Variational Autoencoders

Alejandro Fernández Trillo ,Antoni Łopacz
Universidad Loyola / 3 Matemáticas Aplicadas / Computational intelligence

May 5, 2025

Contents

1 Introduction

In recent years, deep generative models have become a cornerstone of modern machine learning, enabling the creation of realistic data such as images, audio, and text. Among these models, *Variational Autoencoders* (VAEs) stand out as a powerful approach that bridges neural networks with probabilistic modeling.

This paper explores the motivation behind the development of VAEs, starting from a review of traditional autoencoders and their limitations, particularly in the context of data generation and latent space representation. The introduction of VAEs was driven by the need for a more principled, mathematically grounded method to model uncertainty and generate new data in a coherent and structured manner.

As a student of Applied Mathematics, this investigation focuses not only on the conceptual evolution from autoencoders to VAEs, but also on the underlying mathematical framework that supports them—ranging from the construction of probabilistic latent variables to the application of variational inference and the derivation of the loss function based on the Kullback-Leibler divergence.

The ultimate goal of this research is to provide a clear and rigorous understanding of Variational Autoencoders, emphasizing the mathematical ideas that make them both elegant and practical within the field of generative modeling.

2 Traditional Autoencoders

Autoencoders are a type of neural network designed to learn a compressed representation of input data, and then reconstruct it as accurately as possible. They are composed of two main components:

- **Encoder:** maps the input $\mathbf{x} \in \mathbb{R}^d$ to a lower-dimensional latent vector $\mathbf{z} \in \mathbb{R}^k$, with $k < d$.
- **Decoder:** reconstructs the input from the latent vector $\mathbf{z}$, producing $\hat{\mathbf{x}}$.

Mathematically, we can define the encoder and decoder as functions:

$$\begin{aligned}\text{Encoder: } & f_\theta(\mathbf{x}) = \mathbf{z} \\ \text{Decoder: } & g_\phi(\mathbf{z}) = \hat{\mathbf{x}}\end{aligned}$$

where θ and ϕ represent the parameters (weights and biases) of the neural networks used in the encoder and decoder respectively.

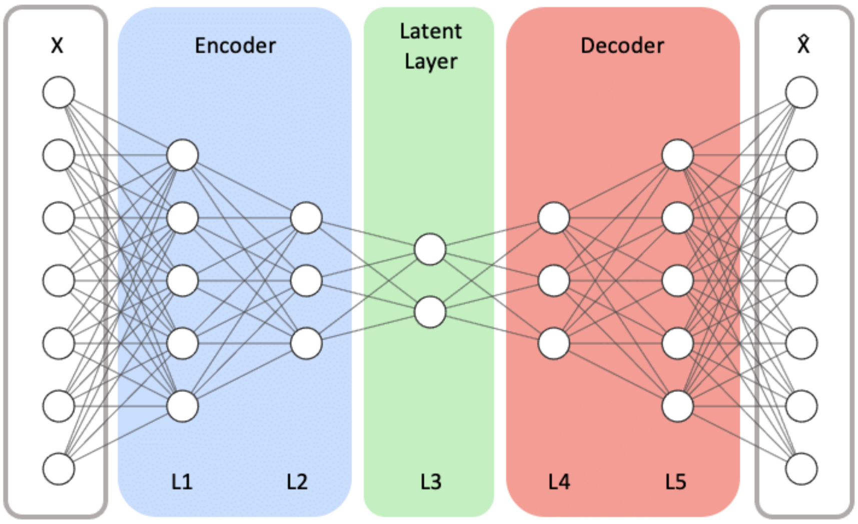


Figure 1: Example architecture of a traditional autoencoder

The training objective is to minimize the reconstruction error, typically measured using the mean squared error (MSE):

$$\mathcal{L}(\mathbf{x}, \hat{\mathbf{x}}) = \|\mathbf{x} - \hat{\mathbf{x}}\|^2$$

By minimizing this loss over a dataset, the autoencoder learns how to encode the most relevant information of the data into a compressed latent space.

Although autoencoders were not originally designed for generative modeling, they inspired further research into creating models capable of both learning efficient representations and generating new data from latent variables.

2.1 Principal limitations of Autoencoders

While traditional autoencoders are effective in learning compressed representations and reconstructing data, they present several significant limitations, especially when considered from a probabilistic and generative modeling perspective:

1. **Unstructured Latent Space:** Traditional autoencoders map each input $\mathbf{x}$ deterministically to a single point $\mathbf{z}$ in the latent space. During training, these points tend to spread out to minimize the reconstruction loss, resulting in a sparse and non-continuous latent space. This means that most of the latent space does not correspond to meaningful or valid outputs.

2. **No Generative Capability:** Since autoencoders do not learn the underlying data distribution, sampling a random latent vector typically results in a meaningless output. They tend to memorize training samples rather than learn a generalizable model, making it difficult to generate realistic variations of data.

3. **Lack of Controlled Generation:** There is no clear way to manipulate or interpret the latent variables in a meaningful manner. Without a structured latent space or

a known prior distribution, one cannot control the nature of generated samples or perform operations like interpolation between data points.

4. **No Uncertainty Modeling:** The encoding process is deterministic, which means the model cannot represent or learn the uncertainty associated with the input representation. This limits its ability to model complex, multimodal distributions.

5. **Risk of Overfitting and Identity Mapping:** If the latent space is too large or the model capacity too high, the autoencoder may learn to replicate the input perfectly without extracting useful features. This leads to overfitting and poor generalization.

These limitations naturally lead to the introduction of *Variational Autoencoders* (VAEs), which reformulate the autoencoder framework using probabilistic principles. In particular, they impose a structured prior (typically a standard Gaussian) over the latent space, enabling meaningful sampling, interpolation, and generation of new data.

3 Motivation for Variational Autoencoders

Once we know that our latent space is restricting us from generating meaningful and diverse data, it becomes clear that traditional autoencoders fall short when it comes to building true generative models. While they can efficiently compress and reconstruct data, they fail to learn a structured representation of the underlying data distribution. This leads to several issues:

In light of the previus limitations mencioned, we need a new framework that not only reconstructs data well but also enables us to:

- Learn a latent space with meaningful structure.
- Sample new data points that resemble the original distribution.
- Incorporate probabilistic reasoning into the encoding process.

This is where **Variational Autoencoders (VAEs)** come into play. They are designed to overcome these challenges by introducing a probabilistic approach to representation learning, where each input is mapped not to a fixed point, but to a distribution in latent space.

In the next section, we will introduce the core ideas behind VAEs and how they achieve these goals.

4 Variational Autoencoders (VAEs)

Variational Autoencoders represent a fundamental shift in how we approach representation learning. Instead of learning a fixed mapping to a single latent point per input, VAEs aim to learn a distribution over latent variables, introducing a probabilistic interpretation of the encoding process.
This idea not only opens the door to controlled data generation and interpolation but also

lays the foundation for a more structured and continuous latent space. What follows is a detailed exploration of the theoretical machinery that makes this possible — combining principles from deep learning, probability, and variational inference.

4.1 previus Concepts

Variational Autoencoders (VAEs) mark a significant evolution in unsupervised learning. Rather than compressing each input into a single deterministic latent vector, VAEs model the latent space as a distribution, typically Gaussian. This probabilistic framework enables the generation of diverse yet coherent outputs and allows for smooth interpolation between data points. By blending concepts from deep learning with variational inference, VAEs introduce a principled way to learn meaningful, continuous, and interpretable latent representations — setting the stage for both theoretical insights and practical generative capabilities.

4.1.1 Bayes Theorem

Given two events A and B, the probability of B conditioned on A is:

$$P(B|A) = \frac{P(B \cap A)}{P(A)} \tag{2.1}$$

In other words, the probability that B occurs given that A has occurred is the probability that both events occur simultaneously, knowing that A has occurred. From this definition of conditional probability, Bayes' theorem can be derived, which is used in obtaining the cost function of the variational auto-encoder. Starting from equation (2.1), we have:

$$P(B \cap A) = P(B|A)P(A)$$
$$P(A \cap B) = P(A|B)P(B)$$

Since the intersection is commutative, that is, $P(A \cap B) = P(B \cap A)$, we have:

$$P(B|A)P(A) = P(A|B)P(B)$$

and by rearranging, we obtain the expression for Bayes' theorem:

$$P(B|A) = \frac{P(A|B)P(B)}{P(A)} \tag{2.2}$$

In equation (2.2), the conditional probabilities $P(B|A)$ and $P(A|B)$ are usually called the posterior probability and likelihood, respectively, the probability $P(B)$ is called the prior probability, and finally, the probability $P(A)$ is called the evidence.

