Autoencoders

ML Instruction Team, Fall 2022

CE Department
Sharif University of Technology

Mahsa Yazdani
Arian Amani

Let's Think

Let's start with a question!

Any ideas?

How can we combine neural networks and unlabeled data to perform compression ?

The Power Of Unsupervised Learning

- Huge datasets compared to supervised learning (No need for labeling)
- Can find previously unknown patterns in data that are impossible with supervised learning

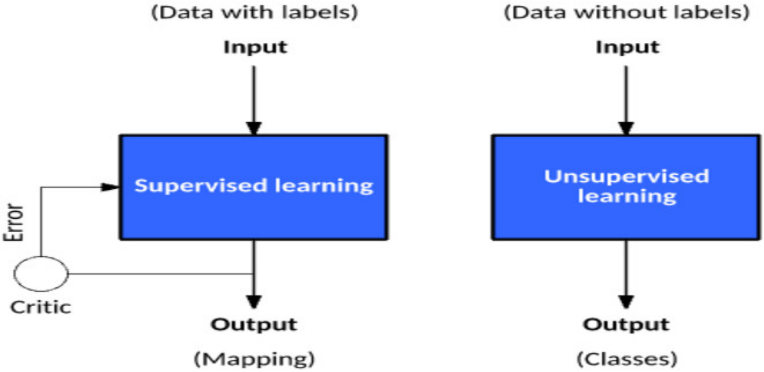
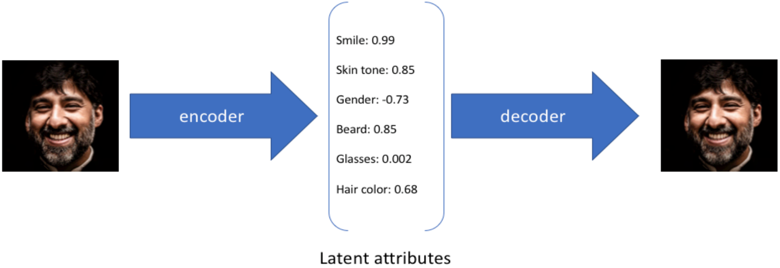


Figure: Types of Machine learning: Deep learning (supervised and unsupervised learning)(Jones [2017]), Source

Applications of Autoencoders

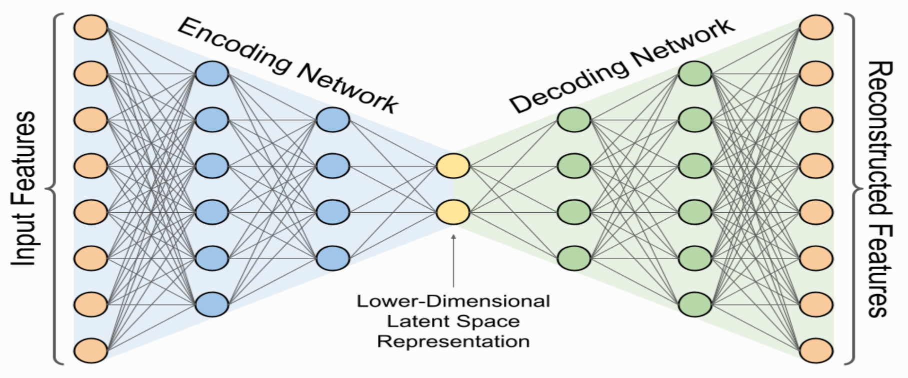
- Compression:
  - AEs can compress our input into a lower dimensional vector and try to reconstruct the original input from that vector



Source

Applications of Autoencoders

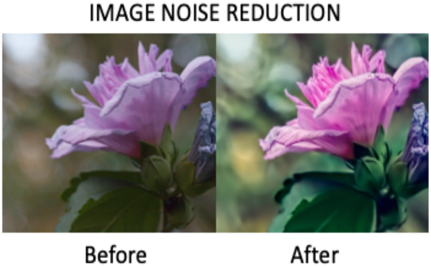
- Dimensionality Reduction:
  - AEs can perform dimensionality reduction



