NPTEL ONLINE CERTIFICATION COURSES

Course Name: Deep Learning

Faculty Name: Prof. P. K. Biswas

Department : E & ECE, IIT Kharagpur

Topic

Lecture 26: Back propagation Learning – Examples II

NPTEL

CONCEPTS COVERED

Concepts Covered:

- Back Propagation Learning in MLP
- Different Loss Functions
- Back Propagation Learning - Example
- Back Propagation – Node Level

NPTEL

Back Propagation Learning an Example

NPTEL

Back Propagation Learning:- Output Layer

$$E = \frac{1}{2}\sum_{j=1}^{2}\left(x_j^2 - t_j\right)^2 \qquad x_j^2 = \frac{1}{1+e^{-\theta_j^2}} \qquad \theta_j^2 = \sum_{i=0}^{2} W_{ij}^2 x_i^1$$

$$\frac{\partial E}{\partial W_{ij}^2} = \frac{\partial E}{\partial x_j^2}\cdot\frac{\partial x_j^2}{\partial \theta_j^2}\cdot\frac{\partial \theta_j^2}{\partial W_{ij}^2} = (x_j^2 - t_j)x_j^2(1-x_j^2)x_i^1$$

We set $\boxed{\delta_j^2 = x_j^2(1-x_j^2)(x_j^2 - t_j)} \Rightarrow \frac{\partial E}{\partial W_{ij}^2} = \delta_j^2 x_i^1$

$$\boxed{W_{ij}^2 \leftarrow W_{ij}^2 - \eta\frac{\partial E}{\partial W_{ij}^2}}$$

NPTEL

Feed Forward Pass

$$W^1 \qquad x_i^0 \qquad \theta_j^1 = \sum W_{ij}^1 x_i^0 \qquad x_j^1 = \frac{1}{1+e^{-\theta_j^1}}$$

$$\begin{bmatrix} 0.5 & 1.5 & 0.8 \\ 0.8 & 0.2 & -1.6 \end{bmatrix} \begin{bmatrix} 1 \\ 0.7 \\ 1.2 \end{bmatrix} = \begin{bmatrix} 2.51 \\ -9.8 \end{bmatrix} \Rightarrow \begin{bmatrix} 0.92 \\ 0.27 \end{bmatrix}$$

$$W^2 \qquad x_i^1 \qquad \theta_j^2 = \sum_i W_{ij}^2 x_i^1 \qquad x_j^2 = \frac{1}{1+e^{-\theta_j^2}}$$

$$\begin{bmatrix} 0.9 & -1.7 & 1.6 \\ 1.2 & 2.1 & -1.-0.26 \end{bmatrix} \begin{bmatrix} 1 \\ 0.92 \\ 0.27 \end{bmatrix} = \begin{bmatrix} -0.232 \\ 3.057 \end{bmatrix} \Rightarrow \begin{bmatrix} 0.44 \\ 0.95 \end{bmatrix}$$

NPTEL

Back Propagation Learning:- Output Layer

$$\delta_j^2 = x_j^2(1-x_j^2)(x_j^2 - t_j)$$

$$\delta_1^2 = x_1^2(1-x_1^2)(x_1^2 - t_1)$$
$$= 0.44*(1-0.44)*(0.44-1)$$
$$= -0.138$$

$$\Rightarrow \frac{\partial E}{\partial W_{11}^2} = \delta_1^2 x_1^1 = -0.126$$

$$\delta_2^2 = x_2^2(1-x_2^2)(x_2^2 - t_2)$$
$$= 0.95*(1-0.95)*(0.95-0)$$
$$= 0.045$$

$$\Rightarrow \frac{\partial E}{\partial W_{12}^2} = \delta_2^2 x_1^1 = 0.04$$

$$W_{11}^2 \leftarrow W_{11}^2 + \eta * 0.126$$

$$W_{12}^2 \leftarrow W_{12}^2 - \eta * 0.04$$

NPTEL

Back Propagation Learning:- Output Layer

$$\frac{\partial E}{\partial W_{21}^{2}} = \delta_1^2 x_2^1 = -0.037 \qquad \frac{\partial E}{\partial W_{22}^{2}} = \delta_2^2 x_2^1 = 0.012$$

$$\frac{\partial E}{\partial W_{01}^{2}} = \delta_1^2 x_0^1 = -1.38 \qquad \frac{\partial E}{\partial W_{02}^{2}} = \delta_2^2 x_0^1 = 0.045$$

NPTEL

Back Propagation Learning:- Hidden Layer

We set $\boxed{\delta_i^k = O_i^k(1-O_i^k)\sum_{j=1}^{M_{k+1}} \partial_j^{k+1} W_{ij}^{k+1}}$ $\Rightarrow \delta_i^1 = x_i^1(1-x_i^1)\sum_{j=1}^{2} \partial_j^2 W_{ij}^2 \Rightarrow \boxed{\frac{\partial E}{\partial W_{ij}^k} = \delta_i^k x_i^{k-1}}$

$$\delta_1^1 = x_1^1(1-x_1^1)\left[\delta_1^2 * W_{11}^2 + \delta_2^2 W_{12}^2\right]$$
$$= 0.92*(1-0.92)[(-0.137)*(-1.7)+0.045*2.1]$$
$$= 0.024$$

$$\delta_2^1 = x_2^1(1-x_2^1)\left[\delta_1^2 * W_{21}^2 + \delta_2^2 W_{22}^2\right]$$
$$= 0.27*(1-0.27)[(-0.137)*0.8+0.045*(-0.2)]$$
$$= -0.02$$