In the case of working with continuous random variables in variational inference, the most common form of this theorem is:

$$p(z|x) = \frac{p(x|z)p(z)}{p(x)} = \frac{p(x|z)p(z)}{\int p(x|z)p(z)dz} \tag{2.3}$$

where the evidence $p(x)$ is the marginal probability of the data.

4.1.2 Kullback-Leibler Divergence

When one wants to determine the difference between two values, for example, two points in space, one alternative is to calculate the distance between those points. In the case of probability distributions, how can one obtain a measure of the difference between them? The Kullback-Leibler divergence[1] provides such a measure. In general, it can be calculated as:

$$D_{\mathrm{KL}}(P\|Q) = \mathbb{E}_{x\sim P}\left[\log \frac{P(x)}{Q(x)}\right] \tag{2.4}$$

One of the properties of this measure, important for obtaining the lower bound used to calculate the cost function of the variational auto-encoder, is that $D_{\mathrm{KL}}(p\|q) \geq 0$. This can be demonstrated using Jensen's inequality, which states that:

$$\mathbb{E}[f(x)] \geq f(\mathbb{E}[x])$$

where $f(\cdot)$ is a convex function. This inequality means that for convex functions, the mean of the function values is always greater than the function value of the mean. In the case of concave functions, such as the logarithm, the inequality works in reverse, as follows:

$$\mathbb{E}[\log x] \leq \log \mathbb{E}[x]$$

4.2 Fundamentals of VAEs

4.2.1 Difference in latent space

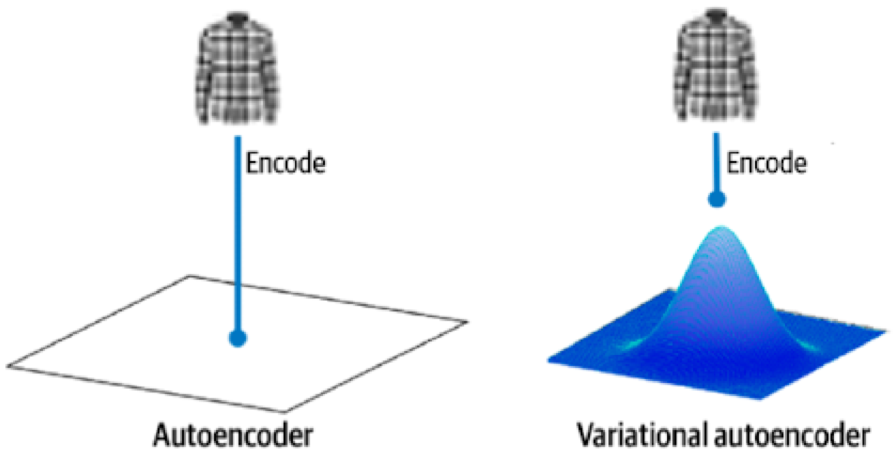


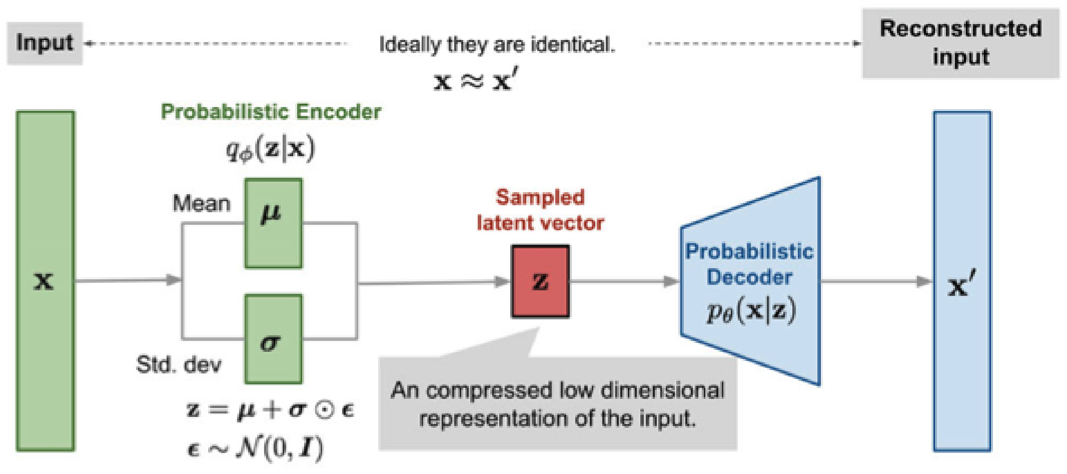
Figure 2: VAEs

While standard autoencoders learn a deterministic mapping from input data to a compressed latent representation, Variational Autoencoders (VAEs) extend this idea by introducing a probabilistic perspective. Instead of encoding each input $\mathbf{x}$ into a fixed latent vector $\mathbf{z}$, VAEs learn to map $\mathbf{x}$ into a distribution over the latent space. This allows the model not only to reconstruct inputs, but also to generate new data by sampling from

[1]Also known as KL divergence or D_{KL}

the learned latent distribution.

4.2.2 VAEs architecture



Input
Ideally they are identical.
$\mathbf{x} \approx \mathbf{x}'$
Reconstructed input
Probabilistic Encoder
$q_\phi(\mathbf{z}|\mathbf{x})$
Mean
$\boldsymbol{\mu}$
$\mathbf{x}$
Std. dev
$\boldsymbol{\sigma}$
$\mathbf{z} = \boldsymbol{\mu} + \boldsymbol{\sigma} \odot \boldsymbol{\epsilon}$
$\boldsymbol{\epsilon} \sim \mathcal{N}(0, \boldsymbol{I})$
Sampled latent vector
$\mathbf{z}$
An compressed low dimensional representation of the input.
Probabilistic Decoder
$p_\theta(\mathbf{x}|\mathbf{z})$
$\mathbf{x}'$

Figure 3: Structure of a Variational Autoencoder. The encoder maps the input $\mathbf{x}$ into a distribution over latent variables, characterized by a mean vector $\boldsymbol{\mu}$ and a standard deviation vector $\boldsymbol{\sigma}$. A latent vector $\mathbf{z}$ is then sampled using the **reparameterization trick** and passed through the decoder to reconstruct the original input $\mathbf{x}'$.

In the diagram above, we can observe the key difference between VAEs and traditional autoencoders. Instead of directly outputting a single latent vector, the encoder (denoted $q_\phi(\mathbf{z}|\mathbf{x})$) outputs the parameters of a Gaussian distribution: $\boldsymbol{\mu}$ and $\boldsymbol{\sigma}$. A latent sample $\mathbf{z}$ is drawn from this distribution using the so-called reparameterization trick, written as:

$$\mathbf{z} = \boldsymbol{\mu} + \boldsymbol{\sigma} \odot \boldsymbol{\epsilon}, \quad \boldsymbol{\epsilon} \sim \mathcal{N}(0, \mathbf{I})$$

This allows the sampling operation to be differentiable and the model to be trained via backpropagation.

The decoder, $p_\theta(\mathbf{x}|\mathbf{z})$, then maps the latent vector back to the input space. The goal is to ensure that the reconstructed input $\mathbf{x}'$ is as close as possible to the original $\mathbf{x}$, while also enforcing a structure on the latent space that encourages meaningful interpolation and generation.

This structure paves the way for powerful applications in generative modeling, while also requiring a deeper theoretical framework — which we now explore.

Note: Figure 4 shows how the latent space in dependent of the input data as we can only generate new data depending on the type of data in our input layer

4.2.3 Problem in distributions links

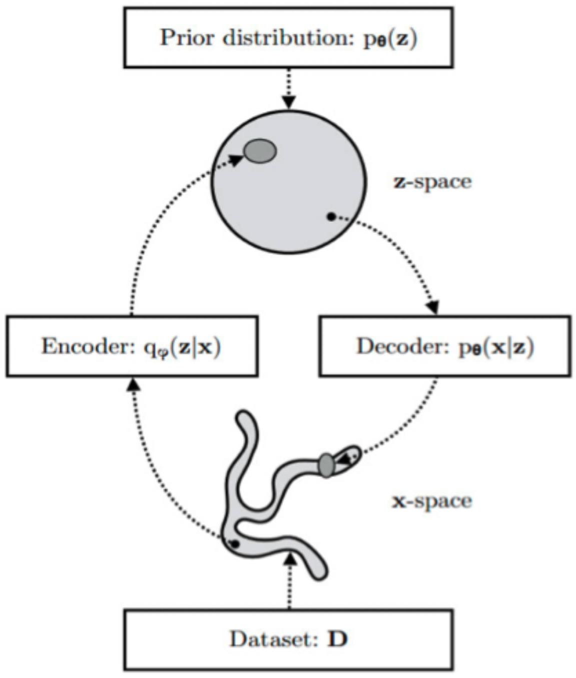

Prior distribution: $p_\theta(\mathbf{z})$
z-space
Encoder: $q_\varphi(\mathbf{z}|\mathbf{x})$
Decoder: $p_\theta(\mathbf{x}|\mathbf{z})$
x-space
Dataset: $\mathbf{D}$

Figure 4: VAEs

In Variational Autoencoders, the encoder maps each input $\mathbf{x}$ not to a fixed point but to a probability distribution over the latent space. This introduces randomness into the reconstruction process, since each output $\mathbf{x}'$ is generated by decoding a sampled latent vector $\mathbf{z}$.

