UNSUPERVISED LEARNING

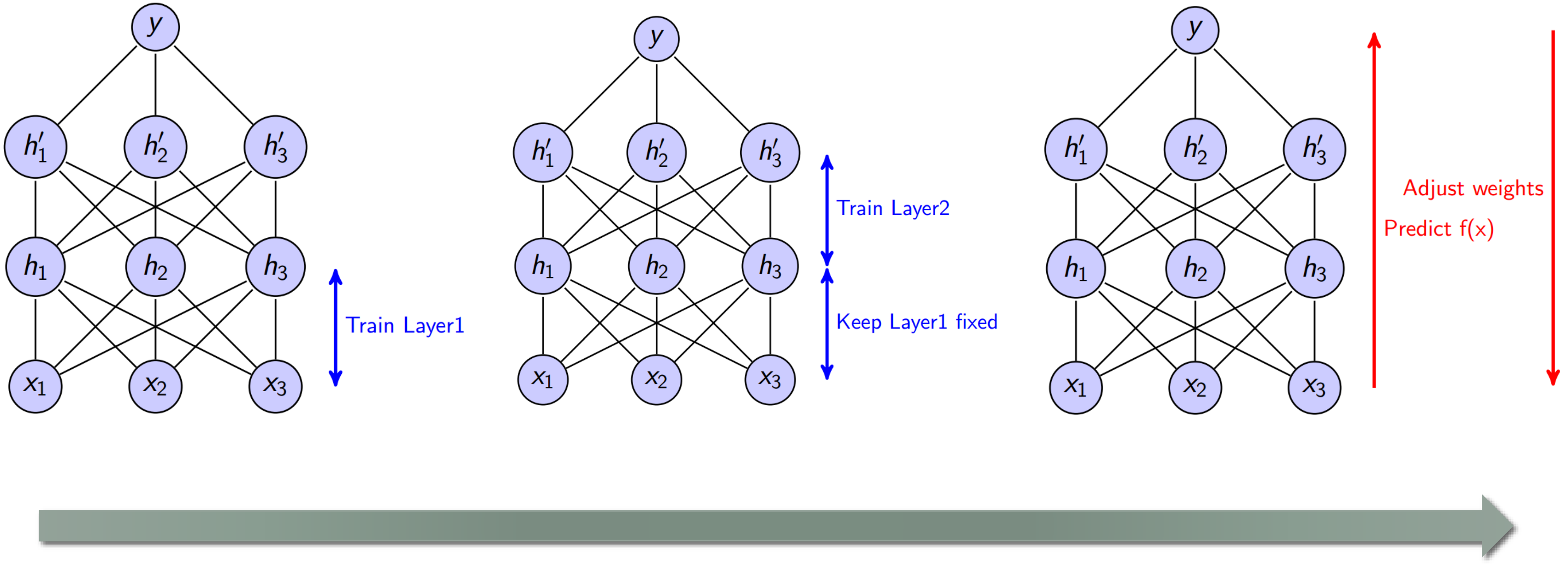
Topics

- Layer-wise (unsupervised) pre-training
  - Restricted Boltzmann Machines
  - Auto-encoders

LAYER-WISE (UNSUPERVISED) PRE-TRAINING

Breakthrough in 2006

- Layer-wise (unsupervised) pre-training



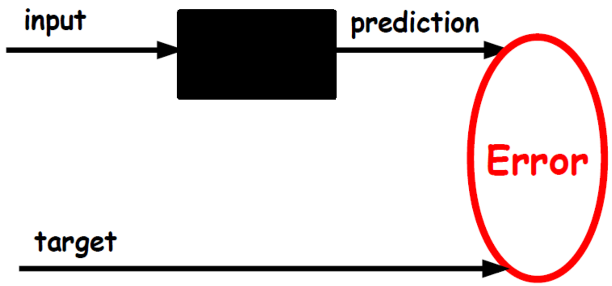
Breakthrough in 2006

- Key idea:
  - Focus on modeling the input $P(X)$ better with each successive layer. Worry about optimizing the task $P(Y|X)$ later.
    - X=observation, Y=label
  - Build generative mode $P(X)$ with a unsupervised manner from unlabeled data.

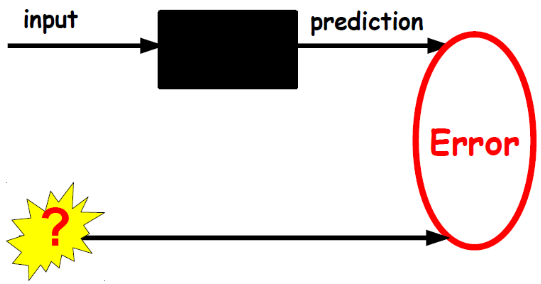
"If you want to do computer vision, first learn computer graphics." -- Geoffrey Hinton

Unsupervised learning

- Supervised learning



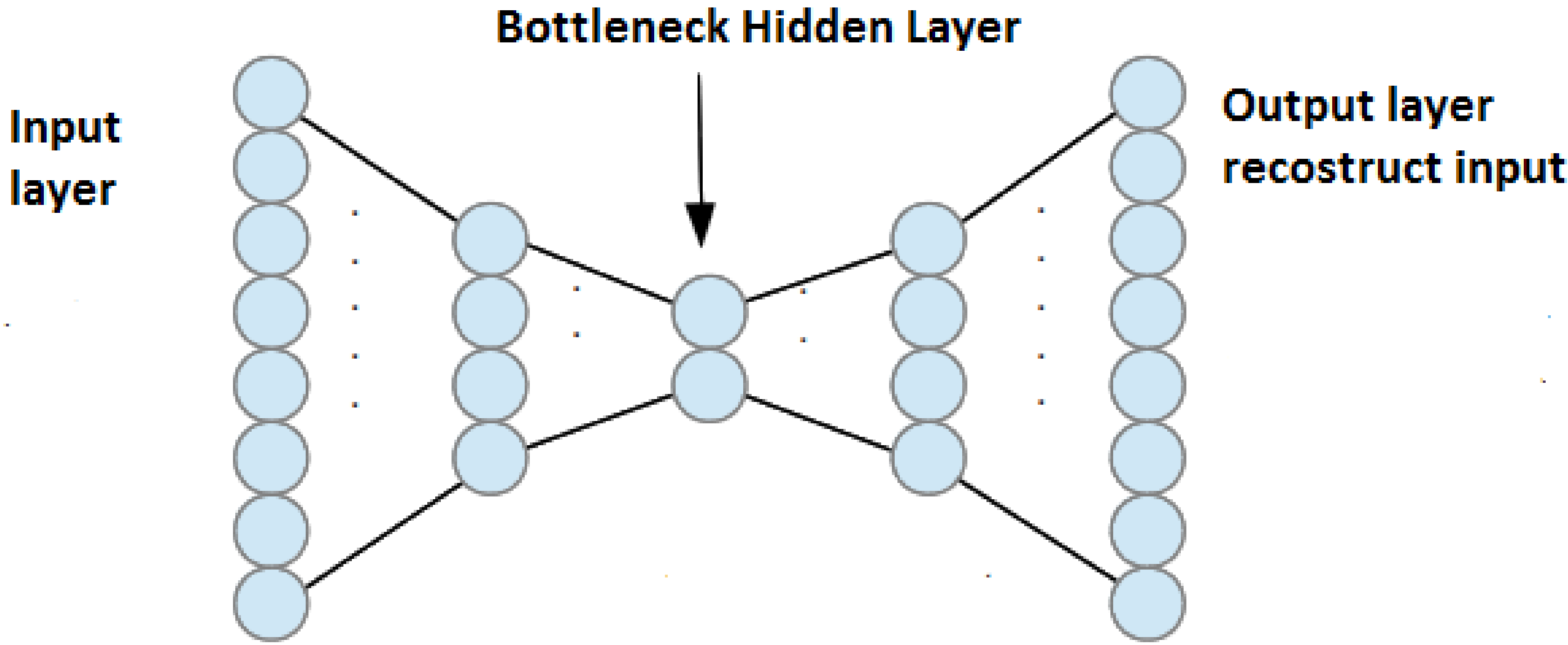
- Unsupervised learning

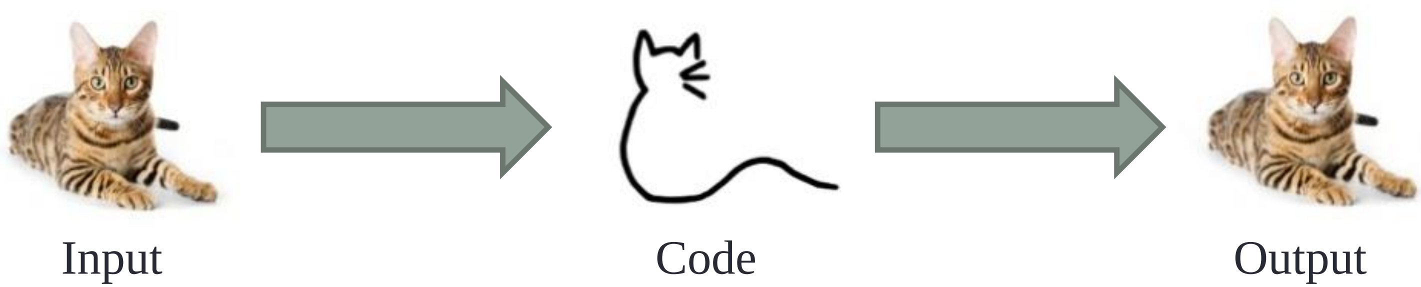


Unsupervised learning

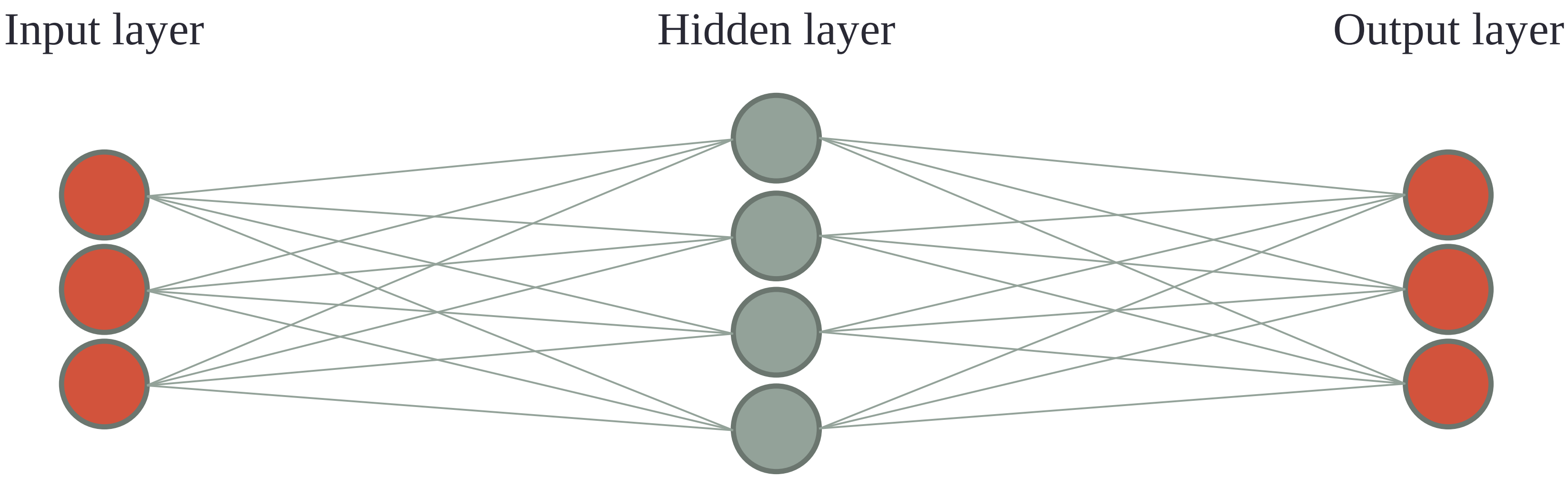
- How to constrain the model to represent training samples better than other data points?
  - [Restricted Boltzmann Machine]
    - Make models that defines the distribution of samples
  - [Auto-encoder with bottleneck]
    - reconstruct the input from the code & make code compact
  - [Sparse auto-encoder]
    - reconstruct the input from the code & make code sparse
  - [Denoising auto-encoder]
    - Add noise to the input or code

Auto-encoder with bottleneck



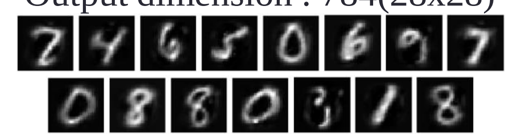


Sparse auto-encoder



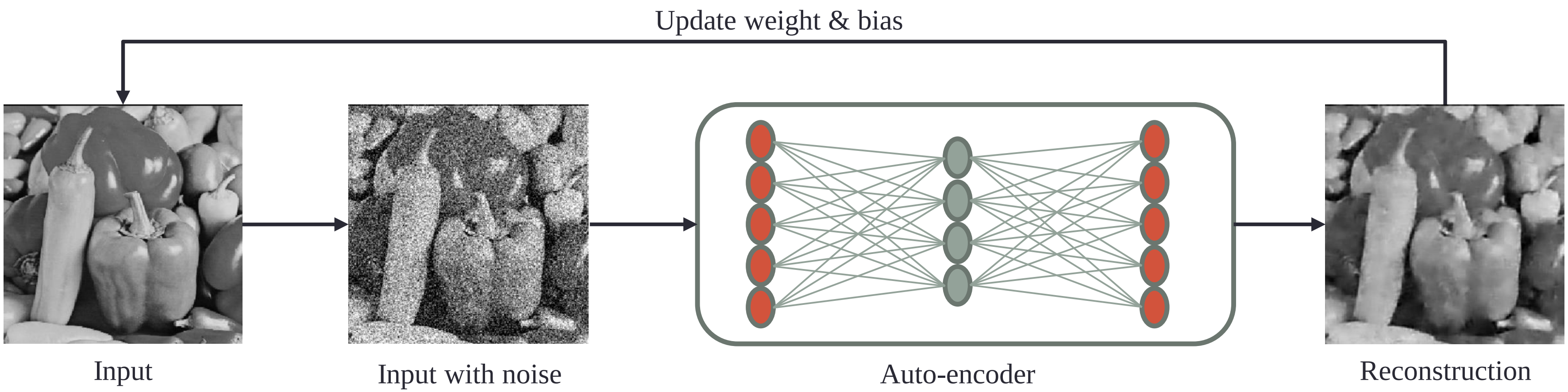
Input : MNIST Dataset
Input dimension : 784(28x28)

Reconstructions
Output dimension : 784(28x28)

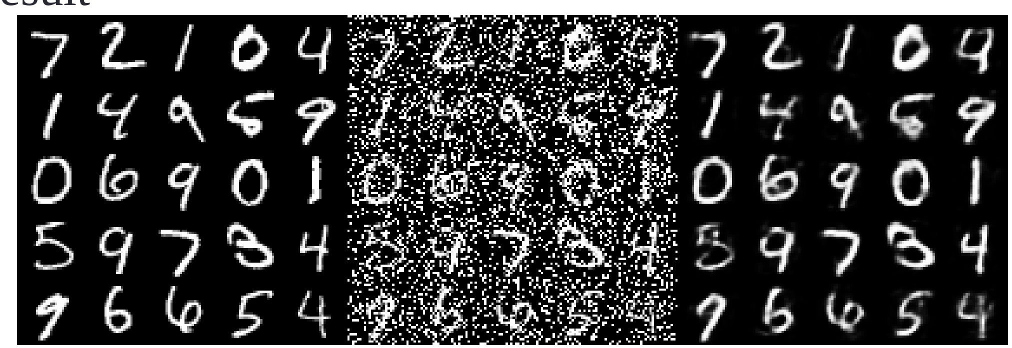


Denoising auto-encoder

Training the denoising auto-encoder



Result

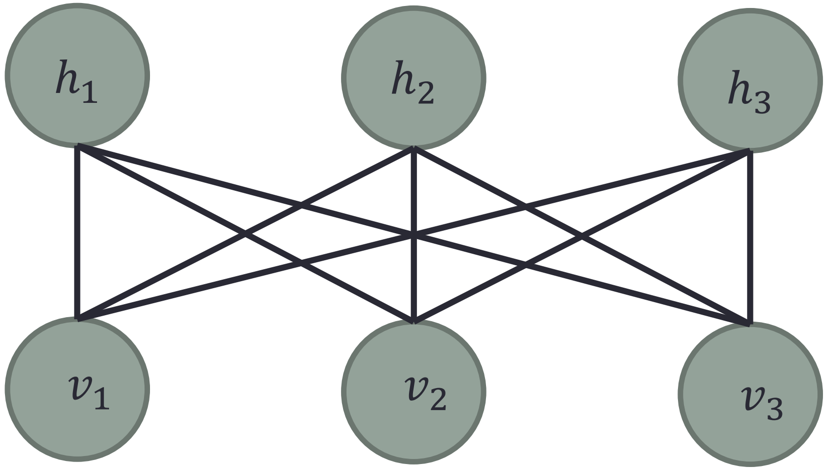


Input Input with noise Reconstruction

RESTRICTED BOLTZMANN MACHINE

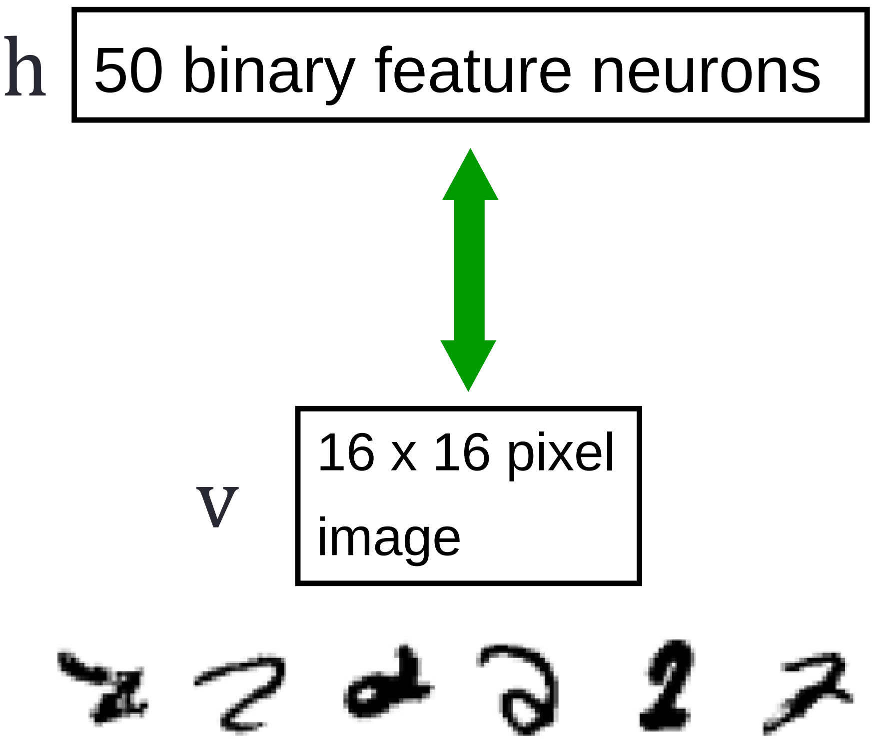
Restricted Boltzmann Machine (RBM)

