Rapid Introduction to Machine Learning/ Deep Learning

Hyeong In Choi

Seoul National University

Lecture 7a
Autoencoder

December 11, 2015

Table of contents

1. Objectives of Lecture 7a
2. Simple autoencoder
   - 2.1. Encoder and decoder
   - 2.2. Training
   - 2.3. Linear autoencoder and PCA
3. Deep autoencoder
   - 3.1. Layerwise training of deep autoencoder (unsupervised learning)
   - 3.2. Classifier (supervised learning)
   - 3.3. Different version of autoencoder
   - 3.4. Why does autoencoder work so well?
   - 3.5. Comparison between deep autoencoder and PCA
4. Denoising autoencoder
   - 4.1. Undercomplete vs. Overcomplete
   - 4.2. Denoising autoencoder
   - 4.3. A glance at manifold learning

1. Objectives of Lecture 7a

Objective 1

Learn the basics of autoencoder, especially deep autoencoder as a stack of single autoencoders

Objective 2

Learn without proof that linear autoencoder is the same as PCA

Objective 3

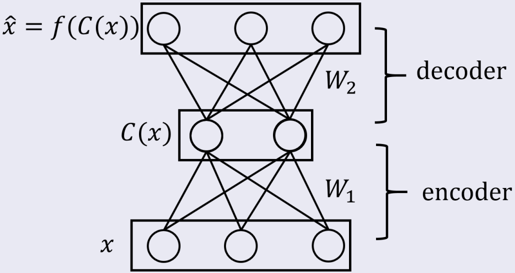
Learn the denoising autoencoder

Objective 4

Understand the idea of manifold learning

2. Simple autoencoder

2.1. Encoder and decoder



- The bottom layer and the top layer has the same number of neurons

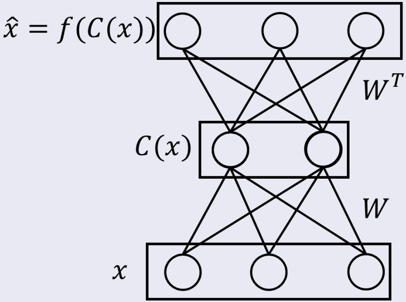
- The middle layer may have

  less neurons: undercomplete autoencoder
  more neurons: overcomplete autoencoder
- W_1 and W_2 are usually (but not always) tied, i.e. $W_2 = W_1^T$

2.2. Training

- connection matrix

$$W_1 = W, W_2 = W^T \quad \text{(tied weight)}$$



- encoding

$$h = \sigma(x)$$

typically, $h = \mathrm{sigm}(Wx + b)$

- decoding

$$\hat{x} = \sigma(h)$$

typically, $\hat{x} = \mathrm{sigm}(W^T h + c)$

- Data $\mathcal{D} = \{x(t)\}_{t=1}^{T}$
- Error
    - Real data

$$\begin{aligned} E &= \frac{1}{2}\sum_t \|\hat{x}(t) - x(t)\|_2^2 \\ &= \frac{1}{2}\sum_t \sum_i |\hat{x}_i(t) - x_i(t)|^2 \end{aligned}$$

- Binary data

$$E = -\sum_t \left\{ x(t) \log \hat{x} + (1 - x(t)) \log(1 - \hat{x}(t)) \right\}$$

- Training algorithm

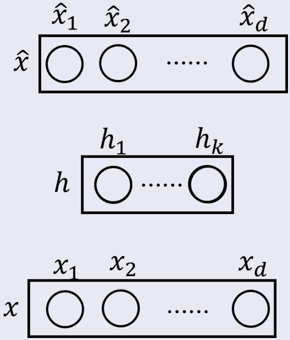
  Use the standard back propagation algorithm of the feedforward network

- Pretraining

  May use RBM pretrainig

2.3. Linear autoencoder and PCA

Data



- If $\sigma(t) = t$,

$$\begin{aligned} h &= h(x) = Wx + b \\ \hat{x} &= W^T h + c = W^T(Wx + b) + c \end{aligned}$$