To ensure that the latent space is smooth and allows meaningful generation, the encoder's distribution $q_\phi(\mathbf{z}|\mathbf{x})$ is encouraged to be close to a simple prior distribution $p_\theta(\mathbf{z})$, typically a standard normal. This regularization improves generalization and prevents overfitting to the training data.

However, a fundamental problem arises: the true posterior distribution $p_\theta(\mathbf{z}|\mathbf{x})$ needed for optimal training is intractable, as it involves an integral that cannot be computed analytically. To overcome this, VAEs introduce a tractable approximation $q_\phi(\mathbf{z}|\mathbf{x})$ using a method called *variational inference*. This allows the model to be trained efficiently by optimizing a surrogate objective, which we now proceed to analyze mathematically.

$$q_\phi(z|x) \approx p_\theta(z|x)$$

4.2.4 Reparameterization trick

One of the main challenges in training models such as variational autoencoders (VAEs) is efficiently adjusting the parameters of both the encoder and decoder, which are typically implemented as neural networks. While these models can usually be trained by computing the gradient of the cost function with respect to their parameters, the sampling of the latent variable z introduces a stochastic component that prevents direct gradient calculation.

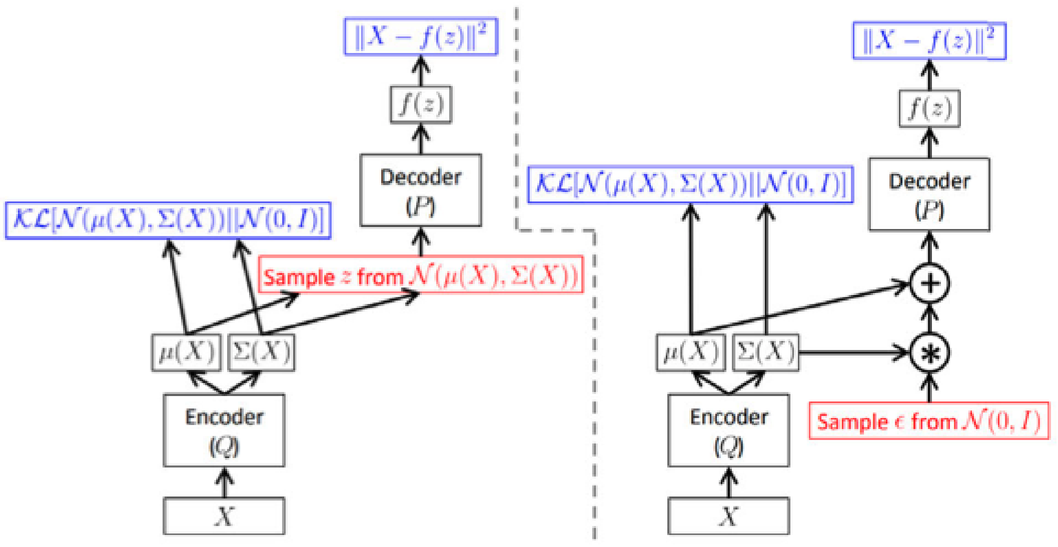

$\|X - f(z)\|^2$
$f(z)$
Decoder (P)
$\mathcal{KL}[\mathcal{N}(\mu(X), \Sigma(X))\|\mathcal{N}(0, I)]$
Sample z from $\mathcal{N}(\mu(X), \Sigma(X))$
$\mu(X)$
$\Sigma(X)$
Encoder (Q)
X
$\|X - f(z)\|^2$
$f(z)$
Decoder (P)
$\mathcal{KL}[\mathcal{N}(\mu(X), \Sigma(X))\|\mathcal{N}(0, I)]$
+
*
$\mu(X)$
$\Sigma(X)$
Sample ϵ from $\mathcal{N}(0, I)$
Encoder (Q)
X

Figure 5: VAEs

The **reparameterization trick** addresses this issue by expressing z as a deterministic function of the model parameters and a random variable sampled from a standard normal distribution, $\mathcal{N}(0, 1)$. This allows gradients to be computed with respect to the model parameters, enabling efficient training.

This trick allows the gradient to pass through the sampling process, making the optimization tractable.

4.3 Learning of the VAEs

The characteristic of the variational autoencoder that distinguishes it from other generative models is the use of a probabilistic model, $p_\theta(z|x)$, which is used to approximate the distribution of the latent space from the distribution of the input data. This approach is known as *amortized inference*. To compute this distribution, Bayes' theorem can be used:

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

However, the denominator in expression 1 is intractable because it requires computing the marginal probability:

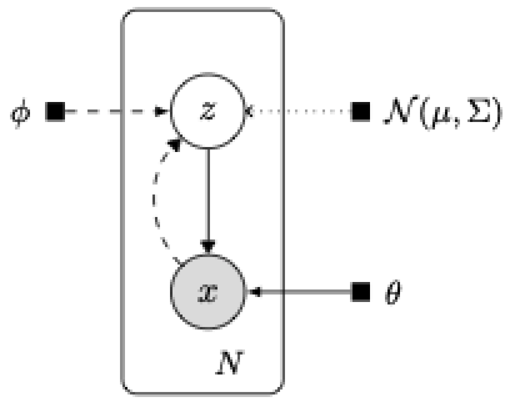
$$p_\theta(x) = \int p_\theta(x|z)p_\theta(z)dz \tag{2}$$

To overcome this difficulty in the variational autoencoder, instead of calculating the posterior distribution $p_\theta(z|x)$, an approximation is sought such that:

$$q_\phi(z|x) \approx p_\theta(z|x)$$

This is as we have seen previously called Variational inference 4.2.3

Graphically, the variational autoencoder can be represented by the Bayesian network shown in Figure 6. In this graph, the generated data are obtained from the latent space using a decoder $p_\theta(x|z)$ and the latent space, which follows a prior distribution $p_\theta(z) \sim \mathcal{N}(\mu, \Sigma)$, is constructed from an encoder $q_\phi(z|x)$.



5

Figure 6: Bayesian graph

4.3.1 Objetive Function

The goal of training a Variational Autoencoder is to find the parameters θ and ϕ that make our model assign high probability to the observed data $\mathbf{x}$. This means we would ideally want to maximize the marginal likelihood $p_\theta(\mathbf{x})$, or more precisely:

$$\max_\theta \log p_\theta(\mathbf{x}) \tag{3}$$

However, computing $p_\theta(\mathbf{x})$ exactly is not tractable because it requires integrating over all possible latent variables $\mathbf{z}$:

$$\log p_\theta(\mathbf{x}) = \log \int p_\theta(\mathbf{x}, \mathbf{z})\, d\mathbf{z} \tag{4}$$

This integral is typically too complex to solve analytically. To handle this, we introduce a new, simpler distribution $q_\phi(\mathbf{z}|\mathbf{x})$ to approximate the true posterior $p_\theta(\mathbf{z}|\mathbf{x})$. This is the core idea behind *variational inference.*

We can now rewrite the log-likelihood using this approximation:

$$\log p_\theta(\mathbf{x}) = \mathbb{E}_{q_\phi(\mathbf{z}|\mathbf{x})}\left[\log p_\theta(\mathbf{x})\right] \tag{5}$$

Although this seems trivial, we can add and subtract the term $\log q_\phi(\mathbf{z}|\mathbf{x})$ inside the expectation to derive something useful:

$$\log p_\theta(\mathbf{x}) = \mathbb{E}_{q_\phi(\mathbf{z}|\mathbf{x})}\left[\log \frac{p_\theta(\mathbf{x}, \mathbf{z})}{q_\phi(\mathbf{z}|\mathbf{x})}\right] + \mathrm{KL}\left(q_\phi(\mathbf{z}|\mathbf{x}) \,\|\, p_\theta(\mathbf{z}|\mathbf{x})\right) \tag{6}$$

Here, $\mathrm{KL}(\cdot\|\cdot)$ refers to the **Kullback–Leibler divergence**, a measure of how different two probability distributions are. It is always non-negative, and zero only when the two distributions are exactly the same. In this case, it measures how far our approximation $q_\phi(\mathbf{z}|\mathbf{x})$ is from the true posterior $p_\theta(\mathbf{z}|\mathbf{x})$.

