CS 6890: Deep Learning

Autoencoders

Razvan C. Bunescu

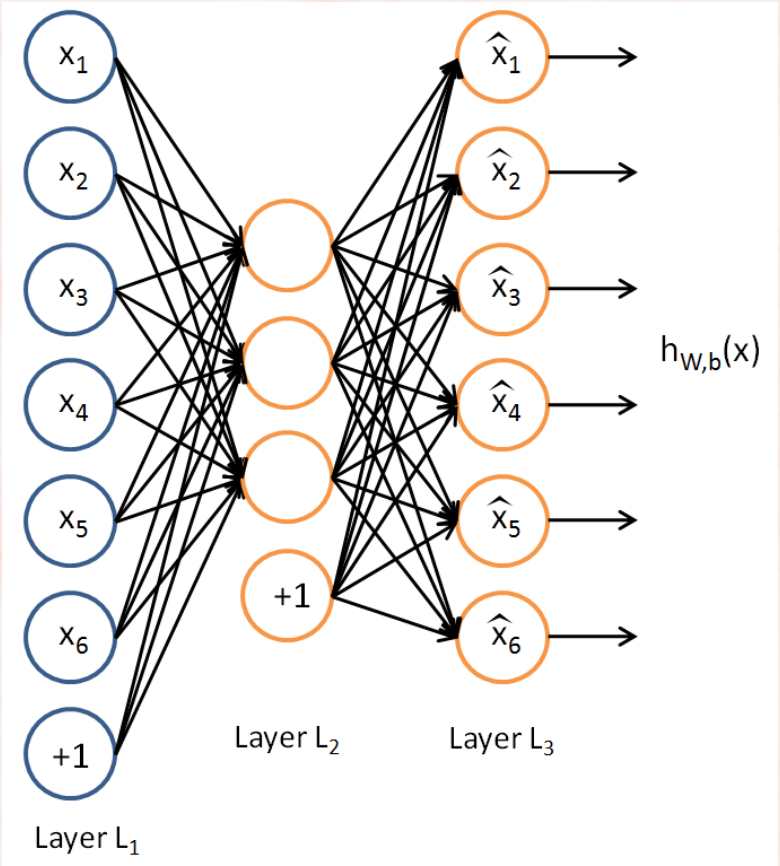
School of Electrical Engineering and Computer Science

bunescu@ohio.edu

Autoencoders

Neural networks with **one hidden layer** that are trained to replicate the input, i.e **target is equal with the input**:

- **Undercomplete**.
- **Overcomplete**:
  - need regularization.



Regularized Autoencoders

1. **Sparse Autoencoders**:
   - Require the code to be sparse.
     - include a sparsity penalty term in the objective.
2. **Denoising Autoencoders**:
   - Use corrupted **x** as input, still use original **x** as target.
3. **Contractive Autoencoders**:
   - Require the code **h** to not change much when input **x** changes.
     - include a Jacobian norm penalty term in the objective.

Sparse Autoencoders

- Compute the average activation of each neuron on the hidden layer:

$$\hat{\rho}_l = \frac{1}{m}\sum_{k=1}^{m} a_l^{(2)}(x^{(k)})$$

- Require the average activations to be low:
  - close to a sparsity parameter ρ = 0.05.

$$KL(\hat{\rho}_l) = KL(\rho \,||\, \hat{\rho}_l) = \rho \log \frac{\rho}{\hat{\rho}_l} + (1-\rho)\log \frac{1-\rho}{1-\hat{\rho}_l}$$

One Hidden Layer with Sparse Activations

- Updated cost function:

sparsity term

$$J(W,b) = \frac{1}{m}\sum_{k=1}^{m} J(W,b,x^{(k)},y^{(k)}) + \frac{\lambda}{2}\|W\|_{2,2}^2 + \beta\sum_{l=1}^{s_2} KL(\hat{\rho}_l)$$

$$J(W,b,x,y) = \frac{1}{2}\left\|h_{W,b}(x) - y\right\|^2$$

$$\|W\|_{2,2}^2 = \sum_{l=1}^{n_l-1}\sum_{i=1}^{s_l}\sum_{j=1}^{s_l+1}\left(W_{ij}^{(l)}\right)^2$$