NPTEL

Back Propagation Learning:- Hidden Layer

$$\frac{\partial E}{\partial W_{11}^1} = \delta_1^1 * x_1^0 = 0.024 * 0.7 = 0.017$$

$$\frac{\partial E}{\partial W_{21}^1} = \delta_2^1 * x_1^0 = -0.02 * 0.7 = -0.014$$

$$\frac{\partial E}{\partial W_{21}^1} = \delta_1^1 * x_2^0 = 0.024 * 1.2 = 0.0288$$

$$\frac{\partial E}{\partial W_{22}^1} = \delta_2^1 * x_2^0 = -0.02 * 1.2 = -0.024$$

$$\frac{\partial E}{\partial W_{01}^1} = \delta_1^1 * x_0^0 = 0.024 * 1 = 0.024$$

$$\frac{\partial E}{\partial W_{02}^1} = \delta_2^1 * x_0^0 = -0.02 * 1 = -0.02$$

$$W_{ij}^1 \leftarrow W_{ij}^1 - \eta \frac{\partial E}{\partial W_{ij}^1}$$

NPTEL

NPTEL ONLINE CERTIFICATION COURSES

NPTEL

Thank you

NPTEL ONLINE CERTIFICATION COURSES

Course Name: Deep Learning

Faculty Name: Prof. P. K. Biswas

Department : E & ECE, IIT Kharagpur

Topic

Lecture 27: Back propagation Learning – Examples

NPTEL

CONCEPTS COVERED

Concepts Covered:

- Back Propagation Learning in MLP
- Back Propagation Learning – Network Level
- Back Propagation – Node Level

NPTEL

Backpropagation at Network Level

NPTEL

Back Propagation Learning at Node Level

NPTEL

Example: Node Architecture

NPTEL

Example: Forward Pass

NPTEL

Example: Backpropagation

NPTEL

Back propagation: Pseudo Code

```
# Set Input
  w=5; x=3; y=2; z=4

# Forward Pass
  v = x*y
  u = w+v
  f = u+z

# Backward Pass
  dfdu = z
  dfdz = u
  dfdw = 1*dfdu   # dudw = 1
  dfdv  = 1*dfdu   # dudv = 1
  dfdx  = y*dfdv   # dvdx = y
  dfdy  = x*dfdv   # dvdy = x
```

NPTEL

Back propagation: Pseudo Code

```
# Set Input
  w=5; x=3; y=2; z=4

# Forward Pass
  v = x*y
  u = w+v
  f = u+z

# Backward Pass
  dfdu = z
  dfdz = u
  dfdw = 1*dfdu   # dudw = 1
  dfdv  = 1*dfdu   # dudv = 1
  dfdx  = y*dfdv   # dvdx = y
  dfdy  = x*dfdv   # dvdy = x
```

NPTEL

Example: Calculate Gradients

NPTEL

Understanding Gradient Backward

$$\frac{\partial L}{\partial f} = g \quad \frac{\partial f}{\partial x} = y; \ \frac{\partial f}{\partial y} = x; \quad \frac{\partial L}{\partial x} = \frac{\partial L}{\partial f}\cdot\frac{\partial f}{\partial x} = gy; \quad \frac{\partial L}{\partial y} = \frac{\partial L}{\partial f}\cdot\frac{\partial f}{\partial y} = gx$$

$$\frac{\partial L}{\partial f} = g \quad \frac{\partial f}{\partial x} = 1; \ \frac{\partial f}{\partial y} = 1; \quad \frac{\partial L}{\partial x} = \frac{\partial L}{\partial f}\cdot\frac{\partial f}{\partial x} = g; \quad \frac{\partial L}{\partial y} = \frac{\partial L}{\partial f}\cdot\frac{\partial f}{\partial y} =$$

$$\frac{\partial L}{\partial f} = g$$

$$\boldsymbol{Case - I: x > y; f = max(x, y) = x}$$

$$\frac{\partial f}{\partial x} = 1; \ \frac{\partial f}{\partial y} = 0; \ \frac{\partial L}{\partial x} = \frac{\partial L}{\partial f}\cdot\frac{\partial f}{\partial x} = g \quad \frac{\partial L}{\partial y} = \frac{\partial L}{\partial f}\cdot\frac{\partial f}{\partial y} = 0$$

$$\boldsymbol{Case - II: x < y; f = max(x, y) = y}$$

$$\frac{\partial f}{\partial x} = 0; \ \frac{\partial f}{\partial y} = 1; \ \frac{\partial L}{\partial x} = \frac{\partial L}{\partial f}\cdot\frac{\partial f}{\partial x} = 0; \quad \frac{\partial L}{\partial y} = \frac{\partial L}{\partial f}\cdot\frac{\partial f}{\partial y} = g$$

NPTEL

Previous example: different approach

$$f = x +$$

$$\frac{\partial L}{\partial x} = g \qquad \frac{\partial L}{\partial y} = g$$

$$\frac{\partial L}{\partial f} =$$

$$f = x$$

$$\frac{\partial L}{\partial x} = gy \qquad \frac{\partial L}{\partial y} = gx$$

NPTEL

Another Example

$$f = x +$$

$$\frac{\partial L}{\partial x} = g \qquad \frac{\partial L}{\partial y} = g$$

$$\frac{\partial L}{\partial f} =$$

$$f = x$$

$$\frac{\partial L}{\partial x} = gy \qquad \frac{\partial L}{\partial y} = gx$$

$$f = \max(x, y)$$

$$\frac{\partial L}{\partial x} = g \text{ if } x > y \qquad \frac{\partial L}{\partial y} = g \text{ if } y > x$$

$$= 0 \text{ otherwise} \qquad = 0 \text{ otherwise}$$

NPTEL

NPTEL ONLINE CERTIFICATION COURSES

NPTEL

Thank you

NPTEL ONLINE CERTIFICATION COURSES

Course Name: Deep Learning

Faculty Name: Prof. P. K. Biswas

Department : E & ECE, IIT Kharagpur

Topic

Lecture 28: Autoencoder

NPTEL

CONCEPTS COVERED

Concepts Covered:

- Back Propagation Learning in MLP
- Autoencoder
  - Undercomplete Autoencoder
  - Autoencoder vs. PCA
  - Sparse Autoencoder
  - Denoising Autoencoder
  - Contractive Autoencoder
  - Convolution Autoencoder

NPTEL

Autoencoder

- Unsupervised Learning where Neural Networks are subject to the task of representation learning.
- Impose a bottleneck in the network
- The bottleneck forces a compressed knowledge representation of the input.

NPTEL

Autoencoder

Assumption:

- High degree of correlation/structure exists in the data.
- For uncorrelated data (input features are independent), then compression and subsequent reconstruction would be difficult.