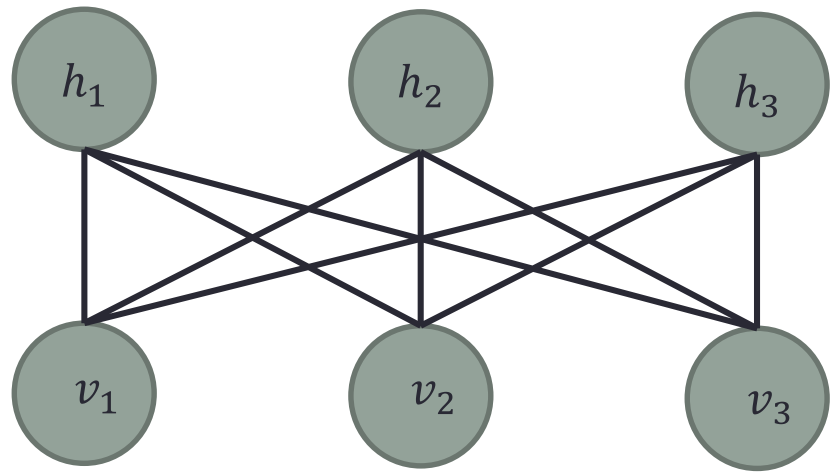
- RBM is a simple energy-based model: $p(\mathrm{v}, \mathrm{h}) = \frac{1}{Z} \exp(-E_\theta(\mathrm{v}, \mathrm{h}))$
  - With only $\mathrm{h} - \mathrm{v}$ interactions: $E_\theta(\mathrm{v}, \mathrm{h}) = -\mathrm{v}^{\mathrm{T}}\mathrm{W}\mathrm{h} - \mathrm{b}^{\mathrm{T}}\mathrm{v} - \mathrm{d}^{\mathrm{T}}\mathrm{h}$
  - here, we assume $\mathrm{h_j}$ and $\mathrm{v_i}$ are binary variables.
  - Normalizer $\mathrm{Z} = \sum_{\mathrm{v,h}} \exp(-E_\theta(\mathrm{v}, \mathrm{h}))$ is called a partition function



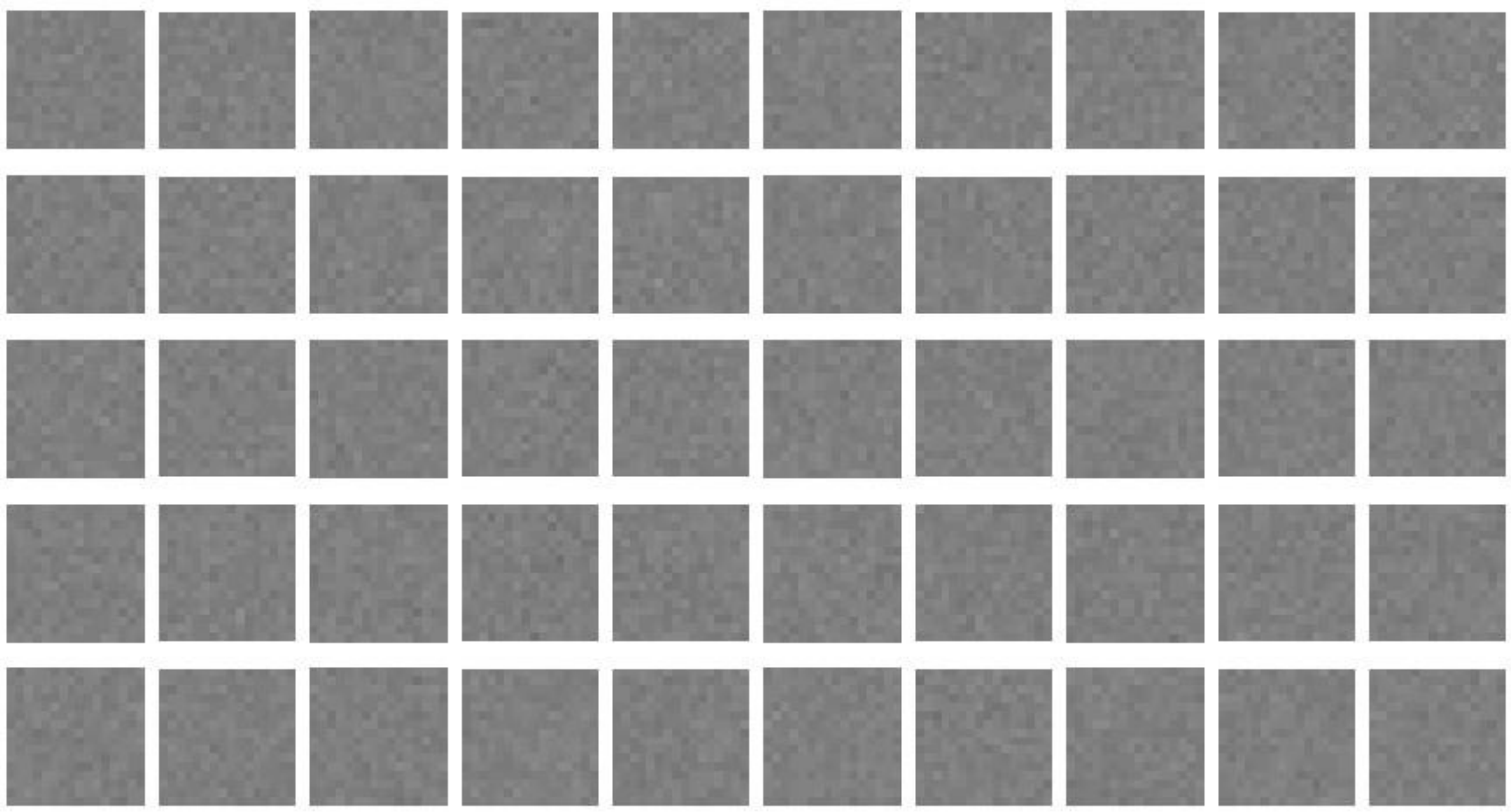
Restricted Boltzmann Machine (RBM)

- The probability that the network assigns to a visible vector v:
  - $p(\mathrm{v}) = \frac{1}{Z}\sum_h \exp(-E_\theta(\mathrm{v},\mathrm{h}))$

