Deep Learning

17 Autoencoders

Dr. Konda Reddy Mopuri
Dept. of AI, IIT Hyderabad
Jan-May 2023

Beyond Classification and Regression

1. Applications such as image synthesis, image-to-image transformations model high-dim signals

Beyond Classification and Regression

1. Applications such as image synthesis, image-to-image transformations model high-dim signals
2. These applications require to learn the meaningful degrees of freedom that constitute the signal

Beyond Classification and Regression

1. Applications such as image synthesis, image-to-image transformations model high-dim signals
2. These applications require to learn the meaningful degrees of freedom that constitute the signal
3. Typically, these degrees of freedom are of lesser dimensions than the signal

Example: Synthesizing Human faces

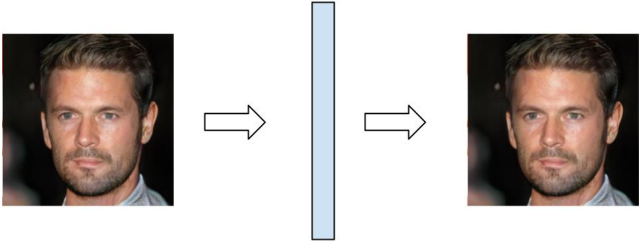
1. For generating new faces, it makes sense to capture a small number of degrees of freedom such as
   - skull size and shape
   - color of skin and eyes
   - features of nose and lips, etc.

Example: Synthesizing Human faces

1. For generating new faces, it makes sense to capture a small number of degrees of freedom such as
   - skull size and shape
   - color of skin and eyes
   - features of nose and lips, etc.
2. Even a comprehensive list of such things will be less than the number of pixels in the image (i.e. resolution)

Example: Synthesizing Human faces

1. If we can model these relatively small number of dimensions, we can synthesize a face with thousands of dimensions



Autoencoder: architecture

1. Feed-forward Neural network that maps a space to itself

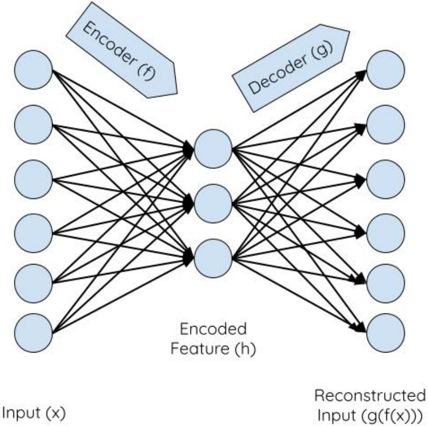
Autoencoder: architecture

1. Feed-forward Neural network that maps a space to itself
2. Trained to map its input to itself (but not an identity function)

Autoencoder: architecture

1. Feed-forward Neural network that maps a space to itself
2. Trained to map its input to itself (but not an identity function)
3. Network consists of two parts: encoder (`f`) and decoder (`g`)

Autoencoder: architecture



Autoencoder: principle

1. Original (input) space is of higher dimensions but the data lies in a manifold of smaller dimension

Autoencoder: principle

1. Original (input) space is of higher dimensions but the data lies in a manifold of smaller dimension
2. Dimension of the latent space is a hyper-parameter chosen from prior knowledge, and/or through heuristics

Autoencoder: principle

1. Original (input) space is of higher dimensions but the data lies in a manifold of smaller dimension
2. Dimension of the latent space is a hyper-parameter chosen from prior knowledge, and/or through heuristics

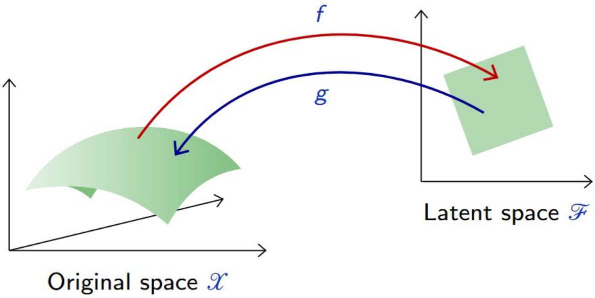


Figure credits: Francois Flueret

Autoencoder: nonlinearity

1. For a binary i/p vector, what could be an appropriate nonlinearity for g?

Autoencoder: nonlinearity

1. For a binary i/p vector, what could be an appropriate nonlinearity for g?
2. For a real i/p vector?

Autoencoder: nonlinearity

1. For a binary i/p vector, what could be an appropriate nonlinearity for g?
2. For a real i/p vector?
3. Nonlinearity for `f`?

