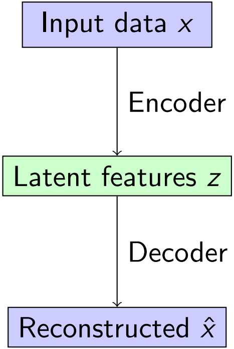
Variational Autoencoders (VAE)

AIT SAID Yassine

November 3, 2024

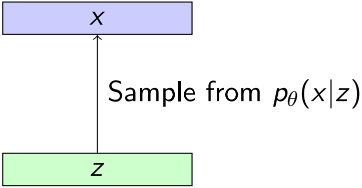
Background: Autoencoders



- Autoencoders learn to reconstruct data and can learn meaningful features, useful for initializing supervised models.
- Latent features capture key variations in the training data. But, can we use an autoencoder to generate new images?

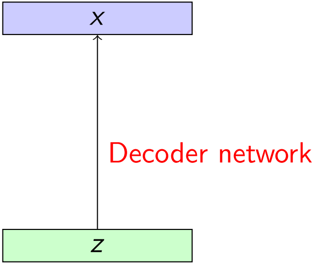
Variational Autoencoders

- A probabilistic extension of autoencoders - enables sampling from the model to generate data.
- Assume the training data $\{x^{(i)}\}_{i=1}^{N}$ is generated from a latent representation z.
- **Intuition:** x is an image, and z is the latent factor used to generate x: including attributes, orientation, etc.



VAE: Model Representation

- Goal: Estimate parameters θ for the generative model.
- Choose a prior $p(z)$, typically a simple distribution (e.g., Gaussian).
- Model $p(x|z)$ with a neural network to capture data complexity.



VAE: Likelihood and Intractability

- The data likelihood $p_\theta(x) = \int p_\theta(x|z)p(z)dz$ is intractable.
- The posterior $p_\theta(z|x) = \frac{p_\theta(x|z)p(z)}{p_\theta(x)}$ is also intractable.

Solution: Define an encoder network $q_\phi(z|x)$ that approximates $p_\theta(z|x)$.

Loss Function

Now, using our encoder and decoder networks, let's derive the (log) data likelihood:

$$
\begin{aligned}
\log p_\theta(x^{(i)}) &= \mathbb{E}_{z\sim q_\phi(z|x^{(i)})}\left[\log p_\theta(x^{(i)})\right] \\
&= \mathbb{E}_{z\sim q_\phi(z|x^{(i)})}\left[\log \frac{p_\theta(x^{(i)}, z)}{q_\phi(z|x^{(i)})}\right] \\
&= \mathbb{E}_{z\sim q_\phi(z|x^{(i)})}\left[\log \frac{p_\theta(x^{(i)}|z)p(z)}{q_\phi(z|x^{(i)})}\right] \\
&= \mathbb{E}_{z\sim q_\phi(z|x^{(i)})}\left[\log p_\theta(x^{(i)}|z)\right] - \mathbb{E}_{z\sim q_\phi(z|x^{(i)})}\left[\log \frac{q_\phi(z|x^{(i)})}{p(z)}\right]
\end{aligned}
$$

Loss Function (continued)

$$\log p_\theta(x^{(i)}) = \mathbb{E}_{z\sim q_\phi(z|x^{(i)})}\left[\log p_\theta(x^{(i)}|z)\right] - D_{KL}(q_\phi(z|x^{(i)})\|p(z)) + D_{KL}(q_\phi(z|x^{(i)})\|p_\theta(z|x^{(i)})) \geq 0$$

Tractable lower bound to optimize with gradients!
$p_\theta(x|z)$ is differentiable, and so is the KL term

$$\mathcal{L}(x^{(i)}, \theta, \phi) = \mathbb{E}_{z\sim q_\phi(z|x^{(i)})}\left[\log p_\theta(x^{(i)}|z)\right] - D_{KL}(q_\phi(z|x^{(i)})\|p(z))$$

Loss Components

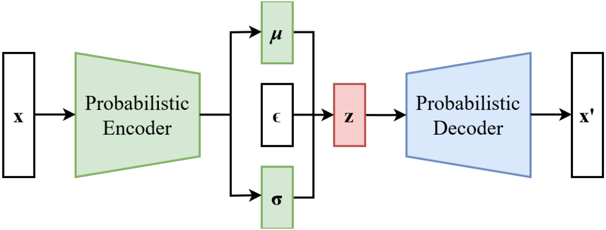
- **Reconstruction Term:** $\mathbb{E}_{q_\phi(z|x)}[\log p_\theta(x|z)]$
- **KL Divergence Term:** $D_{KL}(q_\phi(z|x) \| p(z))$
- These terms balance the accuracy of reconstruction with regularization.

Reparameterization Trick

- Use the reparameterization trick to make sampling differentiable:

$$z = \mu + \sigma \cdot \epsilon \quad \text{where } \epsilon \sim \mathcal{N}(0, I)$$

- Allows gradients to flow through stochastic layers during backpropagation.



