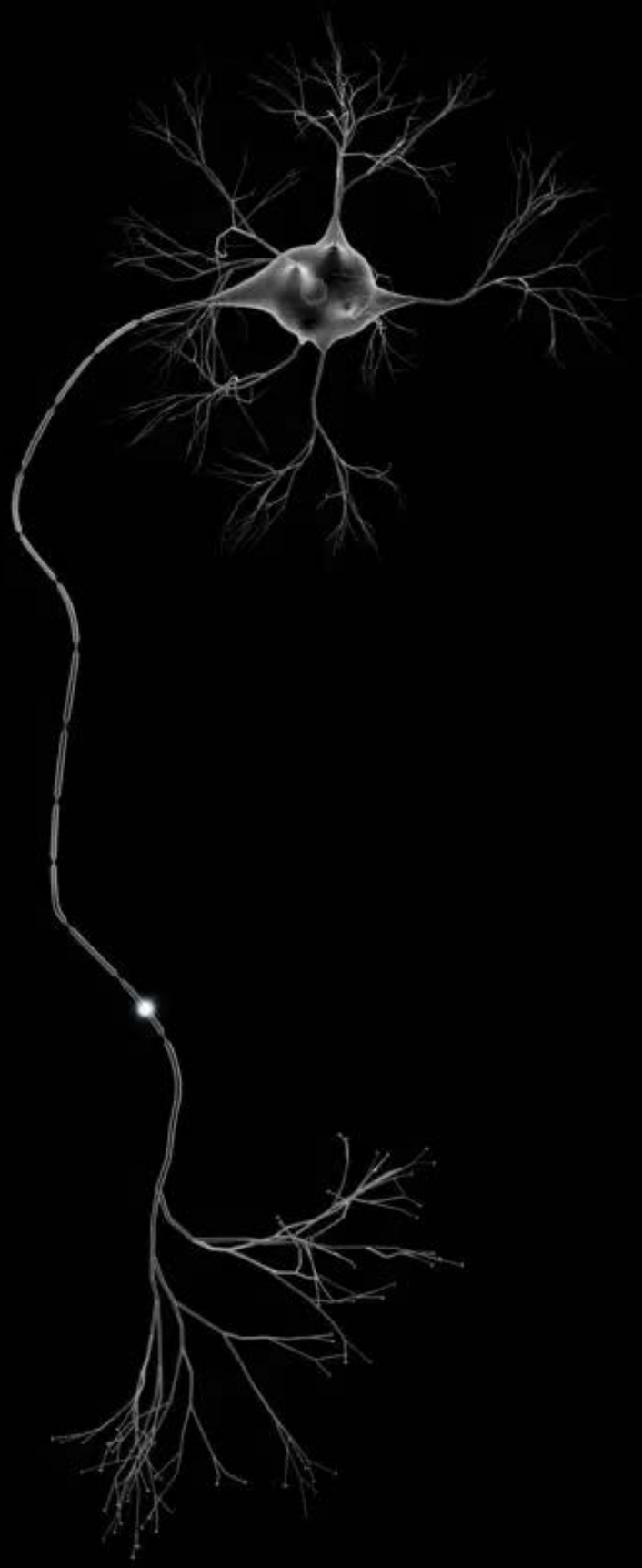
Biologically Inspired Computation

Deep Learning & Convolutional Neural Networks

Joe Marino

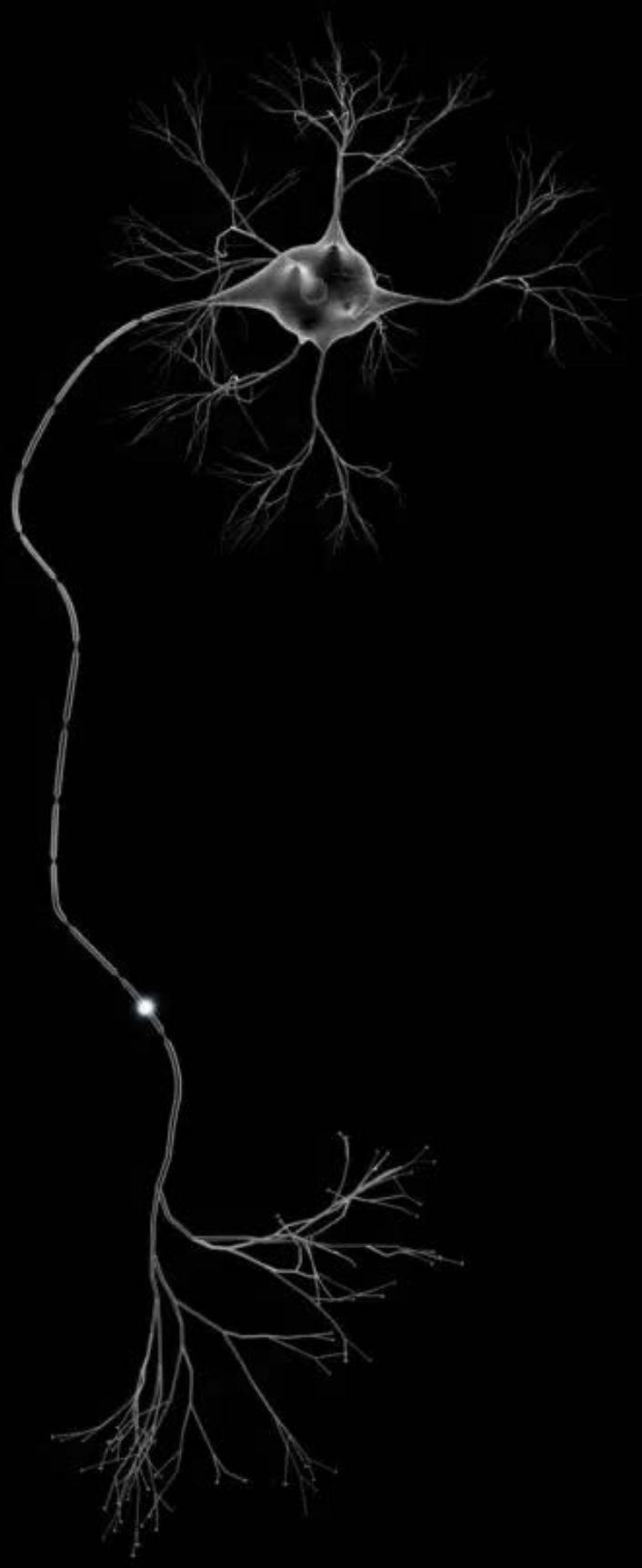
biologically inspired computation



biological intelligence

- flexible
- capable of detecting/ executing/reasoning about high level patterns

- limited by evolutionary constraints
- slow, imperfect



goal:
build machines that have the same capabilities as biological intelligence

use ***inspiration*** from biological intelligence to motivate engineering and design of these machines