$$KL(\hat{\rho}_l) = KL(\rho \,\|\, \hat{\rho}_l) = \rho\log\frac{\rho}{\hat{\rho}_l} + (1-\rho)\log\frac{1-\rho}{1-\hat{\rho}_l}$$

$$\hat{\rho}_l = \frac{1}{m}\sum_{k=1}^{m} a_l^{(2)}(x^{(k)})$$

One Hidden Layer with Sparse Activations

- Updated gradient:

$$\frac{\partial J(W,b)}{\partial W_{ij}^{(1)}} = \frac{1}{m}\sum_{k=1}^{m}\frac{\partial J(W,b,x^{(k)},y^{(k)})}{\partial W_{ij}^{(1)}} + \lambda W_{ij}^{(1)} + \beta\sum_{l=1}^{s_2}\frac{\partial KL}{\partial \hat{\rho}_l}\frac{\partial \hat{\rho}_l}{\partial W_{ij}^{(1)}}$$

$$= \lambda W_{ij}^{(1)} + \frac{1}{m}\sum_{k=1}^{m} a_j^{(1)}\delta_i^{(2)} + \beta\sum_{l=1}^{s_2}\frac{\partial KL}{\partial \hat{\rho}_l}\frac{1}{m}\sum_{k=1}^{m}\frac{\partial a_l^{(2)}}{\partial W_{ij}^{(1)}}$$

$$= \lambda W_{ij}^{(1)} + \frac{1}{m}\sum_{k=1}^{m}\left(a_j^{(1)}\delta_i^{(2)} + \beta\sum_{l=1}^{s_2}\frac{\partial KL}{\partial \hat{\rho}_l}\frac{\partial a_l^{(2)}}{\partial z_l^{(2)}}\frac{\partial z_l^{(2)}}{\partial W_{ij}^{(1)}}\right)$$

$$\frac{\partial z_l^{(2)}}{\partial W_{ij}^{(1)}} = \begin{cases} a_j^1 & \text{if } l = i \\ 0 & \text{if } l \neq i \end{cases}$$

One Hidden Layer with Sparse Activations

- Updated gradient:

$$\frac{\partial J(W,b)}{\partial W_{ij}^{(1)}} = \lambda W_{ij}^{(1)} + \frac{1}{m}\sum_{k=1}^{m}\left(a_j^{(1)}\delta_i^{(2)} + \beta \sum_{l=1}^{s_2} \frac{\partial KL}{\partial \hat{\rho}_l} \frac{\partial a_l^{(2)}}{\partial z_l^{(2)}} \frac{\partial z_l^{(2)}}{\partial W_{ij}^{(1)}} \right)$$

$$= \lambda W_{ij}^{(1)} + \frac{1}{m}\sum_{k=1}^{m}\left(a_j^{(1)}\delta_i^{(2)} + \beta \frac{\partial KL}{\partial \hat{\rho}_i} \frac{\partial a_i^{(2)}}{\partial z_i^{(2)}} a_j^{(1)} \right)$$

$$= \lambda W_{ij}^{(1)} + \frac{1}{m}\sum_{k=1}^{m} a_j^{(1)}\left(\delta_i^{(2)} + \beta\left(-\frac{\rho}{\hat{\rho}_i} + \frac{1-\rho}{1-\hat{\rho}_i} \right) f'(z_i^{(2)}) \right)$$

One Hidden Layer with Sparse Activations

- Updated gradient:

$$\frac{\partial J(W,b)}{\partial W_{ij}^{(1)}} = \lambda W_{ij}^{(1)} + \frac{1}{m}\sum_{k=1}^{m} a_j^{(1)} \left(\delta_i^{(2)} + \beta \left(-\frac{\rho}{\hat{\rho}_i} + \frac{1-\rho}{1-\hat{\rho}_i} \right) f'(z_i^{(2)}) \right)$$

Change from: $\delta_i^{(2)} = \left(\sum_{j=1}^{s_3} W_{ji}^{(2)} \delta_j^{(3)} \right) \times f'(z_i^{(2)})$

to: $\delta_i^{(2)} = \left[\left(\sum_{j=1}^{s_3} W_{ji}^{(2)} \delta_j^{(3)} \right) + \beta \left(-\frac{\rho}{\hat{\rho}_i} + \frac{1-\rho}{1-\hat{\rho}_i} \right) \right] \times f'(z_i^{(2)})$