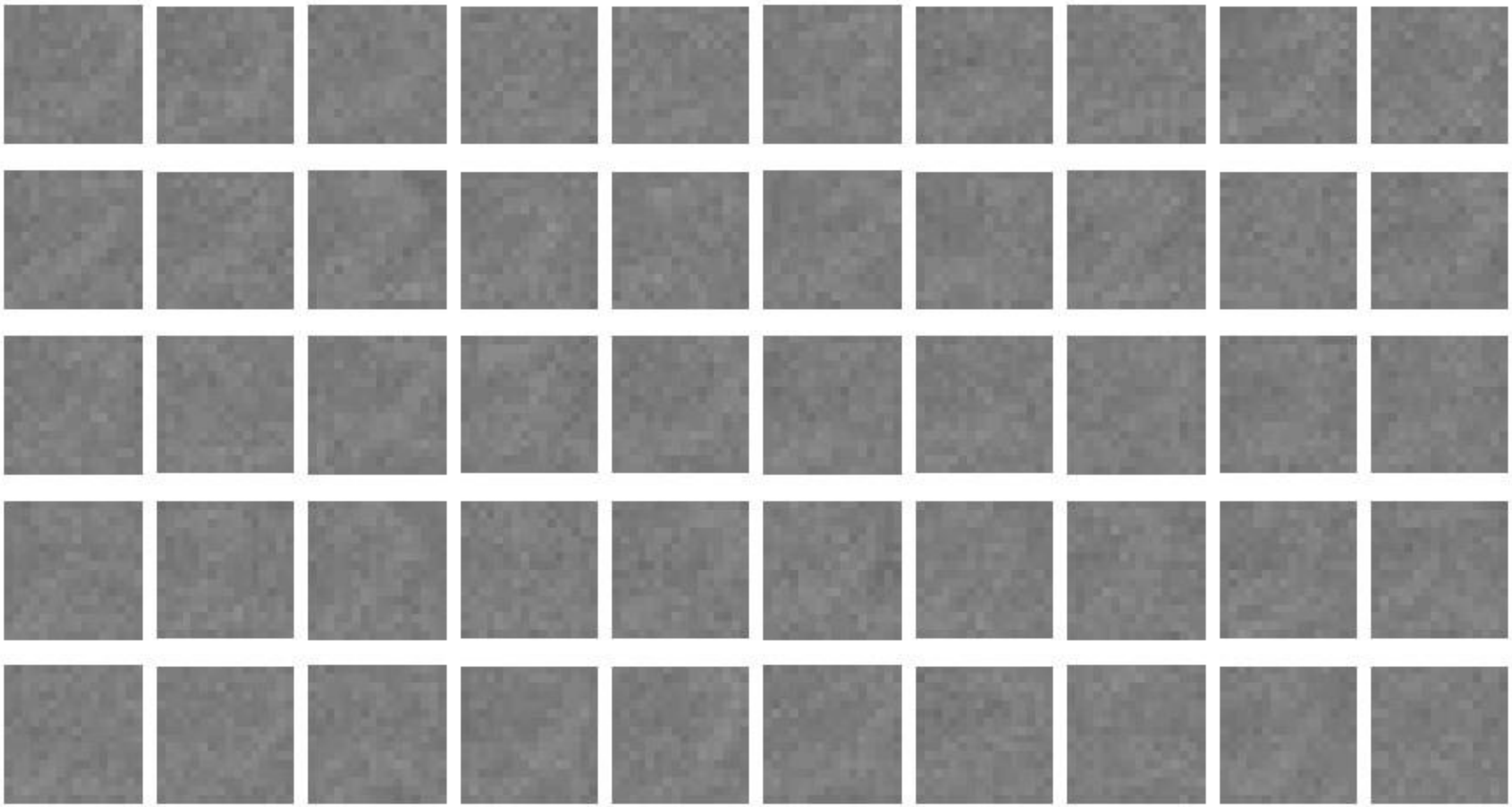


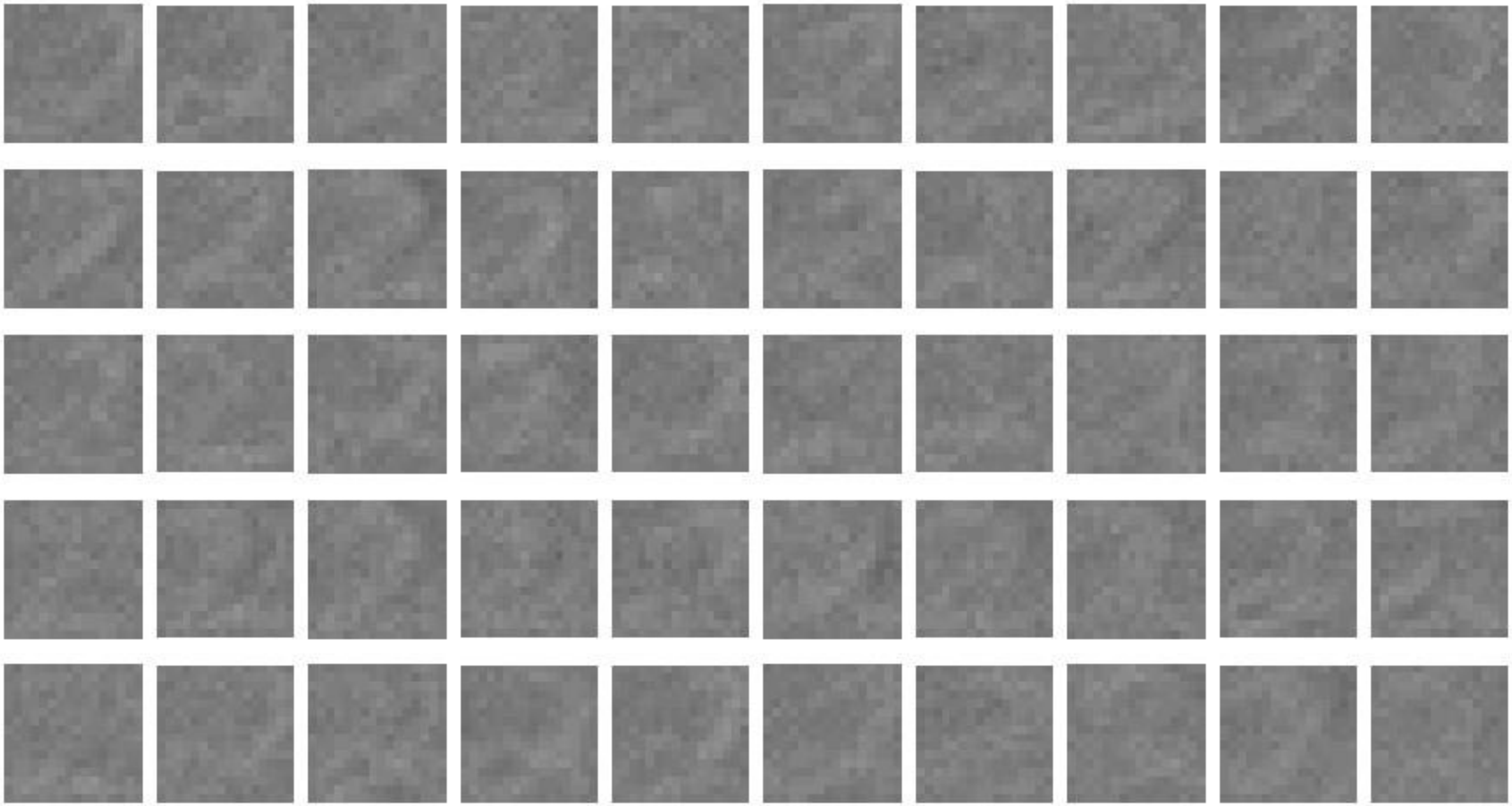


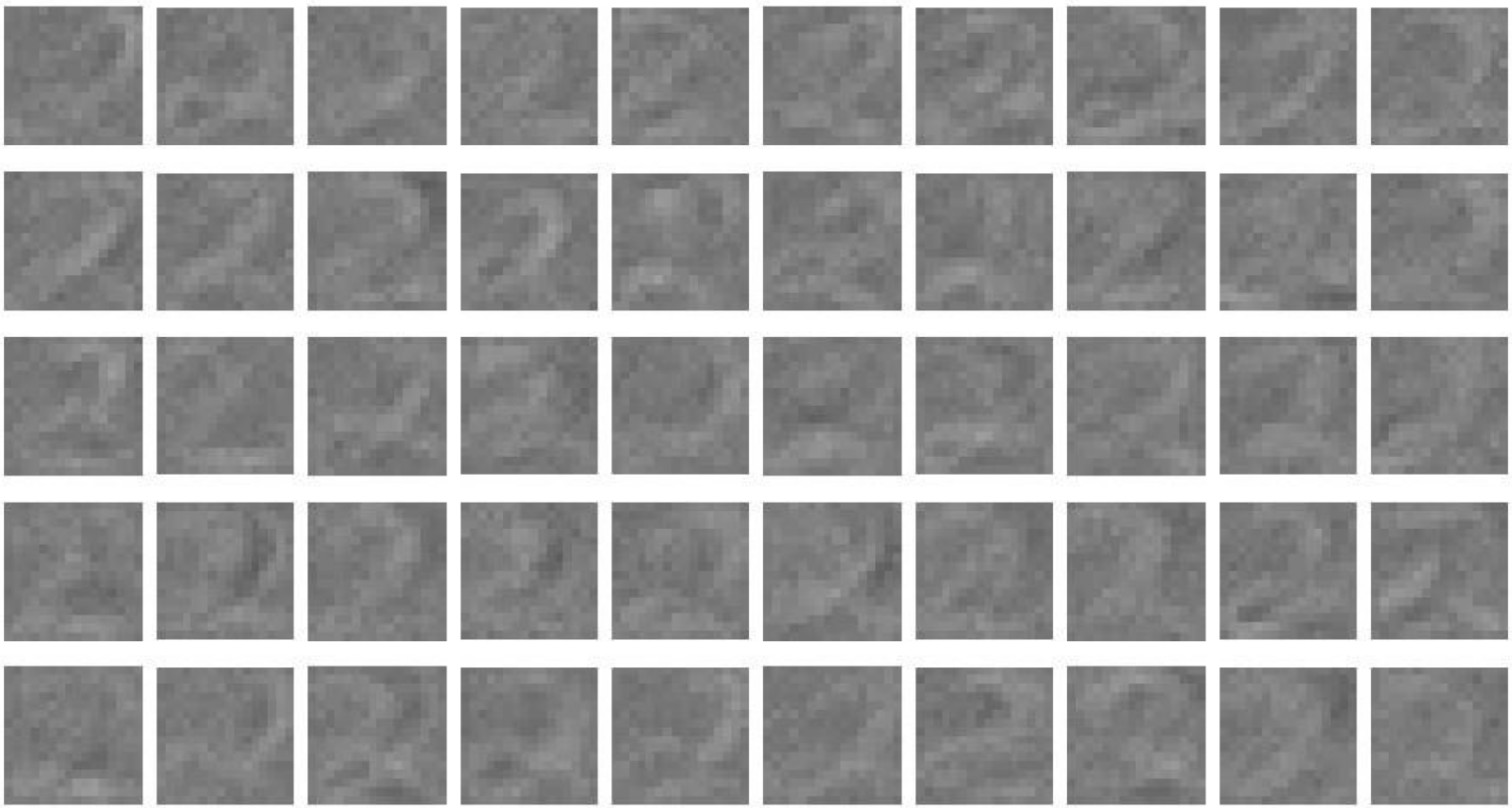
The weights of the 50 feature detectors

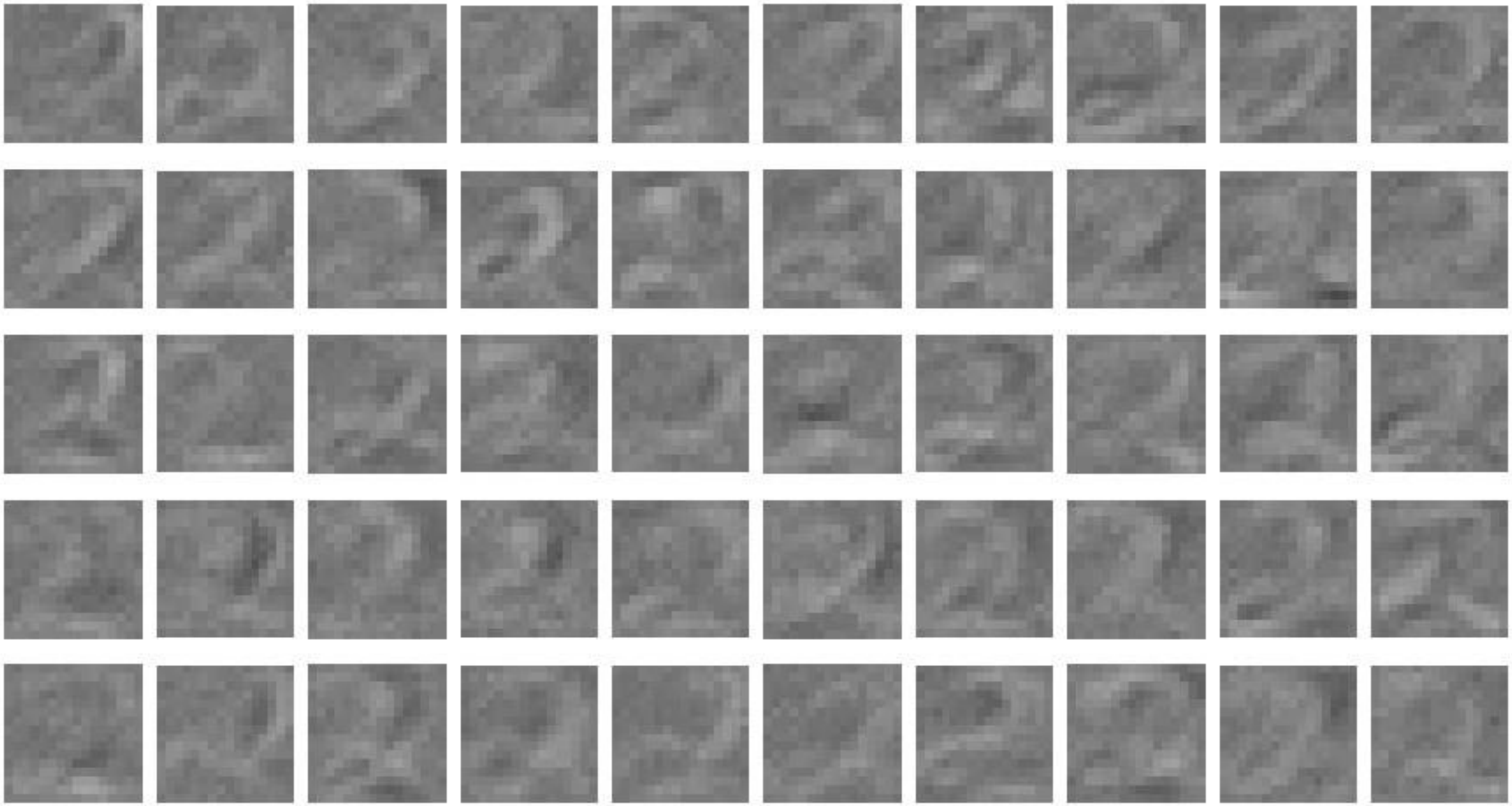


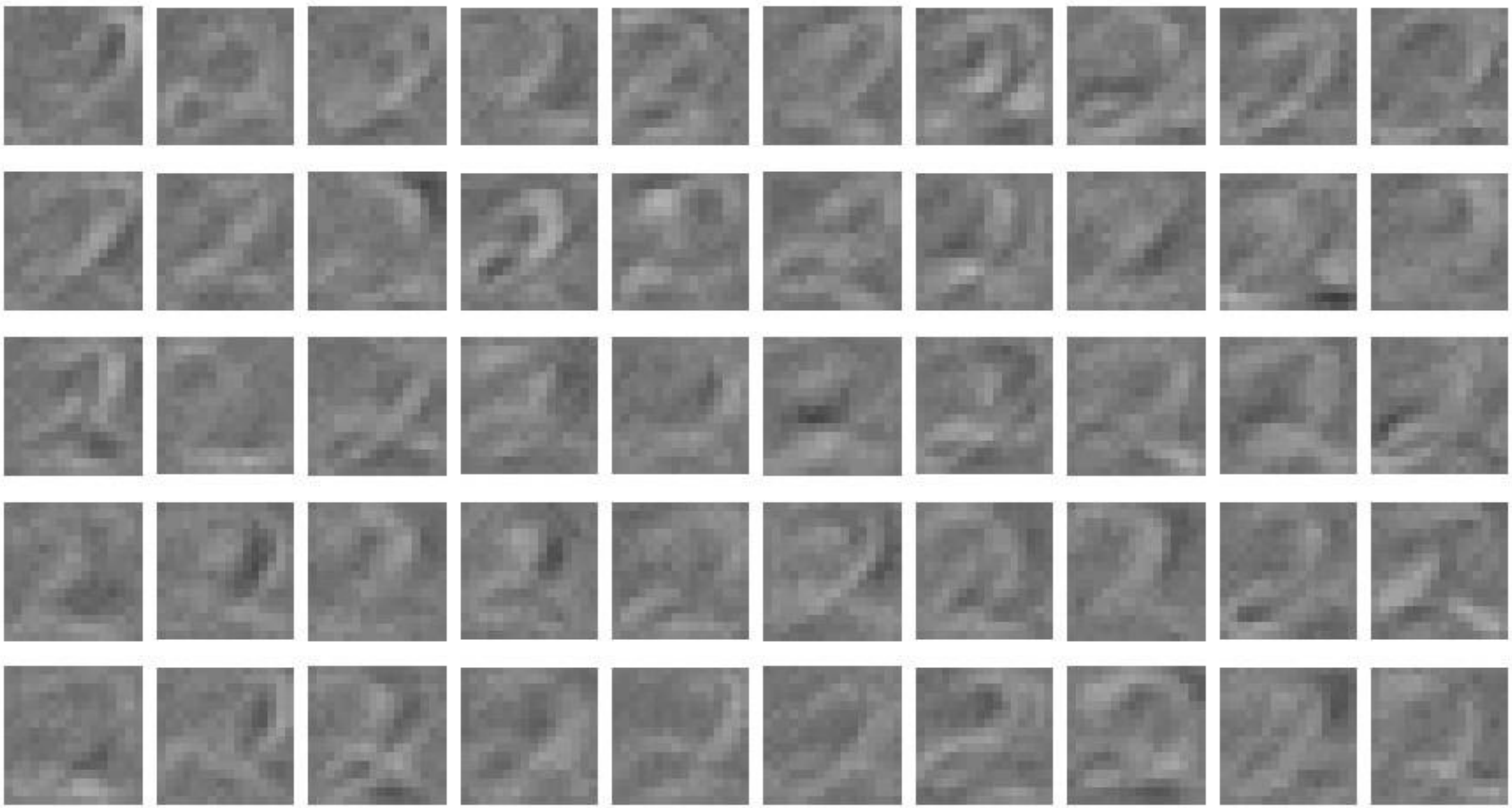
We start with small random weights to break symmetry

