Final Project of Deep Learning

UNIVERSITÉ PARIS-SACLAY

FACULTY OF SCIENCES - MASTER IN DATA SCIENCE

Exploring Digit Generation and Representation with Variational Auto-Encoders (VAEs)

Academic Year 2023-2024

Group: Pablo Mollá Chárlez, Junjie Chen & Pavlo Poliuha

Contents

1 Objectives

The primary objectives of this project are to explore and understand the capabilities of Variational Auto-Encoders (VAEs) in generating and representing digit images effectively.

This exploration includes several key components and computational techniques that are foundational to the operation and effectiveness of VAEs. We aim to address the following topics within the report: VAEs concept, the Monte-Carlo Estimation, the Kullback-Leibler Divergence, the Encoder and Decoder Mechanics, the Python Implementation considered and finally the Latent Space Visualization obtained.

By the end of this project, we expect to gain a comprehensive understanding of the theoretical and practical aspects of Variational Autoencoders, specifically in the context of digit image generation and representation. This knowledge will be demonstrated through detailed discussions, mathematical formulations, and practical Python implementations.

2 Fundamentals of Variational Auto-Encoder Theory

A Variational Auto-Encoder consists of two main parts: the encoder and the decoder. Both parts are typically implemented as neural networks. The encoder compresses the input data into a smaller, encoded representation in a latent space, and the decoder attempts to reconstruct the input data from this compressed representation.

2.1 The Encoder

- **Input Layer**: The input layer of the encoder receives the data. For the image data from MNIST, the input layer has as many units as there are pixels in the image (e.g., 784 for MNIST's 28x28 images).
- **Hidden Layers**: One or more hidden layers transform the input into a more abstract representation. These layers can be fully connected layers, convolutional layers (for image data), or recurrent layers (for sequence data).

- **Latent Space Representation**: The final part of the encoder outputs the parameters of the latent space distribution,in this case, the mean μ and log-variance $log(\sigma^2)$ of a Gaussian distribution. These parameters are used to sample and generate the latent variables z through the reparameterization trick.

2.2 The Decoder

- **Input from Latent Space**: The decoder receives the latent variables z sampled from the distribution defined by the encoder's output.
- **Hidden Layers**: Similar to the encoder, the decoder has one or more hidden layers that transform the latent variables back towards the dimensionality of the original input data. These layers mirror the architecture of the encoder but in reverse, to reconstruct the input data.
- **Output Layer**: The output layer of the decoder produces the reconstruction of the original input. For continuous data like images, the output layer often uses a sigmoid activation function to ensure that the outputs are in the appropriate range (e.g., pixel values between 0 and 1).

2.3 Monte-Carlo Estimation

The Monte Carlo Estimation is a computational algorithm that relies on repeated random sampling to obtain numerical results. It is used to approximate complex integrals or summations that are analytically intractable or computationally expensive. The core principle of Monte Carlo methods is to estimate the properties of an underlying distribution by averaging the properties of randomly drawn samples. This method is particularly powerful in high-dimensional spaces where direct computation of integrals becomes unfeasible.

In the context of estimating an expected value of a function $f(x)$ over a probability distribution $p(x)$, the Monte Carlo estimation can be formally expressed as:

$$\mathbb{E}[f(x)] \approx \frac{1}{N} \sum_{i=1}^{N} f(x_i), \quad x_i \sim p(x)$$

where N is the number of samples, and as approaches infinity, the Monte Carlo estimate converges to the true expected value due to the Law of Large Numbers, $\mathbb{E}[f(x)]$ is the expected value of the function f and x_i are samples drawn from the distribution $p(x)$.

2.3.1 Estimation of Reconstruction Loss

In VAEs, the objective function involves a reconstruction loss term that quantifies how well the decoder is able to reconstruct the input data after encoding and decoding processes. Ideally, this involves computing the expectation over the latent variable z which is typically intractable due to the encoder's complex distribution over z. Hence, Monte Carlo methods are employed:

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

where:

- $q_\phi(z|x)$ is the encoder's distribution of the latent variables given the input x.
- $p_\theta(x|z)$ is the likelihood of observing x given the latent variables z, parameterized by θ.
- $z^{(i)}$ are samples drawn from the latent variable distribution.
- N is the number of samples used for the Monte Carlo estimation.