Now, since the KL term is non-negative, we can drop it to obtain a lower bound:

$$\log p_\theta(\mathbf{x}) \geq \mathbb{E}_{q_\phi(\mathbf{z}|\mathbf{x})}\left[\log \frac{p_\theta(\mathbf{x}, \mathbf{z})}{q_\phi(\mathbf{z}|\mathbf{x})}\right] \tag{7}$$

This expression is known as the **Evidence Lower Bound (ELBO)**. It serves as a surrogate objective function to optimize instead of the intractable $\log p_\theta(\mathbf{x})$.

We now expand the joint probability $p_\theta(\mathbf{x}, \mathbf{z}) = p_\theta(\mathbf{x}|\mathbf{z})p_\theta(\mathbf{z})$:

$$\mathcal{L}(\theta, \phi; \mathbf{x}) = \mathbb{E}_{q_\phi(\mathbf{z}|\mathbf{x})}\left[\log p_\theta(\mathbf{x}|\mathbf{z})\right] - \mathrm{KL}\left(q_\phi(\mathbf{z}|\mathbf{x}) \,\|\, p_\theta(\mathbf{z})\right) \tag{8}$$

This is the final VAE objective function. It has two key terms:

- **Reconstruction term:** $\mathbb{E}_{q_\phi(\mathbf{z}|\mathbf{x})}\left[\log p_\theta(\mathbf{x}|\mathbf{z})\right]$ Encourages the decoder to accurately reconstruct the input $\mathbf{x}$ from latent samples $\mathbf{z}$.
- **Regularization term:** $\mathrm{KL}\left(q_\phi(\mathbf{z}|\mathbf{x}) \,\|\, p_\theta(\mathbf{z})\right)$ Encourages the encoder's latent distribution to be close to a fixed prior (typically a standard normal distribution).

Thus, during training, we maximize this ELBO instead of $\log p_\theta(\mathbf{x})$. When training over a dataset, we typically maximize the sum of ELBOs over all data points:

$$\max_{\theta,\phi} \sum_{i=1}^{N} \mathcal{L}(\theta, \phi; \mathbf{x}^{(i)}) \tag{9}$$

4.3.2 Reparameterizacion trick

To allow gradient-based optimization through the stochastic sampling process introduced in the latent space, we apply the so-called *reparameterization trick*. Instead of sampling $\mathbf{z} \sim \mathcal{N}(\mu_\phi(\mathbf{x}), \sigma^2_\phi(\mathbf{x}))$ directly, we express the latent variable as a deterministic function of μ, σ, and an auxiliary variable:

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

This allows us to backpropagate through $\mathbf{z}$ during training, ensuring the encoder is properly updated.

4.3.3 optimization

The standard training procedure for VAEs involves optimizing the negative Evidence Lower Bound (ELBO), composed of two terms:

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

During training, we encode the input $\mathbf{x}$ into a latent distribution, sample $\mathbf{z}$ using the reparameterization trick, decode it to reconstruct $\mathbf{x}'$, and compute this loss. Optimization is typically performed using stochastic gradient descent or variants such as Adam.

4.3.4 Sampling generation

After training, new data samples can be generated by sampling from the prior distribution $p(\mathbf{z}) = \mathcal{N}(0, \mathbf{I})$ and decoding:

$$\mathbf{z} \sim \mathcal{N}(0, \mathbf{I}), \quad \mathbf{x}' = p_\theta(\mathbf{x}|\mathbf{z})$$

This generative capability is one of the main advantages of VAEs over traditional autoencoders.

4.4 Proof of the KL Divergence Formula

Here, we present the step-by-step proof of the closed-form KL divergence between two Gaussian distributions. Consider the Gaussian approximate posterior $q_\theta(z|x) = \mathcal{N}(\mu, \sigma^2)$ and the prior $p(z) = \mathcal{N}(0, I)$.

Step 1: Defining the Distributions

Let:

$$q_\theta(z|x) = \mathcal{N}(z; \mu, \sigma^2 I), \tag{11}$$

$$p(z) = \mathcal{N}(z; 0, I). \tag{12}$$

Step 2: KL Divergence Formula

The KL divergence between two Gaussian distributions q and p is defined as:

$$\mathrm{KL}(q_\theta(z|x)\|p(z)) = \int q_\theta(z|x) \log \frac{q_\theta(z|x)}{p(z)} dz. \tag{13}$$

Step 3: Expanding the Logarithm

Substitute the Gaussian probability density functions:

$$q_\theta(z|x) = \frac{1}{(2\pi\sigma^2)^{m/2}} \exp\left(-\frac{1}{2\sigma^2}\|z-\mu\|^2\right), \tag{14}$$

$$p(z) = \frac{1}{(2\pi)^{m/2}} \exp\left(-\frac{1}{2}\|z\|^2\right). \tag{15}$$

Taking the logarithm of the ratio:

$$\log \frac{q_\theta(z|x)}{p(z)} = -\frac{1}{2}\left(\sum_{j=1}^{m} \frac{(z_j-\mu_j)^2}{\sigma_j^2} - z_j^2\right) - \sum_{j=1}^{m} \log \sigma_j. \tag{16}$$

Step 4: Calculating the Integral

The expectation of the quadratic terms yields:

$$\mathbb{E}_{q_\theta(z|x)}\left[\|z-\mu\|^2\right] = \sum_{j=1}^{m} \sigma_j^2. \tag{17}$$

and

$$\mathbb{E}_{q_\theta(z|x)}\left[\|z\|^2\right] = \sum_{j=1}^{m} (\mu_j^2 + \sigma_j^2). \tag{18}$$

Combining these results gives the closed-form solution:

$$\mathrm{KL}(q_\theta(z|x)\|p(z)) = \frac{1}{2}\sum_{j=1}^{m}\left(1 + \log(\sigma_j^2) - \mu_j^2 - \sigma_j^2\right). \tag{19}$$

Step 5: Final Result

This closed-form expression of the KL divergence is s one of the main advantages of VAEs over traditional autoencoders.

Differents VAEs architechtures tested

4.5 β-VAE

Variational Autoencoders (VAEs) are generative models that learn a compressed latent representation of data by optimizing a trade-off between reconstruction accuracy and the regularization of the latent space. However, standard VAEs often struggle to produce *disentangled* representations, where individual latent dimensions correspond to semantically meaningful factors of variation in the data.

To address this limitation, **β-VAEs** 8 introduces a hyperparameter $\beta > 0$ to the original VAE loss function. The modified objective becomes:

$$\mathcal{L}(\theta, \phi; \mathbf{x}, \mathbf{z}, \beta) = \mathbb{E}_{q_\phi(\mathbf{z}|\mathbf{x})}[\log p_\theta(\mathbf{x}|\mathbf{z})] - \beta\, D_{KL}(q_\phi(\mathbf{z}|\mathbf{x}) \parallel p(\mathbf{z})), \tag{10}$$

where increasing β strengthens the Kullback–Leibler (KL) divergence penalty, encouraging the posterior $q_\phi(\mathbf{z}|\mathbf{x})$ to align more closely with the unit Gaussian prior $p(\mathbf{z})$. As a result, β-VAEs tend to learn more disentangled latent representations, which are desirable in tasks such as interpretability, transfer learning, and controllable generation.

When $\beta = 1$, the β-VAE reduces to the original VAE formulation. Values of $\beta > 1$ introduce a stricter bottleneck on the latent code, which often leads to a trade-off: improved disentanglement at the cost of lower reconstruction fidelity. Despite this, β-VAEs have become a foundational extension in the study of unsupervised representation learning.

This trade-off between disentanglement and reconstruction quality is most evident when visualizing the learned latent representations. In particular, when β is increased, the latent space tends to align more strongly with the ground-truth generative factors, reducing overlap between different classes or modes in the latent dimensions.

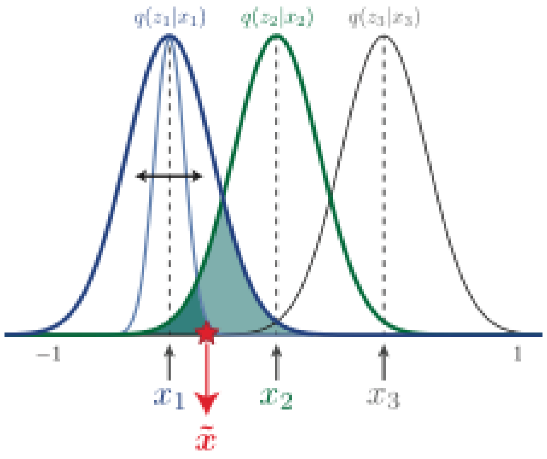


Figure 7: Latent space representations learned by a VAE (left) and a β-VAE (right). With a higher β, the overlap between classes is reduced and the factors become more separated.