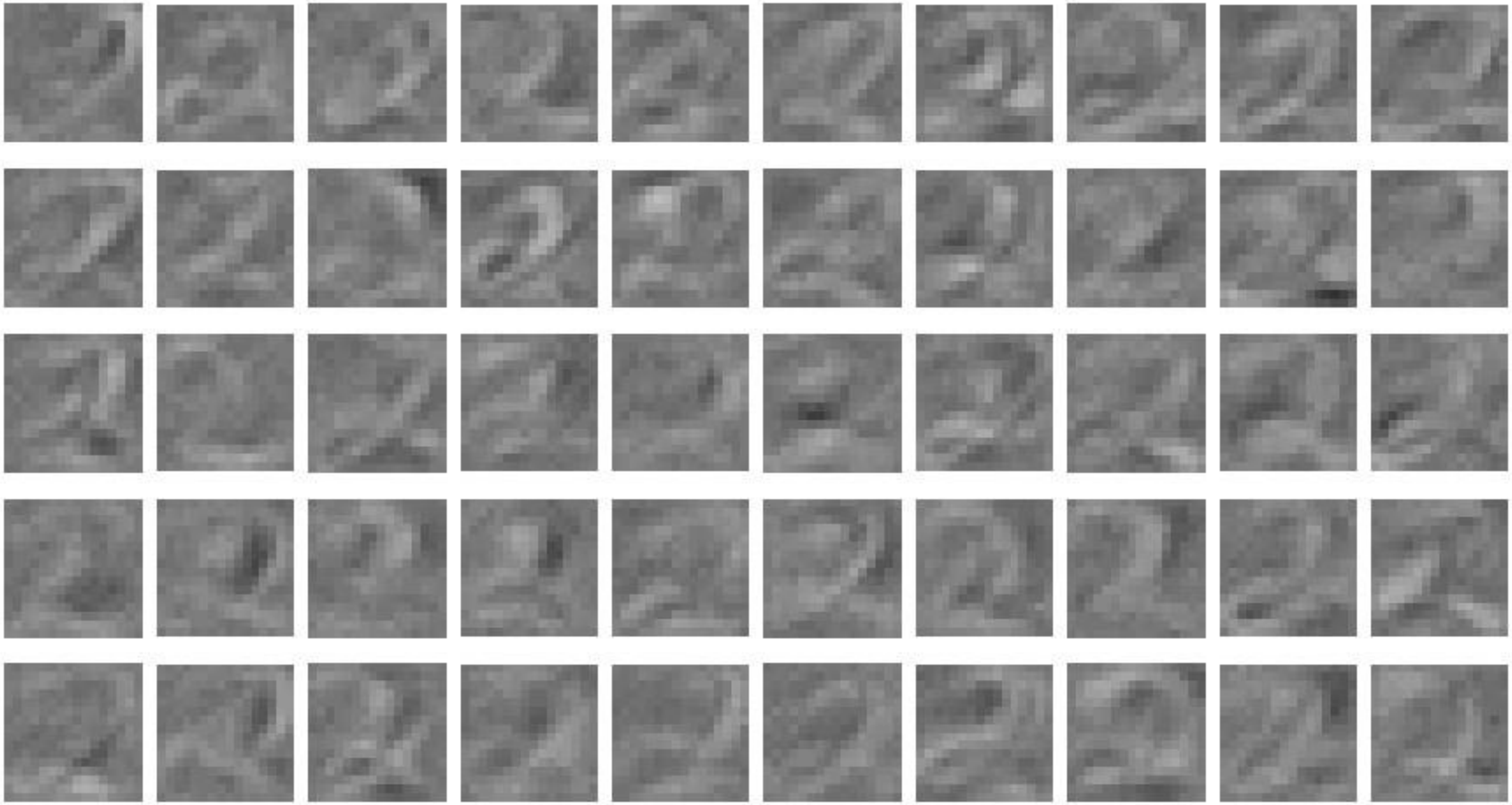


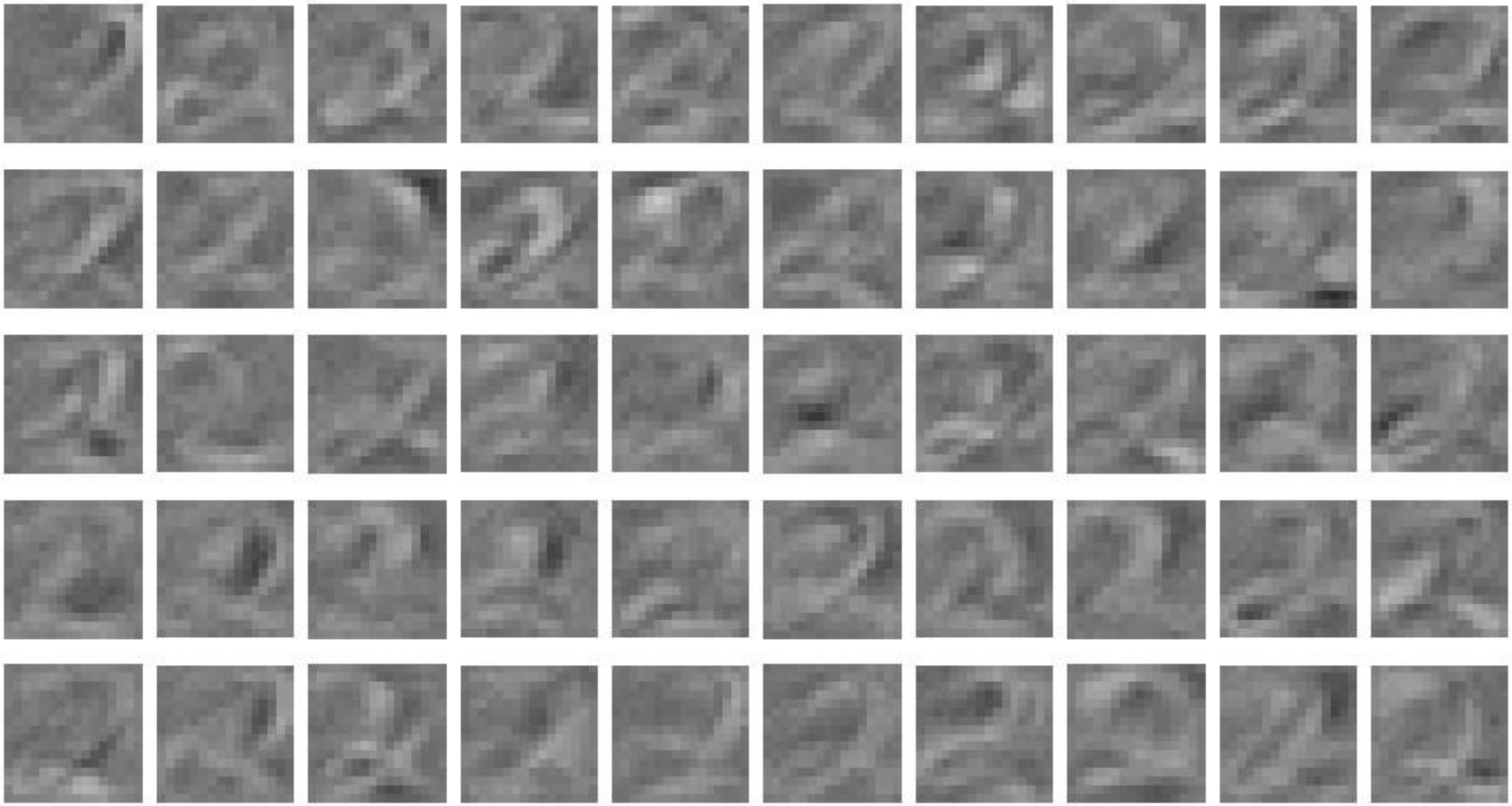


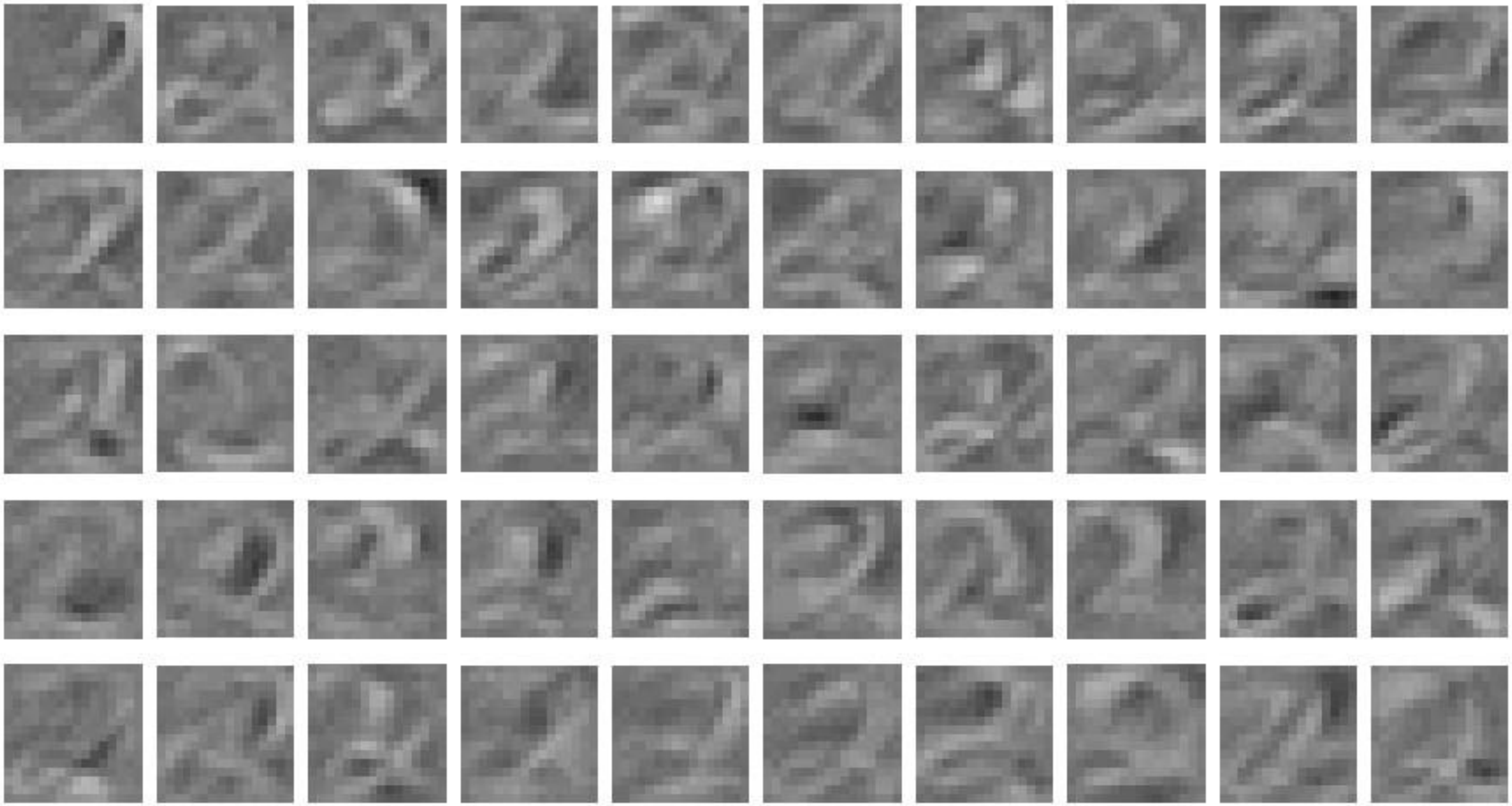


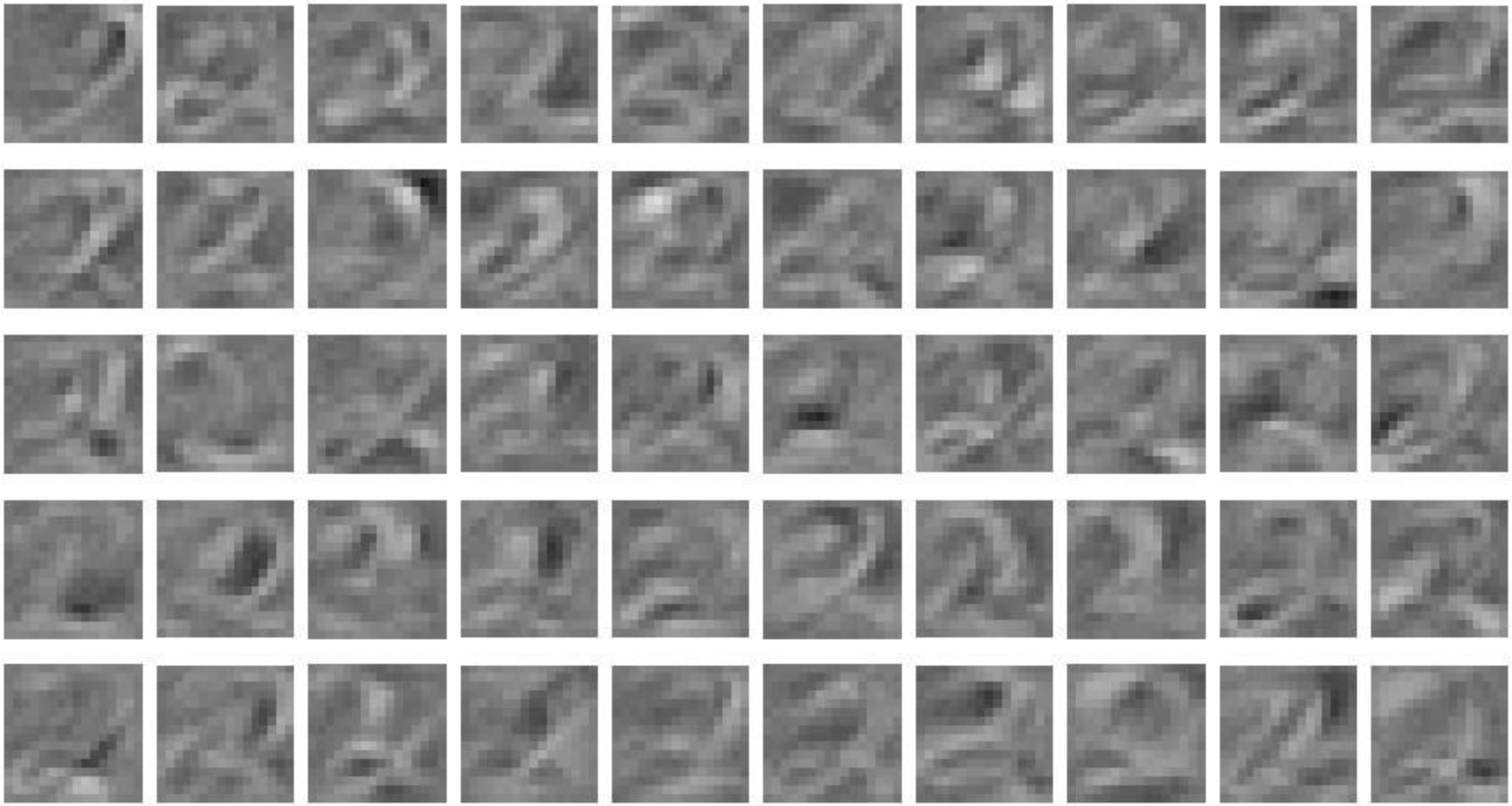


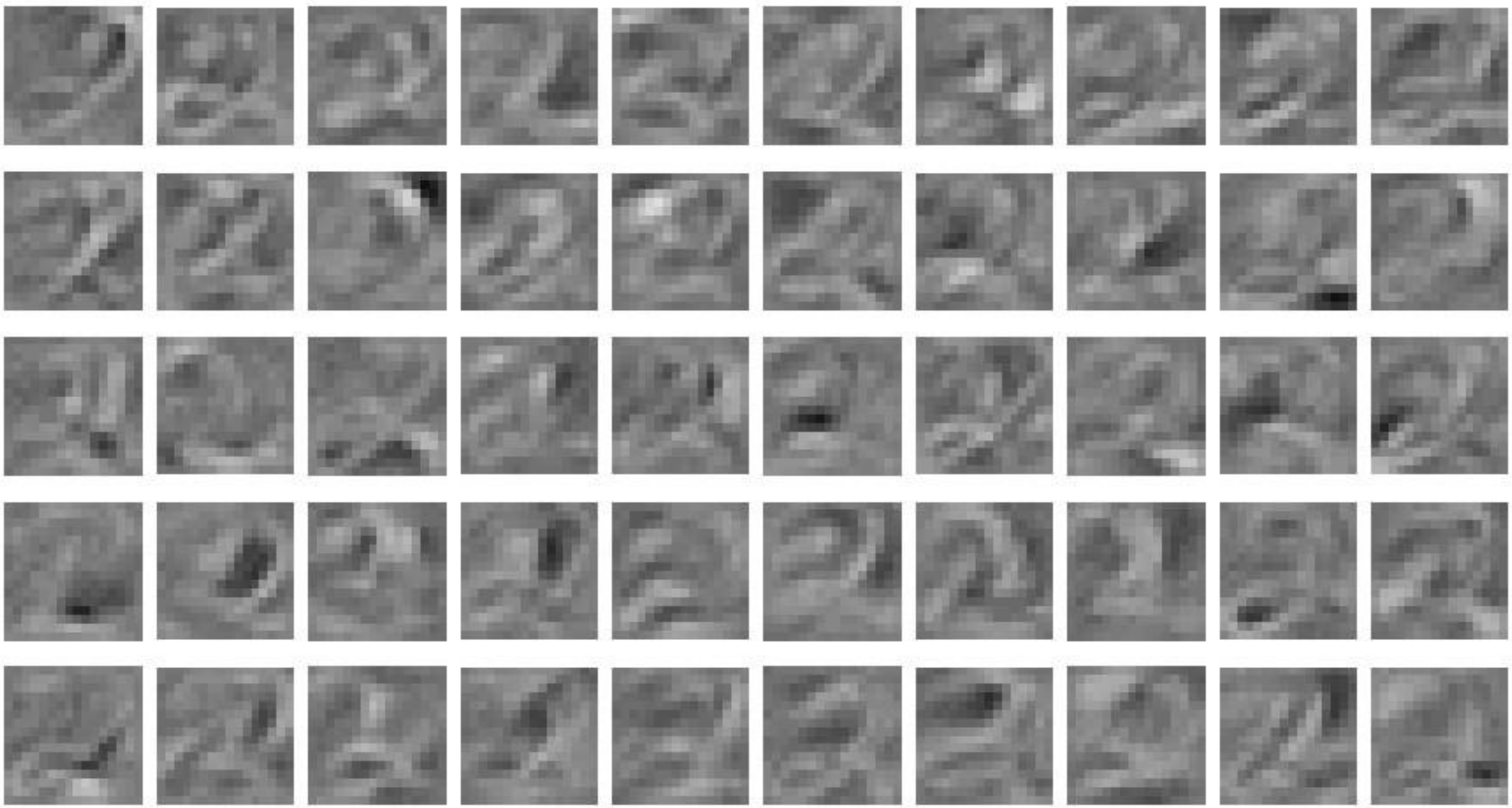


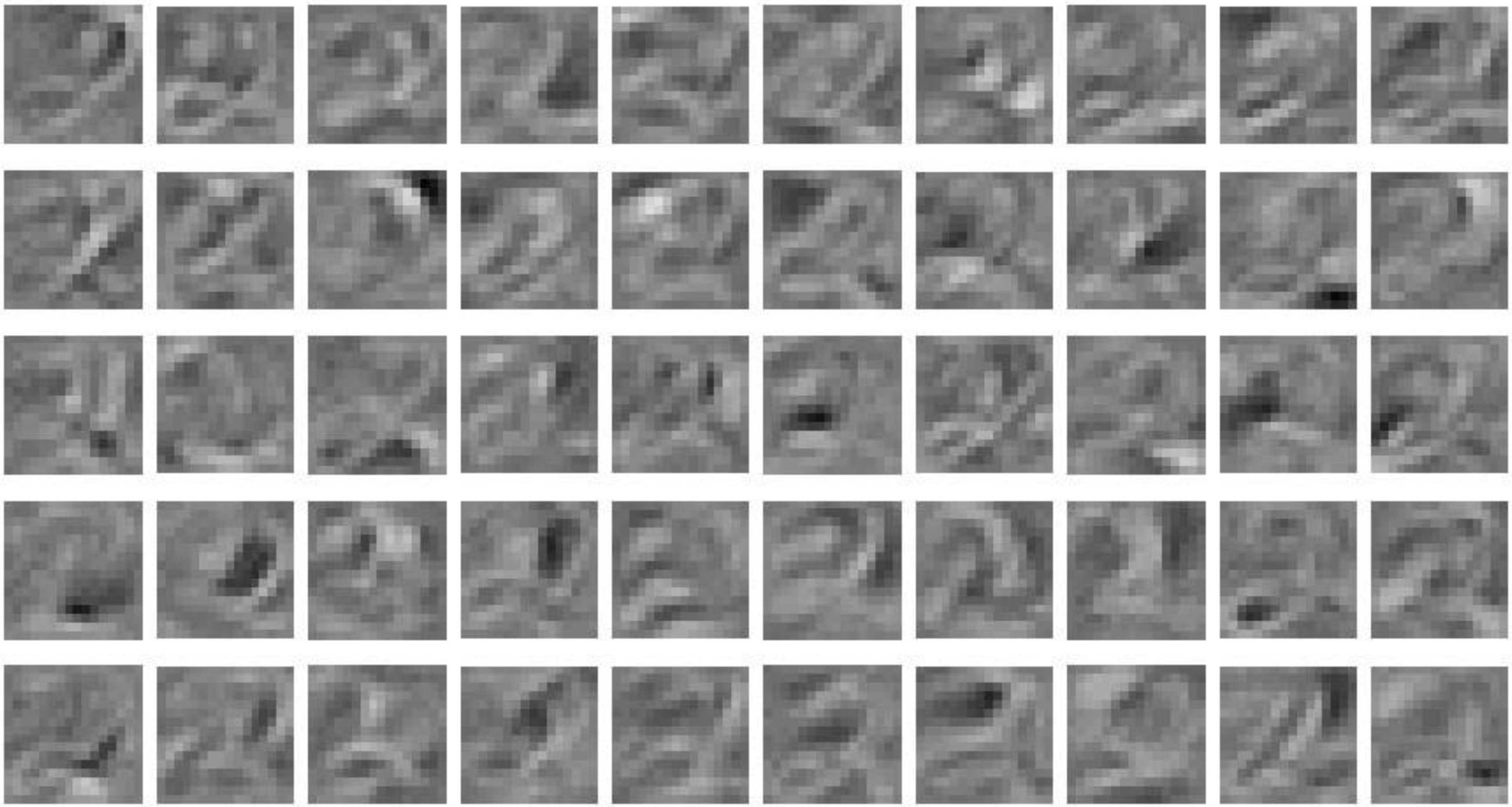


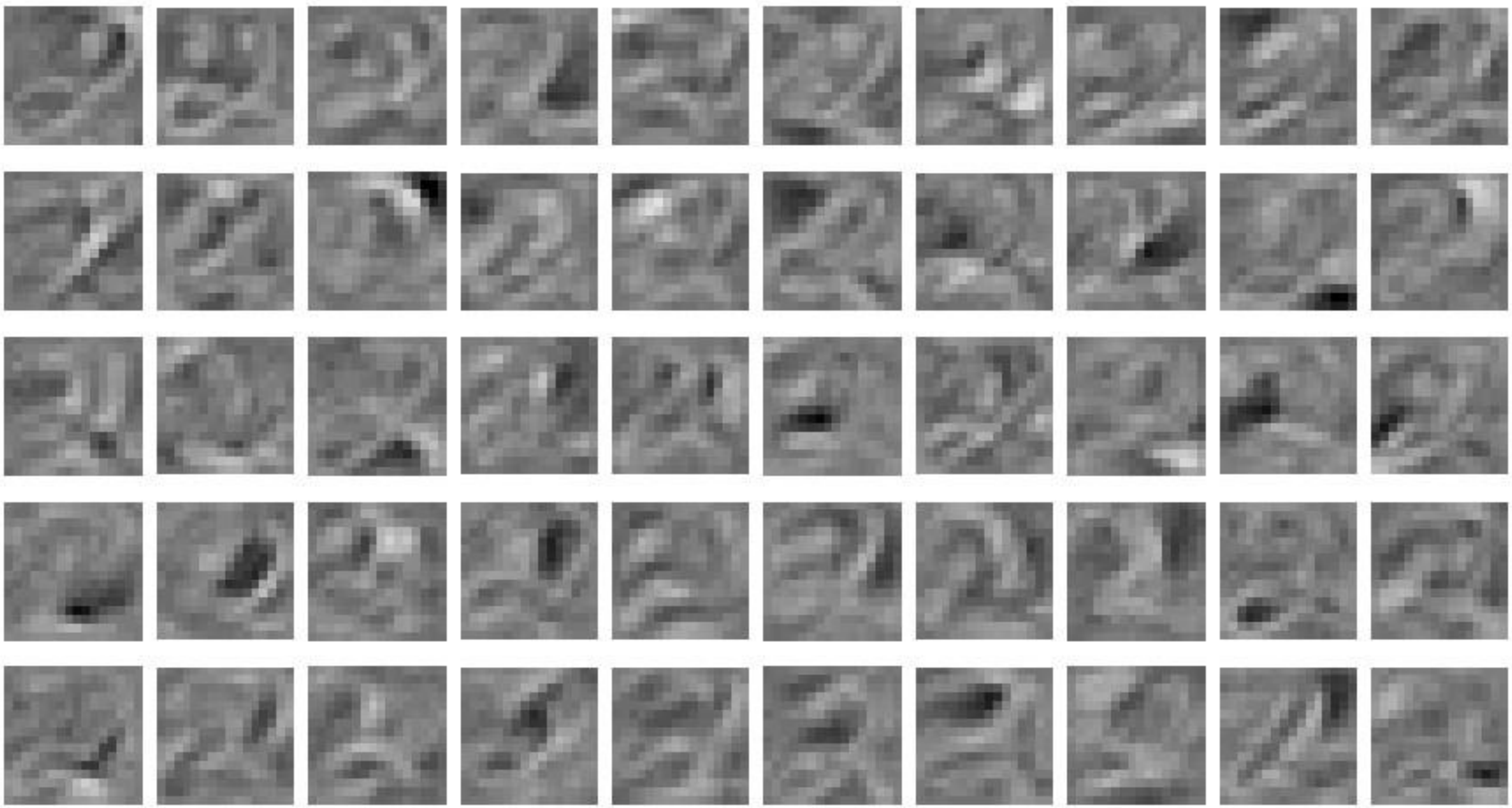