NPTEL

Autoencoder

NPTEL

Autoencoder

NPTEL

Expectation

- Sensitive enough to input for accurate reconstruction
- Insensitive enough that it does not memorize or overfit the training data

$$\text{Loss Function} \Rightarrow L(X, \hat{X}) + \text{Regularizer}$$

NPTEL

Undercomplete Autoencoder

$$L(X, \hat{X}) = \frac{1}{2}\sum_{N}\left\|X - \hat{X}\right\|^2$$

NPTEL

Stacked Autoencoder

$$L(X, \hat{X}) = \frac{1}{2}\sum_{N}\left\|X - \hat{X}\right\|^2$$

NPTEL

Autoencoder

NPTEL

NPTEL ONLINE CERTIFICATION COURSES

NPTEL

Thank you

NPTEL ONLINE CERTIFICATION COURSES

Course Name: Deep Learning

Faculty Name: Prof. P. K. Biswas

Department : E & ECE, IIT Kharagpur

Topic

Lecture 29: Autoencoder vs. PCA

NPTEL

CONCEPTS COVERED

Concepts Covered:

- Autoencoder
  - Undercomplete Autoencoder
  - Autoencoder vs. PCA
  - Deep Autoencoder Training
  - Sparse Autoencoder
  - Denoising Autoencoder
  - Contractive Autoencoder
  - Convolution Autoencoder

NPTEL

Autoencoder

- Unsupervised Learning.
- Representation learning.
- Impose a bottleneck in the network.
- The bottleneck forces a compressed knowledge representation of the input.

NPTEL

Autoencoder

Assumption:

- High degree of correlation/structure exists in the data.
- For uncorrelated data (input features are independent), then compression and subsequent reconstruction would be difficult.

NPTEL

Autoencoder

NPTEL

Undercomplete Autoencoder

$$L(X, \hat{X}) = \frac{1}{2}\sum_{N}\left\|X - \hat{X}\right\|^2$$

NPTEL

Autoencoder

NPTEL

Deep Autoencoder

$$L(X, \hat{X}) = \frac{1}{2}\sum_{N}\left\|X - \hat{X}\right\|^2$$

NPTEL

Autoencoder
vs.
PCA

NPTEL

What is PCA?

NPTEL

NPTEL ONLINE CERTIFICATION COURSES

NPTEL

Thank you

NPTEL ONLINE CERTIFICATION COURSES

Course Name: Deep Learning

Faculty Name: Prof. P. K. Biswas

Department : E & ECE, IIT Kharagpur

Topic

Lecture 30: Autoencoder vs. PCA

NPTEL

CONCEPTS COVERED

Concepts Covered:

- Autoencoder
  - Undercomplete Autoencoder
  - Autoencoder vs. PCA
  - Deep Autoencoder Training
  - Sparse Autoencoder
  - Denoising Autoencoder
  - Contractive Autoencoder
  - Convolution Autoencoder

NPTEL

Autoencoder
vs.
PCA

NPTEL

What is PCA?

NPTEL

PCA

$$X = \left\{ \begin{bmatrix} 3 \\ 4 \end{bmatrix}, \begin{bmatrix} 4 \\ 3 \end{bmatrix}, \begin{bmatrix} 4 \\ 4 \end{bmatrix}, \begin{bmatrix} 4 \\ 5 \end{bmatrix}, \begin{bmatrix} 5 \\ 4 \end{bmatrix}, \begin{bmatrix} 5 \\ 5 \end{bmatrix}, \begin{bmatrix} 5 \\ 6 \end{bmatrix}, \begin{bmatrix} 6 \\ 5 \end{bmatrix} \right\}$$

$$\mu_X = \begin{bmatrix} 4.5 \\ 4.5 \end{bmatrix}$$

NPTEL

PCA

$$X = \left\{ \begin{bmatrix} 3 \\ 4 \end{bmatrix}, \begin{bmatrix} 4 \\ 3 \end{bmatrix}, \begin{bmatrix} 4 \\ 4 \end{bmatrix}, \begin{bmatrix} 4 \\ 5 \end{bmatrix}, \begin{bmatrix} 5 \\ 4 \end{bmatrix}, \begin{bmatrix} 5 \\ 5 \end{bmatrix}, \begin{bmatrix} 5 \\ 6 \end{bmatrix}, \begin{bmatrix} 6 \\ 5 \end{bmatrix} \right\} \qquad \mu_X = \begin{bmatrix} 4.5 \\ 4.5 \end{bmatrix}$$

$$(X_1 - \mu_X)(X_1 - \mu_X)^t = \begin{bmatrix} -1.5 \\ -0.5 \end{bmatrix} \begin{bmatrix} -1.5 & -0.5 \end{bmatrix} = \begin{bmatrix} 2.25 & 0.75 \\ 0.75 & 0.25 \end{bmatrix}$$

$$(X_2 - \mu_X)(X_2 - \mu_X)^t = \begin{bmatrix} 0.25 & 0.75 \\ 0.75 & 2.25 \end{bmatrix}$$

NPTEL

PCA

$$C_X = \begin{bmatrix} 0.75 & 0.375 \\ 0.375 & 0.75 \end{bmatrix}$$

$$\begin{vmatrix} 0.75-\lambda & 0.375 \\ 0.375 & 0.75-\lambda \end{vmatrix} = 0 \Rightarrow \lambda_1 = 1.125 \;\&\; \lambda_2 = 0.375$$

$$\lambda_1 \Rightarrow e_1 = \frac{1}{\sqrt{2}}\begin{bmatrix} 1 \\ 1 \end{bmatrix} \qquad \lambda_2 \Rightarrow e_2 = \frac{1}{\sqrt{2}}\begin{bmatrix} 1 \\ -1 \end{bmatrix}$$

NPTEL

PCA

PCA

ORIGINAL

1

5

25

NPTEL

What does PCA do?

NPTEL

Deep Autoencoder

$$L(X, \hat{X}) = \frac{1}{2}\sum_{N}\left\|X - \hat{X}\right\|^2$$

NPTEL

Autoencoder vs. PCA

NPTEL

Experimental Setup for Dimensionality Reduction

- Dataset used: MNIST (a large database of handwritten digits)
  - Total train Images: $60{,}000$
  - Total test Images: $10{,}000$
  - Image dimension: $28 \times 28\ (784)$
  - Dimensionality reduction: $784 \rightarrow 2$
  - Reconstruction: $784 \rightarrow 30$
- Optimizer used: Adam (Learning rate-10^{-4})
- Loss Function: Mean Squared Error
- Trained for 100 iterations

NPTEL

Source: G. E. Hilton and R. R. Salakhutdinov: "Reducing the Dimensionality of Data with Neural Networks", Science, Vol 313, July 2006, pp. 504-507.

