

- A denoising encoder simply corrupts the input data using a probabilistic process ($P(\tilde{x}_{ij}|x_{ij})$) before feeding it to the network
- A simple $P(\tilde{x}_{ij}|x_{ij})$ used in practice is the following

$$P(\tilde{x}_{ij} = 0|x_{ij}) = q$$

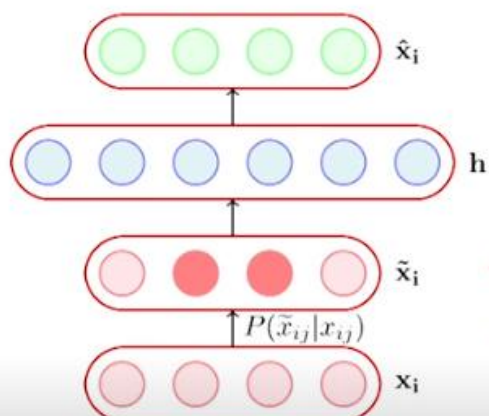
$$P(\tilde{x}_{ij} = x_{ij}|x_{ij}) = 1 - q$$

- In other words, with probability q the input is flipped to 0 and with probability $(1 - q)$ it is retained as it is

3:02 / 26:17

Mitesh M. Khapra

CS7015 (Deep Learning) - Lecture 7



For example, it will have to learn to reconstruct a corrupted x_{ij} correctly by relying on its interactions with other elements of \mathbf{x}_i .

- How does this help ?
- This helps because the objective is still to reconstruct the original (uncorrupted) \mathbf{x}_i

$$\arg \min_{\theta} \frac{1}{m} \sum_{i=1}^m \sum_{j=1}^n (\hat{x}_{ij} - x_{ij})^2$$

- It no longer makes sense for the model to copy the corrupted $\tilde{\mathbf{x}}_i$ into $h(\tilde{\mathbf{x}}_i)$ and then into $\hat{\mathbf{x}}_i$ (the objective function will not be minimized by doing so)
- Instead the model will now have to capture the characteristics of the data correctly.

6:12 / 26:17

Mitesh M. Khapra

CS7015 (Deep Learning) - Lecture 7

We will now see a practical application in which AEs are used and then compare Denoising Autoencoders with regular autoencoders

Task: Hand-written digit recognition

3	1	8	5	5	1	1	8	9	5
8	4	1	5	9	5	6	2	3	1
6	7	3	9	8	5	0	7	1	0
8	0	1	1	4	4	4	2	7	5
4	9	7	7	8	0	4	1	0	0

Figure: MNIST Data

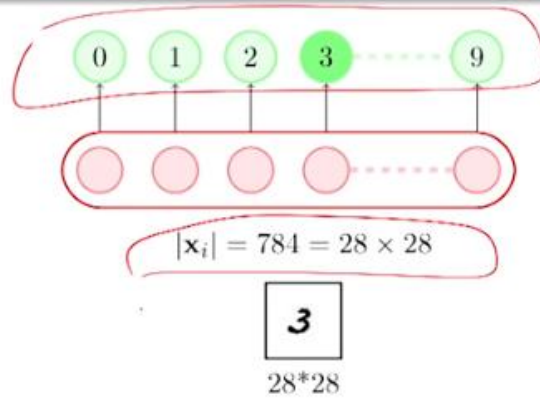


Figure: Basic approach (we use raw data as input features)

Task: Hand-written digit recognition

3	1	8	5	5	1	1	8	9	5
8	4	1	5	9	5	6	2	3	1
6	7	3	9	8	5	0	7	1	0
8	0	1	1	4	4	4	2	7	5
4	9	7	7	8	0	4	1	0	0