Autoencoder: Objective

1. Enforces the reconstructed o/p to be very similar to i/p

Autoencoder: Objective

1. Enforces the reconstructed o/p to be very similar to i/p
2. Loss function takes care of this via training

Autoencoder: Objective

1. Let p be the data distribution in the input space, autoencoder is characterized with the following loss

$$\mathbb{E}_{x\sim p} \left\| x - g \circ f(x) \right\|^2 \approx 0$$

Autoencoder: Objective

1. Let p be the data distribution in the input space, autoencoder is characterized with the following loss

$$\mathbb{E}_{x \sim p} \left\| x - g \circ f(x) \right\|^2 \approx 0$$

2. Training: finding the parameters for the encoder $(f(\cdot; w_f))$ and decoder $(g(\cdot; w_g))$ optimizing the empirical loss

$$\hat{w}_f, \hat{w}_g = \underset{w_f, w_g}{\operatorname{argmin}} \frac{1}{N} \sum_n \left\| x_n - g(f(x_n; w_f); w_g) \right\|^2$$

Autoencoder: Objective

1. For binary i/p, we may interpret the reconstructions as probabilities (with a sigmoid nonlinearity)

Autoencoder: Objective

1. For binary i/p, we may interpret the reconstructions as probabilities (with a sigmoid nonlinearity)
2. Hence, we may use BCE loss for training

Autoencoder: Connection to PCA

1. f and g are linear functions (data is normalized $x_i = \frac{1}{\sqrt{|X|}}(x_i - \mu)$) $\rightarrow$ optimal solution is PCA

Autoencoder: Connection to PCA

1. f and g are linear functions (data is normalized $x_i = \frac{1}{\sqrt{|X|}}(x_i - \mu)$) $\rightarrow$ optimal solution is PCA
2. Better results can be made possible with sophisticated transformations such as deep neural networks $\rightarrow$ Deep Autoencoders

Deep Autoencoders

Top row: original data samples
Bottom row: corresponding reconstructed samples (single ReLU layer of dimension 32)
Figure credits:Keras blog

Autoencoder: Regularization

1. idea is to help generalization

Autoencoder: Regularization

1. idea is to help generalization
2. Over-complete autoencoders (more hidden units than the i/p dimension) $\rightarrow$ parameter explosion and prone to overfitting

Autoencoder: Regularization

1. idea is to help generalization
2. Over-complete autoencoders (more hidden units than the i/p dimension) $\rightarrow$ parameter explosion and prone to overfitting
3. Even the under-complete configurations benefit from regularization

Autoencoder: Regularization

1. idea is to help generalization
2. Over-complete autoencoders (more hidden units than the i/p dimension) $\rightarrow$ parameter explosion and prone to overfitting
3. Even the under-complete configurations benefit from regularization
4. Simplest is to add l_2 regularization term to the objective

Autoencoder: Regularization

1. idea is to help generalization
2. Over-complete autoencoders (more hidden units than the i/p dimension) $\rightarrow$ parameter explosion and prone to overfitting
3. Even the under-complete configurations benefit from regularization
4. Simplest is to add l_2 regularization term to the objective
5. Tie the weights, i.e., $w_g = w_f^T$

Besides dimensionality reduction

1. Autoencoders can capture the dependencies across signal components

Besides dimensionality reduction

1. Autoencoders can capture the dependencies across signal components
2. This can help to restore the missing components from an input

Besides dimensionality reduction

1. In this scenario, we may ignore the encoder/decoder architecture

Besides dimensionality reduction

1. In this scenario, we may ignore the encoder/decoder architecture
2. Goal in this case is not to learn a ϕ such that $\phi(X) \approx X$

Besides dimensionality reduction

1. In this scenario, we may ignore the encoder/decoder architecture
2. Goal in this case is not to learn a ϕ such that $\phi(X) \approx X$
3. It is to learn a ϕ such that $\phi(\tilde{X}) \approx X$, where $\tilde{X}$ is a perturbed version of X