MNIST Data set: Example

By Josef Steppan - Own work, CC BY-SA 4.0,
https://commons.wikimedia.org/w/index.php?curid=64810040

Autoencoder converges to PCA

PCA

2-Layer Autoencoder

$$784 \rightarrow 2 \rightarrow 784$$

NPTEL

Source: G. E. Hilton and R. R. Salakhutdinov: “Reducing the Dimensionality of Data with Neural Networks”, Science, Vol 313, July 2006, pp. 504-507.

Deep vs. Shallow autoencoder

784
↓
2 (Sigmoid)
↓
784

2-Layered AE with
Sigmoid Non-Linearity

Deep AE without any
Non-Linear Activations

784
↓
1000
↓
500
↓
250
↓
2
↓
250
↓
500
↓
1000
↓
784

NPTEL

Source: G. E. Hilton and R. R. Salakhutdinov: “Reducing the Dimensionality of Data with Neural Networks”, Science, Vol 313, July 2006, pp. 504-507.

Deep Autoencoder with Non-Linear Activations

Principal component analysis (PCA)

Deep Autoencoder with Non-Linear Activations

$$
\begin{array}{c}
784\ (ReLU) \\ \downarrow \\
1000\ (ReLU) \\ \downarrow \\
500\ (ReLU) \\ \downarrow \\
250\ (ReLU) \\ \downarrow \\
\boldsymbol{2\ (Sigmoid)} \\ \downarrow \\
250\ (ReLU) \\ \downarrow \\
500\ (ReLU) \\ \downarrow \\
1000\ (ReLU) \\ \downarrow \\
784
\end{array}
$$

Source: G. E. Hilton and R. R. Salakhutdinov: “Reducing the Dimensionality of Data with Neural Networks”, Science, Vol 313, July 2006, pp. 504-507.

Autoencoder for Dimensionality Reduction

Articles from Reuter corpus were mapped to a 2 dimensional vector

PCA

Autoencoder

NPTEL

Source: G. E. Hilton and R. R. Salakhutdinov: “Reducing the Dimensionality of Data with Neural Networks”, Science, Vol 313, July 2006, pp. 504-507.

Reconstruction from Latent Space

Original

Autoencoder

PCA

784 $(ReLU)$
↓
1000 $(ReLU)$
↓
500 $(ReLU)$
↓
250 $(ReLU)$
↓
30 $(ReLU)$
↓
250 $(ReLU)$
↓
500 $(ReLU)$
↓
1000 $(ReLU)$
↓
784

Source: G. E. Hilton and R. R. Salakhutdinov: “Reducing the Dimensionality of Data with Neural Networks”, Science, Vol 313, July 2006, pp. 504-507.

Reconstruction from Latent Space

(a)

(b)

(c)

(a) Original (b) 30-D AE (c) 30-D PC

Source: G. E. Hilton and R. R. Salakhutdinov: “Reducing the Dimensionality of Data with Neural Networks”, Science, Vol 313, July 2006, pp. 504-507.

NPTEL

NPTEL ONLINE CERTIFICATION COURSES

NPTEL

Thank you

NPTEL ONLINE CERTIFICATION COURSES

Course Name: Deep Learning

Faculty Name: Prof. P. K. Biswas

Department : E & ECE, IIT Kharagpur

Topic

Lecture 31: Autoencoder Training

NPTEL

CONCEPTS COVERED

Concepts Covered:

- Autoencoder
  - Undercomplete Autoencoder
  - Autoencoder vs. PCA
  - Deep Autoencoder Training
  - Sparse Autoencoder
  - Denoising Autoencoder
  - Contractive Autoencoder
  - Convolution Autoencoder

NPTEL

Autoencoder Training

NPTEL

Layer by Layer Pretraining

NPTEL

Autoencoder Training

NPTEL

Autoencoder Training

NPTEL

Autoencoder Training

NPTEL

Autoencoder Training

NPTEL

Autoencoder Training

NPTEL

Autoencoder Training

NPTEL

Autoencoder Training

NPTEL

Autoencoder Training

NPTEL

Autoencoder Training

NPTEL

Autoencoder Training

NPTEL

Applications

NPTEL

NPTEL ONLINE CERTIFICATION COURSES

NPTEL

Thank you

NPTEL ONLINE CERTIFICATION COURSES

Course Name: Deep Learning

Faculty Name: Prof. P. K. Biswas

Department : E & ECE, IIT Kharagpur

Topic

Lecture 32: Autoencoder Variants

NPTEL

CONCEPTS COVERED

Concepts Covered:

- Autoencoder
  - Undercomplete Autoencoder
  - Autoencoder vs. PCA
  - Deep Autoencoder Training
  - Sparse Autoencoder
  - Denoising Autoencoder
  - Contractive Autoencoder
  - Convolution Autoencoder

NPTEL

Sparse Autoencoder

NPTEL

Sparse Autoencoder

- Interesting features can be learnt even when number of nodes in the hidden layer is large.
- Introduce sparsity constraint on the hidden layer nodes that penalize activations within a layer.
- Network learns encoding-decoding that relies on activating a small number of neurons.