Source

Applications of Autoencoders

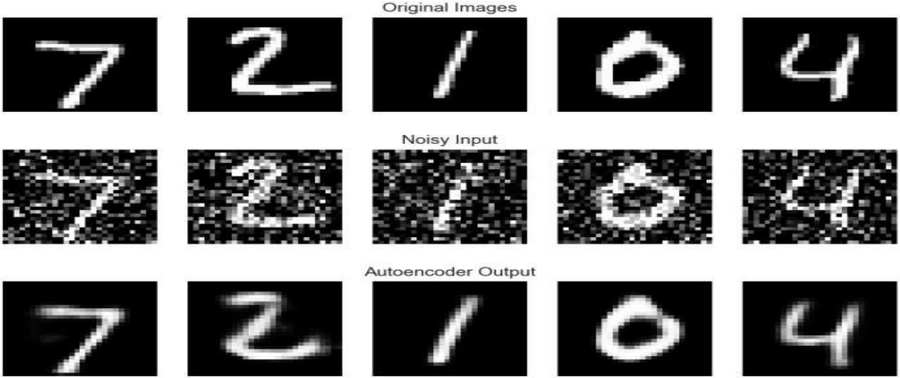
- Image coloring and noise reduction



Source

Applications of Autoencoders

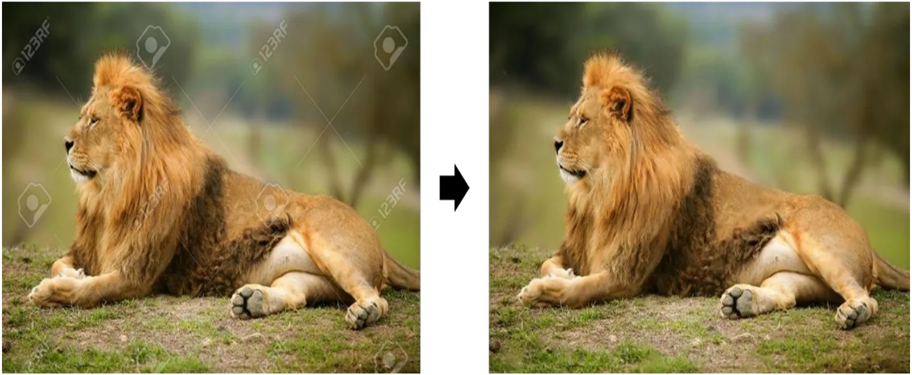
- Noise reduction



Source

Applications of Autoencoders

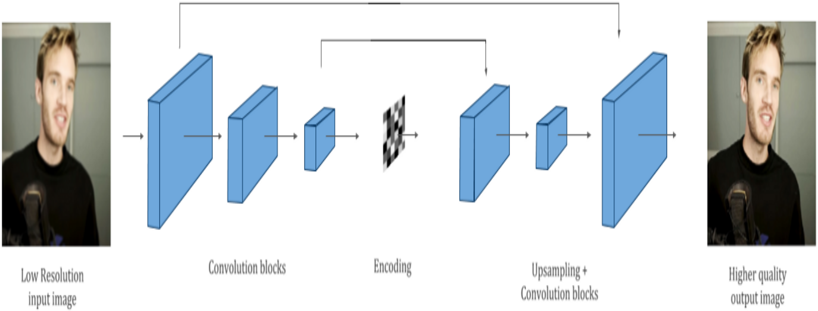
- Watermark removal



Source

Applications of Autoencoders

- Super-Resolution



Source

What is an Autoencoder?

- An autoencoder is a type of artificial neural network, capable of learning a low dimensional representation of the input data (codings), without supervision (unlabeled training data - unsupervised learning)
- Autoencoders take an input X and try to predict X. We use a bottleneck layer with a smaller dimension compared to the input, to use as the coding.

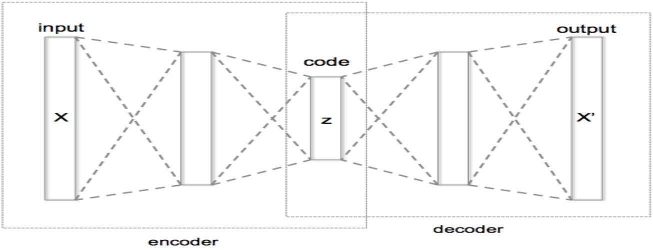


Figure: Schematic structure of an autoencoder with 3 fully connected hidden layers. The code (z - bottleneck) is the most internal layer, Source