This figure illustrates how the latent representations evolve as the KL penalty increases. On the left, a standard VAE tends to produce entangled encodings with overlapping regions across classes. On the right, a β-VAE trained with a larger β successfully learns more factorized and interpretable representations, with lower class overlap.

Such disentangled latent spaces are beneficial for downstream tasks like classification, generation, or interpretability, especially in low-data regimes or settings where semantic control is required.

4.6 Planar Flow VAE

Standard Variational Autoencoders (see Section4.2.2) typically assume that the posterior distribution over the latent variables is Gaussian and diagonal. This assumption simplifies training, but limits the model's ability to approximate complex, multimodal posteriors.

Rezende and Mohamed10 propose using *normalizing flows* to increase the flexibility of the approximate posterior while keeping the model tractable. A normalizing flow is a sequence of invertible and differentiable transformations that progressively reshape a simple initial distribution (e.g., Gaussian) into a more complex one.

One particular example introduced in their work is the **planar flow**, a lightweight transformation of the form:

$$f(z) = z + u \cdot \tanh(w^\top z + b),$$

where $u, w \in \mathbb{R}^d$, and $b \in \mathbb{R}$ are learnable parameters. This transformation bends or warps the space in a way that allows the final latent variable z_K to follow a much richer distribution than the initial $z_0 \sim \mathcal{N}(0, I)$.

Visually, each flow adds structure and curvature to the latent space, making it better suited to match the true posterior. The composition of several planar flows retains a closed-form expression for the log-determinant Jacobian, which is necessary to compute the KL divergence in the variational objective.

This method improves the flexibility of variational inference without significantly increasing the number of parameters or computational cost, and laid the foundation for later developments in flow-based generative models.

5 Metrics used in the code for evaluation

MSE and **KLD** are metrics mainly used when the goal is to *reconstruct* images, like in Variational Autoencoders (VAEs). MSE checks how close the reconstructed image is to the original one, while KLD helps organize the model's latent space. On the other hand, **FID** and **IS** are used for evaluating models that *generate* new images. These metrics focus on how realistic and diverse the generated images are, helping us understand how well the model creates new, believable content.

5.1 Comparison between Mean Squared Error (MSE) and Kullback-Leibler Divergence (KLD)

Mean Squared Error (MSE) evaluates the quality of the image reconstruction by measuring the average squared difference between the original image x and its reconstruction $\hat{x}$. Mathematically, it is defined as:

$$\mathrm{MSE}(x, \hat{x}) = \frac{1}{n} \sum_{i=1}^{n} (x_i - \hat{x}_i)^2 \tag{11}$$

In the context of *Variational Autoencoders* (VAEs), minimizing the MSE corresponds to maximizing the log-likelihood under the assumption that data are modeled with a fixed-variance Gaussian distribution.

Kullback-Leibler Divergence (KLD) measures the discrepancy between two probability distributions: the approximate posterior $q(z|x)$ and the prior $p(z)$. It is formally defined as:

$$D_{\mathrm{KL}}(q(z|x) \,||\, p(z)) = \mathbb{E}_{q(z|x)} \left[\log \frac{q(z|x)}{p(z)} \right] \tag{12}$$

In VAEs, the KLD term acts as a regularizer that pushes the latent representations z to be close to the desired prior (typically a standard normal distribution $\mathcal{N}(0, I)$), enabling coherent generation of new samples.

The following table summarizes the main differences between MSE and KLD:

Aspect	**MSE**	**KLD**
What it measures	Reconstruction error pixel by pixel	Divergence between probability distributions
Mathematical definition	$\frac{1}{n}\sum_{i=1}^{n}(x_i - \hat{x}_i)^2$	$\mathbb{E}_{q(z\|x)}\left[\log \frac{q(z\|x)}{p(z)}\right]$
Main objective	Accurate reconstruction of inputs	Organization of the latent space
High value impact	Blurry or inaccurate reconstructions	Disorganized latent space
Relative importance	Key for reconstruction tasks	Key for sample generation tasks

Table 1: Comparison between MSE and KLD in VAEs.

Conclusion: In practice, achieving a good VAE performance requires a balance between the two terms. While MSE ensures precise input reconstruction, KLD guarantees a structured and smooth latent space, which is essential for generating coherent new samples.

5.2 Fréchet Inception Distance (FID)

The **Fréchet Inception Distance (FID)** is a metric used to evaluate the quality of generated images in models like **Variational Autoencoders (VAEs)**. FID compares the distributions of features extracted from real and generated images, considering both the mean and covariance of these features.

Mathematical Definition of FID:

$$\text{FID} = \|\mu_r - \mu_g\|_2^2 + \text{Tr}(\Sigma_r + \Sigma_g - 2(\Sigma_r\Sigma_g)^{1/2})$$

Where:
- μ_r is the mean vector of features extracted from real images. - μ_g is the mean vector of features extracted from generated images. - Σ_r is the covariance matrix of features from real images. - Σ_g is the covariance matrix of features from generated images.

FID in VAEs:

For **Variational Autoencoders (VAEs)**, FID is used to measure how close the generated images are to real images in terms of their feature distributions. The process involves:

1. **Feature Extraction**: A pre-trained **Inception model** extracts features from both real and generated images.

2. **Statistics Calculation**: The mean and covariance of these features are computed for both sets of images.

3. **Fréchet Distance Calculation**: The formula calculates the distance between the real and generated feature distributions. A low FID indicates that the generated images closely match the real ones, while a high FID suggests significant differences.

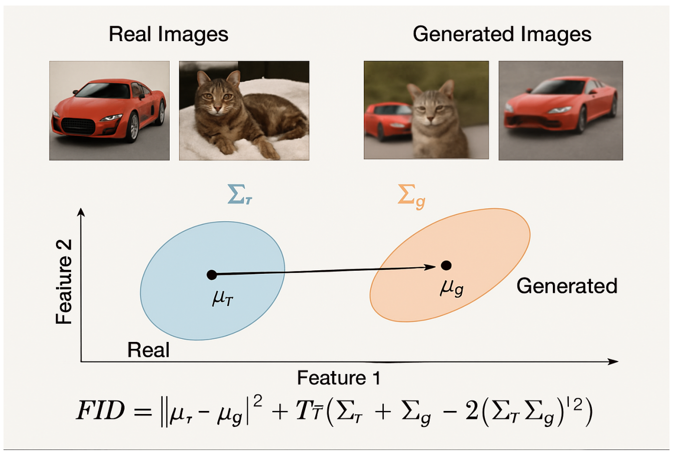


Figure 8: FID in relation with generate images

Interpretation of FID: - **Low FID**: The generated images are statistically similar to real images, suggesting a well-trained VAE. - **High FID**: The generated images differ from real images, indicating the model's failure to capture the true data distribution.

In summary, FID is a valuable metric for assessing how realistic the images generated by VAEs are by comparing the statistical properties of real and generated images.

5.3 Inception Score (IS)

The **Inception Score (IS)** is another metric used to evaluate the quality of images generated by models like **Variational Autoencoders (VAEs)**. Unlike FID, the IS focuses on how well the generated images can be classified by a pre-trained Inception model. High IS values indicate that the generated images are not only realistic but also diverse.

Mathematical Definition of IS:

The Inception Score is defined as:

$$\text{IS}(x) = \exp\left(\mathbb{E}_x\left[D_{\text{KL}}\left(p(y|x)\|p(y)\right)\right]\right)$$

Where:
- $p(y|x)$ is the conditional probability distribution over classes for an image x generated by the model.
- $p(y)$ is the marginal probability distribution over the classes of the Inception model.
- D_{KL} is the **Kullback-Leibler (KL) divergence**, which measures the difference between two probability distributions.

IS in VAEs:

For **Variational Autoencoders (VAEs)**, the Inception Score is used to measure two important aspects of generated images:

1. **Diversity**: The KL divergence term in the IS equation ensures that the model produces images that are not concentrated in a single class. High diversity in the generated images results in a high IS value.

2. **Realism**: The expected KL divergence term ensures that the model's generated images are classified with high confidence into a single class, indicating that the generated images look realistic.

Figure 9: Sample of generated images achieving an Inception Score of 900.15. The maximum achievable Inception Score is 1000, and the highest achieved in the literature is on the order of 10.

Interpretation of IS:
- **High IS**: A high Inception Score means that the generated images are not only realistic but also diverse, as they are confidently classified into different classes.
- **Low IS**: A low Inception Score indicates that the generated images are either unrealistic or lack diversity, being concentrated in a single class.