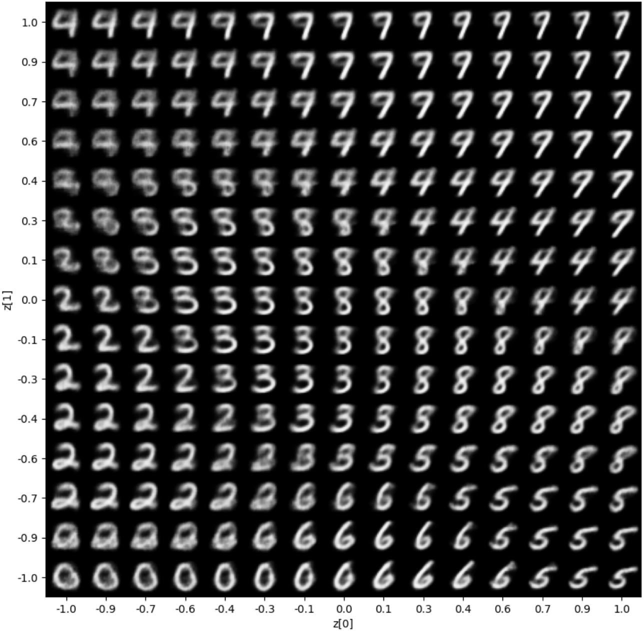
MNIST Dataset

- **MNIST (Modified National Institute of Standards and Technology)** :
  - A dataset of 70,000 handwritten digit images.
  - Each image is grayscale with a size of 28×28 pixels.
- **Data Structure** :
  - **60,000 images** for training.
  - **10,000 images** for testing.
  - Digits range from 0 to 9, each with multiple handwritten representations.
- **Use in Machine Learning** :
  - Popular for testing machine learning and deep learning models.
  - Used for tasks such as classification, image generation, and dimensionality reduction.
  - Serves as a benchmark to evaluate algorithm performance on handwritten digit recognition.

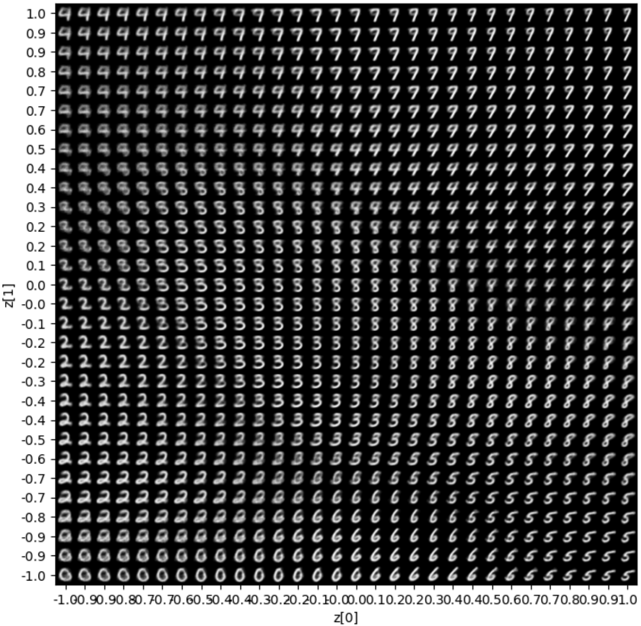
VAE Architecture

- **Encoder** :
  - Transforms an input image of size 28×28 into a low-dimensional (2D) latent vector.
  - Uses convolutional layers to extract features, followed by dense layers to produce the parameters *z_mean* and *z_log_var* of the latent distribution.
- **Reparameterization Trick** :
  - Generates a sample z in the latent space using $z = \mu + \sigma \cdot \epsilon$.
  - Makes sampling differentiable, allowing training via backpropagation.
- **Decoder** :
  - Takes the latent vector z and reconstructs a 28×28 image.
  - Uses transposed convolutional layers to progressively upsample the latent vector to the original image size.
- **Complete VAE Model** :
  - Combines the encoder and decoder to form a trainable model.
  - Uses a total loss function.

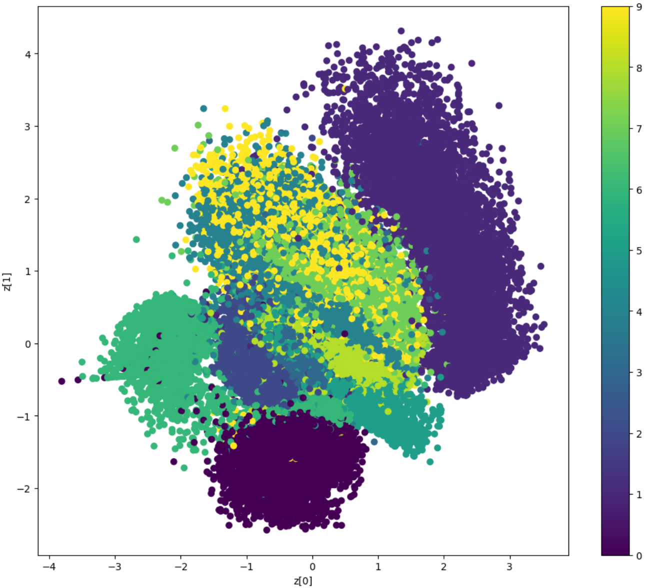
Latent Space with a 15 $\times$ 15 Grid of Digits



Latent Space with a 30 × 30 Grid of Digits



Class Representation of Input Data using z_{mean}



Conclusion

- **Variational Autoencoders (VAEs)** provide a powerful framework for learning meaningful, low-dimensional representations of data.
- By combining **probabilistic modeling** with **neural networks**, VAEs can:
  - Reconstruct data with high fidelity,
  - Generate new, realistic samples,
  - Capture latent structures and variations within the data.
- **Applications and Impact**:
  - Widely used in image generation, anomaly detection, and semi-supervised learning.
  - Their flexibility makes VAEs a cornerstone in generative models, with applications extending to text, audio, and other modalities.
- **Future Directions**:
  - Improvements in training stability and sampling techniques.
  - Combining VAEs with other architectures (e.g., GANs) for enhanced generative capabilities.