Then gradient is: $\frac{\partial J(W,b)}{\partial W_{ij}^{(1)}} = \lambda W_{ij}^{(1)} + \frac{1}{m}\sum_{k=1}^{m} a_j^{(1)} \delta_i^{(2)}$

One Hidden Layer with Sparse Activations

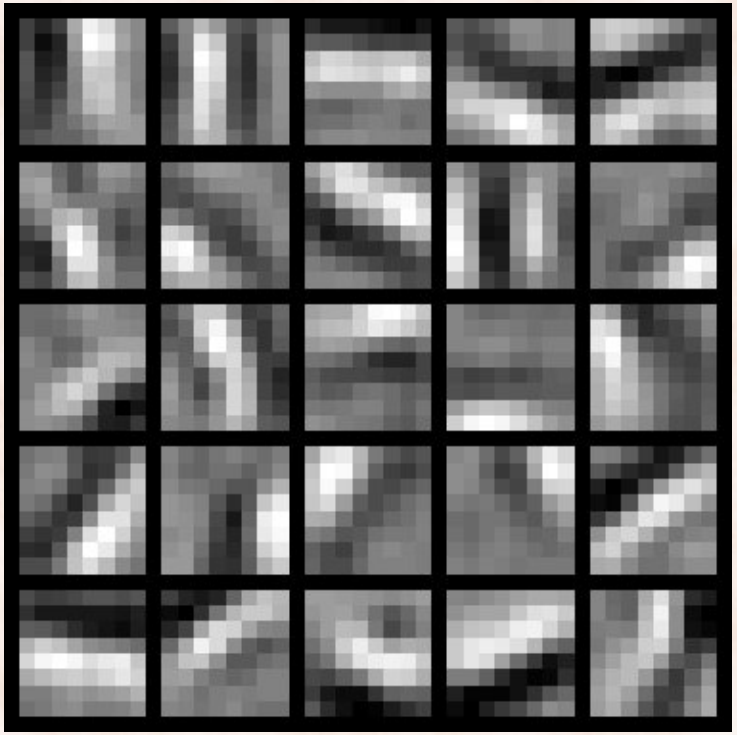
- Updated gradient:

$$\frac{\partial J(W,b)}{\partial W_{ij}^{(1)}} = \lambda W_{ij}^{(1)} + \frac{1}{m}\sum_{k=1}^{m} a_j^{(1)}\delta_i^{(2)}$$