NPTEL

Regularizing Activations not the Weights

Sparsity Constraint

$a_j^h \rightarrow$ Activation of jth Neuron in hidden layer h

$a_j^h \rightarrow 1 \Rightarrow$ Neuron is active

Average activation $\rightarrow \hat{\rho}_j = \frac{1}{m}\sum_{i=1}^{m} a_j^h(x_i)$

Constraint $\rightarrow \hat{\rho}_j = \rho$

$\rho \rightarrow$ sparsity parameter (typically a small value)

NPTEL

Sparsity Constraint

$$\text{Regularizer}: \quad \sum_{j=1}^{N_h}\left[\rho \log\frac{\rho}{\hat{\rho}_j} + (1-\rho)\log\frac{1-\rho}{1-\hat{\rho}_j}\right] \quad \Rightarrow \quad \sum_{j=1}^{N_h} KL(\rho \,\|\, \hat{\rho}_j)$$

$$J_{sparse}(W) = L(X, \hat{X}) + \lambda \sum_{j} KL(\rho \,\|\, \hat{\rho}_j)$$

NPTEL

KL Divergence

NPTEL

https://www.jeremyjordan.me/autoencoders/

Sparsity Constraint

$$\delta_i^k = O_i^k(1-O_i^k)\sum_{j=1}^{M_{k+1}} \partial_j^{k+1} W_{ij}^{k+1}$$

$$\delta_i^k = O_i^k(1-O_i^k)\left[\left(\sum_{j=1}^{M_{k+1}} \partial_j^{k+1} W_{ij}^{k+1}\right) + \lambda\left(-\frac{\rho}{\hat{\rho}_i} + \frac{1-\rho}{1-\hat{\rho}_i}\right)\right]$$

NPTEL

Denoising Autoencoder

NPTEL

Denoising Autoencoder

- The Autoencoder learns a generalizable encoding-decoding scheme.
- An approach:- while training use corrupt data as input but output as uncorrupted original data.
- The model can not memorize the training data as input and target output is not same any more
- The Model learns a vector field to map the input data towards a low dimensional manifold.

NPTEL

NPTEL ONLINE CERTIFICATION COURSES

NPTEL

Thank you

NPTEL ONLINE CERTIFICATION COURSES

Course Name: Deep Learning

Faculty Name: Prof. P. K. Biswas

Department : E & ECE, IIT Kharagpur

Topic

Lecture 33: Autoencoder Variants

NPTEL

CONCEPTS COVERED

Concepts Covered:

- Autoencoder
  - Undercomplete Autoencoder
  - Autoencoder vs. PCA
  - Deep Autoencoder Training
  - Sparse Autoencoder
  - Denoising Autoencoder
  - Contractive Autoencoder
  - Convolution Autoencoder

NPTEL

Denoising Autoencoder

NPTEL

Denoising Autoencoder

- The Autoencoder learns a generalizable encoding-decoding scheme.
- An approach:- while training use corrupt data as input but output as uncorrupted original data.
- The model can not memorize the training data as input and target output is not same any more
- The Model learns a vector field to map the input data towards a low dimensional manifold.

NPTEL

Denoising Autoencoder

$$\hat{X} = r(X)$$
$$= g(f(\tilde{X}))$$

NPTEL

What is Manifold?

Manifold Learning

$r(x) - x$ Vector field

NPTEL

Contractive Autoencoder

NPTEL

Contractive Autoencoder

- For similar inputs- learned encoding (compressed domain representation should also be very similar.
- Hidden layer activation variation with input data should be small.

Effectively the Model learns to contract a neighborhood of Inputs to a small neighborhood of Outputs

NPTEL

Regularization

$$\|A\|_F = \sqrt{\sum_{j=1}^{m}\sum_{i=1}^{N_h}|a_{ij}|^2}$$

$$J = \begin{bmatrix} \frac{\partial a_1^h(X)}{\partial x_1} & \frac{\partial a_1^h(X)}{\partial x_2} & .. & \frac{\partial a_1^h(X)}{\partial x_m} \\ \frac{\partial a_2^h(X)}{\partial x_1} & \frac{\partial a_2^h(X)}{\partial x_2} & .. & \frac{\partial a_2^h(X)}{\partial x_m} \\ .. & .. & .. & .. \\ \frac{\partial a_{N_h}^h(X)}{\partial x_1} & \frac{\partial a_{N_h}^h(X)}{\partial x_2} & .. & \frac{\partial a_{N_h}^h(X)}{\partial x_m} \end{bmatrix}$$

$$L(X,\hat{X}) + \lambda\sum_{i=1}^{N_h}\|\nabla_X a_i^h(X)\|^2$$

NPTEL

Applications

NPTEL

NPTEL ONLINE CERTIFICATION COURSES

NPTEL

Thank you

NPTEL ONLINE CERTIFICATION COURSES

Course Name: Deep Learning

Faculty Name: Prof. P. K. Biswas

Department : E & ECE, IIT Kharagpur

Topic

Lecture 34: Convolutional Neural Network

NPTEL

CONCEPTS COVERED

Concepts Covered:

- CNN
  - Convolution
  - Linear Time Invariant (LTI) System
  - Linear Shift Invariant (LSI) System
  - Cross Correlation

NPTEL

LTI/
LSI

NPTEL

Convolution

NPTEL

Convolution

-1	-2	-1
0	0	0
1	2	1

-1	0	1
-2	0	2
-1	0	1

NPTEL

Convolution

Convolution

NPTEL

Convolution

NPTEL

NPTEL ONLINE CERTIFICATION COURSES

NPTEL

Thank you

NPTEL ONLINE CERTIFICATION COURSES

Course Name: Deep Learning

Faculty Name: Prof. P. K. Biswas

Department : E & ECE, IIT Kharagpur

Topic

Lecture 35: Cross Correlation

NPTEL

CONCEPTS COVERED

Concepts Covered:

- CNN
  - Convolution
  - Linear Time Invariant (LTI) System
  - Linear Shift Invariant (LSI) System
  - Cross Correlation

NPTEL

Convolution

NPTEL

Convolution

-1	-2	-1
0	0	0
1	2	1

-1	0	1
-2	0	2
-1	0	1

NPTEL

Convolution

NPTEL

Convolution

NPTEL

Cross Correlation

NPTEL

Cross Correlation

1	1	1	1	1	1
1	20	2	2	2	1
1	2	3	3	2	1
1	2	3	3	2	1
1	2	2	2	2	1
1	1	1	1	1	1

g

3	3	2
3	3	2
2	2	2

f

47	54	56	20	18	12
54	87	94	40	34	21
56	90	107	54	44	24
20	40	54	56	44	24
18	34	43	44	37	22
12	21	24	24	22	15

C_{fg}

NPTEL

Normalized Cross Correlation

$$C_{fg} \Big/ \left[\sum_{u} \sum_{v} g^{2}(x+u, y=v) \right]^{1/2}$$

NPTEL

Cross Correlation

$$\left[\sum_{u}\sum_{v} g^2(x+u, y+v)\right]^{\frac{1}{2}}$$

20.07	20.19	20.30	3.87	3.46	2.64
20.19	20.54	20.80	6.08	5.09	3.46
20.27	20.76	21.26	7.48	6.08	3.87
3.87	5.91	7.48	7.48	6.08	3.87
3.46	5.09	6.08	6.08	5.09	3.46
2.64	3.46	3.87	3.87	3.46	2.64