Besides dimensionality reduction

భారతీయ సాంకేతిక విజ్ఞాన సంస్థ హైదరాబాద్
भारतीय प्रौद्योगिकी संस्थान हैदराबाद
Indian Institute of Technology Hyderabad

1. In this scenario, we may ignore the encoder/decoder architecture
2. Goal in this case is not to learn a ϕ such that $\phi(X) \approx X$
3. It is to learn a ϕ such that $\phi(\tilde{X}) \approx X$, where $\tilde{X}$ is a perturbed version of X
4. This is referred to as a **Denoising Autoencoder**

Denoising Autoencoder

1. This can be illustrated with an additive Gaussian noise in case of a 2D signal and MSE

$$\hat{w} = \underset{w}{\operatorname{argmin}} \frac{1}{N} \sum_{n=1}^{N} \|x_n - \phi(x_n + \epsilon_n; w)\|^2 ,$$

where x_n are data samples and ϵ_n are Gaussian random noise

Denoising Autoencoder

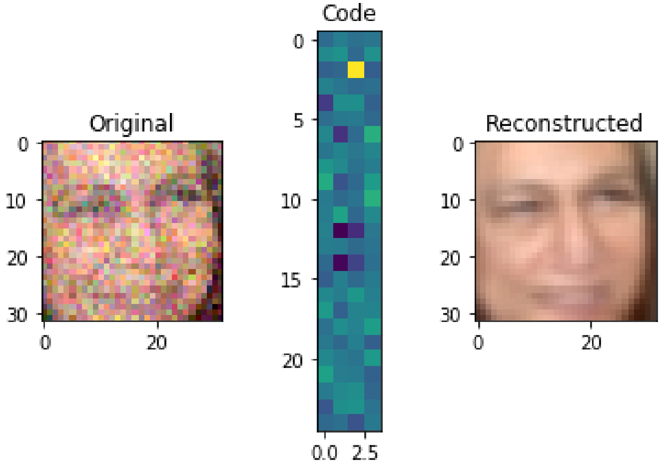


Figure credits: Ali Abdelal, https://stackabuse.com/

Sparse Autoencoders

1. Tries to enforce the hidden neurons to be inactive mostly

Sparse Autoencoders

1. Tries to enforce the hidden neurons to be inactive mostly
2. Restricts the freedom of the parameters by forcing them to fire sparsely

Sparse Autoencoders

1. Tries to enforce the hidden neurons to be inactive mostly
2. Restricts the freedom of the parameters by forcing them to fire sparsely
3. Uses a sparsity parameter (ρ) (typically close to 0, say 0.01)
4. Enforces the mean neuron activation $(\hat{\rho}_l)$ to be close to ρ

Sparse Autoencoders

1. Mean activation: $\hat{\rho}_l = \frac{1}{m} \sum_{i=1}^{m} f(x_i)_l$

Sparse Autoencoders

1. Mean activation: $\hat{\rho}_l = \frac{1}{m}\sum_{i=1}^{m} f(x_i)_l$
2. $R(w) = \sum_{l=1}^{k} \rho \log \frac{\rho}{\hat{\rho}_l} + (1-\rho)\log \frac{1-\rho}{1-\hat{\rho}_l}$

Sparse Autoencoders

1. Mean activation: $\hat{\rho}_l = \frac{1}{m}\sum_{i=1}^{m} f(x_i)_l$
2. $R(w) = \sum_{l=1}^{k} \rho \log \frac{\rho}{\hat{\rho}_l} + (1-\rho)\log\frac{1-\rho}{1-\hat{\rho}_l}$
3. k - dimension of hidden layer
   m -size of training dataset

Contractive Autoencoders

1. Prevents an autoencoder from learning an identity function

Contractive Autoencoders

1. Prevents an autoencoder from learning an identity function
2. $R(w) = |\frac{\partial f}{\partial x}|_F$

Contractive Autoencoders

1. Prevents an autoencoder from learning an identity function
2. $R(w) = |\frac{\partial f}{\partial x}|_F$
3. Competition (in the latent/hidden layer) b/w 'being sensitive' and 'not sensitive' to the i/p variations

Contractive Autoencoders

1. Prevents an autoencoder from learning an identity function
2. $R(w) = |\frac{\partial f}{\partial x}|_F$
3. Competition (in the latent/hidden layer) b/w 'being sensitive' and 'not sensitive' to the i/p variations
4. Ends up capturing only the important variations in the i/p (something like PCA)