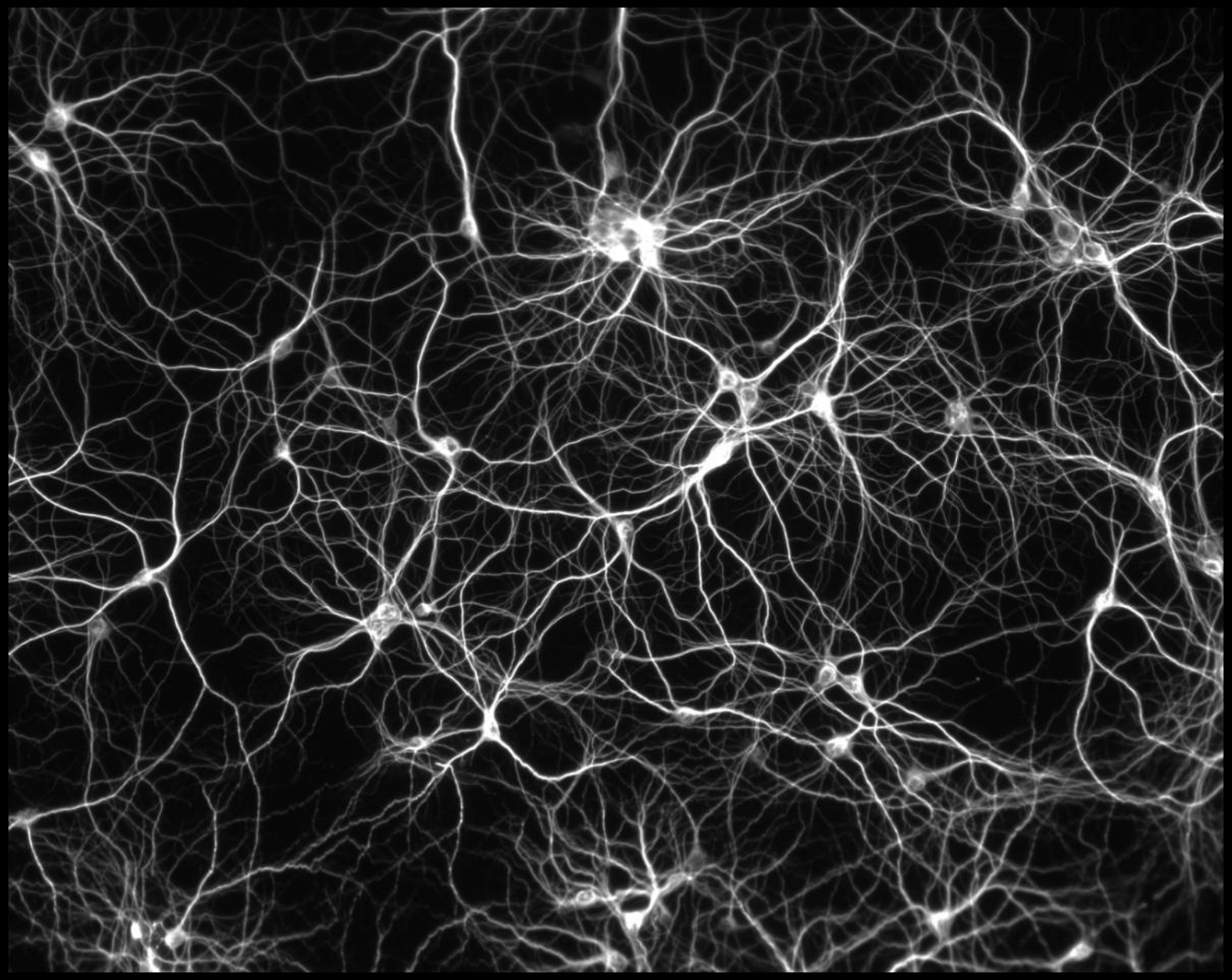
inputs: pre-synaptic signals

output: spike

function: non-linear depolarization

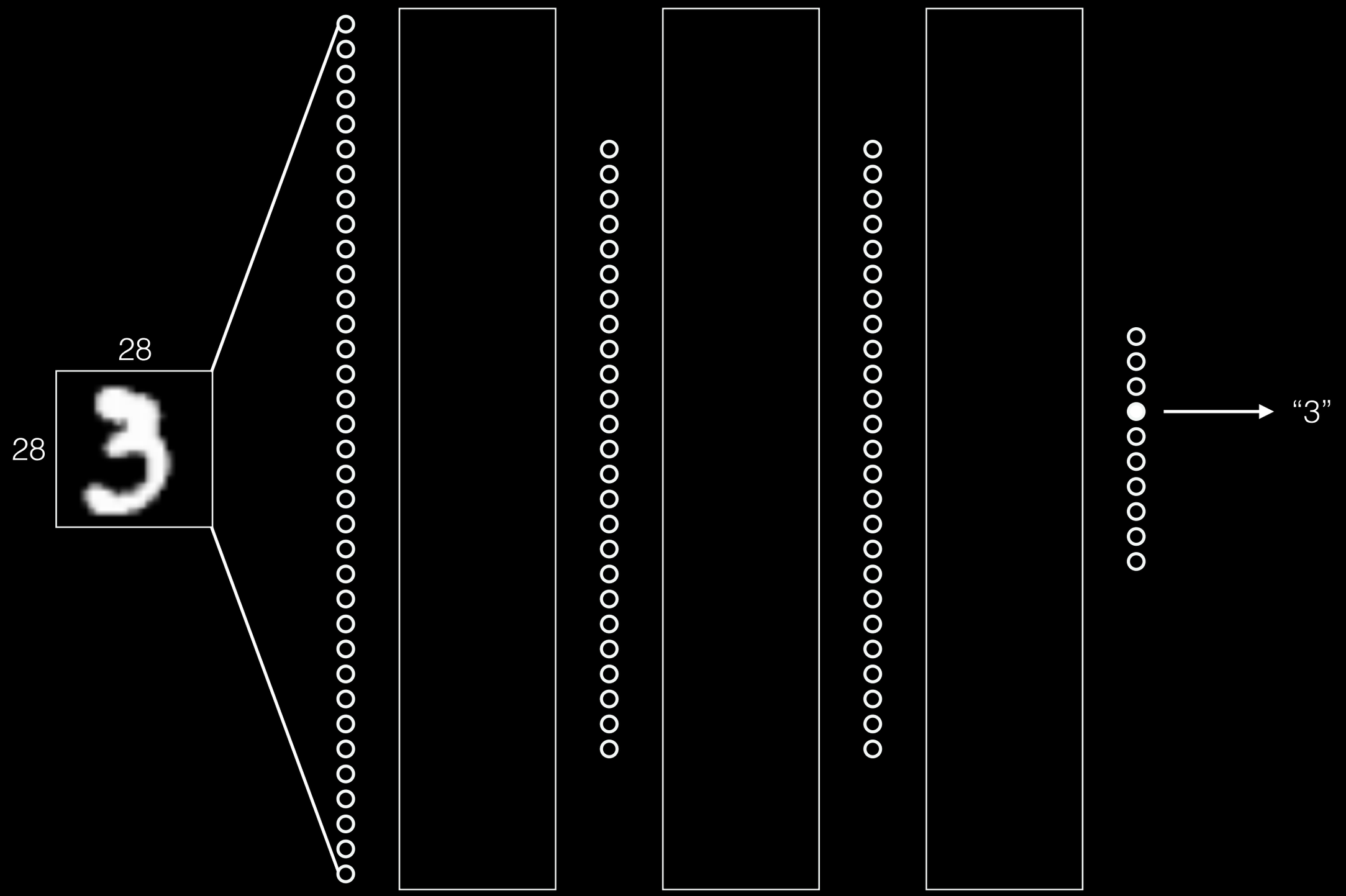
output = **function** (**inputs**)

$$x_{out} = \sigma\left(\mathbf{w}^T \mathbf{x}_{in}\right)$$

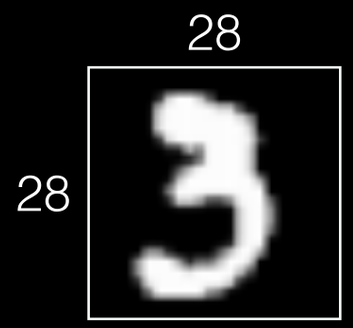


output = **function** (**inputs**)

$$\mathbf{x}_\ell = \sigma\left(\mathbf{w}_\ell^T \mathbf{x}_{\ell-1}\right)$$

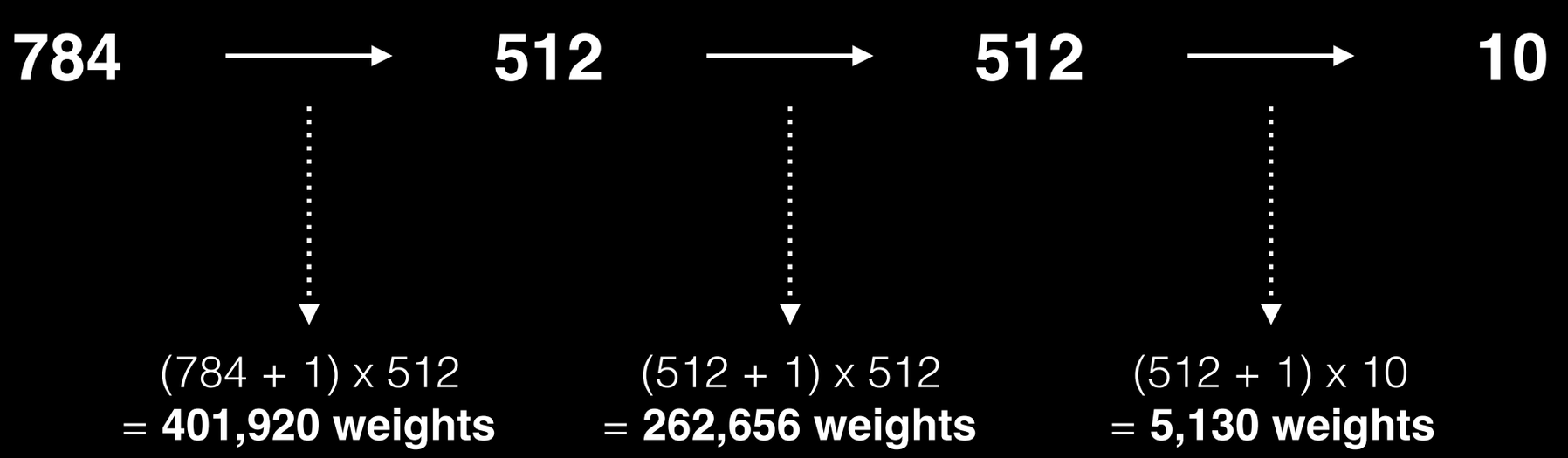


multi-layer perceptron



28 x 28 x 1 = **784 inputs**
(*h* x *w* x *channels*)

Network Architecture:



401,920 + 262,656 + 5,130 = **669,706 weights**

~ 1,000x as many weights as inputs

Natural Images

600

375

375 x 600 x 3 = **675,000 inputs** ⟶ **675,000,000 weights?**

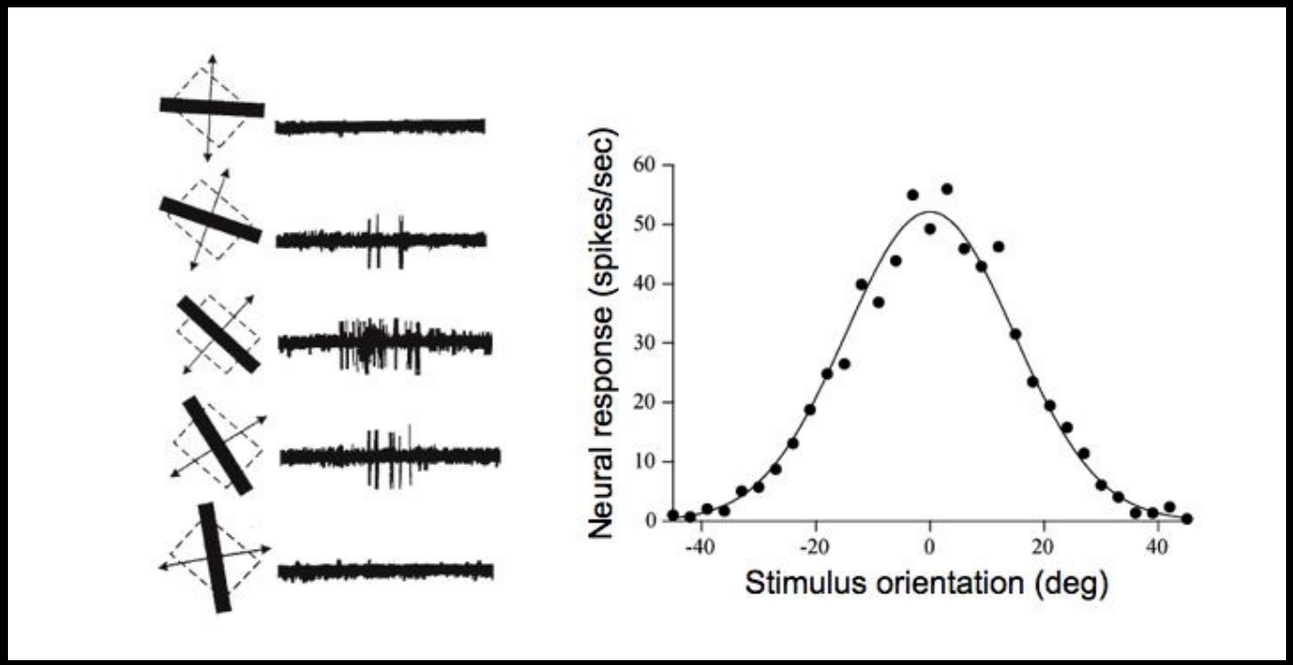
Additional Difficulties

- large space of high level concepts
- more variety of patterns
- complex spatial relationships

Biological Inspiration

How do animals recognize visual stimuli?

Hubel & Wiesel - 1950s



- recorded responses of neurons in primary visual cortex (V1) to simple stimuli
- found selectivity to bars of specific orientation

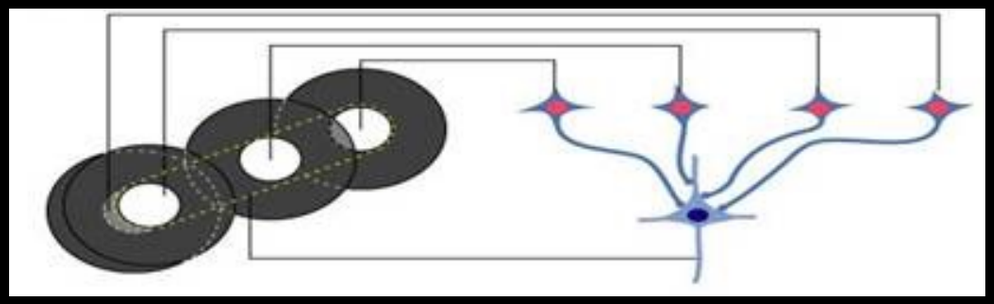
Hubel & Wiesel, 1959

Biological Inspiration

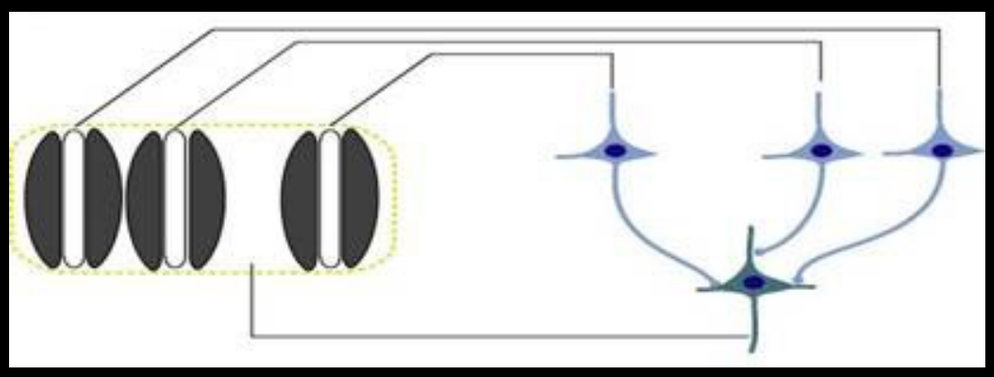
How do animals recognize visual stimuli?