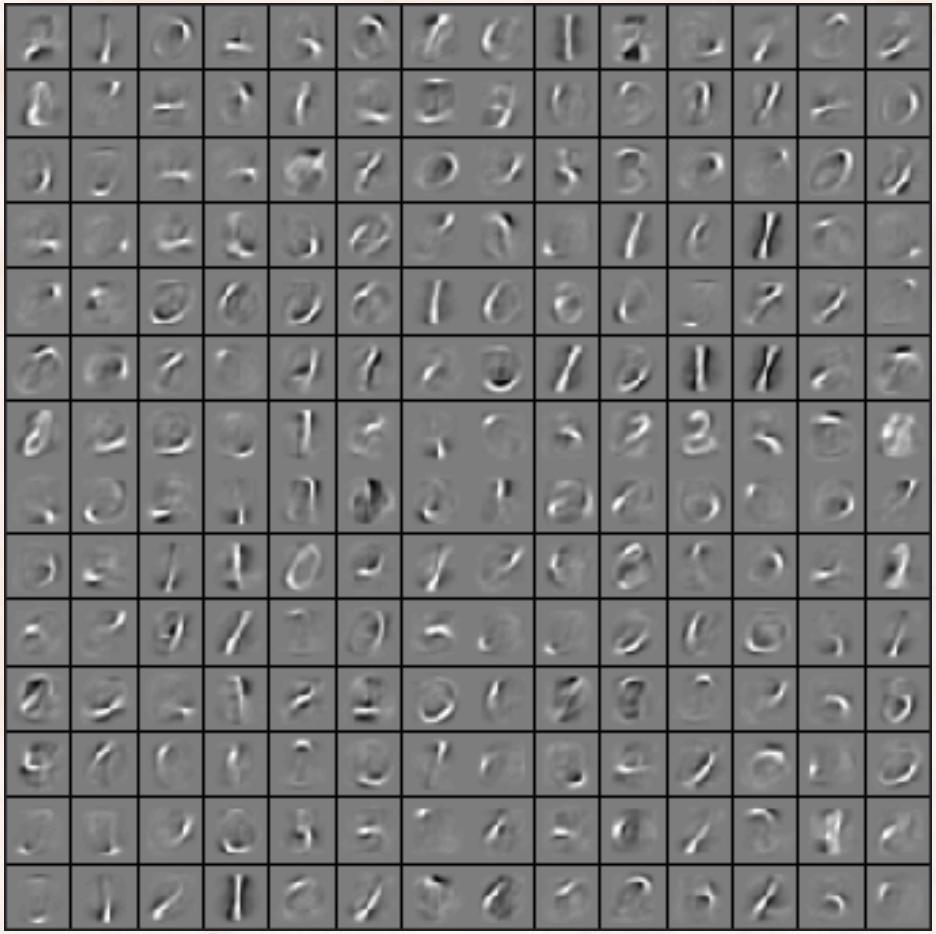
$$\frac{\partial J(W,b,x^{(k)},y^{(k)})}{\partial W_{ij}^{(1)}}$$

$$\delta_i^{(2)} = \left[\left(\sum_{j=1}^{s_3} W_{ji}^{(2)}\delta_j^{(3)}\right) + \beta\left(-\frac{\rho}{\hat{\rho}_i} + \frac{1-\rho}{1-\hat{\rho}_i}\right)\right] \times f'(z_i^{(2)})$$