NPTEL

Cross Correlation

$$C_{fg} \Big/ \left[\sum_u \sum_v g^2(x+u, y=v)\right]^{1/2}$$

2.34	2.67	2.75	5.16	5.2	4.54
2.67	4.2	4.51	6.5	6.67	6.06
2.76	4.33	5.03	7.2	7.23	6.2
5.16	6.76	7.21	7.48	7.23	6.2
5.2	6.67	7.07	7.23	7.26	6.35
4.54	6.06	6.20	6.20	6.35	5.68

1	1	1	1	1	1
1	20	2	2	2	1
1	2	3	3	2	1
1	2	3	3	2	1
1	2	2	2	2	1
1	1	1	1	1	1

NPTEL

NPTEL ONLINE CERTIFICATION COURSES

NPTEL

Thank you

Deep Computer Vision

Alexander Amini

MIT Introduction to Deep Learning

January 7, 2025

“To know what is where by looking.”

MIT 6.869

To discover from images what is present in the world, where things are, what actions are taking place, to predict and anticipate events in the world

The rise and impact of computer vision

Robotics

Mobile computing

Accessibility

Biology & Medicine

Autonomous driving

Impact: Facial Detection & Recognition

Impact: Self-Driving Cars

Amini+ *ICRA* 2019.

Impact: Medicine, Biology, Healthcare

Breast cancer

COVID-19

Skin cancer

MIT 6.S191

Impact: Accessibility

What Computers "See"

MIT 6.S191

Images are Numbers

MIT 6.S191

Images are Numbers

157	153	174	168	150	152	129	151	172	161	155	156
155	182	163	74	75	62	33	17	110	210	180	154
180	180	50	14	34	6	10	33	48	106	159	181
206	109	5	124	131	111	120	204	166	15	56	180
194	68	137	251	237	239	239	228	227	87	71	201
172	105	207	233	233	214	220	239	228	98	74	206
188	88	179	209	185	215	211	158	139	75	20	169
189	97	165	84	10	168	134	11	31	62	22	148
199	168	191	193	158	227	178	143	182	106	36	190
205	174	155	252	236	231	149	178	228	43	95	234
190	216	116	149	236	187	86	150	79	38	218	241
190	224	147	108	227	210	127	102	36	101	255	224
190	214	173	66	103	143	96	50	2	109	249	215
187	196	235	75	1	81	47	0	6	217	255	211
183	202	237	145	0	0	12	108	200	138	243	236
195	206	123	207	177	121	123	200	175	13	96	218

MIT 6.191

Images are Numbers

157	153	174	168	150	152	129	151	172	161	155	156
155	182	163	74	75	62	33	17	110	210	180	154
180	180	50	14	34	6	10	33	48	106	159	181
206	109	5	124	131	111	120	204	166	15	56	180
194	68	137	251	237	239	239	228	227	87	71	201
172	105	207	233	233	214	220	239	228	98	74	206
188	88	179	209	185	215	211	158	139	75	20	169
189	97	165	84	10	168	134	11	31	62	22	148
199	168	191	193	158	227	178	143	182	106	36	190
205	174	155	252	236	231	149	178	228	43	95	234
190	216	116	149	236	187	86	150	79	38	218	241
190	224	147	108	227	210	127	102	36	101	255	224
190	214	173	66	103	143	96	50	2	109	249	215
187	196	235	75	1	81	47	0	6	217	255	211
183	202	237	145	0	0	12	108	200	138	243	236
195	206	123	207	177	121	123	200	175	13	96	218

What the computer sees

157	153	174	168	150	152	129	151	172	161	155	156
155	182	163	74	75	62	33	17	110	210	180	154
180	180	50	14	34	6	10	33	48	106	159	181
206	109	5	124	131	111	120	204	166	15	56	180
194	68	137	251	237	239	239	228	227	87	71	201
172	105	207	233	233	214	220	239	228	98	74	206
188	88	179	209	185	215	211	158	139	75	20	169
189	97	165	84	10	168	134	11	31	62	22	148
199	168	191	193	158	227	178	143	182	106	36	190
205	174	155	252	236	231	149	178	228	43	95	234
190	216	116	149	236	187	86	150	79	38	218	241
190	224	147	108	227	210	127	102	36	101	255	224
190	214	173	66	103	143	96	50	2	109	249	215
187	196	235	75	1	81	47	0	6	217	255	211
183	202	237	145	0	0	12	108	200	138	243	236
195	206	123	207	177	121	123	200	175	13	96	218

An image is just a matrix of numbers [0,255]!
i.e., 1080x1080x3 for an RGB image

Tasks in Computer Vision

- **Regression**: output variable takes continuous value
- **Classification**: output variable takes class label. Can produce probability of belonging to a particular class

MIT 6S191

High Level Feature Detection

Let's identify key features in each image category

Nose,
Eyes,
Mouth

Wheels,
License Plate,
Headlights

Door,
Windows,
Steps

MIT 6.S191

Manual Feature Extraction

Domain knowledge → Define features → Detect features to classify

Problems?

MIT 6.S191

Manual Feature Extraction

Domain knowledge → Define features → Detect features to classify

Viewpoint variation

Illumination conditions

Scale variation

Deformation

Occlusion

Background clutter

Intra-class variation

Manual Feature Extraction

Domain knowledge → Define features → **Detect features to classify**

Viewpoint variation

Scale variation

Deformation

Occlusion

Illumination conditions

Background clutter

Intra-class variation

Learning Feature Representations

Can we learn a **hierarchy of features** directly from the data instead of hand engineering?

Low level features

Edges, dark spots

Mid level features

Eyes, ears, nose

High level features

Facial structure

MIT 6S191

Learning Visual Features

MIT 6.S191

Fully Connected Neural Network

Fully Connected Neural Network

Input:

- 2D image
- Vector of pixel values

Fully Connected:

- Connect neuron in hidden layer to all neurons in input layer
- No spatial information!
- And many, many parameters!