Simple and Complex Cells

simple cells combine lower level features (on/off ganglion responses) within a receptive field to select for more complex features



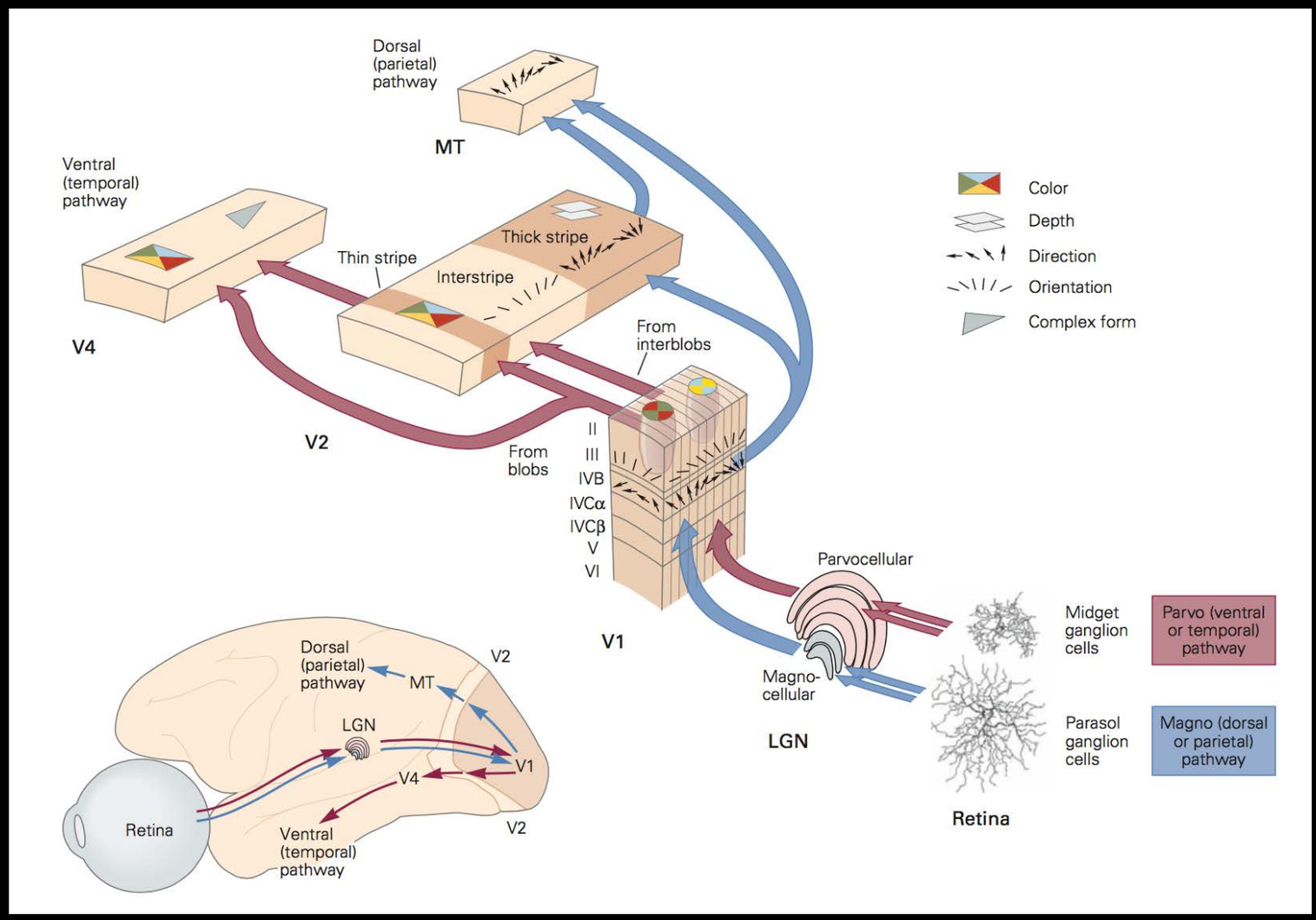
complex cells combine responses from simple cells within a larger receptive field to develop translation invariance



Biological Inspiration

How do animals recognize visual stimuli?

Hierarchical Processing of Visual Features



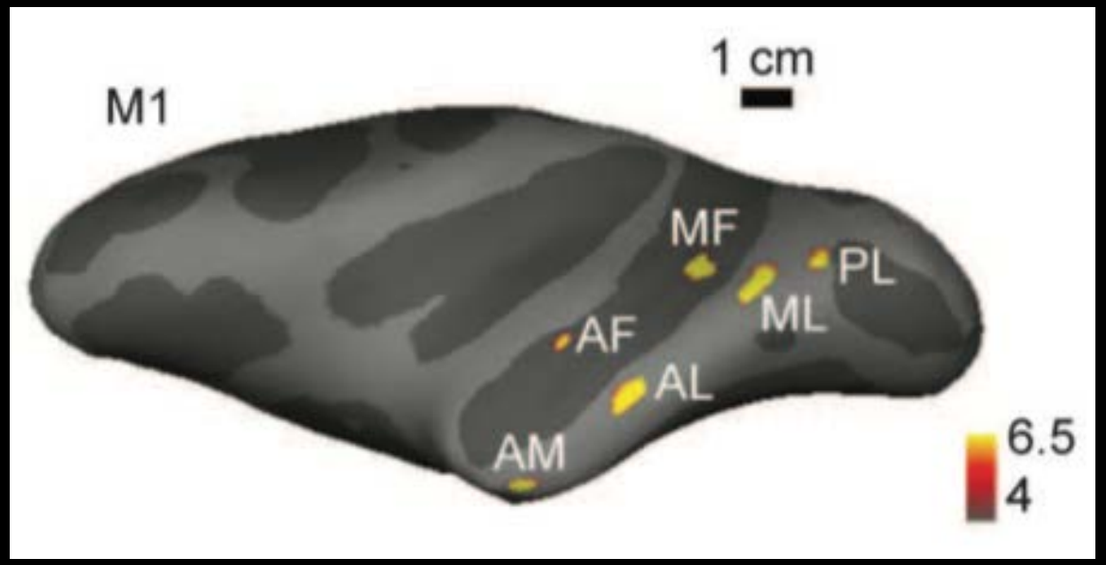
Kandel et al., 2012

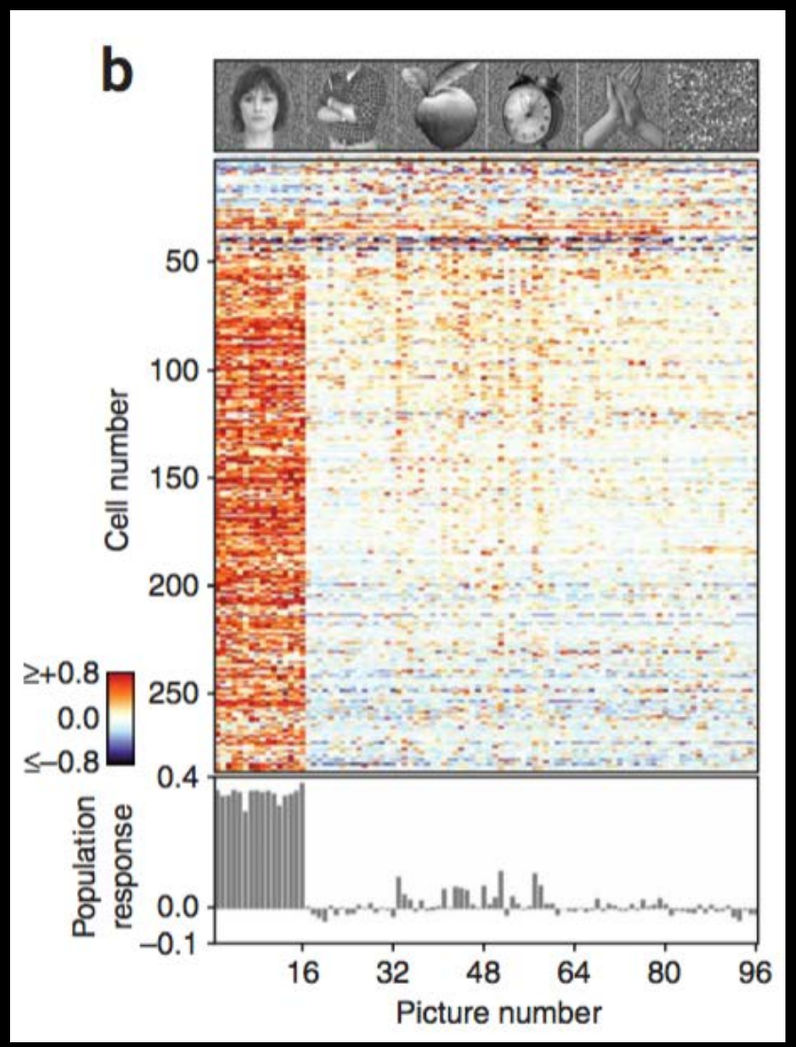
Biological Inspiration

How do animals recognize visual stimuli?

Highly Inter-Connected High Level Visual Areas

face patches

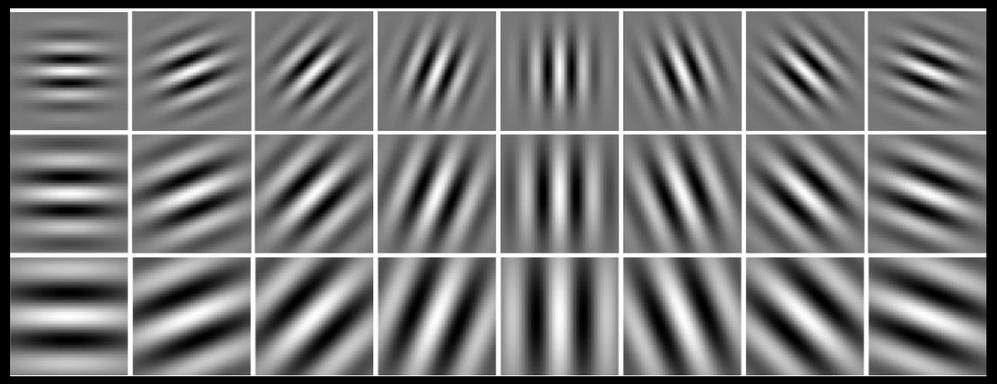




Friewald et al., 2009 & 2010

Engineering Motivation

Natural images can be decomposed into a relatively small set of low level patterns, i.e. *filters*.

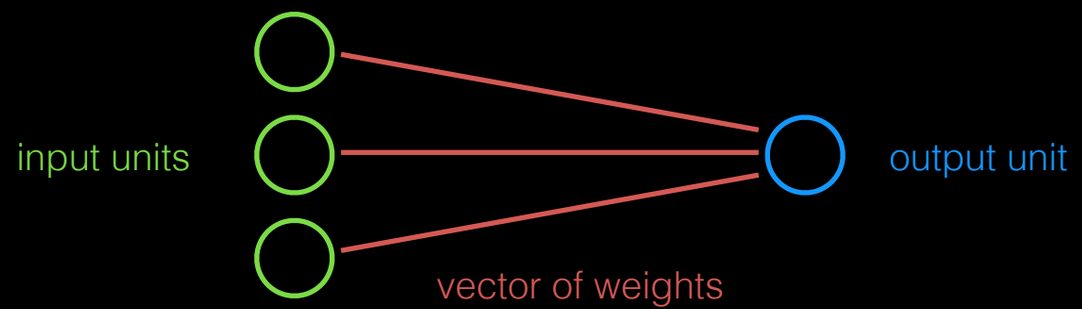


Objects are translation invariant. It's not the absolute positions of patterns that matters, but rather the *relative positions*.

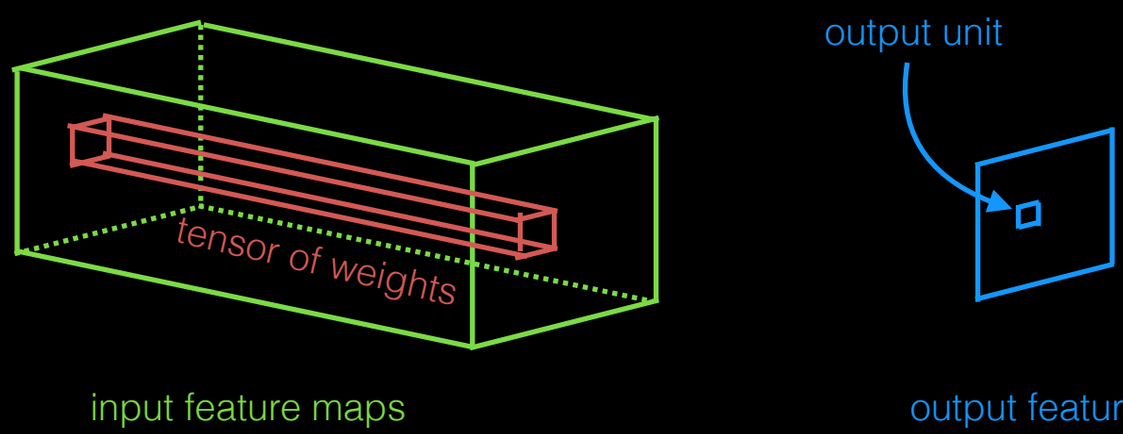
⟶ **Exploit the redundancy within the input by sharing weights within the network.**

Convolution

In a multi-layer perceptron, each layer contains a set of units. Each unit operates over all units in the previous layer through a vector of weights.