Natural Features

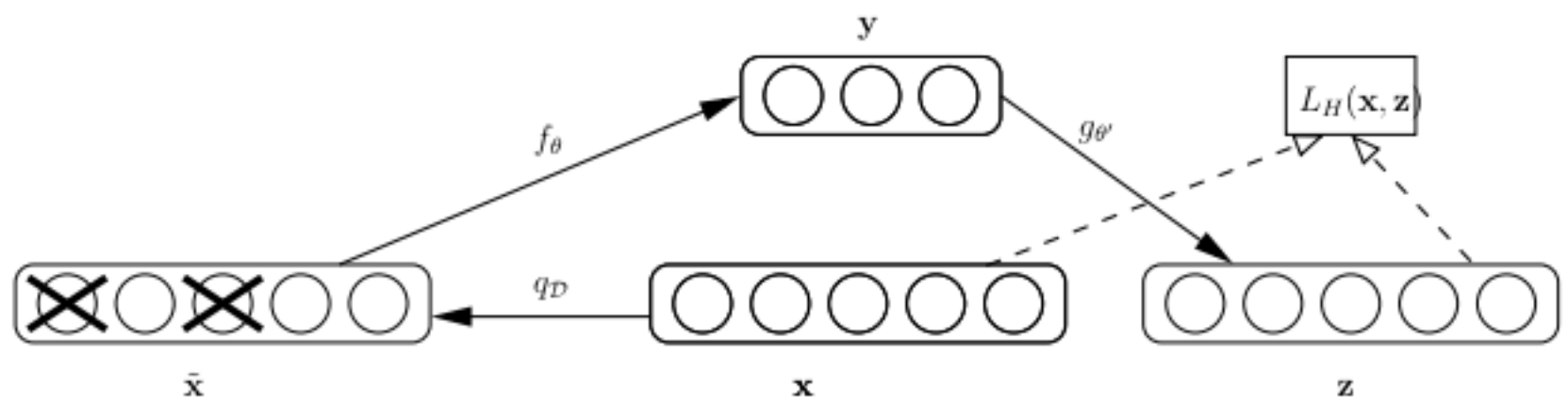


Digit Features



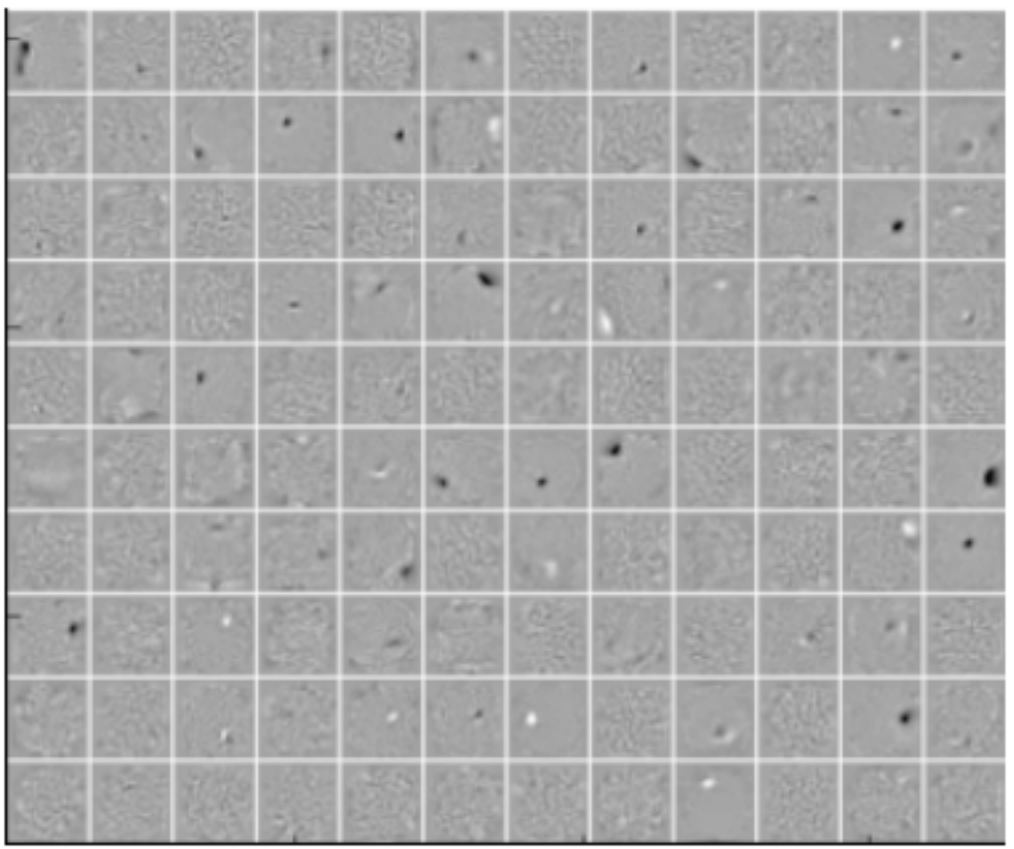
Denoising Autoencoder

[Vincent et al., ICML'08]