In summary, the Inception Score evaluates both the **realism** and **diversity** of generated images by measuring how confidently and diversely they can be classified by a pre-trained Inception model.

5.4 Comparison between FID and IS

The Fréchet Inception Distance (FID) and Inception Score (IS) are both important metrics for evaluating generative models, particularly in the context of image generation. While both metrics focus on assessing the quality of generated images, they do so in different ways.

Fréchet Inception Distance (FID): FID measures the distance between the distributions of real and generated images in the feature space of an Inception-v3 model. It compares the mean and covariance of feature distributions extracted from the real and generated images, with a lower FID indicating that the generated images are closer to real images.

The mathematical expression for FID is:

$$\text{FID} = \|\mu_r - \mu_g\|^2 + \text{Tr}\left(\Sigma_r + \Sigma_g - 2(\Sigma_r\Sigma_g)^{\frac{1}{2}}\right)$$

Where μ_r and Σ_r are the mean and covariance of the real image distribution, and μ_g and Σ_g are those for the generated distribution.

Inception Score (IS): The Inception Score evaluates the clarity and diversity of generated images by examining the conditional class distribution produced by an Inception-v3 model. It uses the Kullback-Leibler (KL) divergence to quantify the difference between the predicted class distribution for each image and the overall distribution of all classes.

The Inception Score is calculated as:

$$\text{IS} = \exp\left(\mathbb{E}_x\left[D_{\text{KL}}(p(y|x) \parallel p(y))\right]\right)$$

Where $p(y|x)$ is the conditional class probability, and $p(y)$ is the marginal class distribution.

Key Differences:

- **FID** considers both the mean and covariance of the feature distributions, making it sensitive to both the quality and the diversity of the generated images.
- **IS** focuses on the clarity and diversity of images based on their classification in a pre-trained model, but does not directly capture the full distribution of the images.
- **FID** is generally considered more robust because it accounts for both the quality and diversity of the generated images, while **IS** might favor models that produce sharp but less diverse images.

Although both metrics are widely used, FID tends to be more reliable for comprehensive evaluation in the context of generative models due to its consideration of distributional differences between real and generated images.

6 Model evaluation

In this section we are going to explian and visualize the VAEs results on the famous dataset called : Fashion MNIST

6.1 VAE Architectures Used

In this project, 14 different architectures of **Variational Autoencoders (VAEs)** were designed and trained using the **Fashion-MNIST** dataset. The goal was to study the effect of three key components on the model's performance:

- The size of the **latent space** z
- The β parameter for regularization in Beta-VAEs
- The use of **planar flows** to increase the expressiveness of the latent distribution

Summary of Architectures

The following 14 models were instantiated:

- **Vanilla VAEs:**
  - `vanilla_z8`
  - `vanilla_z32`
  - `vanilla_z64`
  - `vanilla_z128`
- **Beta-VAEs:** with $\beta \in \{0.25, 2, 4\}$
  - `beta0.25_z32`, `beta2_z32`, `beta4_z32`
  - `beta0.25_z64`, `beta2_z64`, `beta4_z64`
- **Flow VAEs:** with number of planar flow layers $L \in \{4, 8\}$
  - `flow4_z32`, `flow8_z32`
  - `flow4_z64`, `flow8_z64`

Key Parameter Descriptions

- **Latent space dimension z:** controls the size of the latent representation. Values of $z = 8, 32, 64, 128$ were tested to assess the model's ability to compress and reconstruct meaningful features.
- **Beta parameter β:** used in Beta-VAEs to scale the KL divergence term in the loss function, controlling the trade-off between reconstruction accuracy and latent space regularization. The standard VAE corresponds to $\beta = 1$. Lower values such as $\beta = 0.25$ prioritize reconstruction quality by relaxing the regularization constraint, while higher values like $\beta = 2$, $\beta = 4$, or $\beta = 5$ enforce stronger disentanglement by increasing the weight of the KL divergence term, as motivated by .
- **Planar flows:** used in Flow-VAEs to transform the latent variable via a sequence of invertible functions, increasing the flexibility of the approximate posterior. Configurations with 4 and 8 planar flow layers were explored.

6.2 Quantitative Results

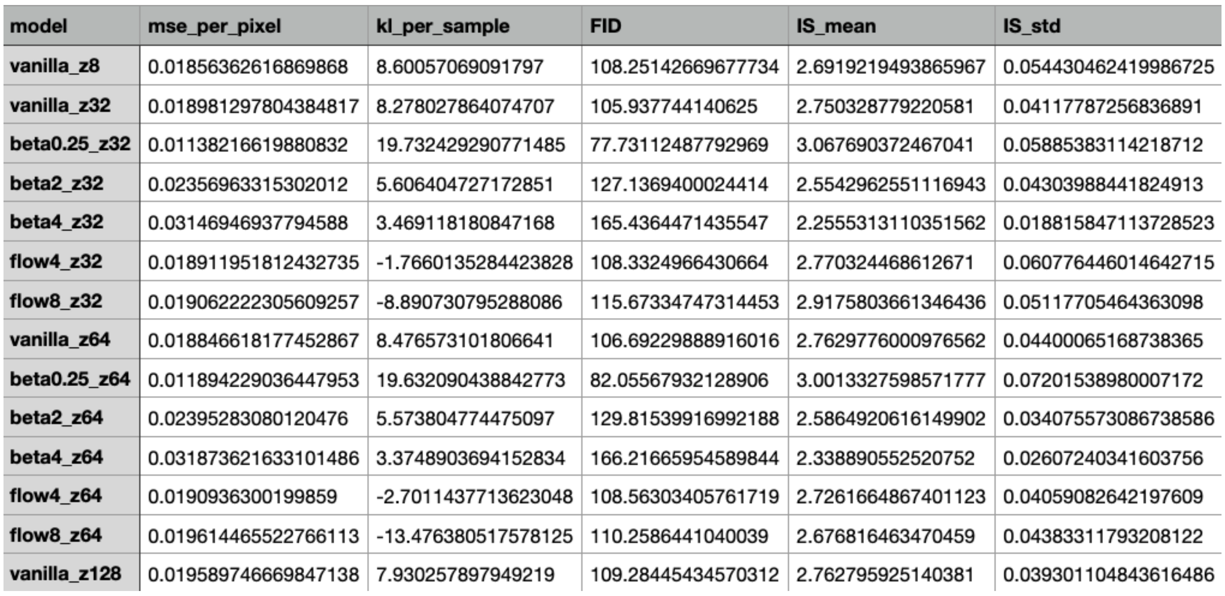
6.2.1 Evaluation Metrics

Here we report the performance of all 14 VAE models using four main metrics:

- **MSE per pixel:** Mean squared error computed between original and reconstructed images.
- **KL divergence:** Measures how close the approximate posterior is to the prior distribution.
- **FID (Fréchet Inception Distance):** Evaluates the quality and realism of generated images.
- **IS (Inception Score):** Measures both image quality and diversity by assessing how confidently a classifier predicts labels for generated images, and how diverse those predictions are. We report the mean and standard deviation across multiple splits to capture generation stability.

6.2.2 Metric Comparison Across Models

Present a table here:



model	mse_per_pixel	kl_per_sample	FID	IS_mean	IS_std
vanilla_z8	0.01856362616869868	8.60057069091797	108.25142669677734	2.6919219493865967	0.05443046241998725
vanilla_z32	0.018981297804384817	8.278027864074707	105.937744140625	2.750328779220581	0.04117787256836891
beta0.25_z32	0.01138216619880832	19.732429290771485	77.73112487792969	3.067690372467041	0.05885383114218712
beta2_z32	0.02356963315302012	5.606404727172851	127.1369400024414	2.5542962551116943	0.04303988441824913
beta4_z32	0.03146946937794588	3.469118180847168	165.4364471435547	2.2555313110351562	0.018815847113728523
flow4_z32	0.018911951812432735	-1.7660135284423828	108.3324966430664	2.770324468612671	0.060776446014642715
flow8_z32	0.019062222305609257	-8.890730795288086	115.67334747314453	2.9175803661346436	0.05117705464363098
vanilla_z64	0.018846618177452867	8.476573101806641	106.69229888916016	2.7629776000976562	0.04400065168738365
beta0.25_z64	0.011894229036447953	19.632090438842773	82.05567932128906	3.0013327598571777	0.07201538980007172
beta2_z64	0.02395283080120476	5.573804774475097	129.81539916992188	2.5864920616149902	0.034075573086738586
beta4_z64	0.031873621633101486	3.3748903694152834	166.21665954589844	2.338890552520752	0.02607240341603756
flow4_z64	0.0190936300199859	-2.7011437713623048	108.56303405761719	2.7261664867401123	0.04059082642197609
flow8_z64	0.019614465522766113	-13.476380517578125	110.2586441040039	2.676816463470459	0.04383311793208122
vanilla_z128	0.019589746669847138	7.930257897949219	109.28445434570312	2.762795925140381	0.039301104843616486

Figure 10: VAEs metrics

Quantitative Results Analysis

Across the 14 architectures tested, we observe that:

ALL the model last about 38-39 seconds

- **Reconstruction Quality:** The model `beta0.25_z32` achieves the lowest MSE per pixel, indicating the highest reconstruction fidelity. Increasing β generally leads to higher MSE, consistent with stronger regularization.