Figure: MNIST Data

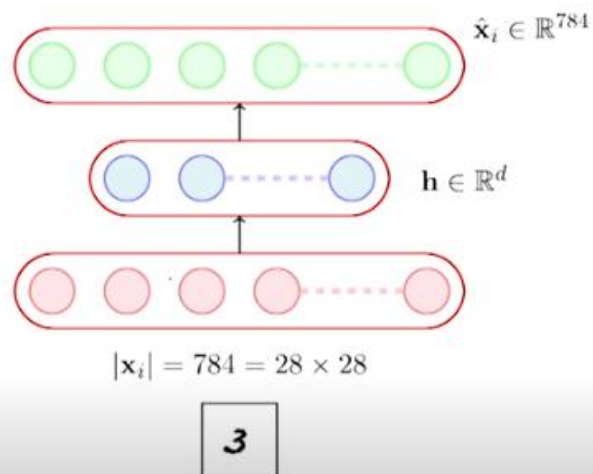


Figure: AE approach (first learn important characteristics of data)

Task: Hand-written digit recognition

3 1 8 5 5 1 1 8 9 5
8 4 1 5 9 5 6 2 3 1
6 7 3 9 8 5 0 7 1 0
8 0 1 1 4 4 4 2 7 5
4 9 7 7 8 0 4 1 0 0

Figure: MNIST Data

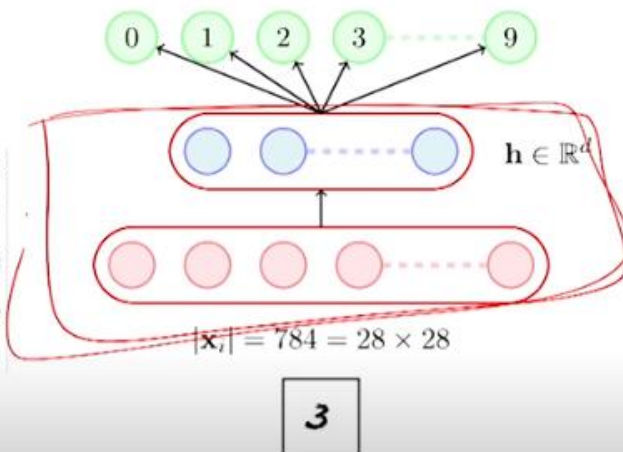


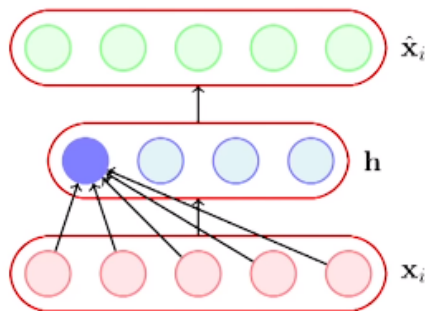
Figure: AE approach (and then train a classifier on top of this hidden representation)

We will now see a way of visualizing AEs and use this visualization to compare different AEs

12:15 / 26:17

Mitesh M. Khapra

CS7015 (Deep Learning) : Lecture 7



$$\begin{aligned} \max_{\mathbf{x}_i} \quad & \{w_1^T \mathbf{x}_i\} \\ \text{s.t.} \quad & \|\mathbf{x}_i\|^2 = \mathbf{x}_i^T \mathbf{x}_i = 1 \end{aligned}$$

- We can think of each neuron as a filter which will fire (or get maximally) activated for a certain input configuration \mathbf{x}_i
- For example,

$$\mathbf{h}_1 = \sigma(W_1^T \mathbf{x}_i) \text{ [ignoring bias } b]$$

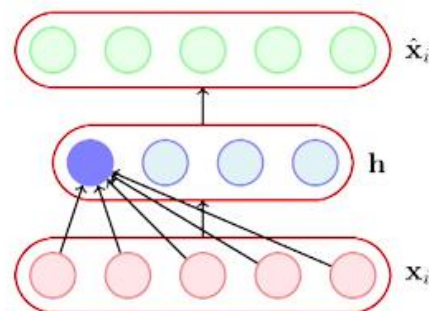
Where W_1 is the trained vector of weights connecting the input to the first hidden neuron