Autoencoders: Architecture

- Autoencoders consist of 3 parts:
  1. Encoder : Function $f(x)$ that transforms input x to the latent variable z
  2. Bottleneck : Single layer of neurons that represent the encoding of our input, therefore the most important part of our model
  3. Decoder : Function $h(z)$ that tries to reconstruct input x from encoded latent variable $z \rightarrow h(z) = h(f(x)) = x'$

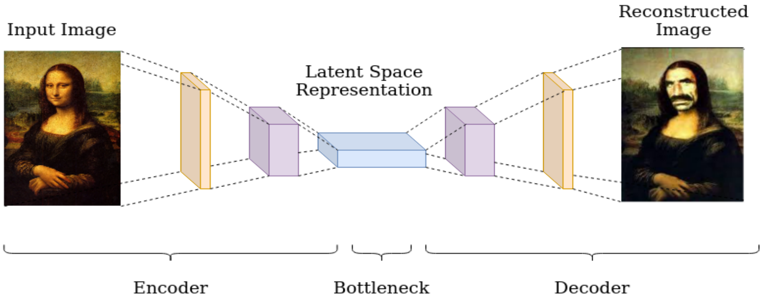


Figure: Autoencoder Architecture (with a little joke :)), Source

Linear vs. Non-Linear

- If your AE uses only linear activations and MSE loss function:
  - It will be performing PCA (Principal Component Analysis)
  - So we need non-linearity to get the most out of autoencoders

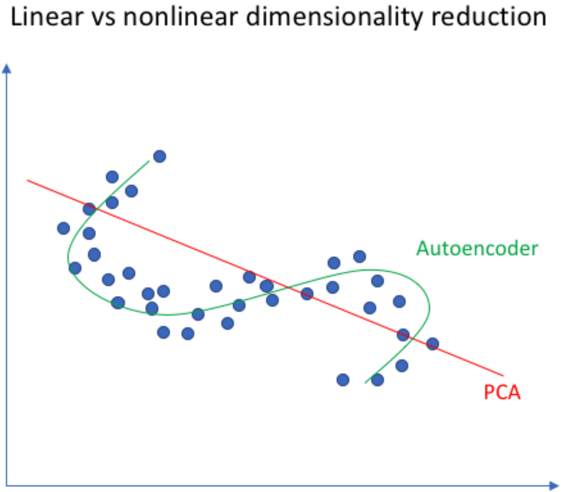
```
encoder = tf.keras.models.Sequential([tf.keras.layers.Dense(2,
    input_shape=input_shape)])

decoder = tf.keras.models.Sequential([tf.keras.layers.Dense(
    input_shape, input_shape=[2])])

pca_autoencoder = tf.keras.models.Sequential([encoder, decoder
    ])

pca_autoencoder.compile(loss="mse", optimizer=tf.keras.
    optimizers.SGD())
```

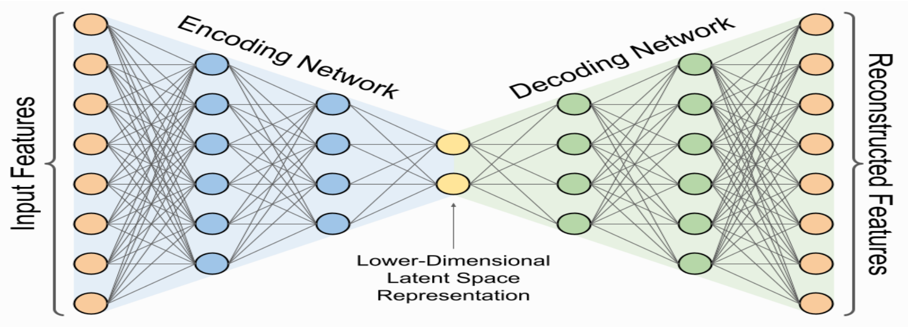
Linear vs. Non-Linear



Source

Stacked Autoencoders

- Autoencoders can have multiple hidden layers to learn more complex encoding/decoding functions - deep autoencoders
- Although, using too deep networks, can cause overfitting
  - Your model will just memorize points in the coding space for each training data, instead of learning a good latent representation of them



Source

Loss and Training

- We're trying to reconstruct our input
- Common loss functions for training autoencoders are:
  - L2: $loss(x, x') = \sum_{i=1}^{m}(x^{(i)} - x'^{(i)})^2$
  - Cross-Entropy