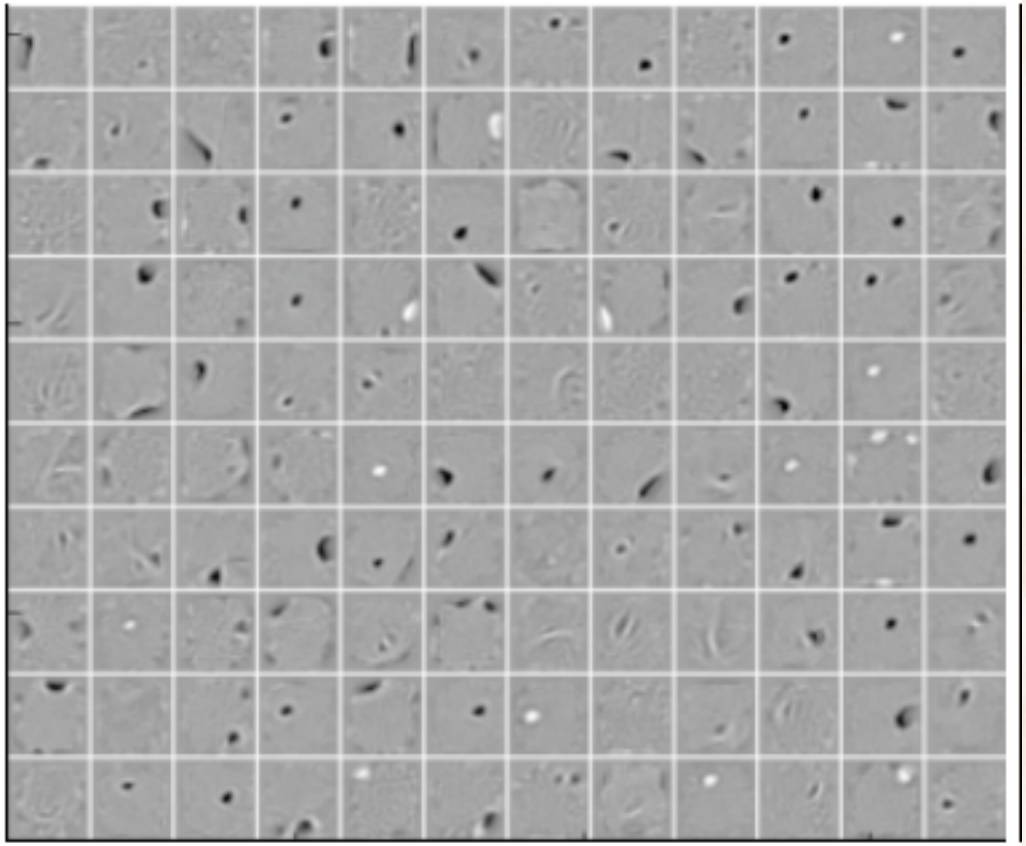
- Learned representations should be robust to partial destruction of the input:
  - Choose a desired proportion ν of destruction.
  - Select ν × |*x*| components of *x* at random and set them to 0 => corrupted ~*x*.
  - Use corrupted ~*x* as input to the encoder, which produces *y*.
  - "Ask" decoder to use *y* to reproduce the original x.