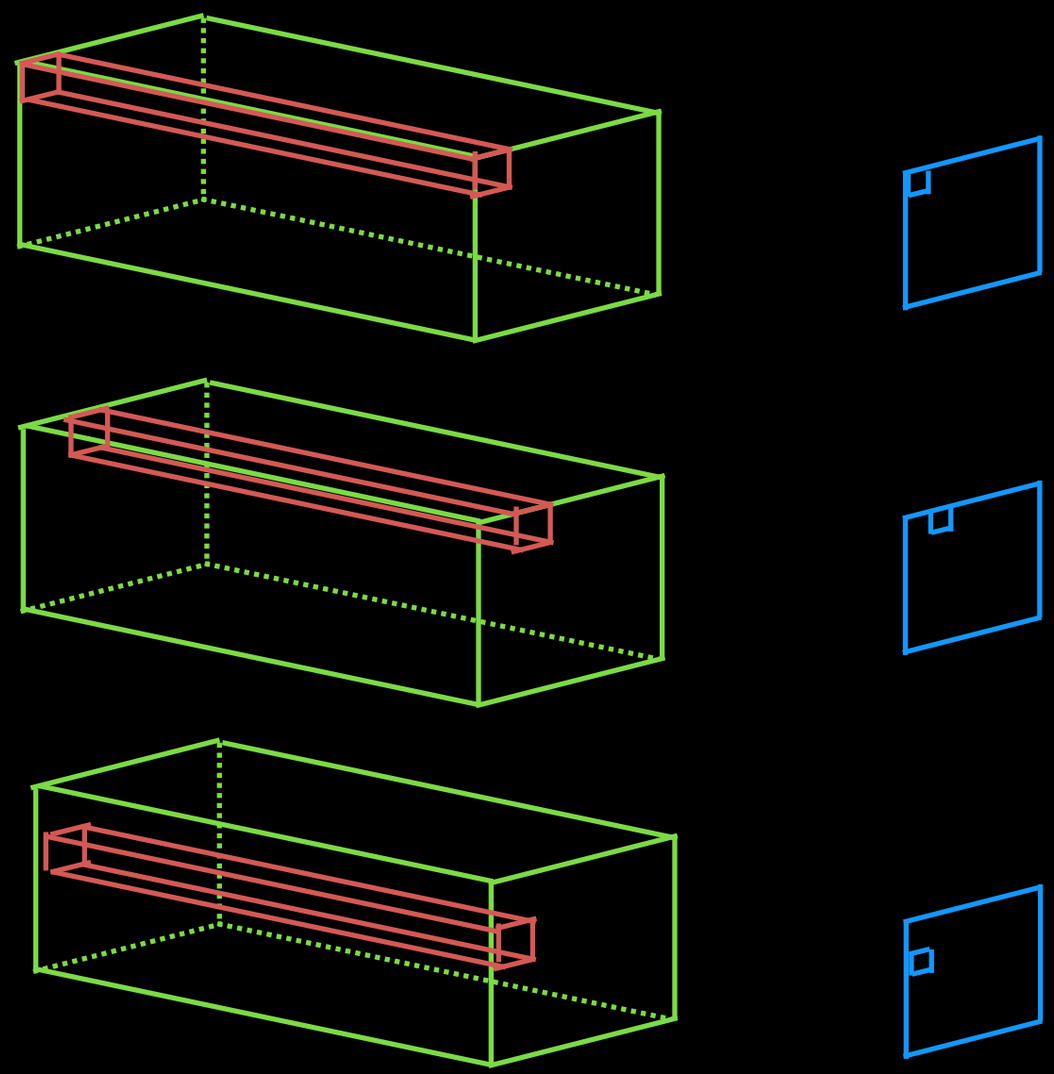


In a convolutional neural network, each convolutional layer contains a set of feature maps. Each feature map operates over all feature maps in the previous layer through a tensor of weights, a *filter*.



Convolution

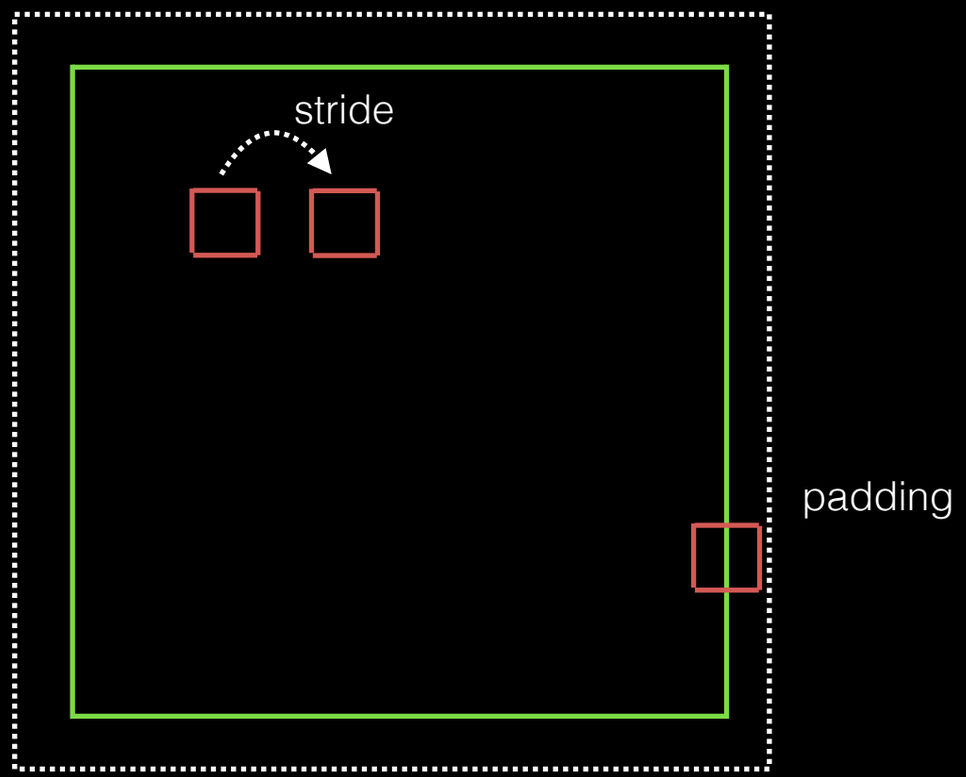
A feature map is a matrix of units. We calculate a feature map by **convolving** the corresponding *filter* with the previous layer's feature maps. This is just a tensor dot product of the filter with the previous feature maps.



Convolution

The **stride** of a convolution is the step size by which you convolve each filter with the input feature maps. This can be used to decrease the spatial size of output feature maps.

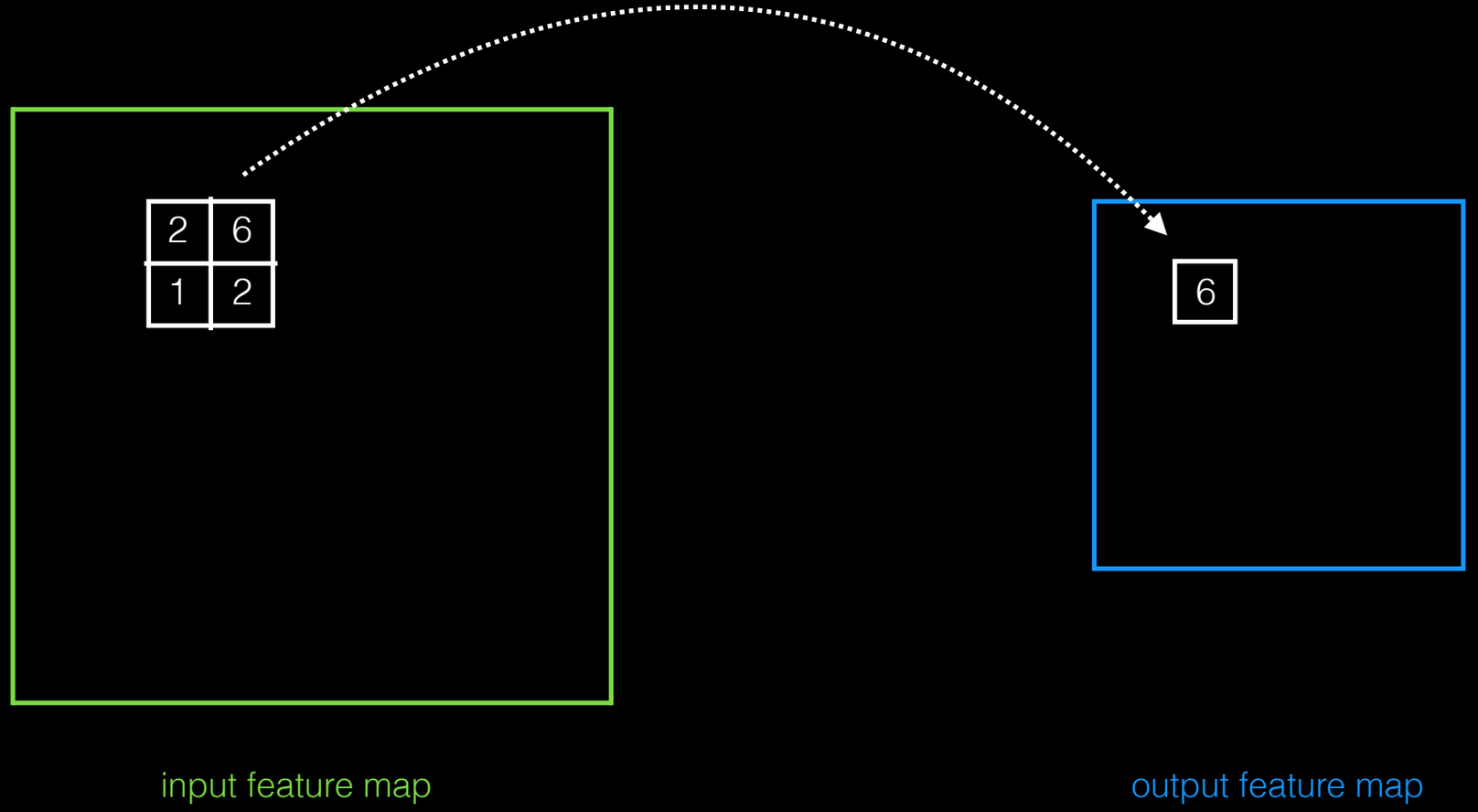
The **padding** of a convolution is the amount of space to place around the boundaries of the input feature maps. This can be used to maintain the spatial size of output feature maps.



Pooling

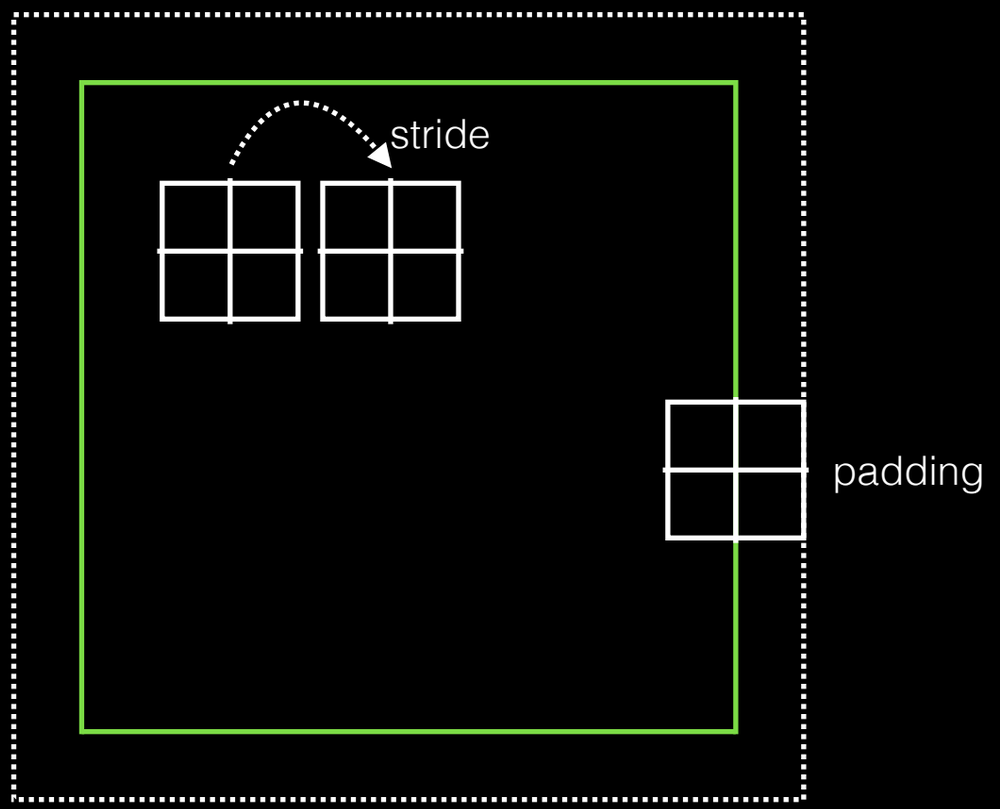
Convolutional layers allow us to be selective to features within the input image. We also want **translation invariance** with respect to these features.

