameters $\phi$ and $\theta$ randomly ;
2  while not converged do
3      for each mini-batch $\mathcal{B} \subset X$ of size $M$ do
4          for each $x \in \mathcal{B}$ do
5              Sample $\epsilon^{(l)} \sim \mathcal{N}(0, I)$ for $l = 1, \ldots, L$ ;
6              Compute $z^{(l)} = \mu_\phi(x) + \sigma_\phi(x) \odot \epsilon^{(l)}$ ;
7              Compute $\hat{L}_{\text{reconst}} = \frac{1}{L} \sum_{l=1}^{L} \log p_\theta(x|z^{(l)})$ ;
8              Compute $L_{\text{KL}} = -\frac{1}{2} \sum_j (1 + \log \sigma_j^2 - \mu_j^2 - \sigma_j^2)$ ;
9              Compute $\mathcal{L}(x) = \hat{L}_{\text{reconst}} - L_{\text{KL}}$ ;
10         Compute gradient $\nabla_{\phi,\theta} \frac{1}{M} \sum_{x \in \mathcal{B}} \mathcal{L}(x)$ ;
11         Update $\phi \leftarrow \phi + \alpha \cdot \nabla_\phi \mathcal{L}$ ;
12         Update $\theta \leftarrow \theta + \alpha \cdot \nabla_\theta \mathcal{L}$ ;
13 return Trained encoder $q_\phi(z|x)$ and decoder $p_\theta(x|z)$ ;
```

Both algorithms has the same base model, maintaining the original architecture of two hidden layers (500) with tanh activations. The weights were initialized randomly from a normal distribution. The training was done with 50 epochs, with a latent dimension of 20 and the Adam optimizer with a learning rate of 0.001. The loss during training can be seen in figure 4. The obtained results are summarized in table 1. A notable difference between the two algorithms is visible when it comes to generating random samples: figure 1 was generated by the AEVB algorithm, while figure 2 was generated by the Wake-Sleep algorithm. Moreover, for different latent space dimensions, we obtained different performances, visible in figure 3. Also, for the different dimensionalities of the latent space, we plotted some random samples from the learned generative models, visible in figure 5.

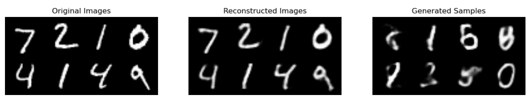


Figure 1: Visualization of AEVB results

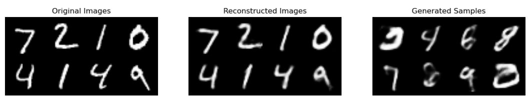


Figure 2: Visualization of Wake-Sleep results

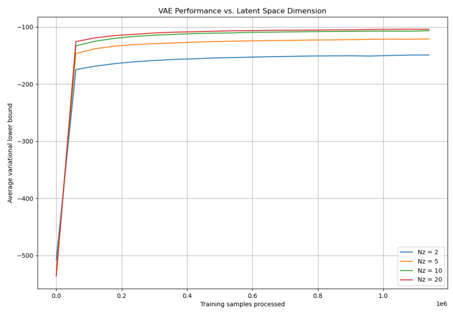


Figure 3: VAE Performance vs. Latent Space Dimension

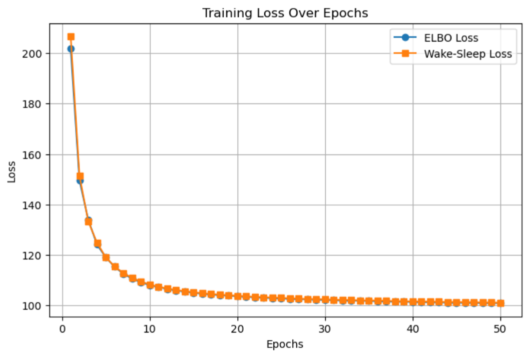


Figure 4: Training loss

Algorithm 2: Wake-Sleep Algorithm

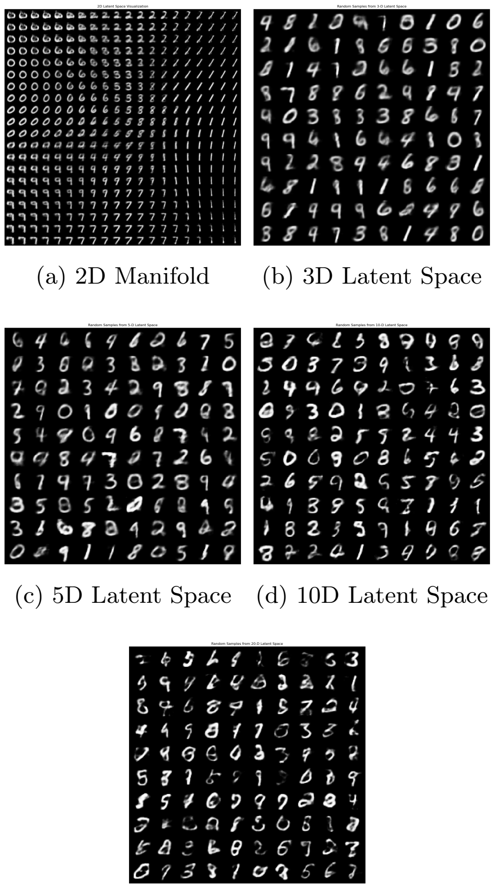
```