- What values of \mathbf{x}_i will cause \mathbf{h}_1 to be maximum (or maximally activated)
- Suppose we assume that our inputs are normalized so that $\|\mathbf{x}_i\| = 1$

14:18 / 26:17

Mitesh M. Khurana

CS7015 (Deep Learning) - Lecture 7



$$\begin{aligned} \max_{\mathbf{x}_i} \quad & \{w_1^T \mathbf{x}_i\} \\ \text{s.t.} \quad & \|\mathbf{x}_i\|^2 = \mathbf{x}_i^T \mathbf{x}_i = 1 \end{aligned}$$

Solution: $\mathbf{x}_i = \frac{w_1}{\sqrt{w_1^T w_1}}$

- Thus the inputs

$$\mathbf{x}_i = \frac{W_1}{\sqrt{W_1^T W_1}}, \frac{W_2}{\sqrt{W_2^T W_2}}, \dots, \frac{W_n}{\sqrt{W_n^T W_n}}$$

will respectively cause hidden neurons 1 to n to maximally fire

- Let us plot these images (\mathbf{x}_i 's) which maximally activate the first k neurons of the hidden representations learned by a vanilla autoencoder and different denoising autoencoders
- These \mathbf{x}_i 's are computed by the above formula using the weights ($W_1, W_2 \dots W_k$) learned by the respective autoencoders

16:34 / 26:17

Mitesh M. Khurana

CS7015 (Deep Learning) - Lecture 7



7/55

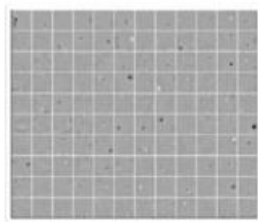


Figure: Vanilla AE
(No noise) ✓

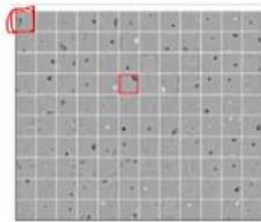


Figure: 25% Denoising
AE (q=0.25) ✓

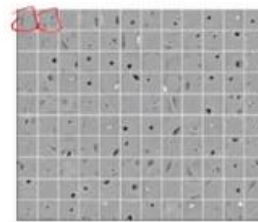


Figure: 50% Denoising
AE (q=0.5) ✓

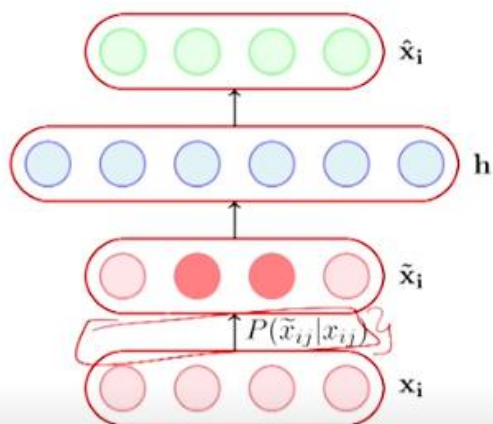
- The vanilla AE does not learn many meaningful patterns



20:24 / 26:17

Mitesh M. Khapra

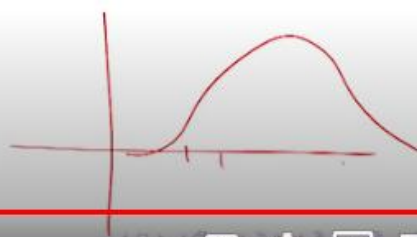
CS7015 (Deep Learning) - Lecture 7



- We saw one form of $P(\tilde{x}_{ij}|x_{ij})$ which flips a fraction q of the inputs to zero
- Another way of corrupting the inputs is to add a Gaussian noise to the input

$$\tilde{x}_{ij} = x_{ij} + \mathcal{N}(0, 1)$$

- We will now use such a denoising AE on a different dataset and see their performance



24:30 / 26:17

CC

Settings

Fullscreen

Help

More