We can sub-sample the maximum values of the feature maps to retain only the (invariant) high-level details. This is called **max pooling.**



Pooling

Pooling also contains a **stride** and **padding**, which are analogous to convolution. A larger stride decreases the feature map's spatial size more. Padding preserves the edges.



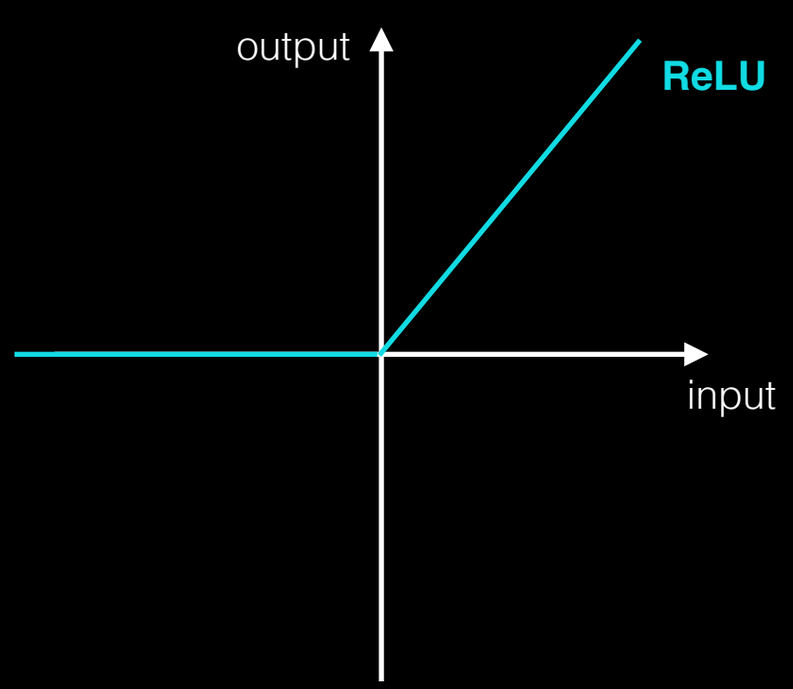
Other
(Biologically Inspired)
Tricks

Rectified Linear Units (ReLU)

Sigmoid non-linearities lead to vanishing gradients during backpropagation in deep networks.

Instead, use rectified linear units (ReLU). This non-linearity does not suffer from vanishing gradients, allowing for deeper networks. *However, it also has the negative effect of linearizing the network.*

$$\text{ReLU}(x) = \max(x, 0)$$

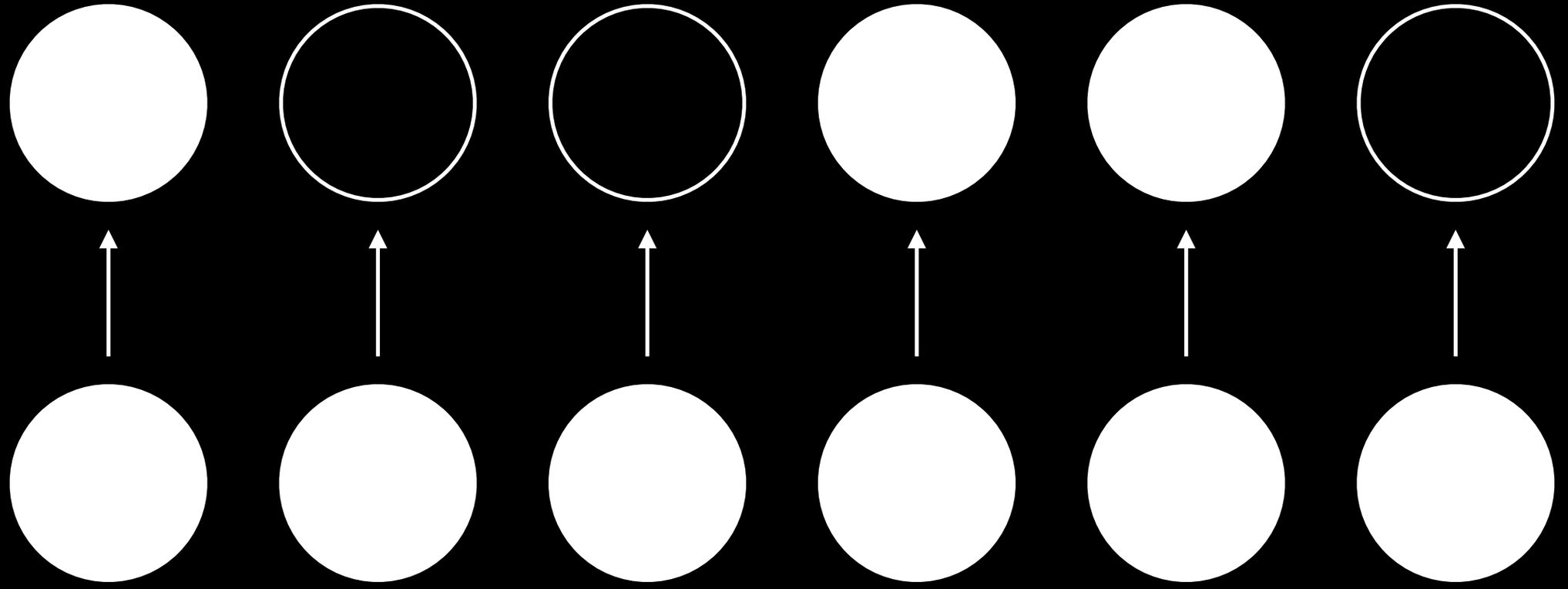


Nair et al., 2010

Dropout

With large networks, it is easier to overfit to the training data. Units may start to co-adapt during training, in which they depend heavily on each other.

Remedy this by using **dropout**, randomly turning off units. This prevents the units from co-adapting, effectively creating an ensemble of networks within one network.



Srivastava et al., 2014

Normalization

It often helps to **normalize** the units to a fixed mean and variance, capturing only the relative differences in the activations rather than their absolute values. This also has the effect of preventing *co-variate shift*, allowing for faster training.

There are multiple ways to normalize the units. The most popular method is **batch normalization**.

batch

$$\mathcal{B} = \{x_1, ..., x_m\}$$

batch norm output

$$y_i = \mathrm{BN}_{\gamma,\beta}(x_i)$$

$$\mu_{\mathcal{B}} \leftarrow \frac{1}{m}\sum_{i=1}^{m} x_i$$ batch mean

$$\sigma^2_{\mathcal{B}} \leftarrow \frac{1}{m}\sum_{i=1}^{m}(x_i - \mu_{\mathcal{B}})^2$$ batch variance

$$\hat{x}_i \leftarrow \frac{x_i - \mu_{\mathcal{B}}}{\sqrt{\sigma^2_{\mathcal{B}} + \epsilon}}$$ normalize

$$y_i \leftarrow \gamma\hat{x}_i + \beta$$ scale and shift

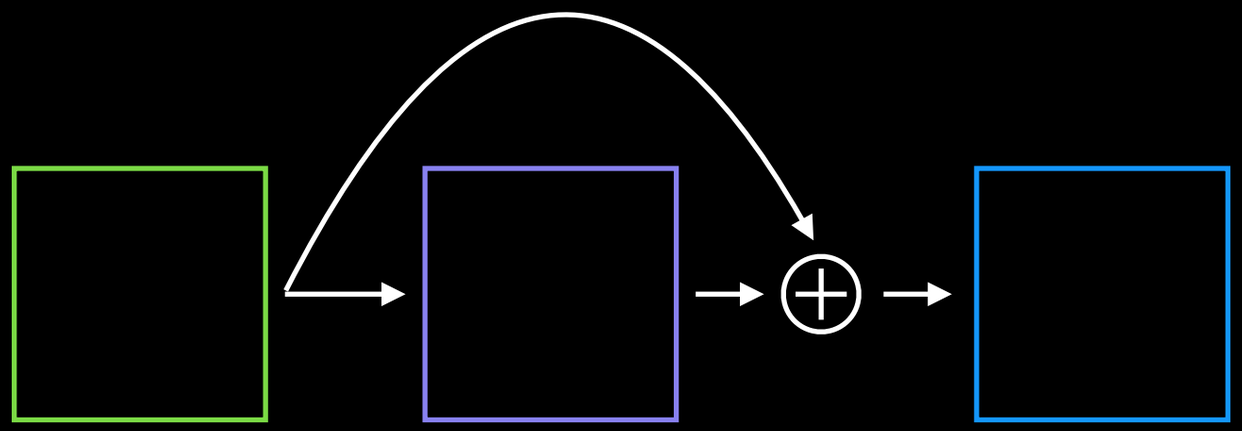
Ioffe et al., 2015

Residual Connections

It is difficult to train very deep networks: it becomes more difficult to avoid local minima. For this reason, we can introduce **residual connections**, in which the *activations are added to their input* at each layer.

Each layer learns a *residual function,* allowing the network to maintain important features at deeper layers.

$$\mathbf{x}_L = \mathbf{x}_0 + \sum_{\ell=0}^{L-1} \mathcal{F}\left(\mathbf{x}_\ell, W_\ell\right)$$