   Data: Training dataset $X$, recognition network $q_\phi(h|x)$, generative network $p_\theta(x|h)$
   Result: Trained recognition and generative networks
1  Initialize: Parameters $\phi$ and $\theta$ randomly;
2  while not converged do
       // Wake Phase: Learn Recognition Network
3      for each mini-batch $\mathcal{B} \subset X$ of size $M$ do
4          for each $x \in \mathcal{B}$ do
5              Compute hidden representation $h = q_\phi(x)$;
6              Compute reconstruction $\hat{x} = p_\theta(h)$;
7              Compute reconstruction loss $L_{\text{recon}} = \|\hat{x} - x\|^2$;
8              Update recognition network parameters $\phi$ to minimize $L_{\text{recon}}$;
       // Sleep Phase: Learn Generative Network
9      for number of sleep iterations do
10         Sample random hidden state $h \sim p(h)$;
11         Generate synthetic data $\hat{x} = p_\theta(h)$;
12         Compute prior loss $L_{\text{prior}} = \|h\|^2$;
13         Update generative network parameters $\theta$ to minimize $L_{\text{prior}}$;
14 return Trained recognition network $q_\phi(h|x)$ and generative network $p_\theta(x|h)$;
```

Metric	Wake-Sleep	AEVB
Average Loss	101.1612	101.1053
Average BCE	76.1481	76.4414
Average KLD	25.0131	24.6638

Table 1: Metrics Comparison

5 Conclusion

This replication proves that the theoretical findings in [5] are replicable with even fewer learning epochs, with the simple trick of changing the SGD with Adam optimizer. Also, we validate the idea that the wake-sleep algorithm is still comparable with the AEVB algorithm.



(a) 2D Manifold (b) 3D Latent Space

(c) 5D Latent Space (d) 10D Latent Space

(e) 20D Latent Space

Figure 5: Comparison of latent spaces in different dimensions

References

[1] Stack Exchange Community. How is the vae related to the autoencoding variational bayes (aevb) algorithm?, 2021. Accessed: 2025-01-31.

[2] Carl Doersch. Tutorial on variational autoencoders, 2021.

[3] Geoffrey E. Hinton, Peter Dayan, Brendan J. Frey, and Radford M. Neal. The "wake-sleep" algorithm for unsupervised neural networks. *Science*, 268(5214):1158–1161, May 1995.

[4] Diederik P. Kingma and Jimmy Ba. Adam: A method for stochastic optimization, 2017.

[5] Diederik P Kingma and Max Welling. Auto-encoding variational bayes, 2022.