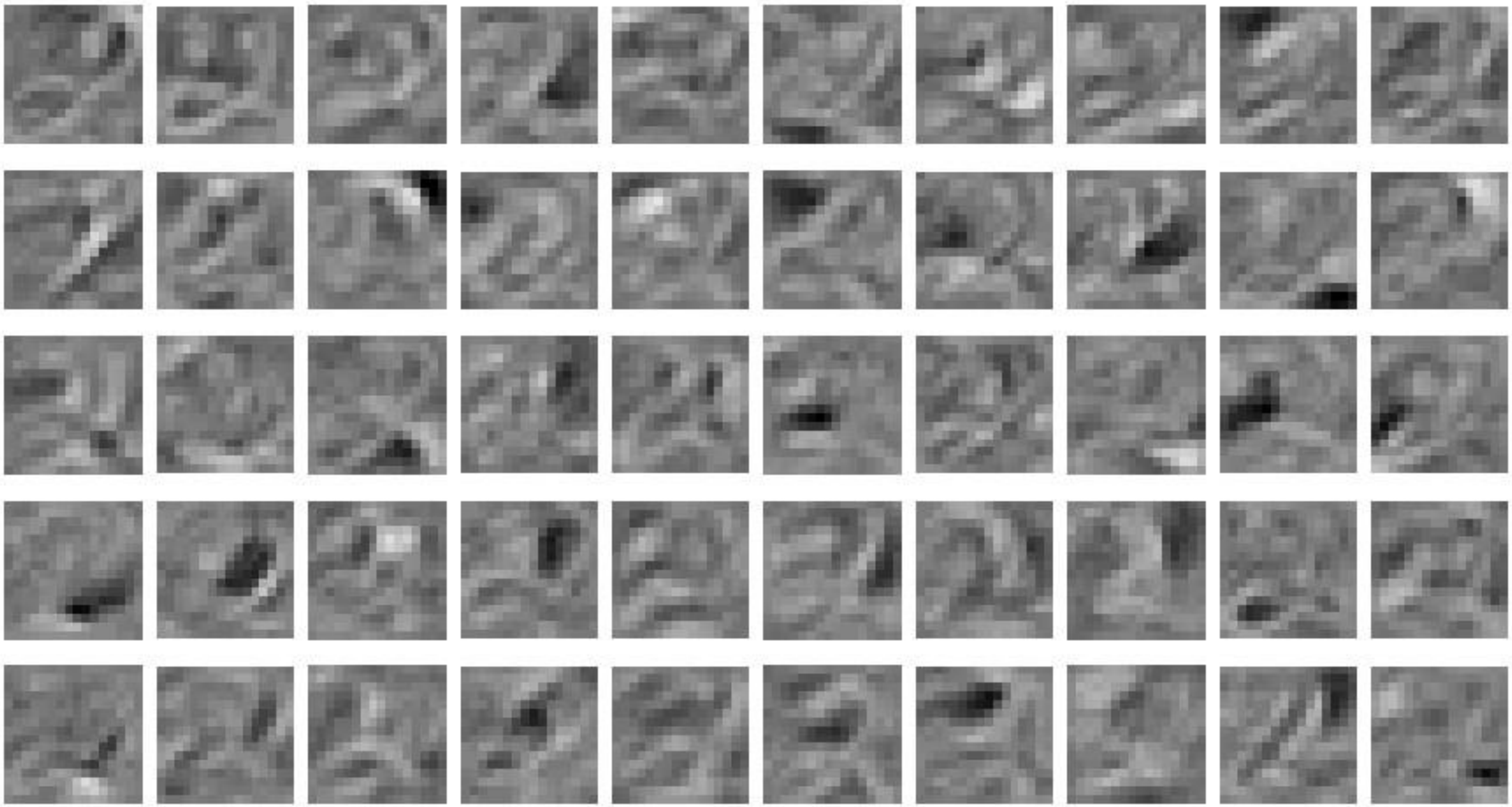


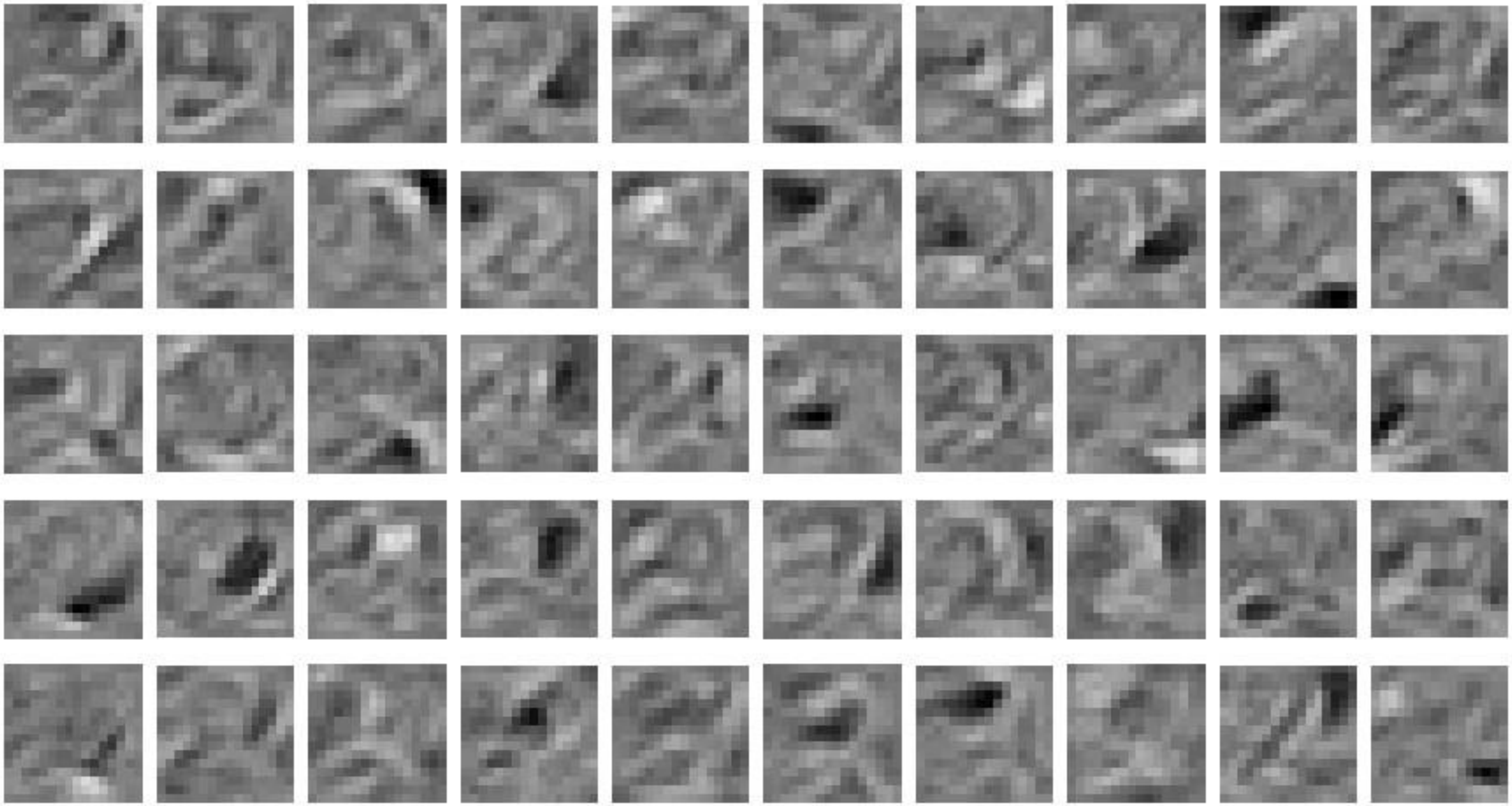


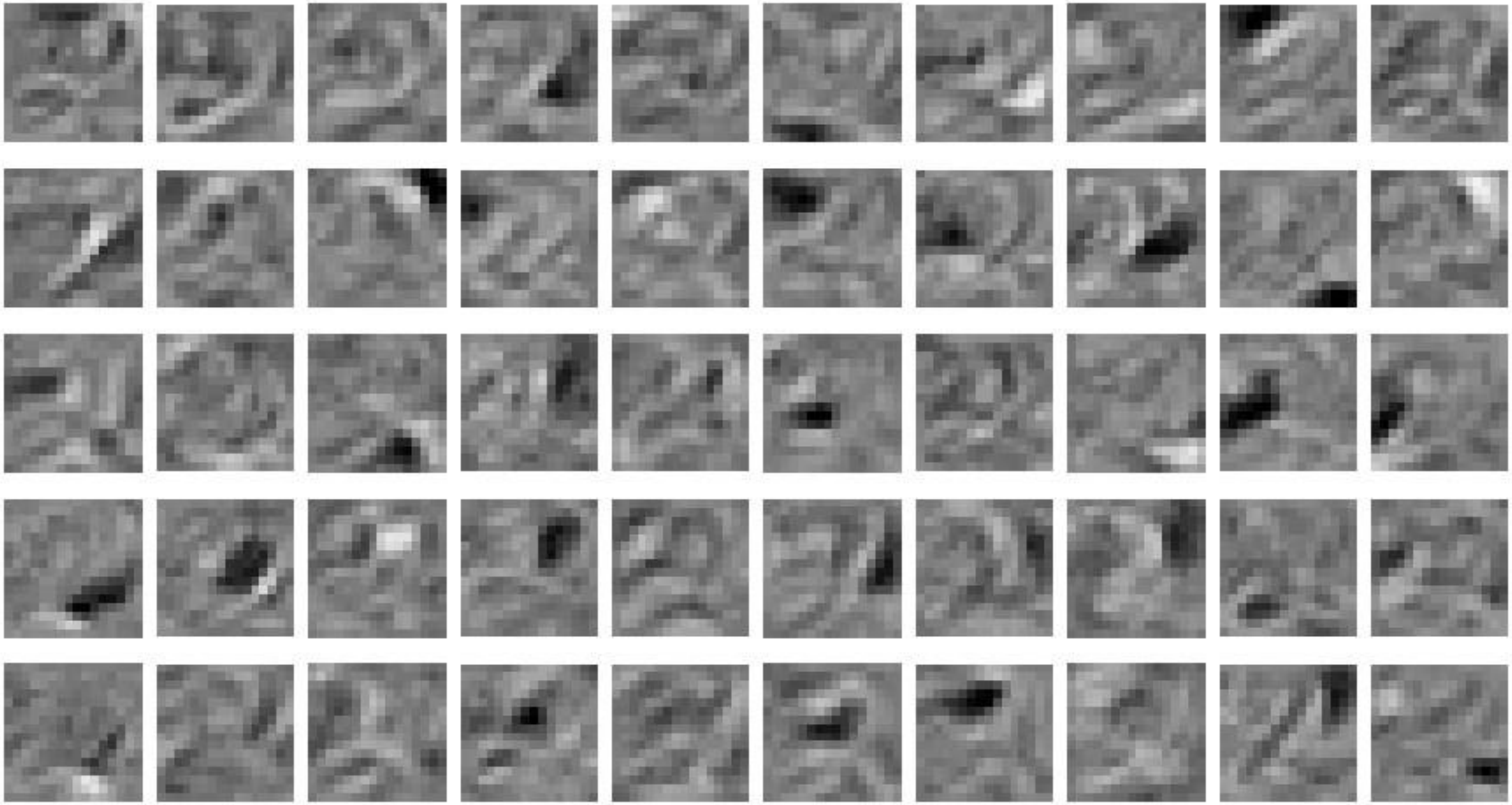


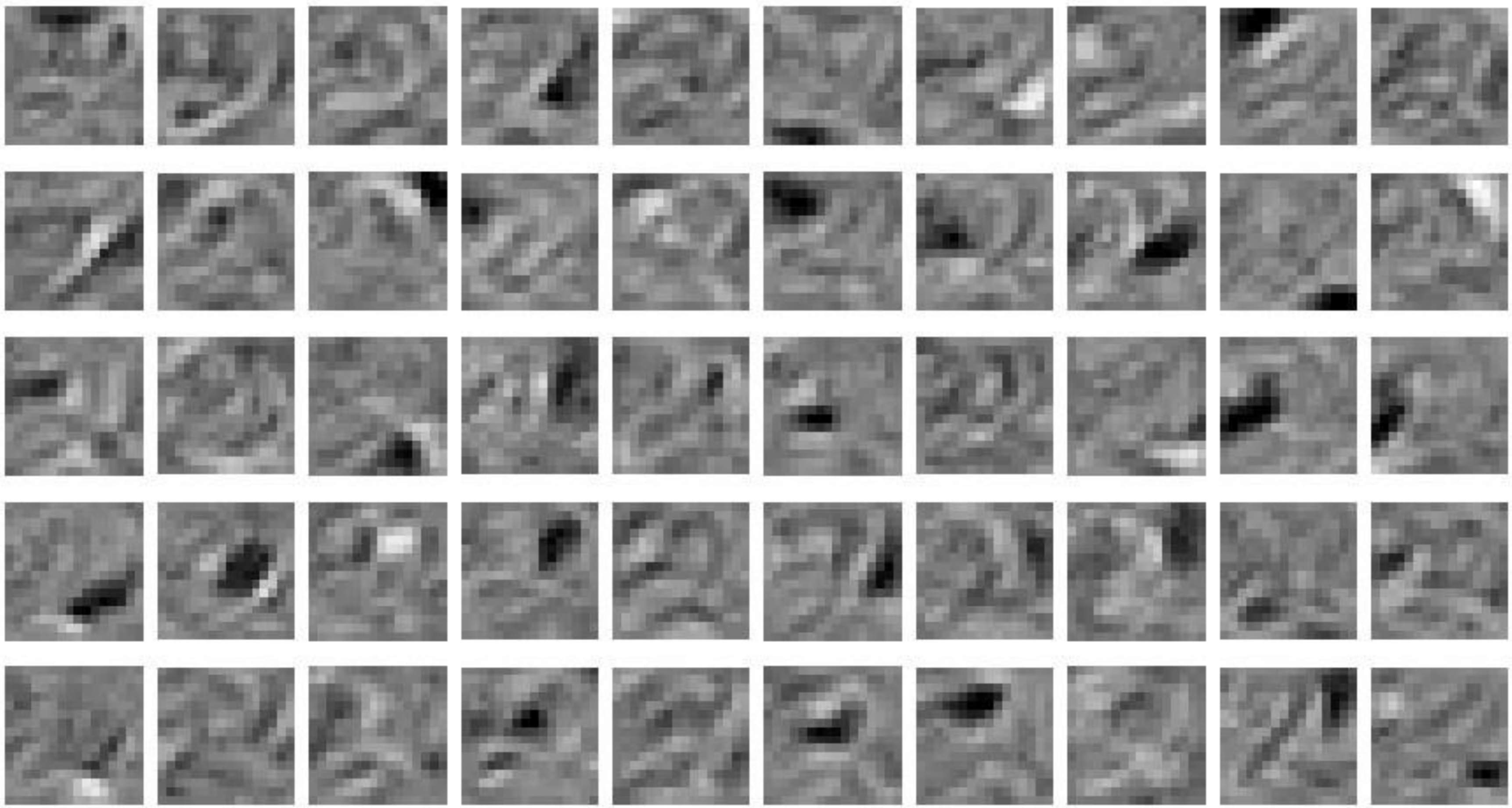




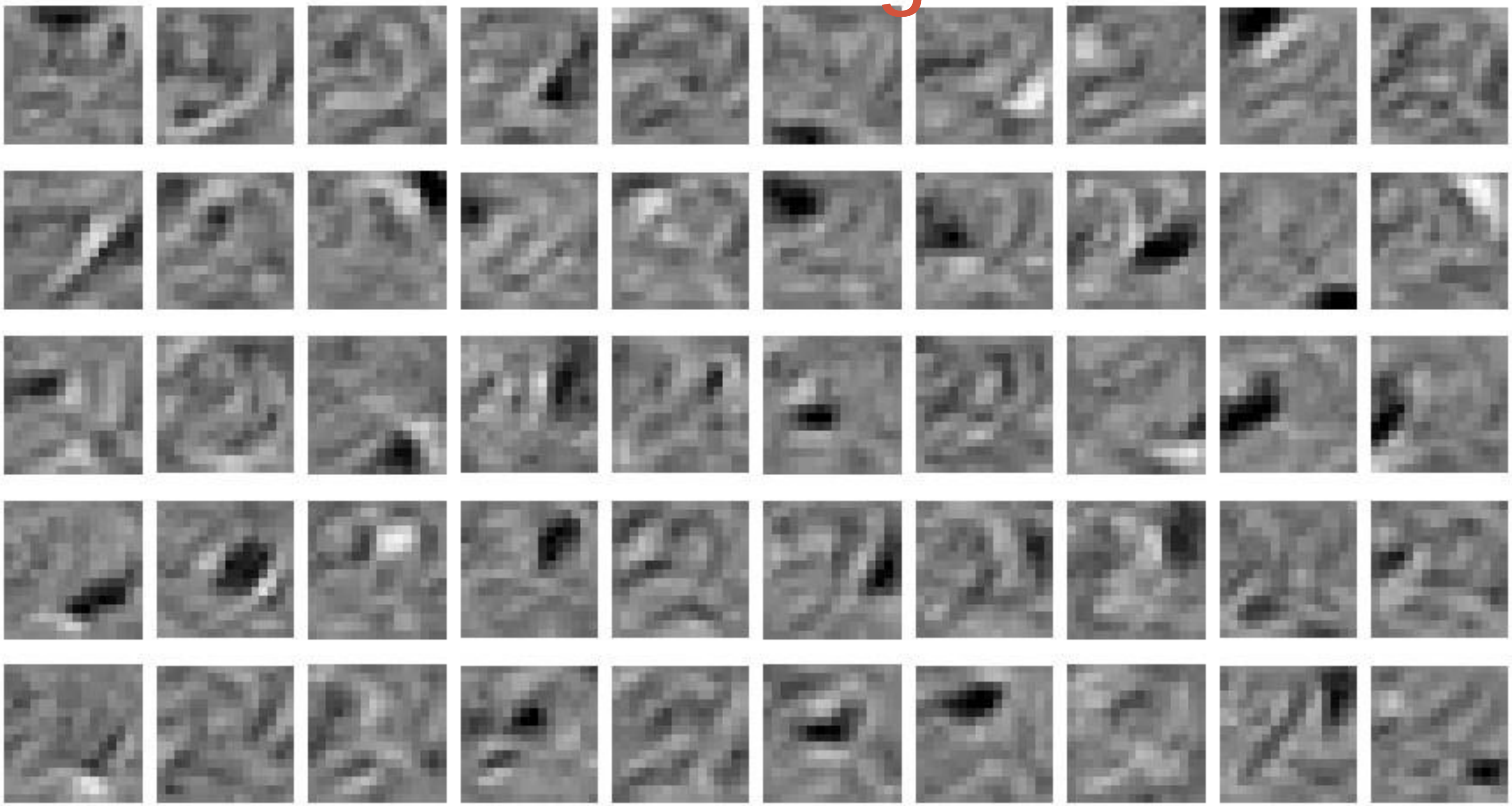








The final 50 x 256 weights



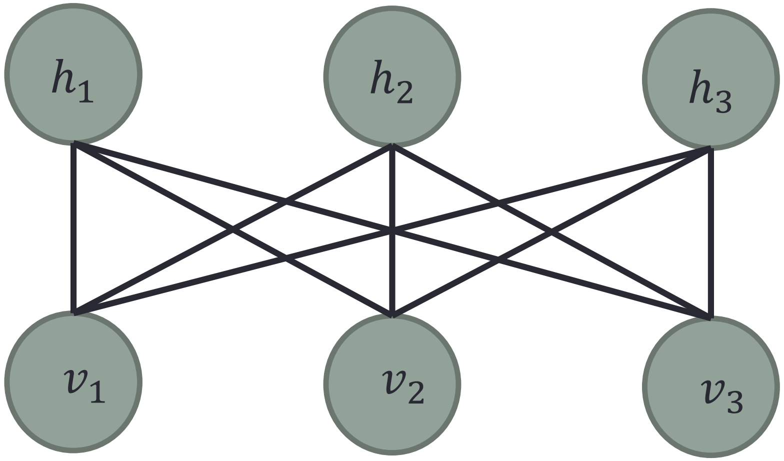
Each neuron grabs a different feature.