MIT 6.S191

Fully Connected Neural Network

Input:

- 2D image
- Vector of pixel values

Fully Connected:

- Connect neuron in hidden layer to all neurons in input layer
- No spatial information!
- And many, many parameters!

How can we use **spatial structure** in the input to inform the architecture of the network?

MIT 6.S191

Using Spatial Structure

Input: 2D image.
Array of pixel values

Idea: connect patches of input to neurons in hidden layer.

Neuron connected to region of input. Only "sees" these values.

Using Spatial Structure

Connect patch in input layer to a single neuron in subsequent layer.
Use a sliding window to define connections.
How can we **weight** *the patch to detect particular features?*

Feature Extraction with Convolution

- Filter of size 4x4 : 16 different weights
- Apply this same filter to 4x4 patches in input
- Shift by 2 pixels for next patch

This "patchy" operation is **convolution**

1) Apply a set of weights – a filter – to extract **local features**

2) Use **multiple filters** to extract different features

3) **Spatially share** parameters of each filter

MIT 6.S191

Feature Extraction and Convolution

A Case Study

MIT 6.S191

X or X?

-1	-1	-1	-1	-1	-1	-1	-1	-1
-1	1	-1	-1	-1	-1	-1	1	-1
-1	-1	1	-1	-1	-1	1	-1	-1
-1	-1	-1	1	-1	1	-1	-1	-1
-1	-1	-1	-1	1	-1	-1	-1	-1
-1	-1	-1	1	-1	1	-1	-1	-1
-1	-1	1	-1	-1	-1	1	-1	-1
-1	1	-1	-1	-1	-1	-1	1	-1
-1	-1	-1	-1	-1	-1	-1	-1	-1

$$\overset{?}{=}$$

-1	-1	-1	-1	-1	-1	-1	-1	-1
-1	-1	-1	-1	-1	-1	1	-1	-1
-1	1	-1	-1	-1	1	-1	-1	-1
-1	-1	1	1	-1	1	-1	-1	-1
-1	-1	-1	-1	1	-1	-1	-1	-1
-1	-1	-1	1	-1	1	1	-1	-1
-1	-1	-1	1	-1	-1	-1	1	-1
-1	-1	1	-1	-1	-1	-1	-1	-1
-1	-1	-1	-1	-1	-1	-1	-1	-1

Image is represented as matrix of pixel values… and computers are literal!
We want to be able to classify an X as an X even if it's shifted, shrunk, rotated, deformed.

Features of X

MIT 6.S191

Filters to Detect X Features

filters

MIT 6.S191

The Convolution Operation

MIT 6.S191

The Convolution Operation

Suppose we want to compute the convolution of a 5x5 image and a 3x3 filter:

1	1	1	0	0
0	1	1	1	0
0	0	1	1	1
0	0	1	1	0
0	1	1	0	0

image

$\otimes$

1	0	1
0	1	0
1	0	1

filter

We slide the 3x3 filter over the input image, element-wise multiply, and add the outputs…

MIT 6.S191

The Convolution Operation

We slide the 3x3 filter over the input image, element-wise multiply, and add the outputs:

MIT 6.S191

The Convolution Operation

We slide the 3x3 filter over the input image, element-wise multiply, and add the outputs:

MIT 6.S191

The Convolution Operation

We slide the 3x3 filter over the input image, element-wise multiply, and add the outputs:

MIT 6.S191

The Convolution Operation

We slide the 3x3 filter over the input image, element-wise multiply, and add the outputs:

MIT 6.S191

The Convolution Operation

We slide the 3x3 filter over the input image, element-wise multiply, and add the outputs:

MIT 6.S191

The Convolution Operation

We slide the 3x3 filter over the input image, element-wise multiply, and add the outputs:

MIT 6.S191

The Convolution Operation

We slide the 3x3 filter over the input image, element-wise multiply, and add the outputs:

MIT 6.S191

The Convolution Operation

We slide the 3x3 filter over the input image, element-wise multiply, and add the outputs:

MIT 6.S191

The Convolution Operation

We slide the 3x3 filter over the input image, element-wise multiply, and add the outputs:

MIT 6.S191

Producing Feature Maps

Original

Sharpen

Edge Detect

"Strong" Edge Detect

Feature Extraction with Convolution

1) Apply a set of weights – a filter – to extract **local features**

2) Use **multiple filters** to extract different features

3) **Spatially share** parameters of each filter

MIT 6.S191

Convolutional Neural Networks (CNNs)

MIT 6.S191

CNNs for Classification

`tf.keras.layers.Conv2D`	1. **Convolution**: Apply filters to generate feature maps.	`torch.nn.Conv2d`
`tf.keras.activations.*`	2. **Non-linearity**: Often ReLU.	`torch.nn.ReLU,…`
`tf.keras.layers.MaxPool2D`	3. **Pooling**: Downsampling operation on each feature map.	`torch.nn.MaxPool2d`

Train model with image data.
Learn weights of filters in convolutional layers.

MIT 6.S191

Convolutional Layers: Local Connectivity

`tf.keras.layers.Conv2D`

`torch.nn.Conv2d`

For a neuron in hidden layer:

- Take inputs from patch
- Compute weighted sum
- Apply bias

MIT 6S191

Convolutional Layers: Local Connectivity

4x4 filter: matrix of weights w_{ij}

$$\sum_{i=1}^{4}\sum_{j=1}^{4} w_{ij}\, x_{i+p,j+q} + b$$

for neuron (p,q) in hidden layer

`tf.keras.layers.Conv2D`

`torch.nn.Conv2d`

For a neuron in hidden layer:

- Take inputs from patch
- Compute weighted sum
- Apply bias

1) applying a window of weights
2) computing linear combinations
3) activating with non-linear function

CNNs: Spatial Arrangement of Output Volume

Layer Dimensions:

$h \; x \; w \; x \; d$

where h and w are spatial dimensions
d (depth) = number of filters

Stride:

Filter step size

Receptive Field:

Locations in input image that a node is path connected to

```
tf.keras.layers.Conv2D( filters=d, kernel_size=(h,w), strides=s )
```

```
torch.nn.Conv2d( in_channels=3, out_channels=d, kernel_size=(h,w), stride=s )
```

Need to specify input dimensionality!