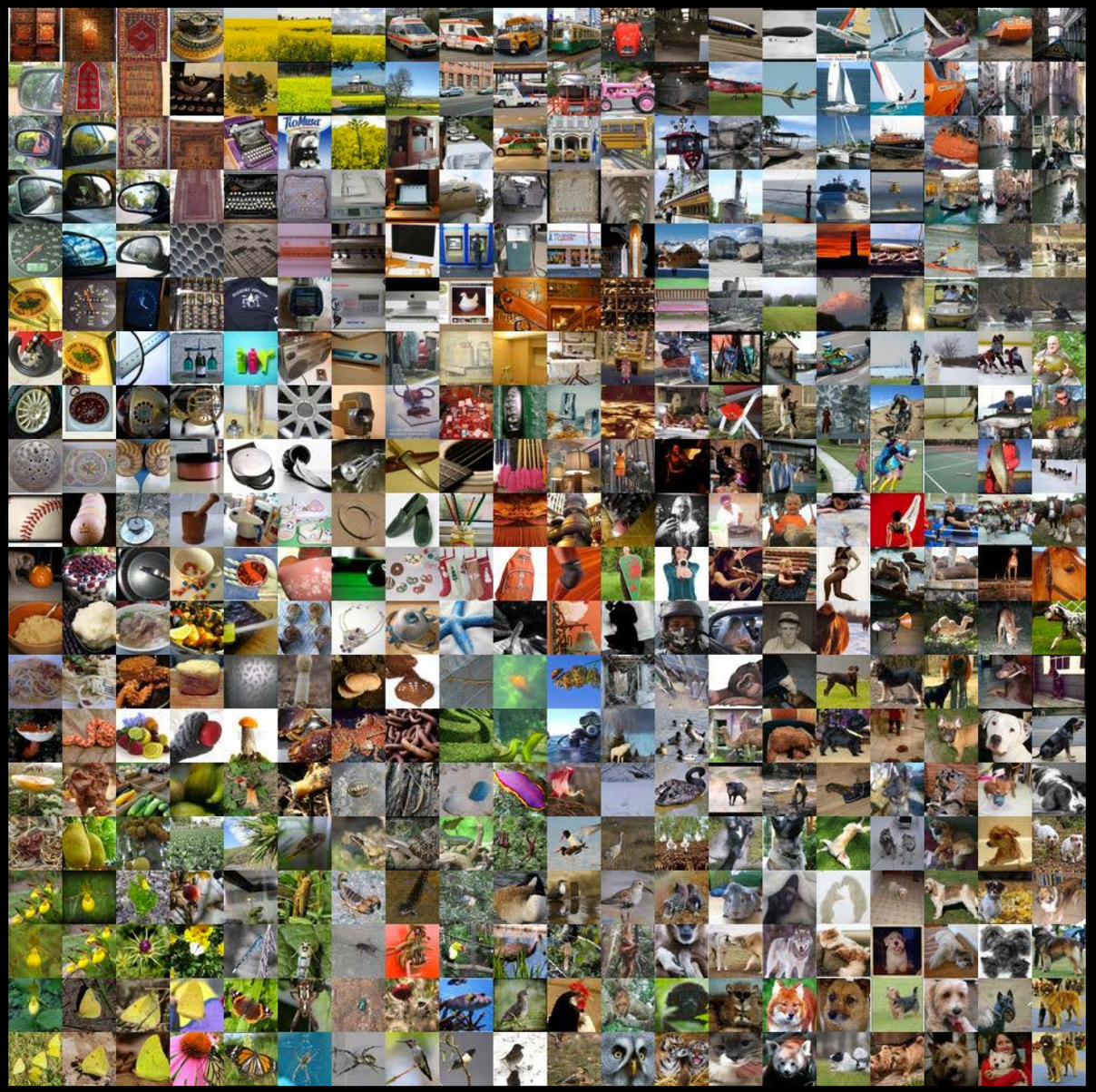


He et al., 2015, 2016

DEMO

Multi-Layer Perceptrons
vs.
Convolutional Neural Networks

Object Classification



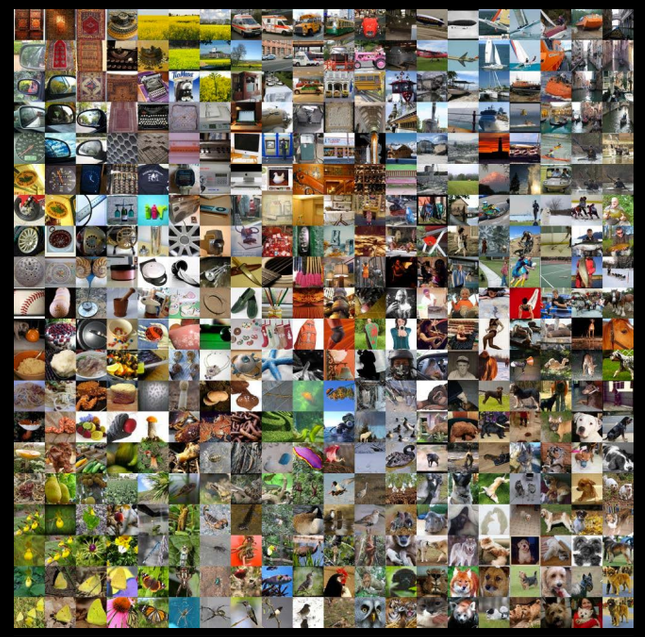
Objects are high-level visual patterns. We want to train computers to recognize these patterns: pedestrian detection, visual search, surveillance, etc.

Object Classification

To build a successful object classifier, we need

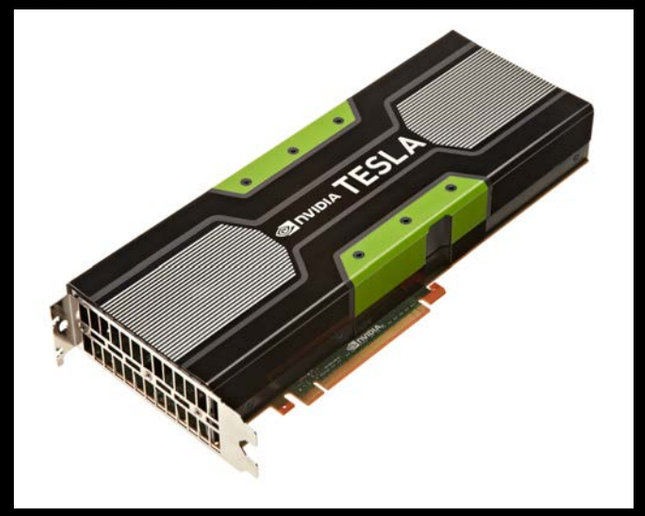
data

- ImageNet —> over 14 million images belonging to over 20,000 object categories



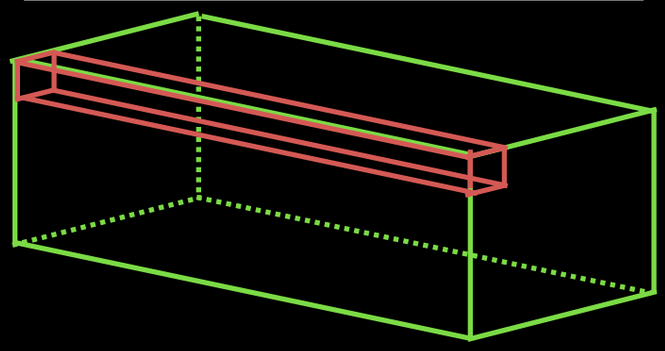
compute hardware

- GPUs allow parallelized computation, resulting in significant speed up over CPU

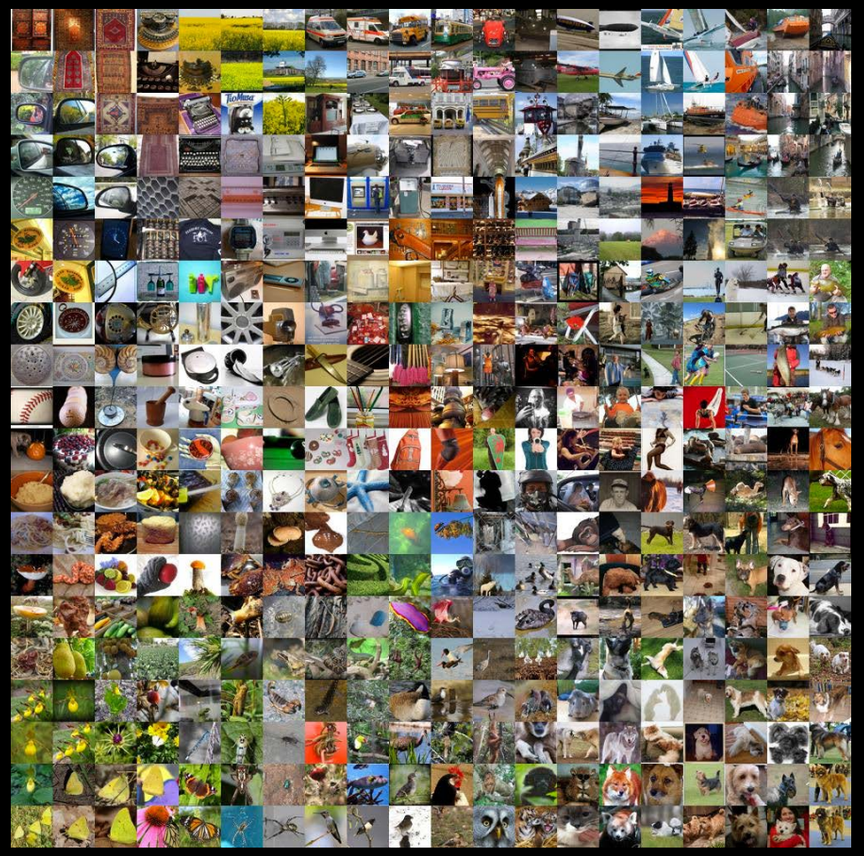


models

- deep convolutional neural networks



ILSVRC



A subset of 1.2 million images from ImageNet is used for the ImageNet Large Scale Visual Recognition Challenge. This competition requires entrants to classify objects from 1,000 different categories.

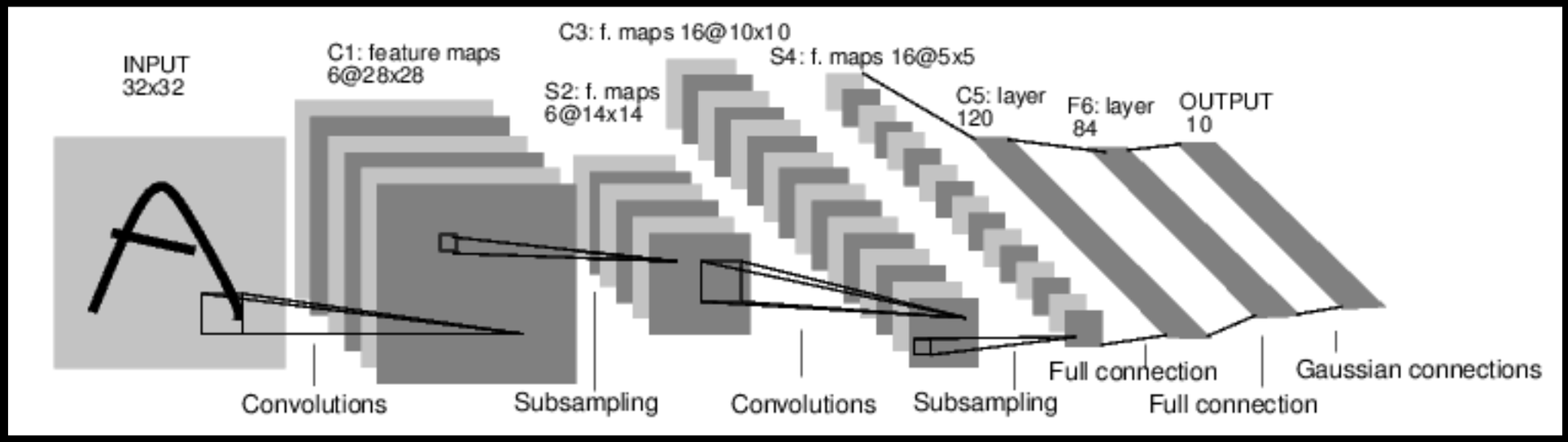
The human top-5 error rate (correct label is not in top 5 guesses) is about **5%**. An estimated 3% of the data is mislabeled.

DEMO

Object Classification

Deep Network Architectures

LeNet - 1989



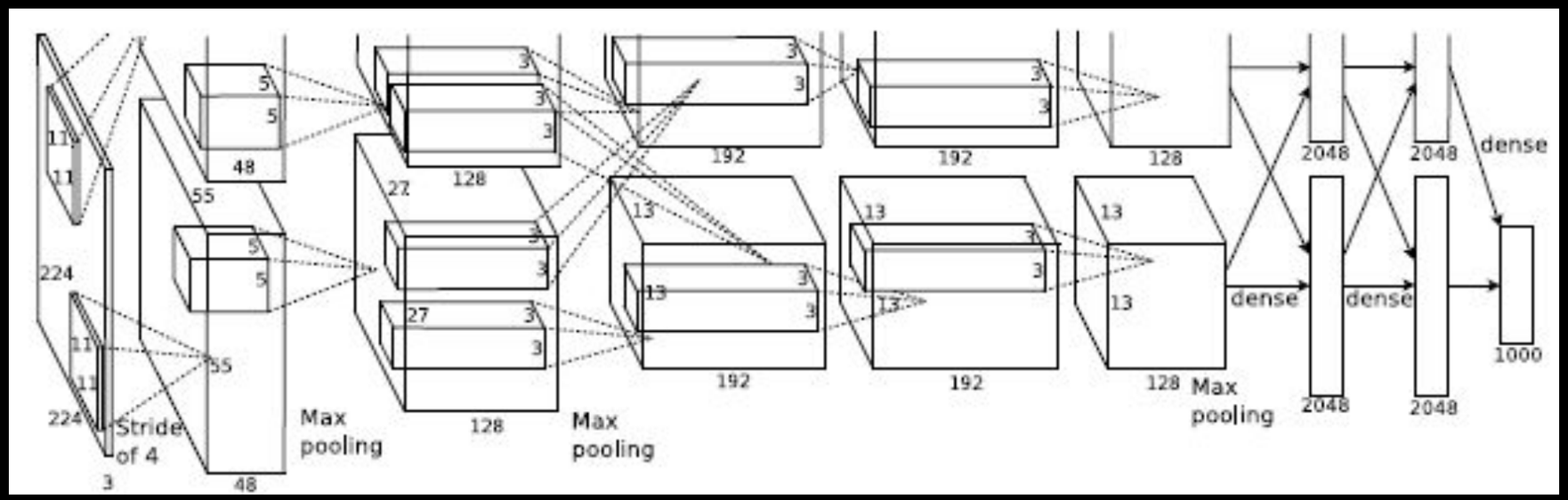
Introduced convolutional neural networks