Learned Filters



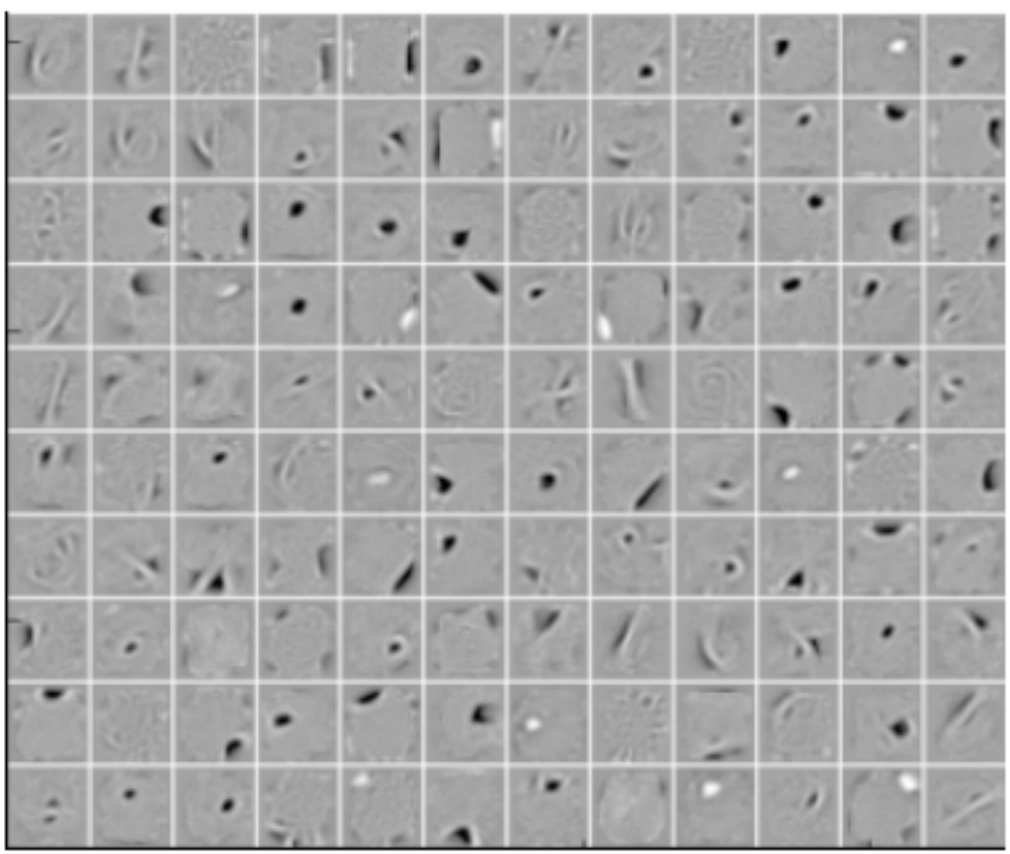
(a) No destroyed inputs

Learned Filters



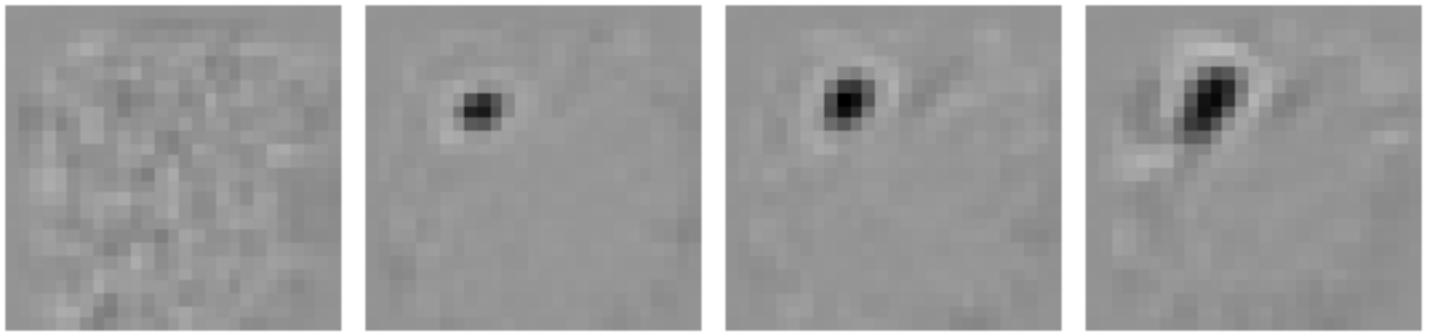
(b) 25% destruction

Learned Filters

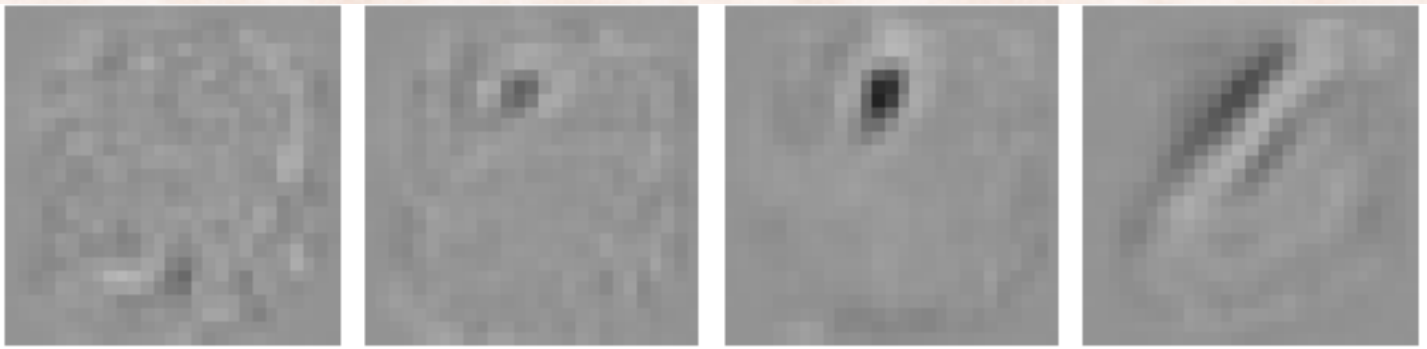


(c) 50% destruction

Learned Filters, Zoomed



(d) Neuron A (0%, 10%, 20%, 50% destruction)



(e) Neuron B (0%, 10%, 20%, 50% destruction)