Latent Representations

1. Consider two samples in the latent space and reconstruct the samples along the line joining these

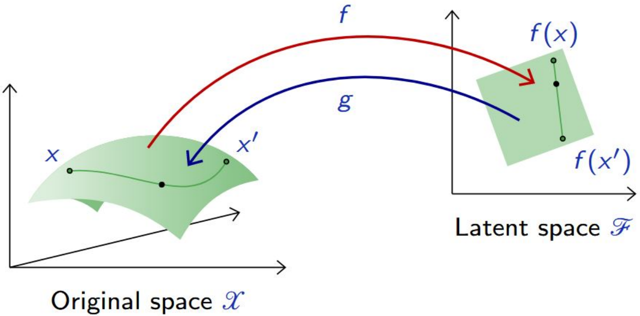


Figure credits: Francois Fleuret

Latent Representations

1. Consider two samples in the latent space and reconstruct the samples along the line joining these
2. $g(\alpha x + (1-\alpha)x')$

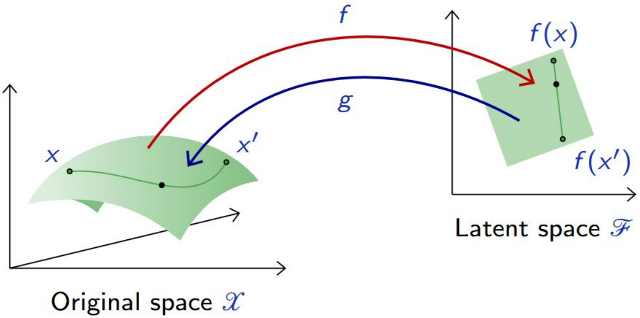
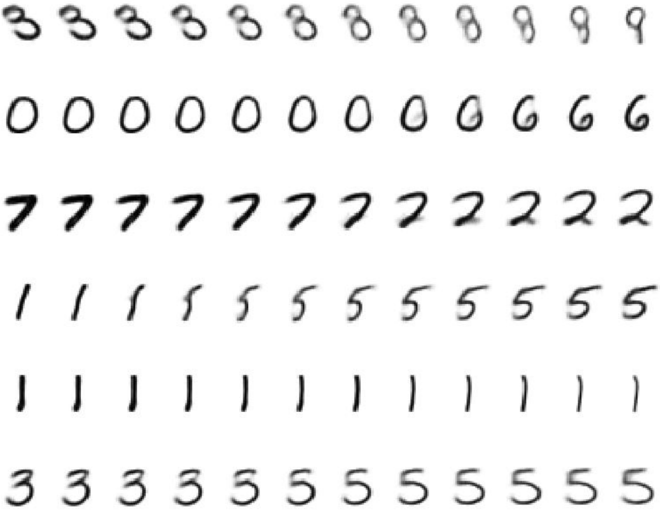


Figure credits: Francois Fleuret

Latent Representations



Generative Modeling by Autoencoder

1. Introduce a density model over the latent space

Generative Modeling by Autoencoder

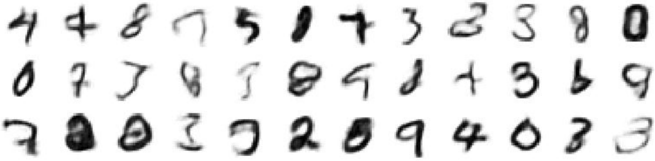
1. Introduce a density model over the latent space
2. Sample there and reconstruct using the decoder g

Generative Modeling by Autoencoder

1. Introduce a density model over the latent space
2. Sample there and reconstruct using the decoder g
3. For instance, use a Gaussian density for modeling the latent space from the training data (estimate mean and a diagonal covariance matrix)

Generative Modeling by Autoencoder

Autoencoder sampling ($d = 8$)



Autoencoder sampling ($d = 16$)

Figure credits: Francois Fleuret

Generative Modeling by Autoencoder

1. Reconstructions are not convincing

Generative Modeling by Autoencoder

1. Reconstructions are not convincing
2. Because the density model is too simple

Generative Modeling by Autoencoder

1. Reconstructions are not convincing
2. Because the density model is too simple
3. Good model still needs to capture the empirical distribution on the data although in a lower dimensional space