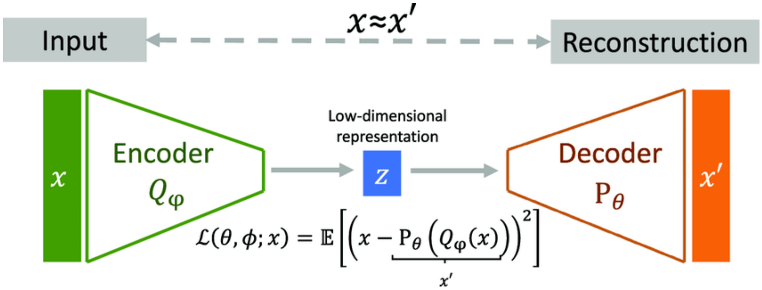


Figure: Schematic of an autoencoder architecture with mean-squared error reconstruction loss.[1]

Pretraining

- You can use an autoencoder for pretraining on supervised problems with few labeled data

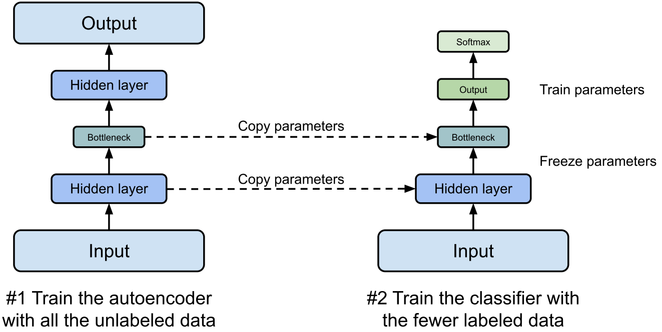
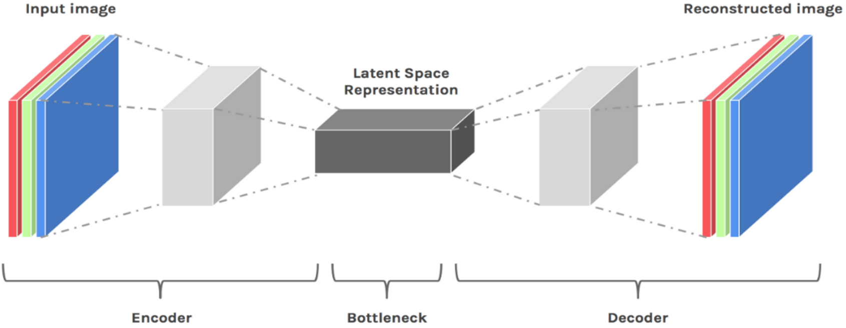


Figure: Using unsupervised learning for pretraining with autoencoders

Autoencoders & Images

- **Are normal autoencoders suitable for working with images?**
  - Encoder: A regular CNN
  - Decoder: A network that uses transpose convolutional layers (or upsampling layers with convolution layers)



Convolutional Encoder-Decoder architecture

Source

Denoising Autoencoders (DAE)

- An AE learns the identity function which brings the risk of overfitting
- A DAE forces the model to learn a more robust latent representation
- It can also be used to efficiently remove noise from noisy images