2.4 Kullback-Leibler Divergence

The Kullback-Leibler (KL) divergence is a measure of how one probability distribution diverges from a second, expected probability distribution. In VAEs, it is used to regularize the encoder by measuring the divergence between the encoder's distribution $q_\phi(z|x)$ and the prior distribution of the latent variables $p(z)$, which is often assumed to be a standard Gaussian $\mathcal{N}(0, I)$:

$$\text{KL Divergence} = \mathbb{E}_{q_\phi(z|x)}\left[\log \frac{q_\phi(z|x)}{p(z)}\right] = -\frac{1}{2}\sum_{k=1}^{K}\left(1 + \log((\sigma_k^{(i)})^2) - (\mu_k^{(i)})^2 - (\sigma_k^{(i)})^2\right)$$

where $\mu_k^{(i)}$ and $\sigma_k^{(i)}$ are the mean and standard deviation of the latent variable z for the k-th dimension of the i-th data point, respectively. This formula results from assuming the encoder's output distribution for z as a Gaussian parameterized by means μ and variances σ^2, and the prior as a standard Gaussian.

2.5 Complete Loss Function

The global loss for a Variational Auto-Encoder (VAE) comprises two components: the reconstruction loss and the Kullback-Leibler (KL) divergence, targeting both accurate input reconstruction and latent space regularization to a standard Gaussian prior. The composite loss is:

$$L(\theta, \phi; x) = L_{\text{recon}}(x, \hat{x}) + \beta \cdot L_{\text{KL}}(\mu, \log \sigma^2)$$

where β balances the terms and:

- $L(\theta, \phi; x)$ denotes the VAE's total loss with decoder parameters θ and encoder parameters ϕ.
- L_{recon} represents the reconstruction loss, quantifying the fidelity between input x and reconstruction $\hat{x}$.
- L_{KL} is the KL divergence, enforcing the encoder's latent variable distribution $q_\phi(z|x)$ to align with the prior distribution $p(z)$, where μ and $\log \sigma^2$ are the encoded latent means and log variances, respectively.

3 Implementation on Python

3.1 Choice in Implementation

- **Architecture**: The current VAE implementation uses fully connected layers for both the encoder and the decoder. While this approach is straightforward and effective for understanding the basic mechanics of VAEs, it's possible to enhance the model's performance and efficiency by incorporating convolutional layers in the encoder and deconvolutional (transposed convolution) layers in the decoder. Convolutional layers are particularly beneficial as they better capture spatial hierarchies and features.
- **Alternatives**: As noted, exploring convolutional VAEs could be a future direction. PyTorch provides robust support for such architectures, which can be particularly advantageous for dealing with larger or more complex image datasets.

3.2 Description of Experiments Conducted

- **Baseline Model**: The initial experiments involved training a simple VAE with two latent dimensions. This setup was chosen to facilitate easy visualization of the latent space and to provide a clear understanding of how well the VAE can cluster and separate different digit classes.
- **Latent Space Dimensionality**: Experiments were also conducted by varying the dimensionality of the latent space to observe the impact on the reconstruction quality and the disentanglement of the latent representations.

3.3 Presentation of Results

We applied the Principal Component Analysis (PCA) to reduce the dimensionality of the latent space to two dimensions to visualize how the different classes of digits were distributed in the latent space. The uploaded image represents the visualization produced by the code. It shows a scatter plot where each point represents a latent encoding of an MNIST digit, colored according to its true class label (0 through 9).

- **Latent Space Overlap**: The scatter plot reveals a significant degree of overlap between different digit classes in the latent space. This suggests that while the VAE has learned a compressed representation of the digits, there is not a clear separation between different classes in the latent space. This is expected because VAEs prioritize reconstruction over disentanglement, and the model was not provided with label information during training (unsupervised).
- **Class Clustering**: Some digit classes may form distinct clusters, while others overlap significantly. For instance, in the provided plot, certain colors appear more grouped together, indicating that some digit representations are more distinct than others in the 2D PCA-reduced latent space.
- **Implications**: The degree of overlap has implications for using VAEs in tasks like semi-supervised learning or clustering. It may suggest that further fine-tuning, a more complex model, or a supervised approach might be necessary for tasks that require clear separation of classes.

3.4 Discuss Difficulties

- **Hyperparameter Tuning**: Determining the optimal size of the latent space, the architecture of the neural networks, and the training parameters (like learning rate and batch size) required multiple experimentation and significantly impacted on performance.
- **Balancing Loss Components**: Finding the right balance between the reconstruction loss and the KL divergence is challenging but crucial for good performance. Too much emphasis on one can detrimentally affect the other.

4 Conclusion

In this project, we successfully implemented a Variational Autoencoder to generate and represent digit images, gaining insights into the latent space structure using PCA visualization. The balance between reconstruction accuracy and latent space regularization was achieved by optimizing the combined loss function, consisting of reconstruction loss and KL divergence. The VAE demonstrated the potential for unsupervised learning of complex data distributions, though the observed overlap in the latent space visualization suggests further refinement could enhance class separation.

5 Appendix