- In this case,

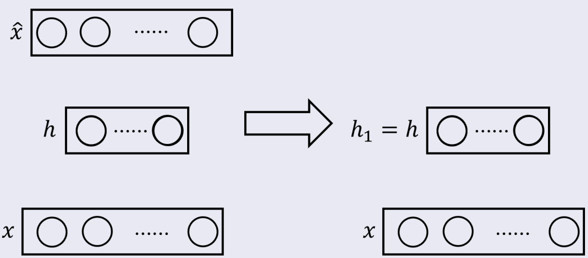
  Going from x to h: Projection to the k highest eigenspace
  Going from h to $\hat{x}$: Truncated x (saving only k highest eigenspace)

3. Deep autoencoder

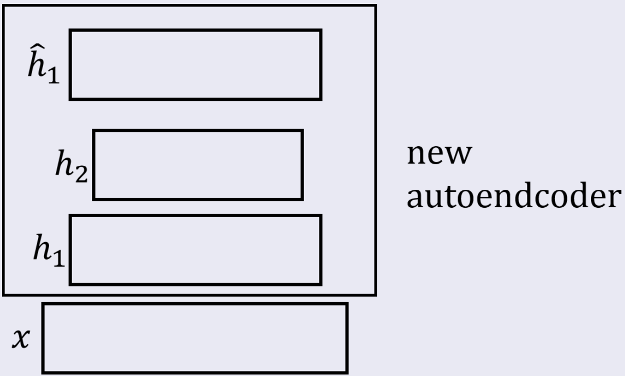
3.1. Layerwise training of deep autoencoder (unsupervised learning)

Stacking autoencoders

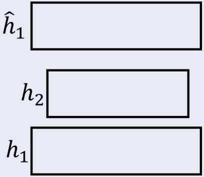
- Remove top layer



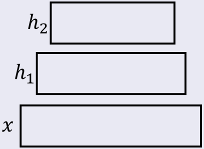
- Use h as a new input and add another autoencoder



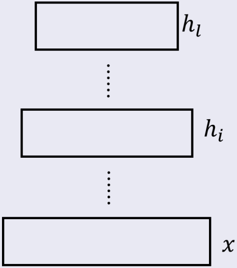
- Train this new autoencoder



- Remove the top layer, the resulting one is

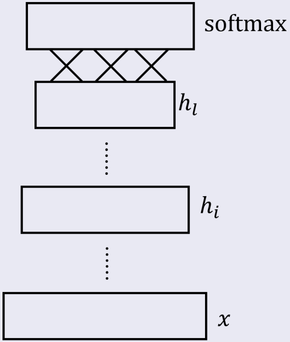


- Keep adding new layers to come up with a multilaye (deep) autoencoder



3.2. Classifier (supervised learning)

- Put a softmax layer on top
- Do the supervised training using the back propagation algorithm



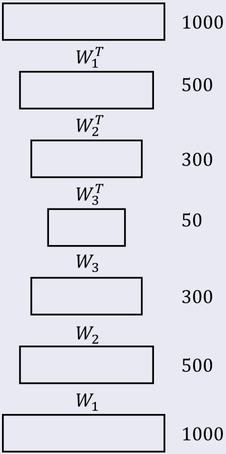
3.3. Different version of autoencoder

Architecture and training

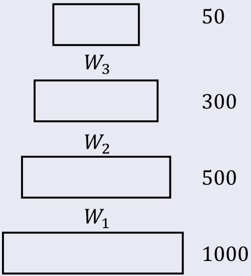
- There are different ways of stacking and training

Example

- Stack them all like this (Example)

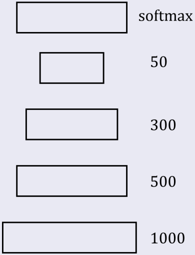


- Each layer is pre-trained (e.g. by RBM)
- Then the whole layer is trained using the back propagation
- But this training process is still an unsupervised training (i.e. don't need data labels)
- Remove the top layers



Classification

- For classification, as before, add softmax layer



- Do the supervised training as above

3.4. Why does autoencoder work so well?

(1) It is a simplest deep neural network, and is easy to understand and train

(2) The unsupervised learning process (autoencoder part) provides good initial weights for supervised learning's back propagation algorithm

(3) The network is trained in such a way that final autoencoder output (the one that is to be fed to the softmax layer) retrains the characteristics of the training data set, while the input that is quite different from training data set produces a nonsensical (incoherent) output in the final autoencoder layer