- **KL Divergence:** As expected, `beta-VAE` variants with higher β show lower KL divergence, while flow-based models yield negative KL estimates due to the nature of the flow transformations.
- **Generative Quality:** The best FID is obtained by `beta0.25_z32`, suggesting that mild regularization ($\beta = 0.25$) enhances generation quality. Flow models also perform competitively.
- **Inception Score:** Although differences are small, `beta0.25_z64` and `flow8_z64` yield the highest IS means, suggesting improved realism and diversity.
- **Efficiency:** Generation times are comparable across models, indicating that architectural complexity (e.g., flow layers) does not drastically impact speed.

6.3 Qualitative Results

6.3.1 Original vs Reconstructed Images

To evaluate reconstruction quality, we compare original images with their reconstructions to see how well it represents the information given as input.

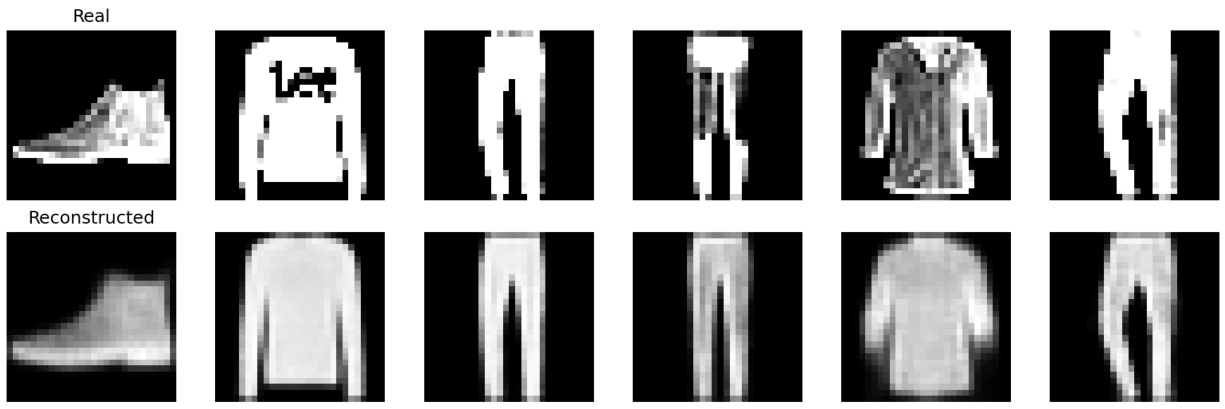


Figure 11: vanilla z=32 samples

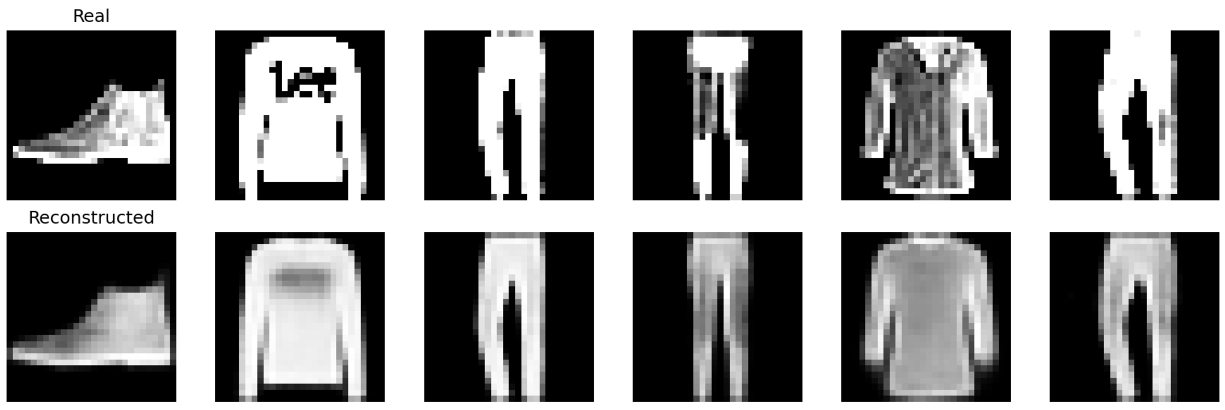


Figure 12: beta = 0.25 z=32 samples

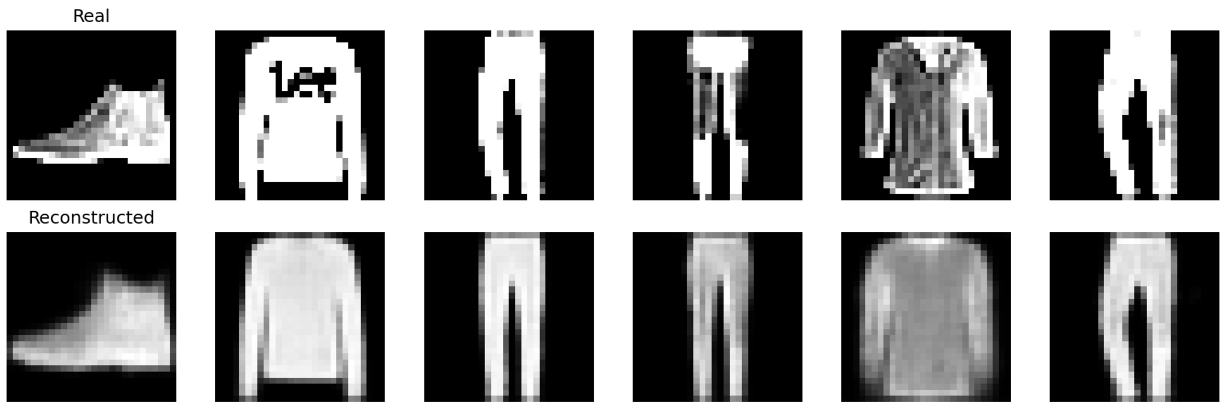


Figure 13: flow4 , z=32

Conclusion on Reconstructed Images

Due to space constraints and the high number of models evaluated, we only include three representative examples of reconstructed images corresponding to the three main VAE architectures explored. These examples were carefully selected to provide a visual summary of the reconstruction quality across the spectrum of model configurations.

Among all tested configurations, the model with $\beta = 0.25$ and latent dimension $Z = 32$ produced the most visually accurate reconstructions. This result aligns with our quantitative metrics and reflects a balanced trade-off between reconstruction fidelity and latent space regularization. A lower β value (closer to 0) gives more importance to the reconstruction loss over the KL divergence, allowing the model to capture finer image details. At the same time, a latent dimension of 32 provides enough capacity for the model to encode the essential image information without overfitting or becoming too constrained.

The combination of these factors results in reconstructed images that closely resemble the originals while still benefiting from a structured latent space suitable for generation and interpolation. Full visual comparisons for all model configurations are available in the

6.3.2 Original vs Generated Images

To evaluate generated quality in order to see how our model can generate images from 0 and still be realistic