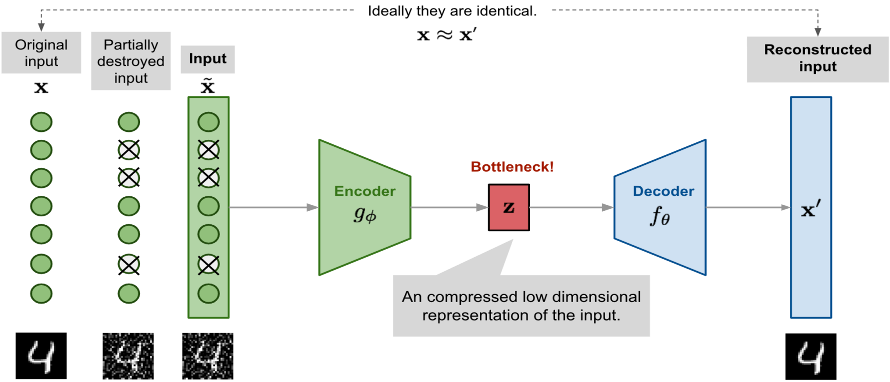
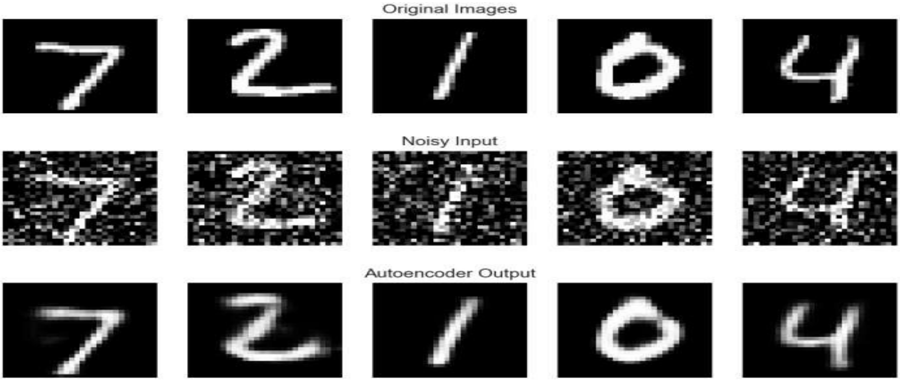


Figure: Illustration of denoising autoencoder model architecture, Source

Denoising Autoencoders

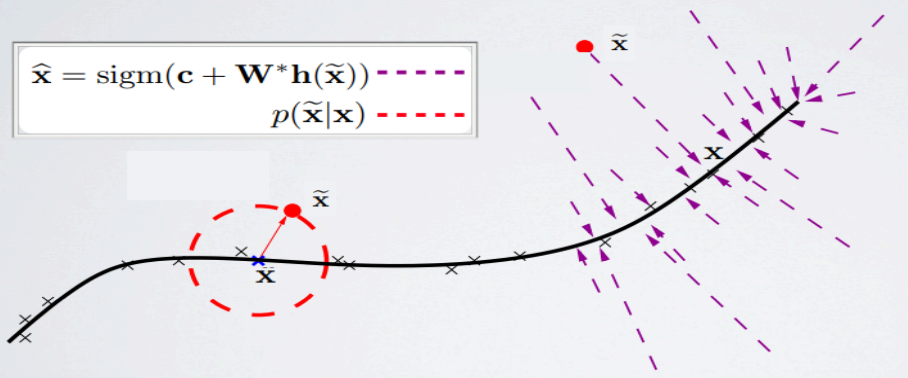
- Results of an autoencoder, predicting what the original data was, without even seeing it



Source

Denoising Autoencoders

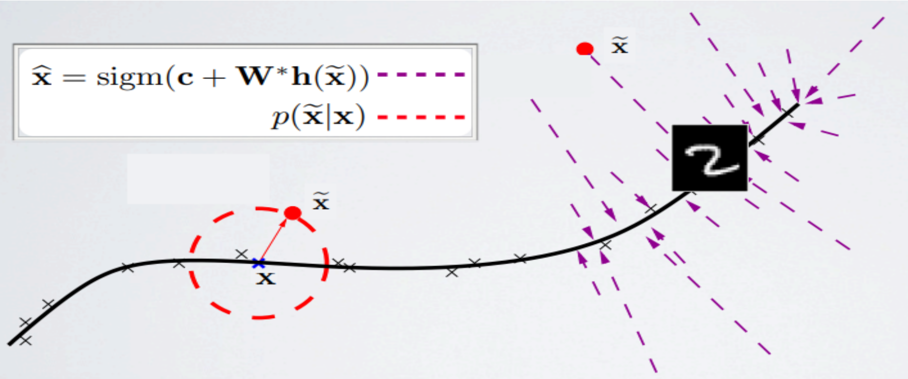
- DAEs won't simply memorize the inputs and outputs (More robustness)
- Intuitively, a DAE learns a projection from a neighborhood of our training data back onto the training data (Refrence)