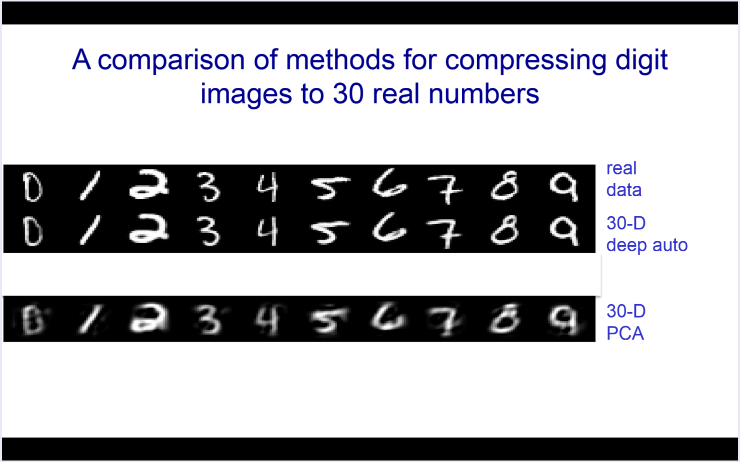
3.5. Comparison between deep autoencoder and PCA

The first really successful deep autoencoders
(Hinton & Salakhutdinov, Science, 2006)

$$784 \xrightarrow{W_1} 1000 \xrightarrow{W_2} 500 \xrightarrow{W_3} 250 \xrightarrow{W_4} \text{30 linear units}$$

$$784 \xleftarrow{W_1^T} 1000 \xleftarrow{W_2^T} 500 \xleftarrow{W_3^T} 250 \xleftarrow{W_4^T} \text{30 linear units}$$

We train a stack of 4 RBM's and then "unroll" them.
Then we fine-tune with gentle backprop.



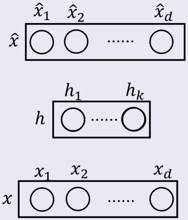
Source: Hinton's Coursera lecture 15

4. Denoising autoencoder

4.1. Undercomplete vs. Overcomplete

Undercomplete autoencoder

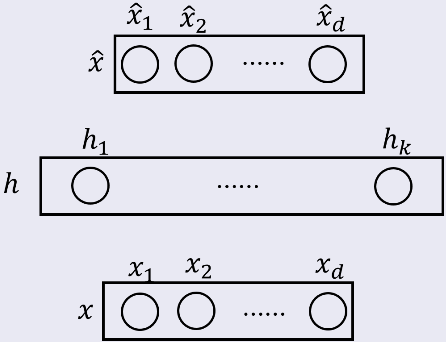
- If $|h| < |x|$, i.e. $k < d$, this autoencoder is called undercomplete



- Undercomplete autoencoder produces a "judiciously compressed" h from the input x

Overcomplete autoencoder

- If $|h| > |x|$, i.e. $k > d$, this autoencoder is called overcomplete

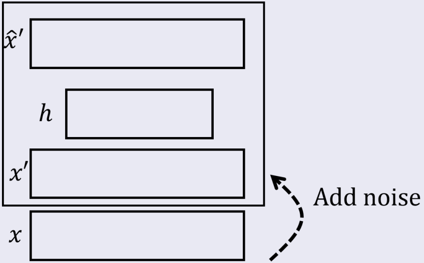


- Normally, the overcomplete autoencoder are not used because x can be copied to a part of h for faithful recreation of $\hat{x}$
- It is, however, used quite often together with the following denoising autoencoder

4.2. Denoising autoencoder

Procedure for denoisiong autoencoder

- Add noise to the input and still try to reproduce faithful output of the autoencoder



- Add noise to x to produce a new input x'
- x' as an input to produce $\hat{x}'$
- Error is calculated as the discrepancy between x and $\hat{x}'$, i.e.

$$
\begin{aligned}
\text{real: } E &= \frac{1}{2}\sum_t \|\hat{x}'(t) - x(t)\|_2^2 \\
&= \frac{1}{2}\sum_t \sum_i |\hat{x}_i(t) - x_i(t)|^2 \\
\text{Binary: } E &= -\sum_t \left\{ x(t)\log \hat{x}' + (1 - x(t))\log(1 - \hat{x}'(t)) \right\}
\end{aligned}
$$

Merits

(1) Denoising deep autoencoder is used as a sort of most popular unsupervised pretraining method

(2) Denoising autoencoder prevents neuron from unduly colluding with each other, i.e. it forces each neuron or a small group of neurons to do its best in reconstructing the input

4.3. A glance at manifold learning