Efficient learning procedure for RBMs

- The probability that the network assigns to a visible vector v:
  - $p(\mathrm{v}) = \frac{1}{Z} \sum_h \exp(-E_\theta(\mathrm{v}, \mathrm{h}))$
  - $\hat{w} = \arg\max_w \sum_n \log p(\mathrm{v}^n)$

- The derivative of the log probability w. r. t. a weight:
  - $\frac{1}{N} \sum_{n=1}^{N} \frac{\partial \log p(\mathrm{v}^n)}{\partial w_{ij}} = < \mathrm{v_i h_j} >_{data} - < \mathrm{v_i h_j} >_{model}$

- Stochastic steepest ascent
  - $\Delta w_{ij} = \epsilon(< \mathrm{v_i h_j} >_{data} - < \mathrm{v_i h_j} >_{model})$



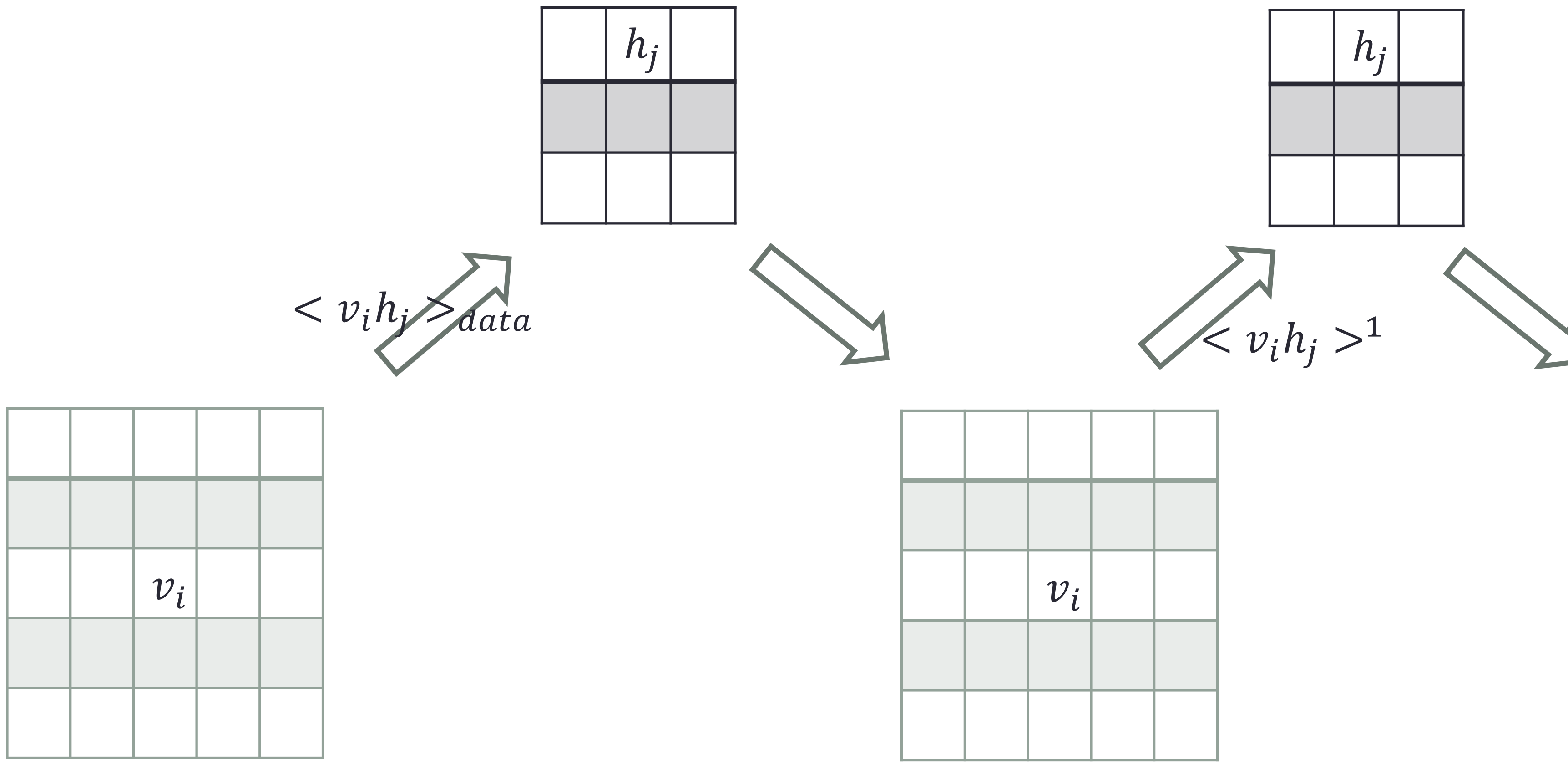
Efficient learning procedure for RBMs

- Computing posteriors
  - $p(\mathrm{h}|\mathrm{v}) = \prod_{j=1} p\left(\mathrm{h}_j \middle| \mathrm{v}\right)$
  - $p(\mathrm{v}|\mathrm{h}) = \prod_{i=1} p(\mathrm{v}_i|h)$

- The absence of direct connections between hidden units in an RBM allows us to have
  - $p\left(h_j = 1 \middle| \mathrm{v}\right) = \sigma(d_j + \sum w_{ij} v_i)$

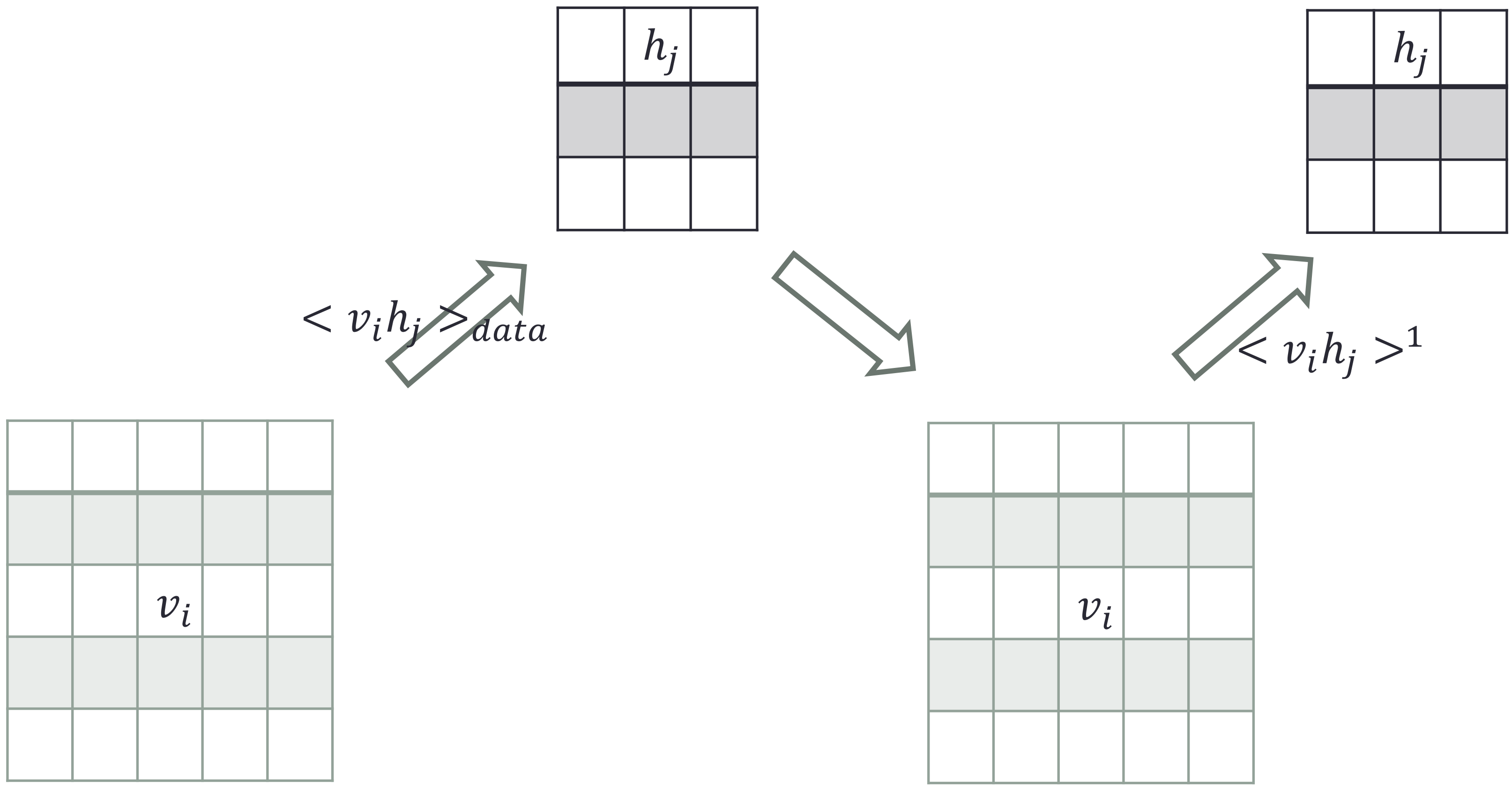
- The absence of direct connects between visible units in an RBM allows us to have
  - $p(v_i = 1|\mathrm{h}) = \sigma(b_i + \sum w_{ij} h_j)$

Efficient learning procedure for RBMs



$$\Delta w_{ij} = \epsilon\left(< v_i h_j >_{data} - < v_i h_j >_{model}\right)$$
$$= \epsilon(< v_i h_j >_{data} - < v_i h_j >^{\infty})$$

Efficient learning procedure for RBMs



$$
\begin{aligned}
\Delta w_{ij} &= \epsilon\big(< v_i h_j >_{data} - < v_i h_j >_{model}\big) \\
&= \epsilon(< v_i h_j >_{data} - < v_i h_j >^{\infty}) \\
&\simeq \epsilon(< v_i h_j >_{data} - < v_i h_j >^{1})
\end{aligned}
$$

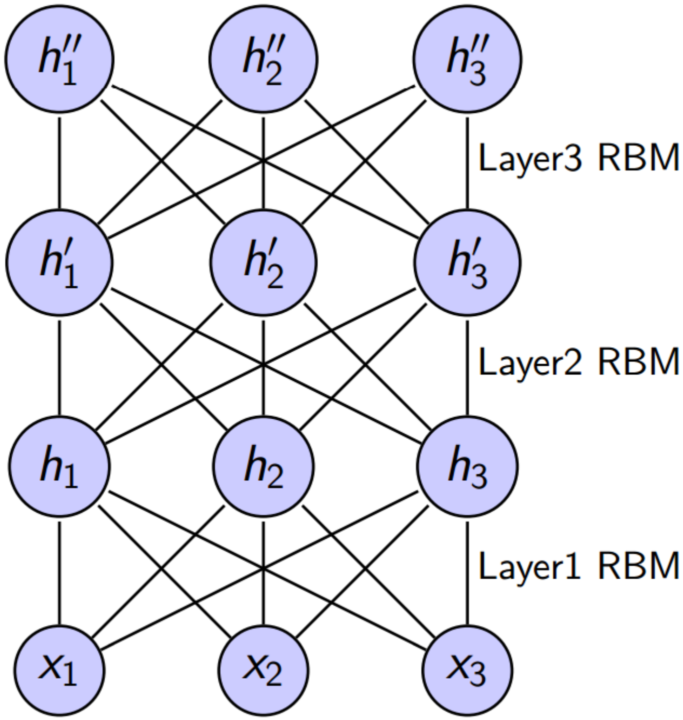
Training RBMs with Contrastive Divergence

- The term $< \mathrm{v_i h_j} >_{model}$ is expensive because it requires sampling (v, h) from the model
- Gibbs sampling (sample v then h iteratively) works, but waiting for convergence at each gradient step is slow.
- Contrastive Divergence is faster: initialize with training point and wait only a few (usually 1) sampling steps
  - Let v be a training point.
  - Sample $\widehat{h_j} \in \{0,1\}$ from $p\left(h_j = 1 \middle| \mathrm{v}\right) = \sigma(d_j + \sum w_{ij} v_i)$
  - Sample $\breve{v_i} \in \{0,1\}$ from $p\left(v_i = 1 \middle| \hat{h}\right) = \sigma(b_i + \sum w_{ij} \widehat{h_j})$
  - Sample $\breve{h_j} \in \{0,1\}$ from $p\left(h_j = 1 \middle| \breve{v}\right) = \sigma\left(b_i + \sum w_{ij} \breve{v_i}\right)$
  - $w_{ij} \leftarrow w_{ij} + \epsilon(v_i \widehat{h_j} - \breve{v_i} \breve{h_j})$

DEEP BELIEF NETWORK

Ruslan Salakhutdinov and Geoffrey Hinton,
"Deep Boltzmann Machine," 2009

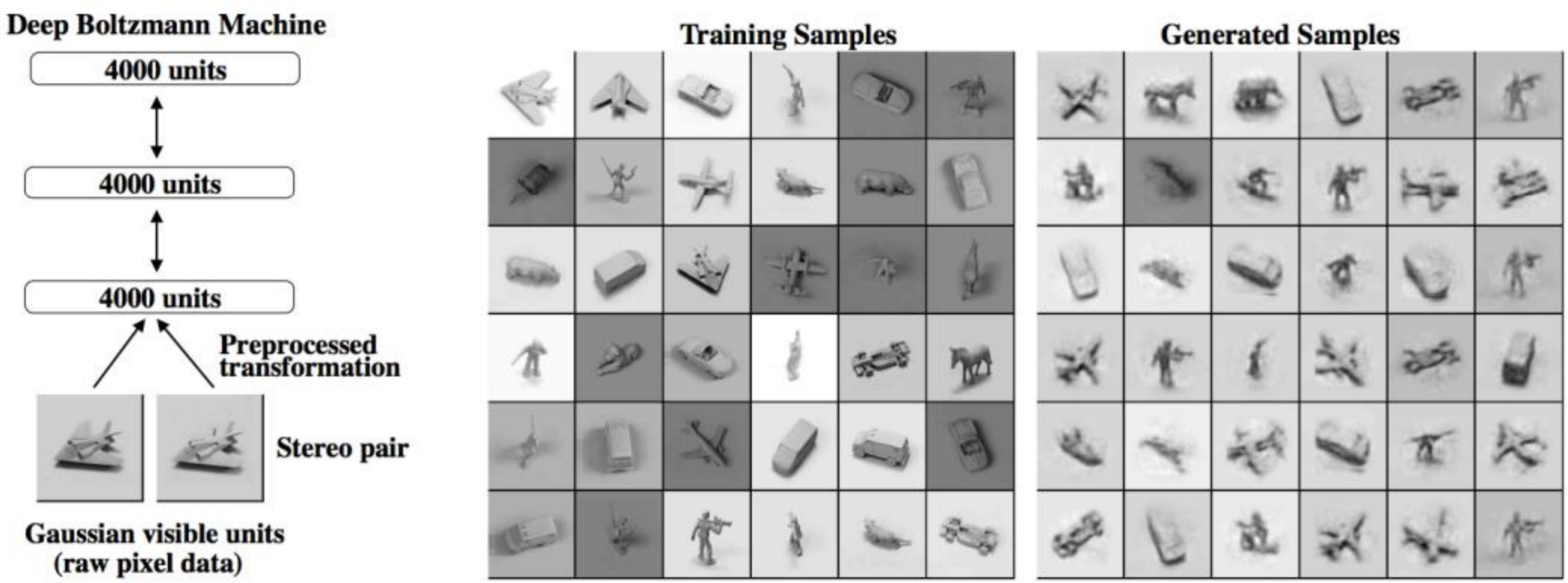
Deep Belief Nets (DBN) = Stacked RBM



- DBN defines a probabilistic generative model
  - $p(x) = \sum_{h,h',h''} p(x|h)p(h|h')\,p(h', h'')$
- Stacked RBMs can also be used to initialize a Deep Neural Network (DNN).

Generating Data from a Deep Generative Model

- After training on 20k images, the generative model of can generate random mages (dimension=8976) that are amazingly realistic!