Source

Denoising Autoencoders

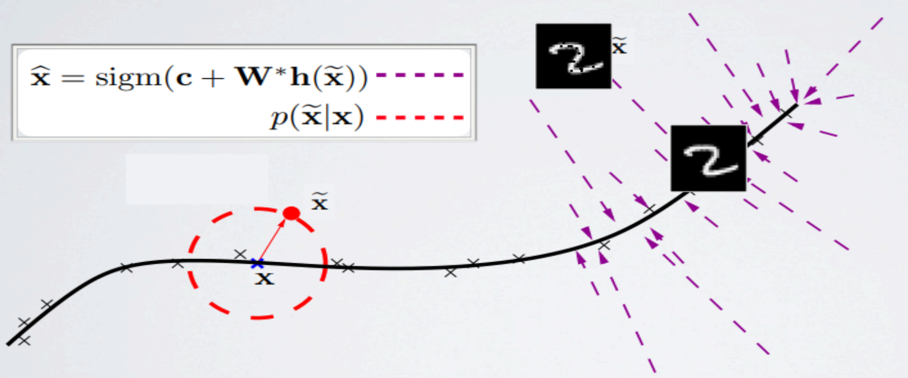
- DAEs won't simply memorize the inputs and outputs (More robustness)
- Intuitively, a DAE learns a projection from a neighborhood of our training data back onto the training data (Refrence)



Source

Denoising Autoencoders

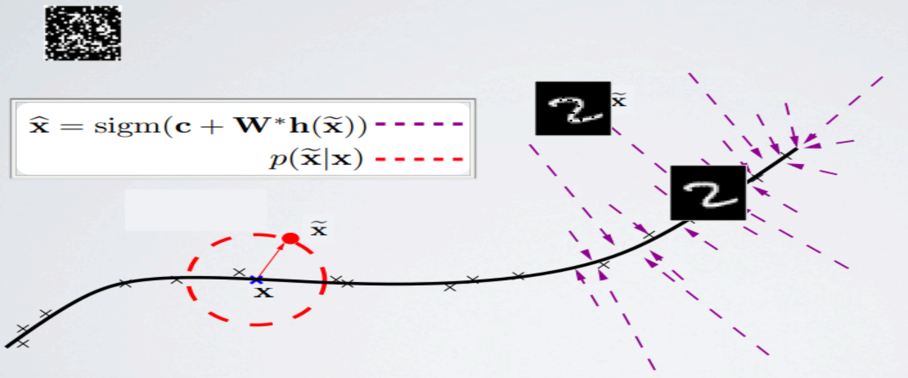
- DAEs won't simply memorize the inputs and outputs (More robustness)
- Intuitively, a DAE learns a projection from a neighborhood of our training data back onto the training data (Refrence)



Source

Denoising Autoencoders

- DAEs won't simply memorize the inputs and outputs (More robustness)
- Intuitively, a DAE learns a projection from a neighborhood of our training data back onto the training data (Refrence)



Source

Autoencoder Generative Models

How can we generate NEW data with Autoencoders??

hint: Autoencoder learns the feature space!

Walking through an example

- We can try to decode a random point from the latent space to generate new data
- The freedom of AEs leads to a severe overfitting which results in some points of the latent space being meaningless once decoded

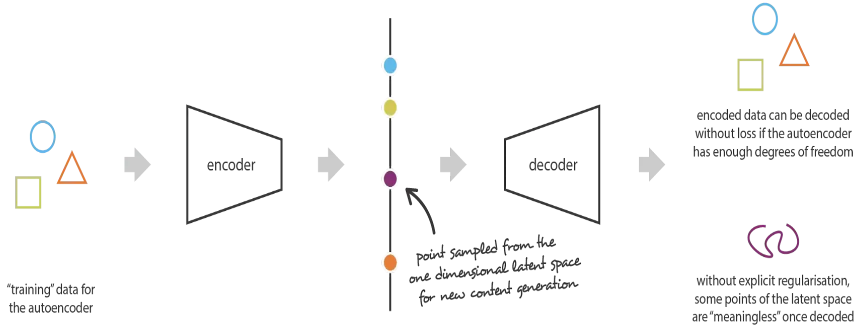


Figure: Irregular latent space prevent us from using autoencoder for new content generation, Source

Walking through an example

- Not all of the points in latent space have meaningful reconstructions

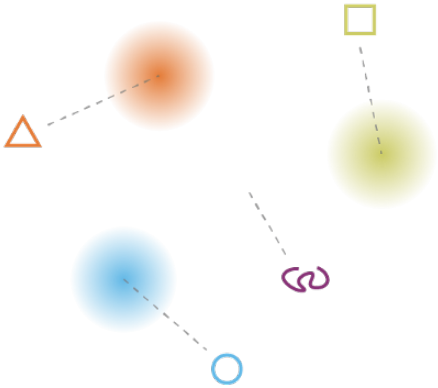


Figure: Latent space and the decoded outputs for some points in it, Source

Walking through an example

- What we want is something like the following picture. So that with sampling from the latent space, we can generate new shapes.