Modeled after Fukushima's Neocognitron

Achieved state-of-the-art performance on MNIST

LeCun et al., 1989

AlexNet - 2012



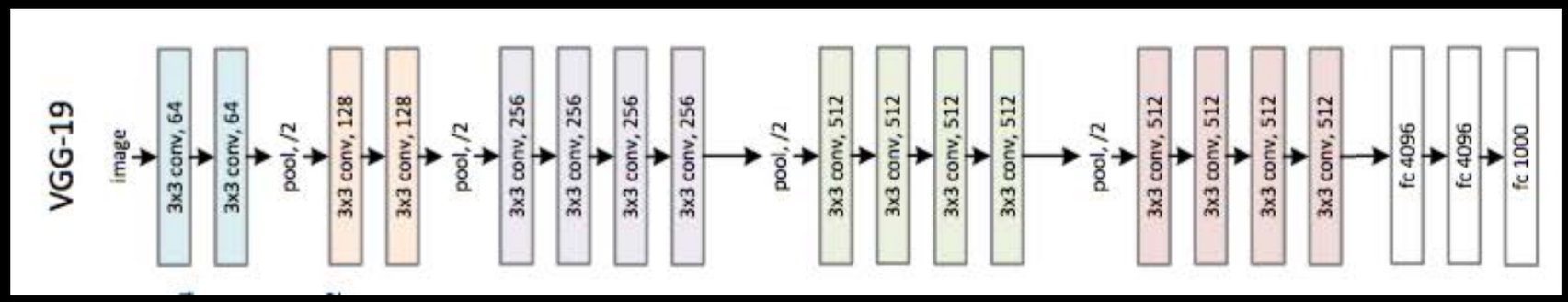
7 layers

Introduced training on GPUs

ILSVRC top-5 error rate: **15.3 %**

Krizhevsky et al., 2012

VGG - 2014



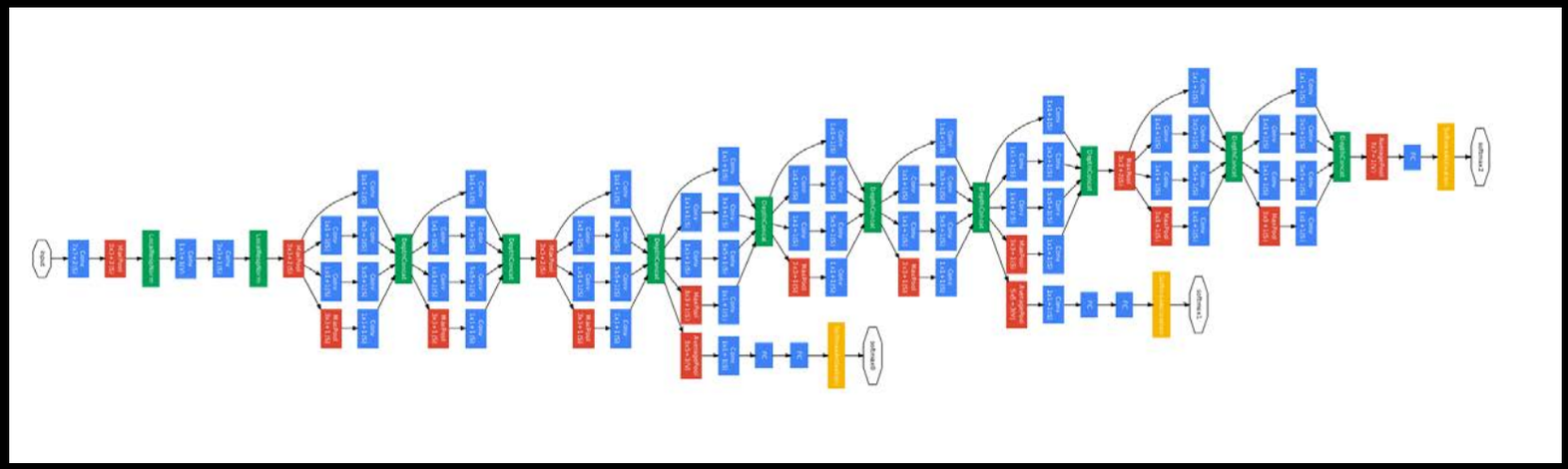
19 Layers

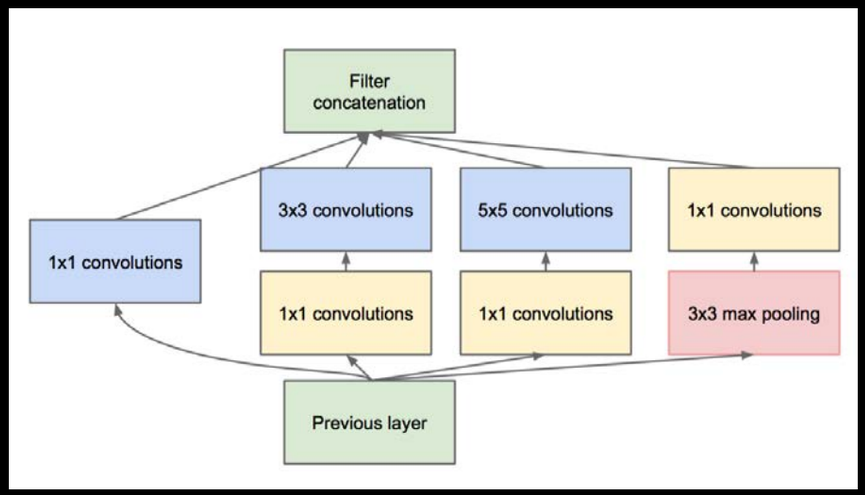
Many layers of convolutions with 3 x 3 filters

ILSVRC top-5 error rate: **7.4 %**

Simonyan et al., 2014

GoogLeNet - 2014





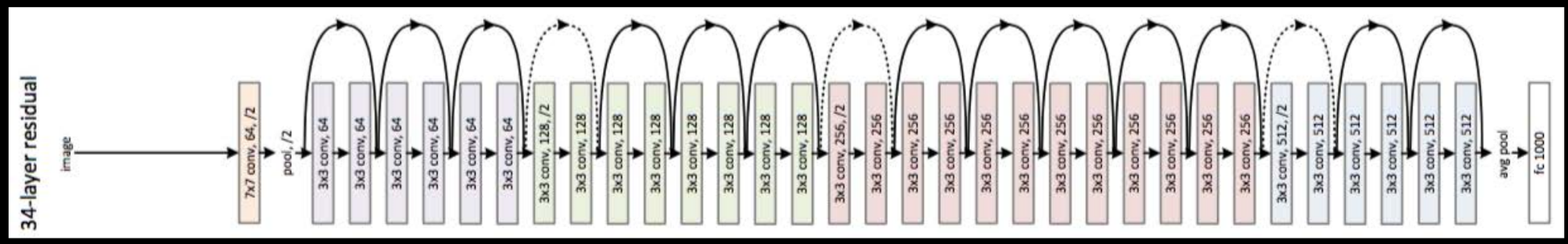
22 Layers

Introduced inception blocks, auxiliary classifiers

ILSVRC top-5 error rate: **6.7 %**

Szegedy et al., 2014

ResNet - 2015



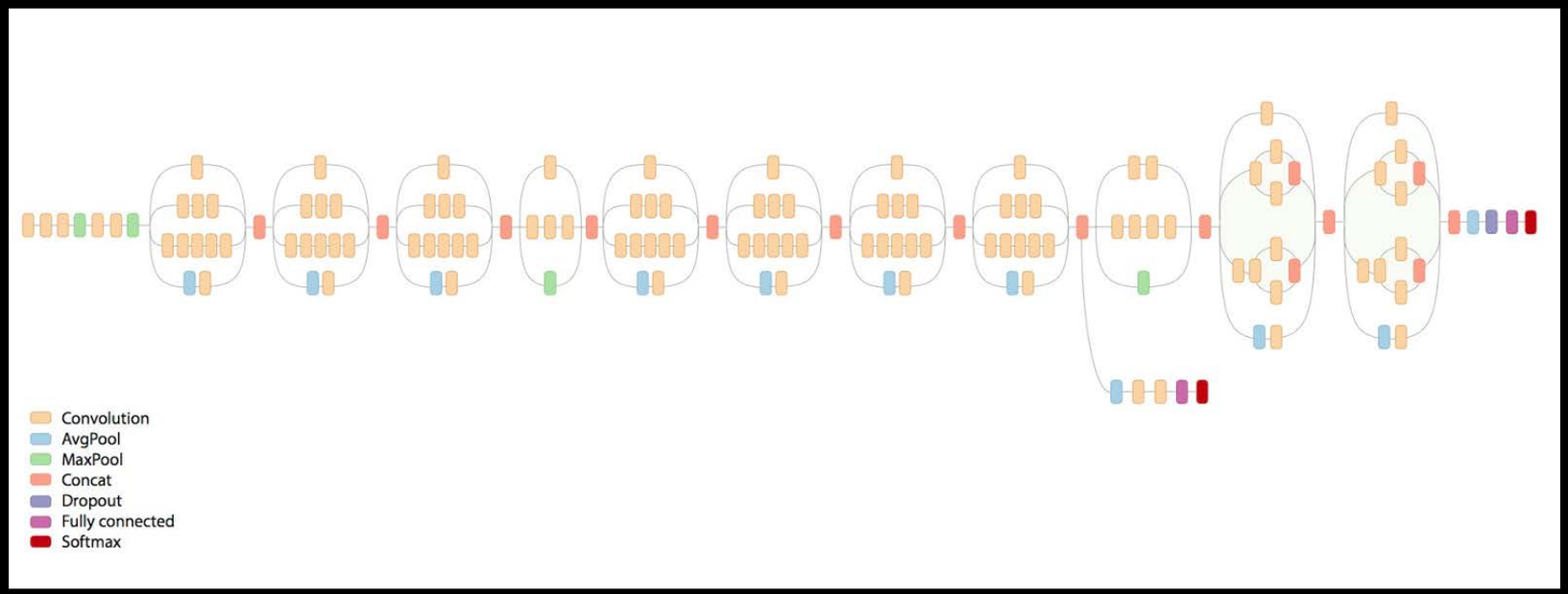
34 Layers

Introduced residual connections.

ILSVRC top-5 error rate: **3.6 %**

He et al., 2014

Inception-ResNet - 2016



53 Layers

Combined residual connections with inception architecture.

ILSVRC top-5 error rate: **3.5 %**

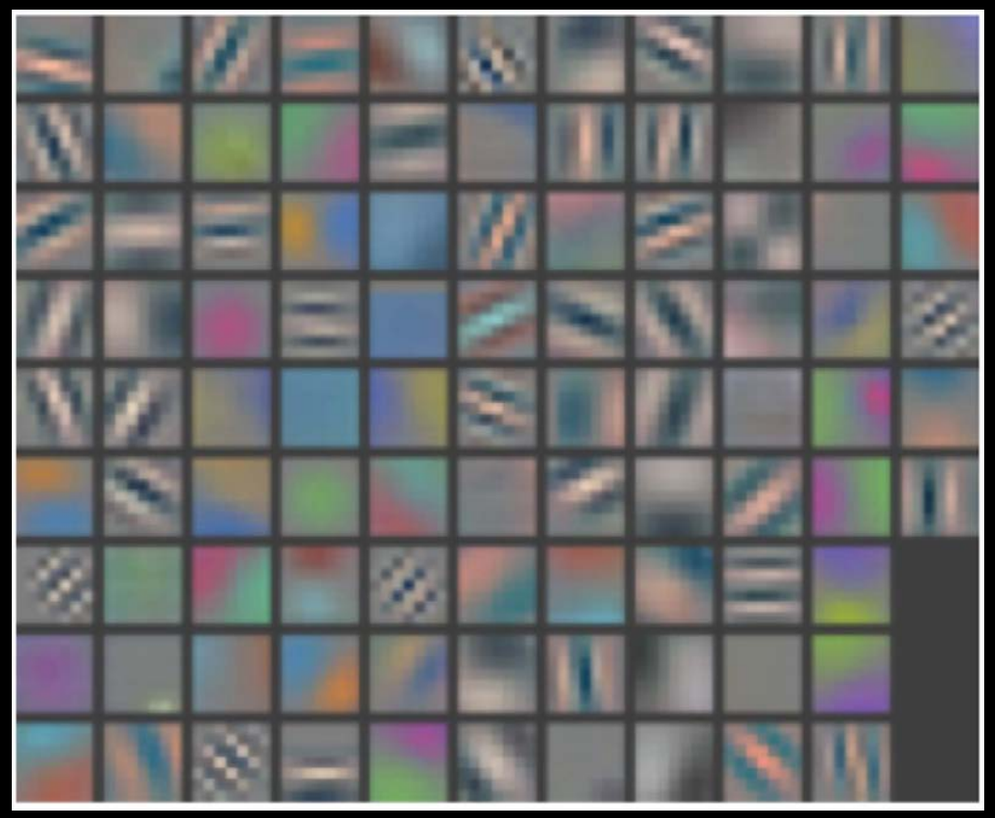
Filter Visualization

These models are clearly performing well on object classification.