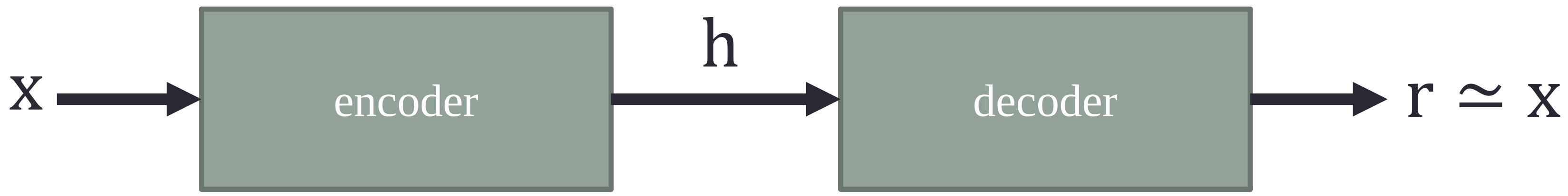


DEEP AUTOENCODER

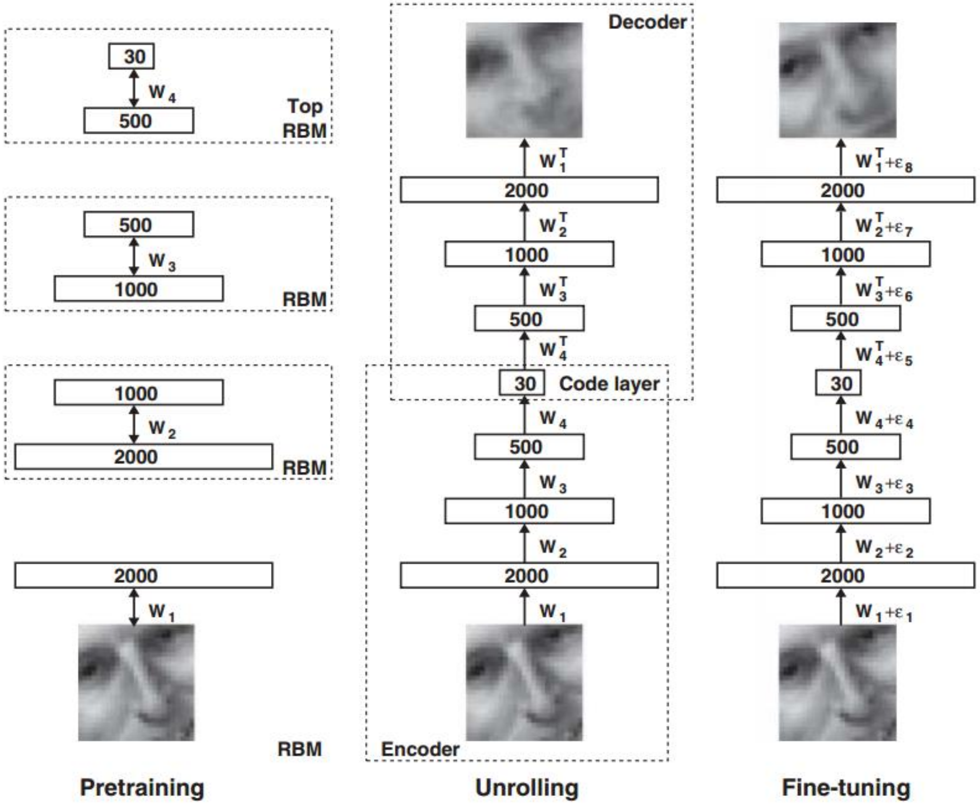
Hinton and Salakhutdinov, “Reducing the Dimensionality of Data with Neural Networks,” Science, 2006

Auto-encoder

- An auto-encoder is trained to encode the input in some representation so that the input can be reconstructed from that representation. Hence the target output is the input itself



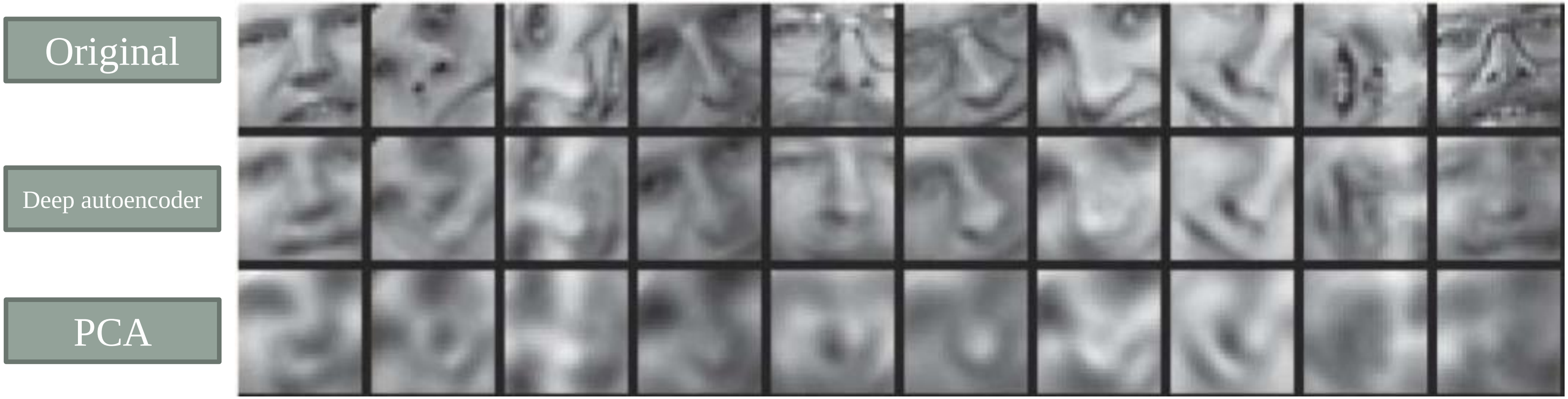
Deep auto-encoder



- Pre-training consists of learning **a stack of RBMs**, each having only one layer of feature detectors. The learned feature activations of one RBM are used as the "data" for training the next RBM in the stack. After the pre-training, the RBMs are "unrolled" to create a deep auto-encoder, which is then fine-tuned using **back-propagation** of error derivatives.

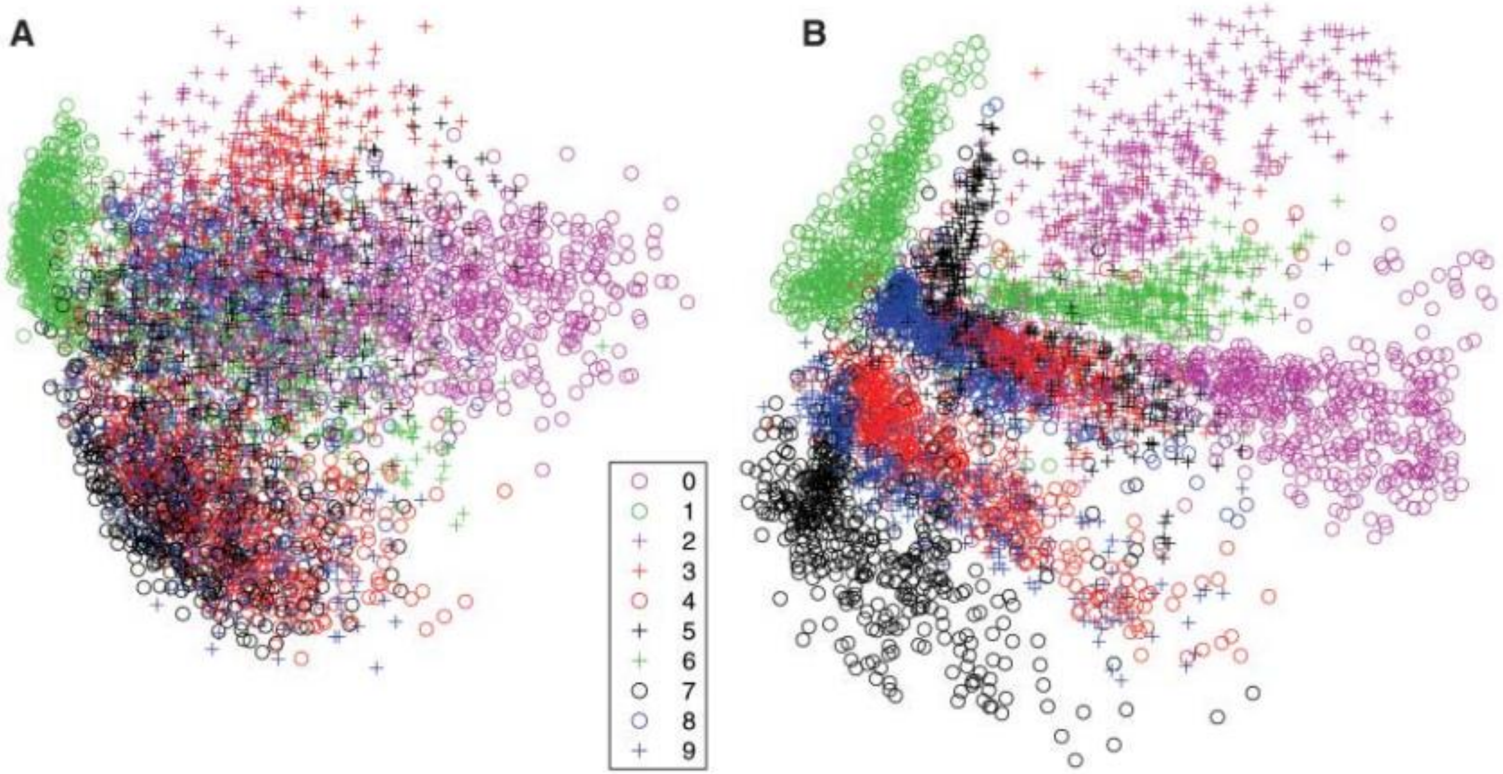
Deep auto-encoder

- Can be used to reduce the dimensionality of the data
  - Better reconstruction than PCA



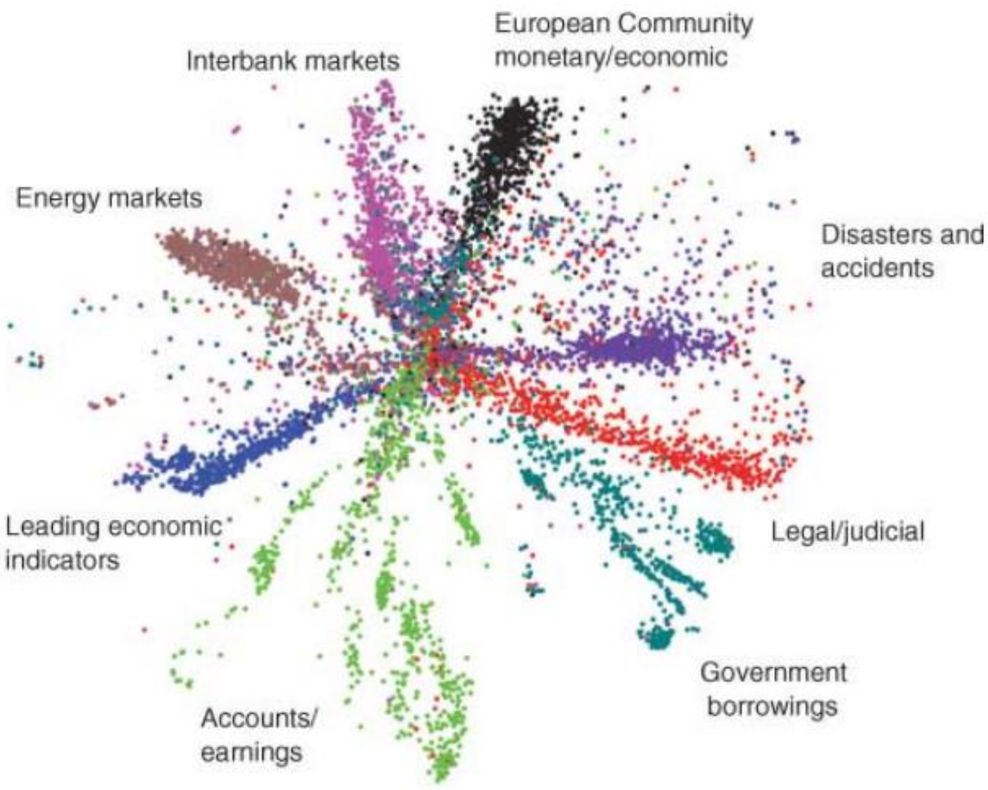
Deep auto-encoder (digit data)

- PCA
- Deep auto-encoder
  - 784-1000-500-250-2



Deep auto-encoder (documents)

- Visualization of the data (dimension reduction to 2D)
  - 2000-500-250-125-2



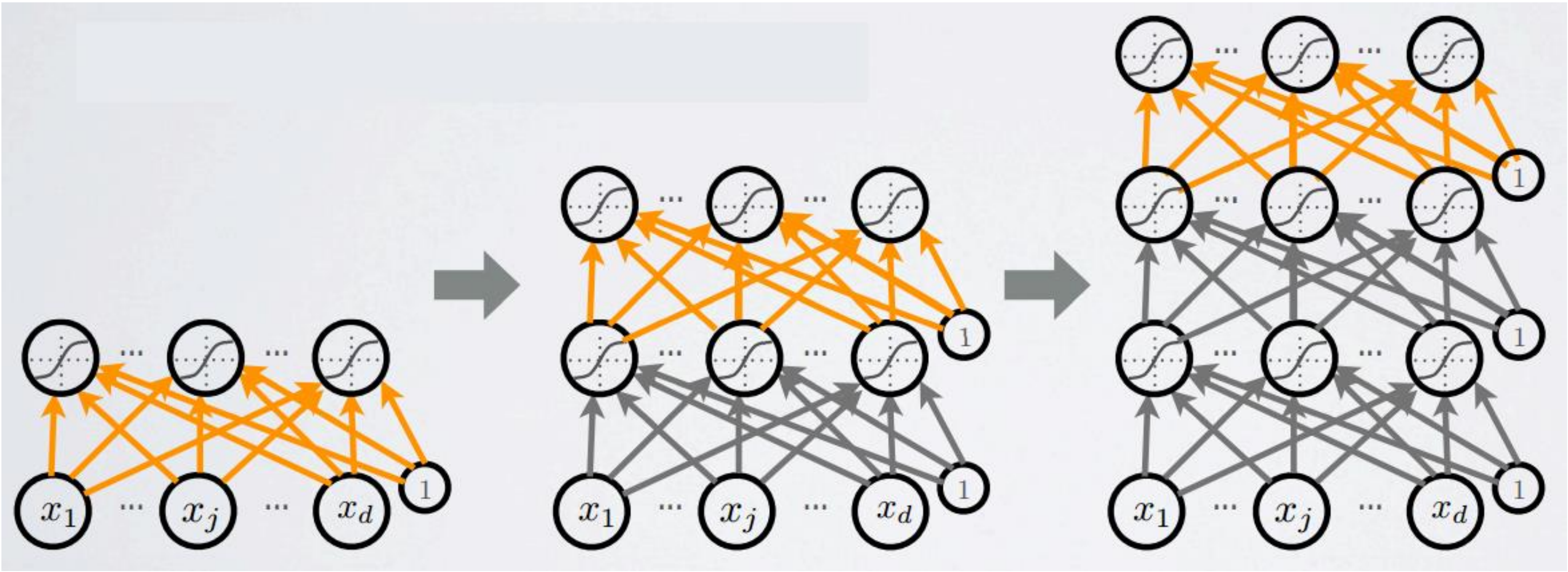
Summary

- Layer-wise pre-training is the innovation that rekindled interest in deep architectures.
- Pre-training focuses on optimizing likelihood on the data, not the target label. First model $p(\mathrm{v})$ to do better $p(\mathrm{y}|\mathrm{v})$.
- Why RBM? $p(\mathrm{h}|\mathrm{x})$ is tractable, so it's easy to stack.
- RBM training can be expensive.
  - Solution: contrastive divergence
- DBN formed by stacking RBMs is a probabilistic generative mode

RBM + SUPERVISED LEARNING

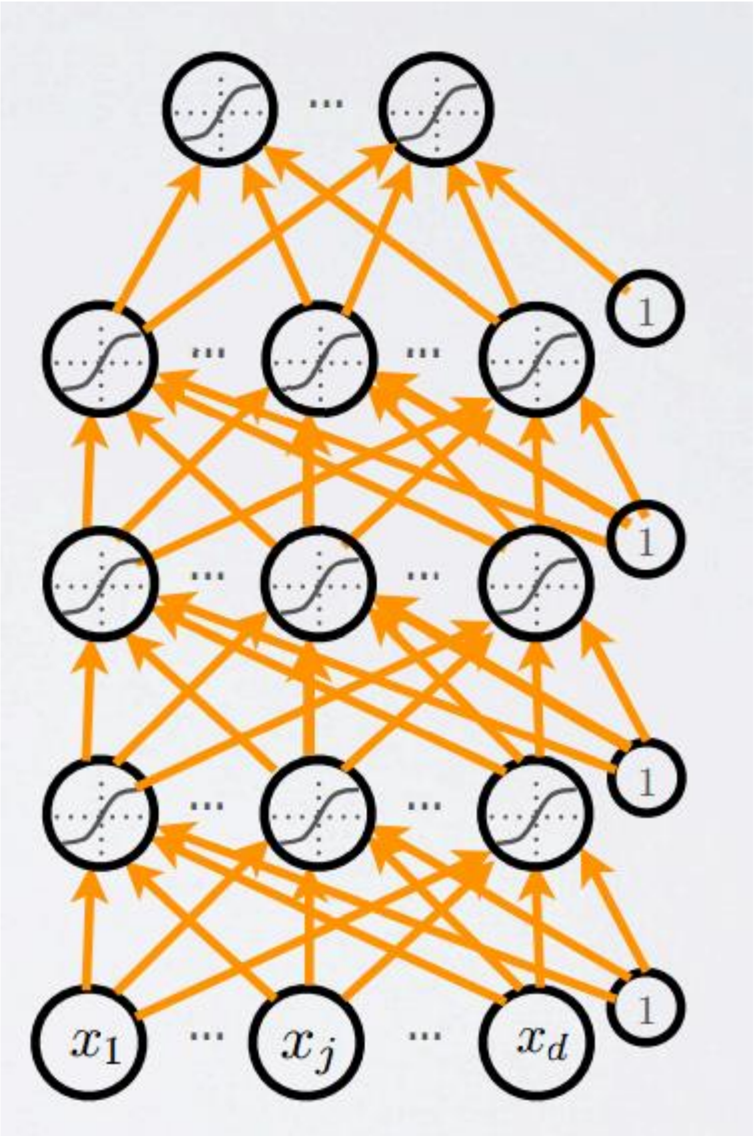
UNSUPERVISED PRE-TRAINING

- We will use a greedy, layer-wise procedure
  - train one layer at a time, from first to last, with unsupervised criterion
  - fix the parameters of previous hidden layers
  - previous layers viewed as feature extraction



FINE-TUNING

- Once all layers are pre-trained
  - add output layer
  - train the whole network using supervised learning
- Supervised learning is performed as in a regular feed-forward network
  - forward propagation, backpropagation and update
- We call this last phase fine-tuning
  - all parameters are "tuned" for the supervised task at hand
  - representation is adjusted to be more discriminative



ONE LEARNING ALGORITHM HYPOTHESIS?
GRANDMOTHER CELL HYPOTHESIS?

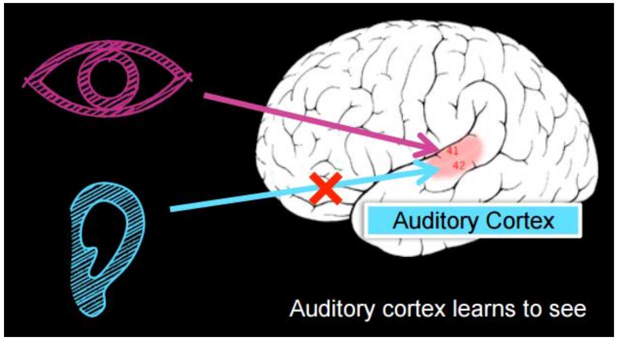
THERE'S A THEORY that human intelligence stems from a single algorithm.