MIT 6.S191

Introducing Non-Linearity

- Apply after every convolution operation (i.e., after convolutional layers)
- ReLU: pixel-by-pixel operation that replaces all negative values by zero. **Non-linear operation!**

Rectified Linear Unit (ReLU)

$$g(z) = \max(0, z)$$

`tf.keras.layers.ReLU`

`torch.nn.ReLU`

Pooling

max pool with 2x2 filters and stride 2

```
tf.keras.layers.MaxPool2D(
    pool_size=(2,2),
    strides=2
)
```

```
torch.nn.MaxPool2d(
    kernel_size=(2,2),
    stride=2
)
```

6	8
3	4

1) Reduced dimensionality
2) Spatial invariance

How else can we downsample and preserve spatial invariance?

MIT 6.S191

Representation Learning in Deep CNNs

Low level features

Edges, dark spots

Conv Layer 1

Mid level features

Eyes, ears, nose

Conv Layer 2

High level features

Facial structure

Conv Layer 3

MIT 6.191

CNNs for Classification: Feature Learning

1. Learn features in input image through **convolution**
2. Introduce **non-linearity** through activation function (real-world data is non-linear!)
3. Reduce dimensionality and preserve spatial invariance with **pooling**

MIT 6.S191

CNNs for Classification: Class Probabilities

- CONV and POOL layers output high-level features of input
- Fully connected layer uses these features for classifying input image
- Express output as **probability** of image belonging to a particular class

$$\text{softmax}(y_i) = \frac{e^{y_i}}{\sum_j e^{y_j}}$$

MIT 6.S191

Putting it all together: CNN in TensorFlow

```
import tensorflow as tf

def generate_model():
    model = tf.keras.Sequential([
        # first convolutional layer
        tf.keras.layers.Conv2D(32, filter_size=3, activation='relu'),
        tf.keras.layers.MaxPool2D(pool_size=2, strides=2),

        # second convolutional layer
        tf.keras.layers.Conv2D(64, filter_size=3, activation='relu'),
        tf.keras.layers.MaxPool2D(pool_size=2, strides=2),

        # fully connected classifier
        tf.keras.layers.Flatten(),
        tf.keras.layers.Dense(1024, activation='relu'),
        tf.keras.layers.Dense(10, activation='softmax')  # 10 outputs
    ])
    return model
```

Putting it all together: CNN in PyTorch

```python
import torch

def generate_model():
    model = nn.Sequential([
        # first and second convolutional layer
        torch.nn.Conv2d(in_channels=3,out_channels=32, filter_size=3),
        torch.nn.ReLU(),
        torch.nn.MaxPool2d(kernel_size=2, stride=2),

        torch.nn.Conv2d(in_channels=32, out_channels=64, filter_size=3),
        torch.nn.ReLU(),
        torch.nn.MaxPool2d(kernel_size=2, stride=2),

        # fully connected classifier
        torch.nn.Flatten(),
        torch.nn.Linear(64*6*6, 1024), # flattened dim after 2 conv layers
        torch.nn.ReLU(),
        torch.nn.Linear(1024, 10),  # 10 outputs
    ])
    return model
```

An Architecture for Many Applications

MIT 6.S191

An Architecture for Many Applications

Classification
Object detection
Segmentation
Probabilistic control

MIT 6.S191

Classification: Breast Cancer Screening

International evaluation of an AI system for breast cancer screening

nature

CNN-based system outperformed expert radiologists at detecting breast cancer from mammograms

Breast cancer case missed by radiologist but detected by AI

Object Detection

Image x

CNN

Taxi

Class label y

Image x

CNN

Label (x, y, w, h)

6.S191

Object Detection

Image x

Output:
taxi: (x1, y1, w1, h1)

Image x

Output:
taxi: (x1, y1, w1, h1)
person: (x2, y2, w2, h2)
person: (x3, y3, w3, h3)
....

Naïve Solution to Object Detection

Problem: Way too many inputs! This results in too many scales, positions, sizes!

Object Detection with R-CNNs

R-CNN algorithm: Find regions that we think have objects. Use CNN to classify.

1. Input image
2. Extract region proposals (~2k)
3. Compute CNN features
4. Classify regions

Problems: 1) Slow! Many regions; time intensive inference.
2) Brittle! Manually defined region proposals.

Faster R-CNN Learns Region Proposals

Classification of regions → object detection

Feature extraction over proposed regions

Region proposal network to learn candidate regions

Learned, data-driven

Image input directly into convolutional feature extractor

Fast! Only input image once!

MIT 6.S191

Semantic Segmentation: Fully Convolutional Networks

FCN: Fully Convolutional Network.
Network designed with all convolutional layers,
with **downsampling** and **upsampling** operations

Input:
3 x H x W

High-res:
D_1 x H/2 x W/2

High-res:
D_1 x H/2 x W/2

Predictions:
H x W

`tf.keras.layers.Conv2DTranspose`

`torch.nn.ConvTranspose2d`

Continuous Control: Navigation from Vision

Raw Perception
I
(ex. camera)

Coarse Maps
M
(ex. GPS)

Possible Control Commands

End-to-End Framework for Autonomous Navigation

Entire model is trained end-to-end **without any human labelling or annotations**

$$L = -\log(P(\theta|I,M))$$

Amini+ ICRA 2019.

Deep Learning for Computer Vision: Impact

Deep Learning for Computer Vision: Summary

Foundations

- Why computer vision?
- Representing images
- Convolutions for feature extraction

CNNs

- CNN architecture
- Application to classification
- ImageNet

Applications

- Segmentation, image captioning, control
- Security, medicine, robotics

MIT 6.S191

MIT 6.S191
Introduction to Deep Learning

Lab 2: Facial Detection Systems

Link to download labs:
introtodeeplearning.com#schedule
github.com/MITDeepLearning/introtodeeplearning

1. Open the lab in Google Colab
2. Start executing code blocks and filling in the #TODOs
3. Need help? Come to 32-123!

Introduction to Deep Learning

Announcements and Reminders

Lecture 4: Generative Modeling starting soon!

Lab 2 begins today!
Prizes for participants!
Enter for a chance to win $$$!

Lots of swag will be available after class tomorrow!

Interested in cutting-edge AI jobs?
introtodeeplearning.com/jobs.html