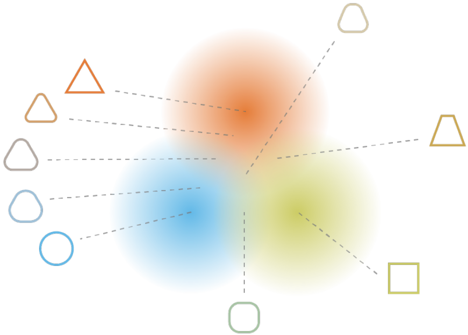
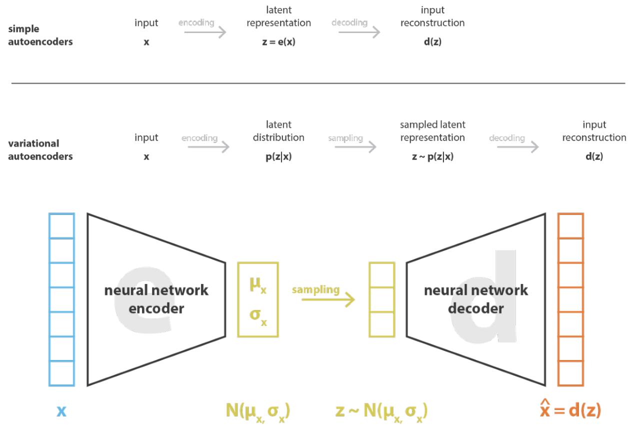


Figure: Regularisation tends to create a “gradient” over the information encoded in the latent space, Source

Variational Autoencoders

Ideas for solving this problem?

Variational Autoencoders



$$\text{loss} = \| x - \hat{x} \|^2 + \text{KL}[\, N(\mu_x, \sigma_x), N(0, I)\,] = \| x - d(z) \|^2 + \text{KL}[\, N(\mu_x, \sigma_x), N(0, I)\,]$$

Image References

- https://lilianweng.github.io/posts/2018-08-12-vae/
- https://emkademy.medium.com/1-first-step-to-generative-deep-learning-with-autoencoders-22bd41e56d18
- Aurelien Geron. 2019. Hands-On Machine Learning with Scikit-Learn, Keras, and TensorFlow: Concepts, Tools, and Techniques to Build Intelligent Systems (2nd. ed.). O'Reilly Media, Inc.
- https://github.com/wojciechmo/vae
- https://ai.googleblog.com/2017/08/making-visible-watermarks-more-effective.html
- https://medium.com/@harishr2301/denoising-autoencoders-996e866e5cd0
- https://subscription.packtpub.com/book/big-data-and-business-intelligence/9781787121089/4/ch04lvl1sec51/setting-up-stacked-autoencoders
- https://www.analyticsvidhya.com/blog/2021/01/auto-encoders-for-computer-vision-an-endless-world-of-possibilities/
- https://towardsdatascience.com/convolutional-autoencoders-for-image-noise-reduction-32fce9fc1763
- https://ift6266h17.files.wordpress.com/2017/03/14_autoencoders.pdf
- https://towardsdatascience.com/understanding-variational-autoencoders-vaes-f70510919f73

References

- Aurelien Geron. 2019. Hands-On Machine Learning with Scikit-Learn, Keras, and TensorFlow: Concepts, Tools, and Techniques to Build Intelligent Systems (2nd. ed.). O'Reilly Media, Inc.
- https://www.cs.toronto.edu/ rgrosse/courses/csc321_2017/slides/lec20.pdf
- https://www.math.purdue.edu/ buzzard/MA598-Spring2019/Lectures/Lec16%20-%20Autoencoders.pptx
- http://cs231n.stanford.edu/slides/2019/cs231n_2019_lecture11.pdf

Refrences

Yasemin Bozkurt, Tristan Bereau, and Joseph Rudzinski.
Interpretable embeddings from molecular simulations using gaussian mixture variational autoencoders.
Machine Learning: Science and Technology, 1, 03 2020.

Thank You!

Any Question?