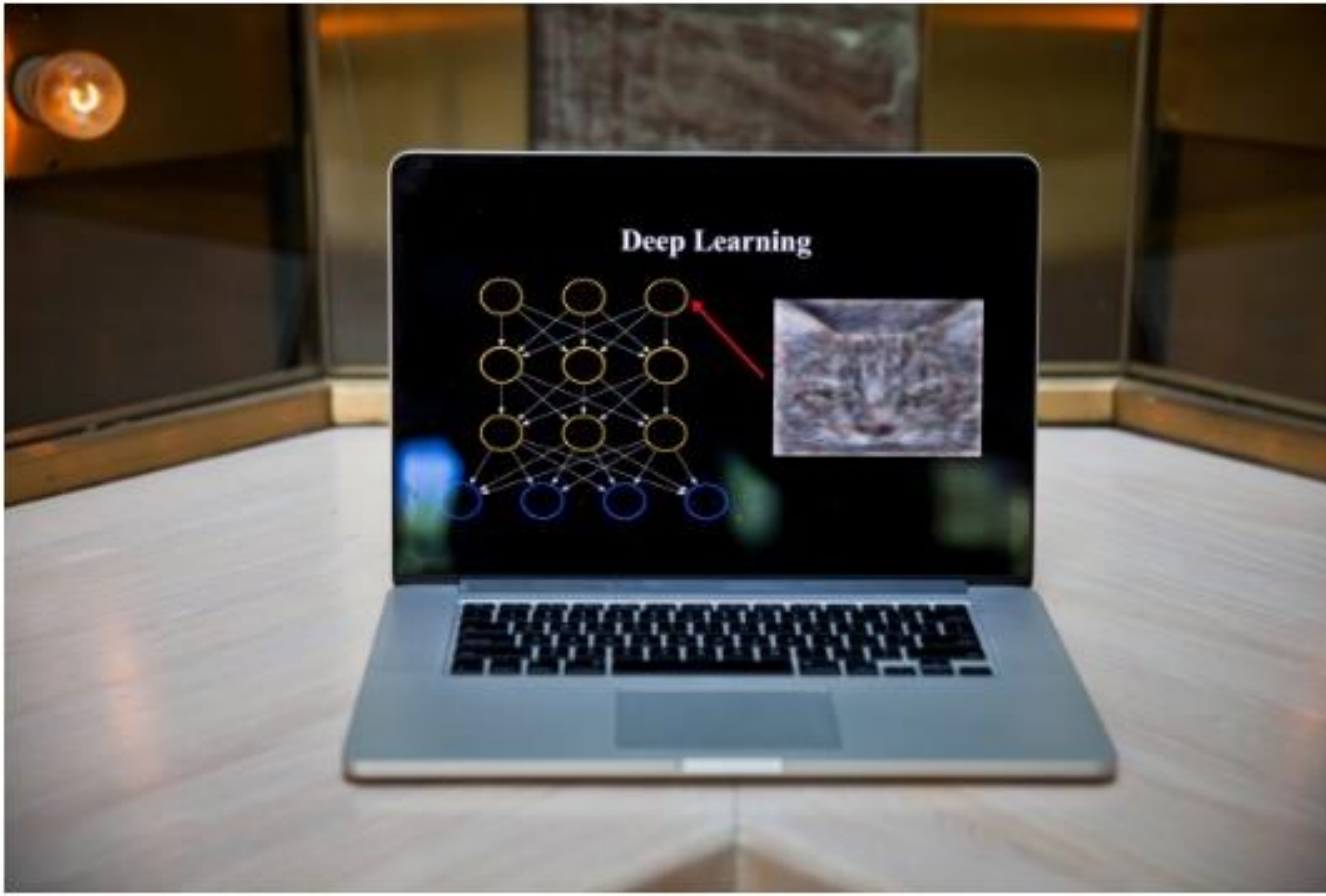
The idea arises from experiments suggesting that the portion of your brain dedicated to processing sound from your ears could also handle sight for your eyes. This is possible only while your brain is in the earliest stages of development, but it implies that the brain is — at its core — a general-purpose machine that can be tuned to specific tasks.

About seven years ago, Stanford computer science professor Andrew Ng stumbled across this theory, and it changed the course of his career, reigniting a passion for artificial intelligence, or AI. "For the first time in my life," Ng says, "it made me feel like it might be possible to make some progress on a small part of the AI dream within our lifetime."

THE MAN BEHIND THE GOOGLE BRAIN: ANDREW NG AND THE QUEST FOR THE NEW AI

Stanford professor Andrew Ng, the man at the center of the Deep Learning movement.
Photo: Ariel Zambelich/Wired





Andrew Ng's laptop explains Deep Learning. *Photo: Ariel Zambelich/Wired*

Google's Artificial Brain Learns to Find Cat Videos

BY WIRED UK 06.26.12 | 11:15 AM | PERMALINK

By Liat Clark, Wired UK

When computer scientists at Google's mysterious X lab built a neural network of 16,000 computer processors with one billion connections and let it browse YouTube, it did what many web users might do — it began to look for cats.

BUILDING HIGH-LEVEL FEATURES USING LARGE SCALE UNSUPERVISED LEARNING

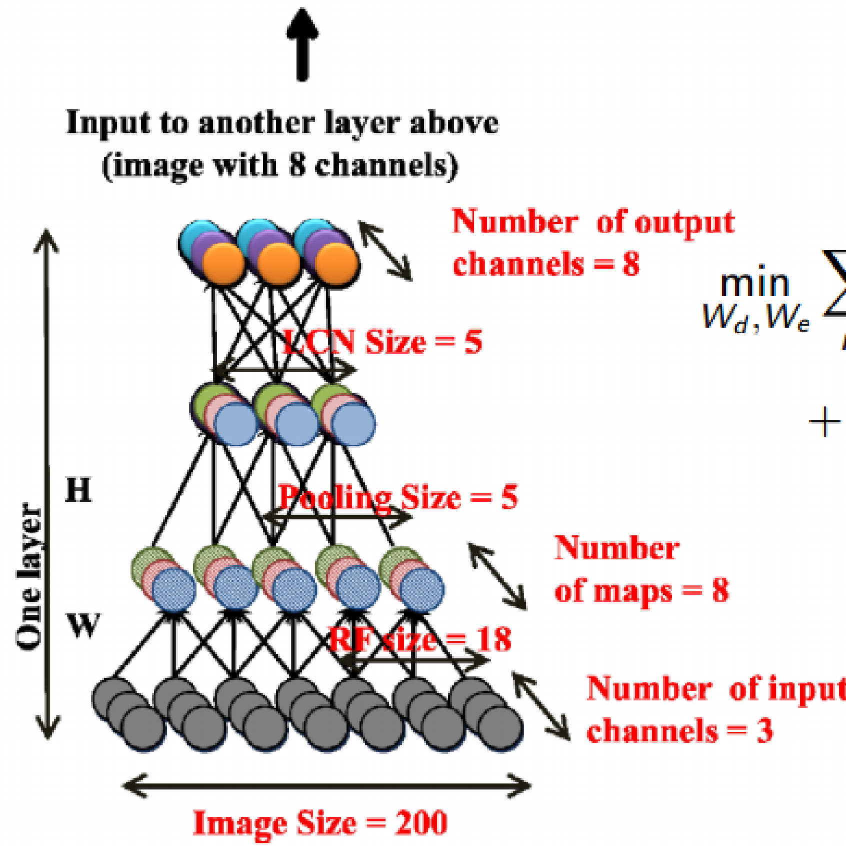
Quoc V. Le et al, ICML, 2012

Motivating Question

- Is it possible to learn high-level features (e.g. face detectors) using only unlabeled images?

- "Grandmother Cell" Hypothesis
  - Grandmother cell: A neuron that lights up when you see or hear your grandmother
    - Lots of interesting (controversial) discussions in the neuroscience literature

Architecture (≃sparse deep auto-encoder)



$$\min_{W_d, W_e} \sum_m ||W_d W_e x^{(m)} - x^{(m)}|| \quad (1)$$

$$+ \sum_{m,k} \sqrt{\epsilon + P_k (W_e x^{(m)})^2} \quad (2)$$

(1): auto-encoder
(2): pooling

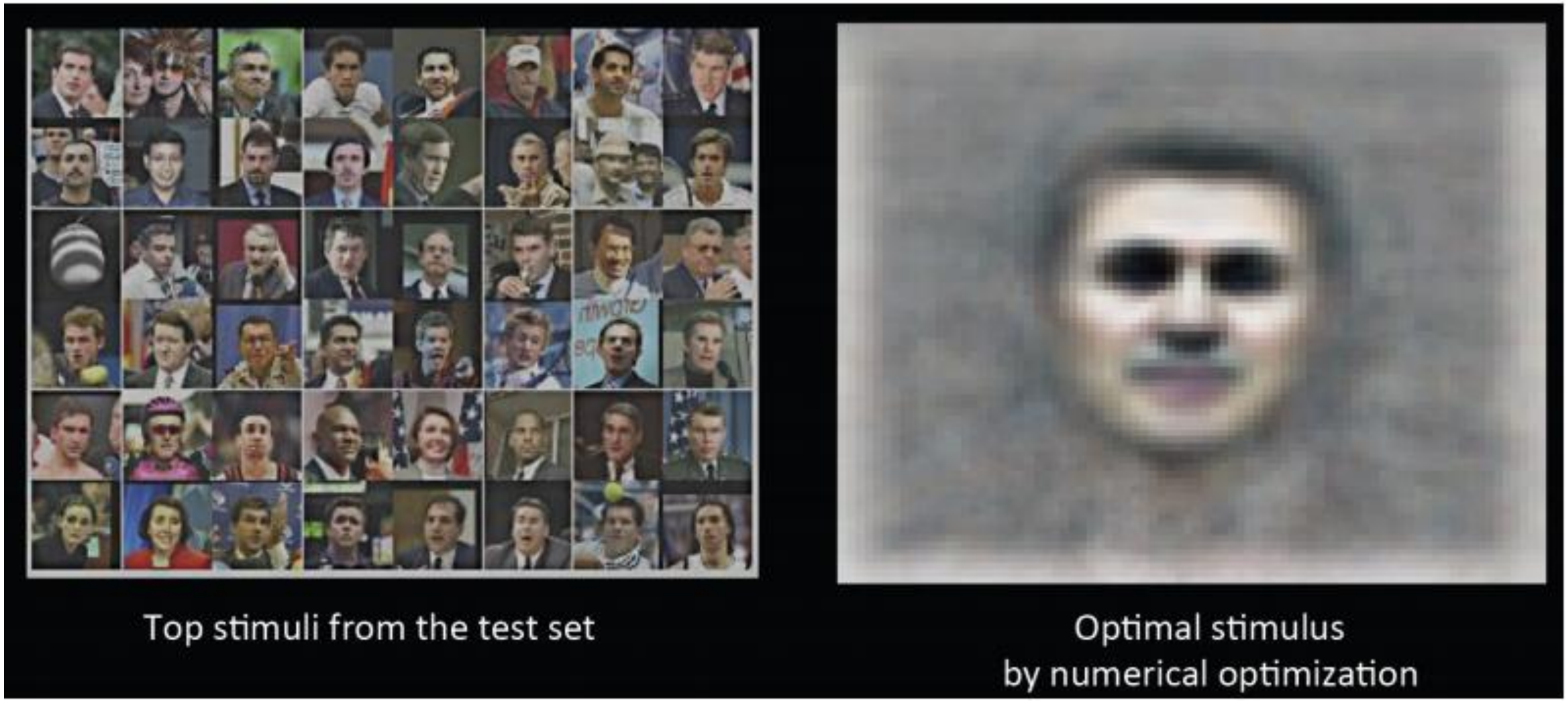
Repeated 3 times to form Deep Architecture
$x^{(m)}$ = image of 200x200 pixels x3 channels

Training

- Using a deep network of 1 billion parameters
  - 10 million images (sampled from Youtube)
  - 1000 machines (16,000 cores) x 3 week.

- Model parallelism
  - Distributing the local weights W_d, W_e in different machines
  - Asynchronous SGD

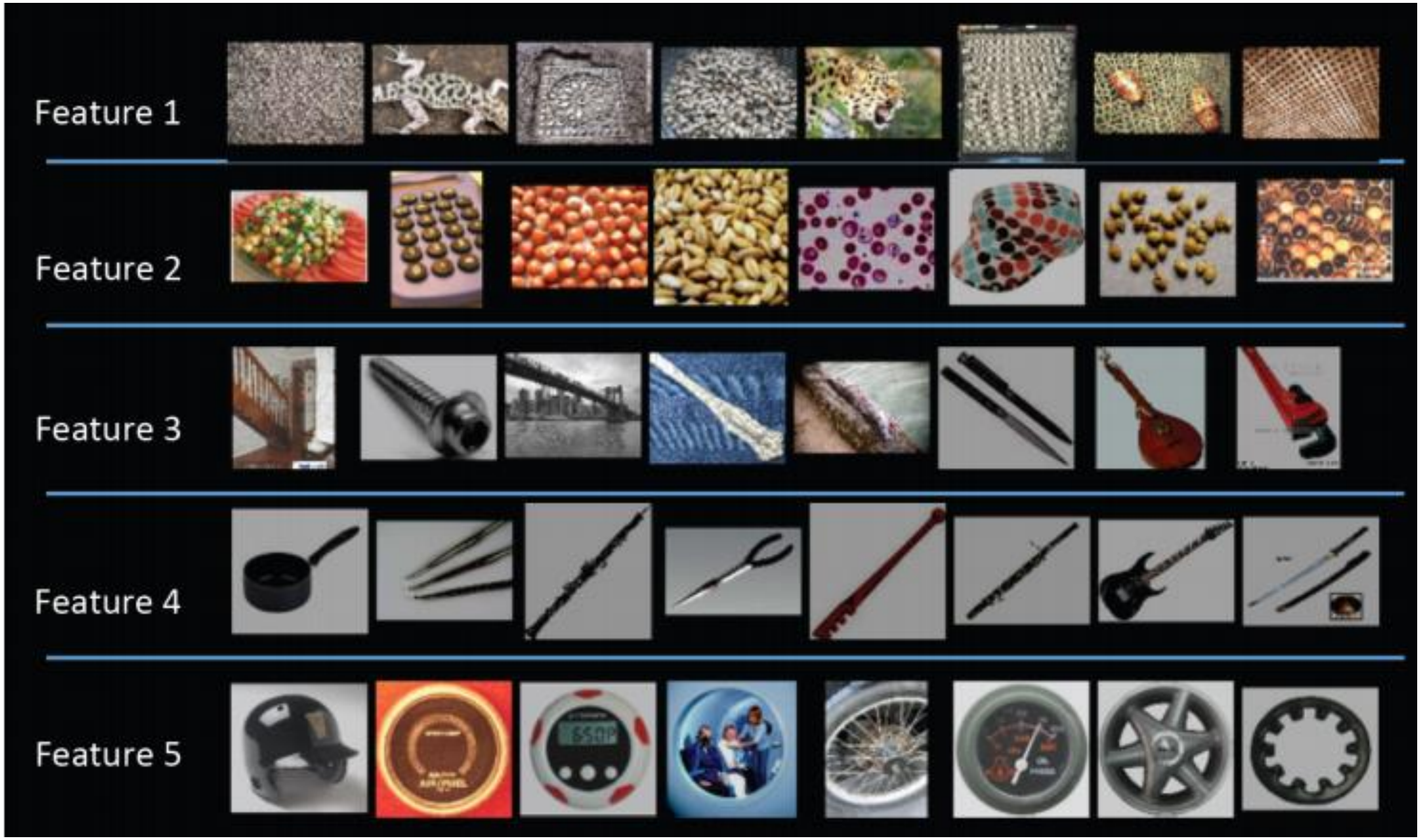
Face neuron


Top stimuli from the test set
Optimal stimulus
by numerical optimization

Cat neuron


Top stimuli from the test set
Optimal stimulus
by numerical optimization

More examples



APPLICATION

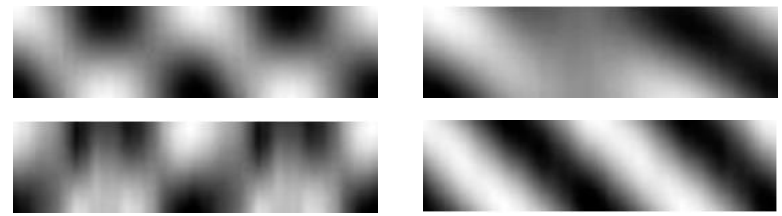
<table>
<tr><td colspan="5" rowspan="4"></td><th colspan="6">Tasks</th></tr>
<tr><th colspan="5">ADAS</th><th></th></tr>
<tr><th colspan="3">Self Driving</th><th></th><th></th><th></th></tr>
<tr><th>Localization</th><th>Perception</th><th>Planning/ Control</th><th>Driver state</th><th>Vehicle Diagnosis</th><th>Smart factory</th></tr>
<tr><td rowspan="8">Methods</td><td rowspan="2">Traditional</td><td colspan="3">Non-machine Learning</td><td>GPS, SLAM</td><td></td><td>Optimal control</td><td></td><td></td><td></td></tr>
<tr><td rowspan="7">Machine-Learning based method</td><td rowspan="4">Supervised</td><td>SVM
MLP</td><td></td><td>Pedestrian detection (HOG+SVM)</td><td></td><td></td><td></td><td></td></tr>
<tr><td rowspan="6">Deep-Learning based</td><td>CNN</td><td></td><td>Detection/ Segmentation/Classification</td><td>End-to-end Learning</td><td></td><td></td><td></td></tr>
<tr><td>RNN (LSTM)</td><td></td><td>Dry/wet road classification</td><td>End-to-end Learning</td><td>Behavior Prediction/ Driver identification</td><td></td><td>*</td></tr>
<tr><td>DNN</td><td></td><td></td><td></td><td></td><td>*</td><td>*</td></tr>
<tr><td colspan="2">Reinforcement</td><td></td><td></td><td>*</td><td></td><td></td><td></td></tr>
<tr><td colspan="2">Unsupervised</td><td></td><td></td><td></td><td></td><td></td><td>*</td></tr>
</table>

Rotor System Diagnosis

Vibration image generation

RK4 test-bed (Labeled)
Power plant (Labeled)

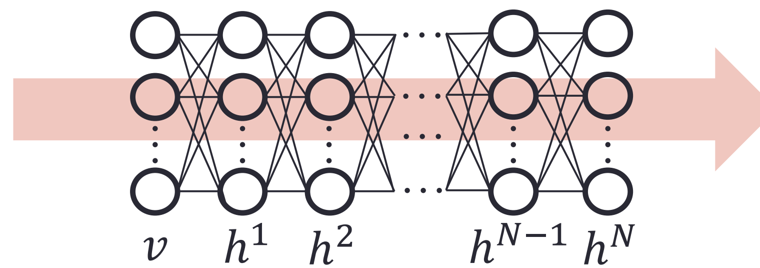
Power plant (Unlabeled)



Vibration images using ODR

High–level feature abstraction

- Greedy layer-wise unsupervised pre-training :
  Restricted Boltzmann Machine in deep belief network (DBN)



Health reasoning

- Classification :
  multi-layer perceptron (MLP)
- Clustering : self-organizing map