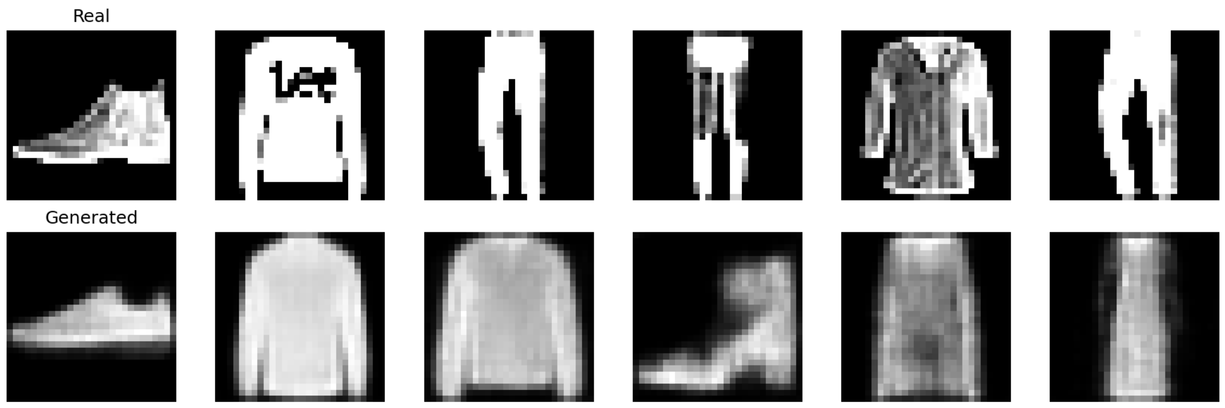


Figure 14: vanilla z=128 samples

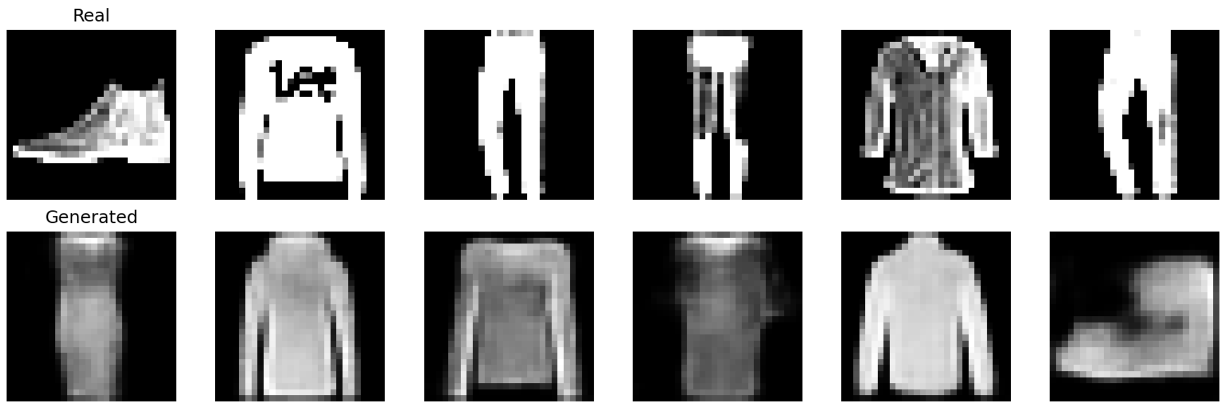


Figure 15: beta = 0.25 z=32 samples

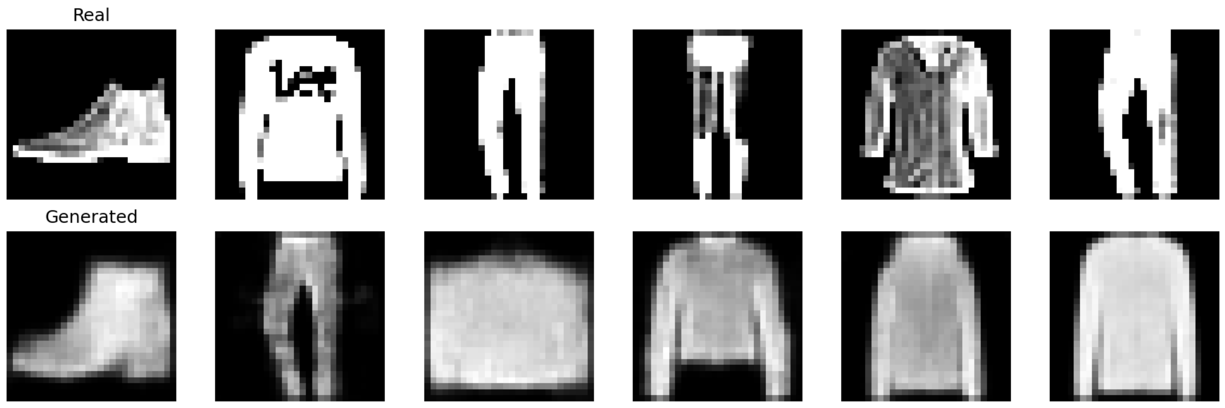


Figure 16: flow4 , z=32

Conclusion on Generated Images

As with the reconstructions, we only include a subset of generated image samples due to space limitations. Among the examples shown, the best generation quality was obtained with the Vanilla VAE using $Z = 128$, while the poorest results came from the Flow4 model with $Z = 32$.

The superior performance of the Vanilla VAE can be attributed to the larger latent space, which allows for richer and more diverse representations during sampling. In contrast, the Flow4 model with a smaller latent dimension appears too constrained, resulting in blurry or repetitive outputs. These differences highlight the importance of both latent capacity and model design when aiming for high-quality generation.

7 Final conclusion

Conclusions

The experiments show that relatively simple Variational Autoencoder models perform well on the Fashion-MNIST dataset for both reconstruction and image generation tasks, even when using a small latent dimension.

Results indicate that increasing the latent dimension from 32 to 64 or 128 brings minimal improvements. Specifically, the MSE, FID and IS change for vanilla VAE by less than 2%, suggesting that the decoder's capacity, rather than latent dimension, is likely the main limiting factor.

The best results are achieved with a lightly regularized β-VAE ($\beta = 0.25$, latent dimension = 32), providing the best balance between reconstruction accuracy and sample quality. With this setup, the mean squared reconstruction error decreases by nearly half compared to the standard VAE, while FID (approximately 78) and Inception Score (IS approximately 3.1) show significant improvement. A short linear warm-up period for β to reach 0.25 seems optimal, as it is strong enough to structure the latent space but gentle enough for the decoder to retain detailed pixel information.

Increasing β significantly beyond this optimal level causes predictable but undesirable side effects. At $\beta = 1$ (vanilla) and $\beta = 4$, the KL divergence decreases and latent factors become more clearly separated, but the visual quality suffers severely. Reconstructions become blurry, FID exceeds 160, and IS drops below 2.6. Therefore, higher β values are only recommended when disentangling latent factors (meaningful latent structure) is more important than image clarity.

Experiments with planar-flow enhanced VAEs show different behavior. Adding flow layers significantly improves the approximate posterior's match to the Gaussian prior, sometimes resulting in negative KL values (planar flows correct the base KL by subtracting the sum of log-Jacobian determinants, so if that sum exceeds the original KL, the net KL can go negative). However, better latent matching does not lead to notably better samples: FID scores remain around 110–115, and IS stays around 2.9. Thus, while planar flows effectively address issues with latent distributions, they do not improve the decoder's capability to generate sharper images. Without a deeper decoder or a more expressive likelihood, the latent improvements from flows do not enhance overall image quality.

8 References

References

[1] C. Pérez Curiel, *Autoencoders variacionales: una aproximación probabilística al aprendizaje no supervisado*, Universidad Politécnica de Madrid, 2022. Available at: https://oa.upm.es/71832/1/TESIS_MASTER_CESAR_PEREZ_CURIEL.pdf

[2] D. P. Kingma and M. Welling, *Auto-Encoding Variational Bayes*, 2014. Available at SSRN: https://ssrn.com/abstract=4999896

[3] J. Zhou, *Mathematics behind Variational Autoencoders*, 2020. Available at: https://medium.com/@j.zh/mathematics-behind-variational-autoencoders-c69297301957 (Accessed: 2025-04-18)

[4] Barratt, S., & Sharma, R. (2018). *A Note on the Inception Score.* arXiv. https://arxiv.org/abs/1801.01973

[5] GeeksforGeeks. (2021). *Role of KL-divergence in Variational Autoencoders.* https://www.geeksforgeeks.org/role-of-kl-divergence-in-variational-autoencoders/

[6] Dhupar, B. (2021). *Loss Functions in Simple Autoencoders: MSE vs. L1 Loss.* Medium. https://medium.com/@bhipanshudhupar/loss-functions-in-simple-autoencoders-mse-vs-l1-loss-4e838ae425b9

[7] Chan, D. A., *et al.* (2024). *Evaluating the Suitability of Inception Score and Fréchet Inception Distance for Assessing Generative Models.* Proceedings of the 7th International Conference on Computational Intelligence and Intelligent Systems (CIIS '24). https://dl.acm.org/doi/10.1145/3708778.3708790

[8] Mack, D. (2018). *A simple explanation of the Inception Score.* Medium. https://medium.com/octavian-ai/a-simple-explanation-of-the-inception-score-372dff6a8c7a

[9] C. P. Burgess, I. Higgins, A. Pal, L. Matthey, N. Watters, G. Desjardins, and A. Lerchner,
Understanding disentangling in β-VAE, arXiv preprint arXiv:1804.03599, 2018.
https://arxiv.org/pdf/1804.03599

[10] D. J. Rezende and S. Mohamed,
Variational Inference with Normalizing Flows,
arXiv preprint arXiv:1505.05770, 2015.
https://arxiv.org/pdf/1505.05770