How to we determine what they have learned?

Need some method of “seeing” inside the model to visualize the information stored in the filters.

The first set of filters is in the image space, so we can visualize these filters directly:



Filter Visualization

For later layers, there are a variety of methods for visualizing the filters. Each method finds an image that maximally activates a particular filter.

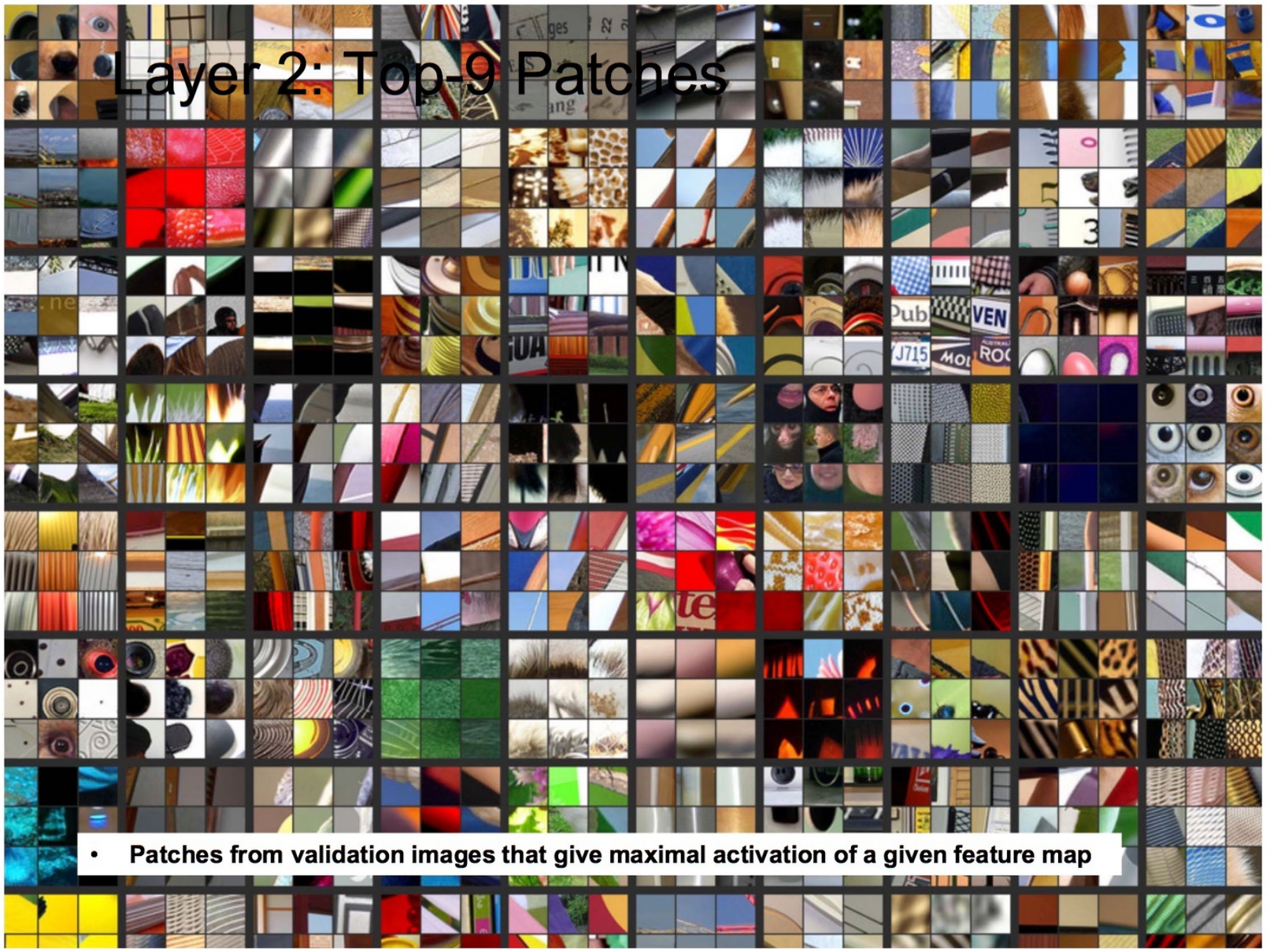
- Maximal images from dataset
  - Feed in all of the images and keep track of which image maximally activates a filter

- Deconvolution
  - Run the network in reverse to get most important features of an image for an activated filter

- Gradient ascent in image space
  - Backpropagate from a filter to the image itself, modifying the image to maximally activate the filter

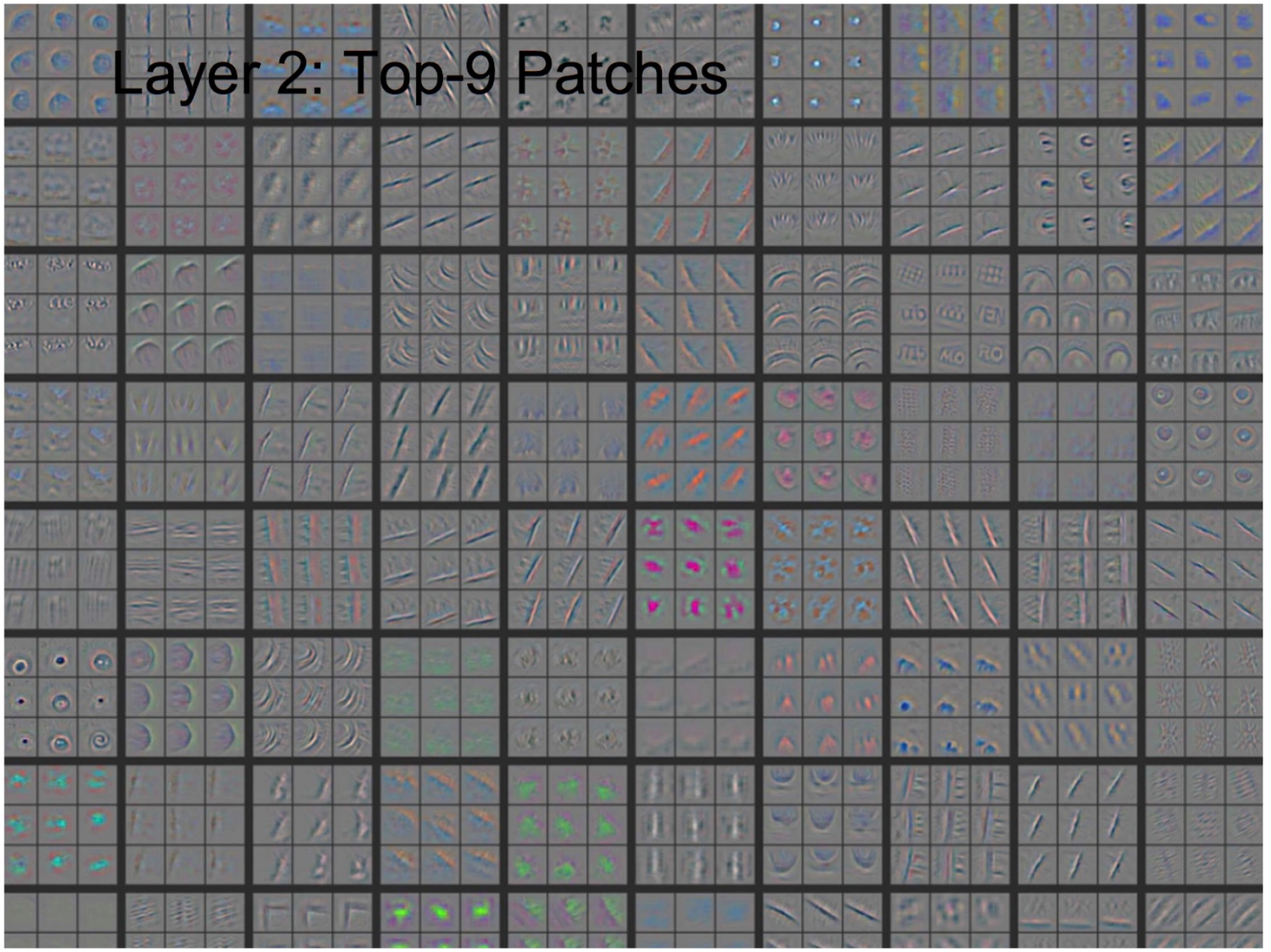
Filter Visualization

Top Image Patches - Layer 2



Filter Visualization

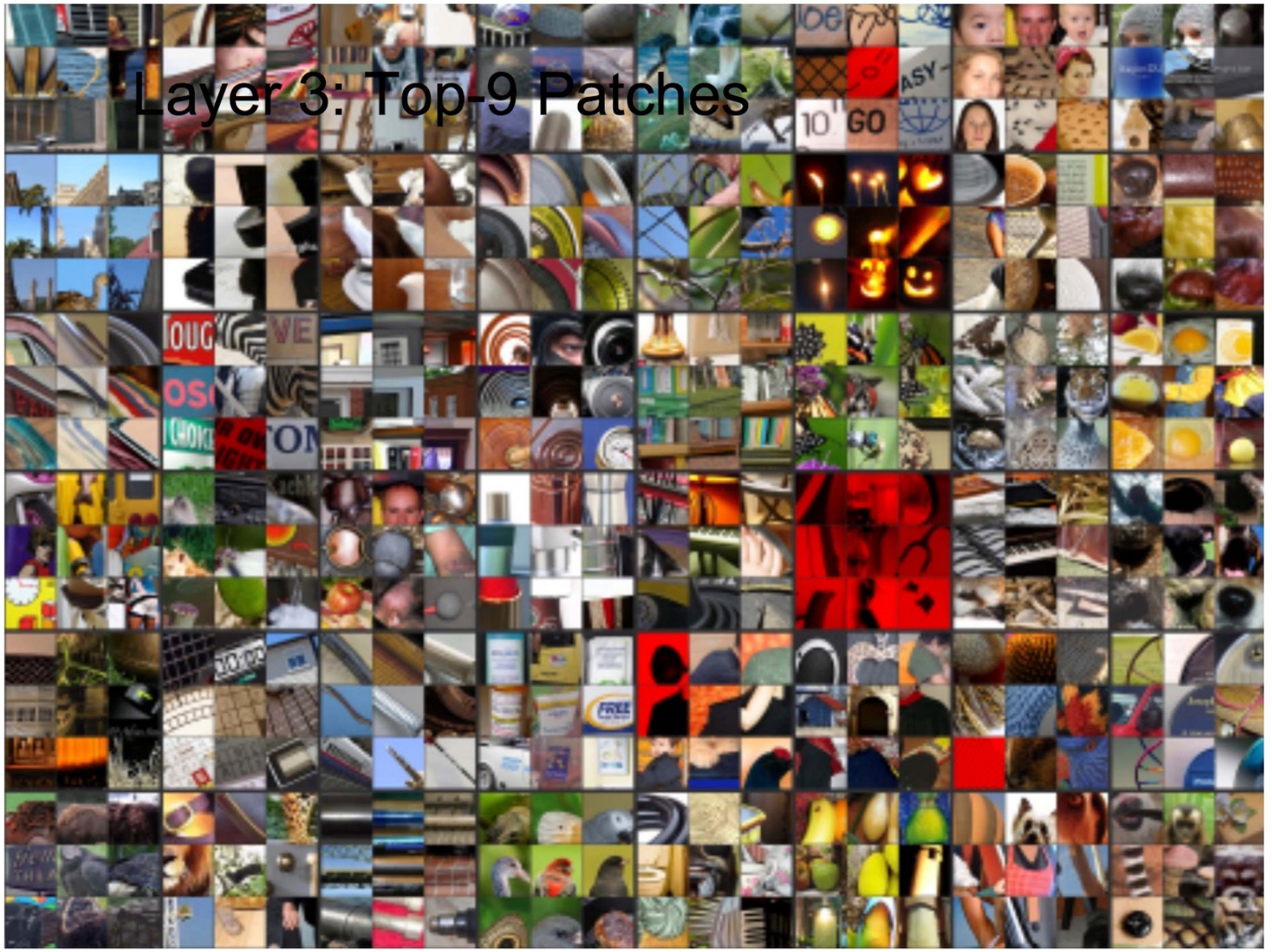
Deconv on Top Image Patches - Layer 2



Matt Zeiler

Filter Visualization

Top Image Patches - Layer 3



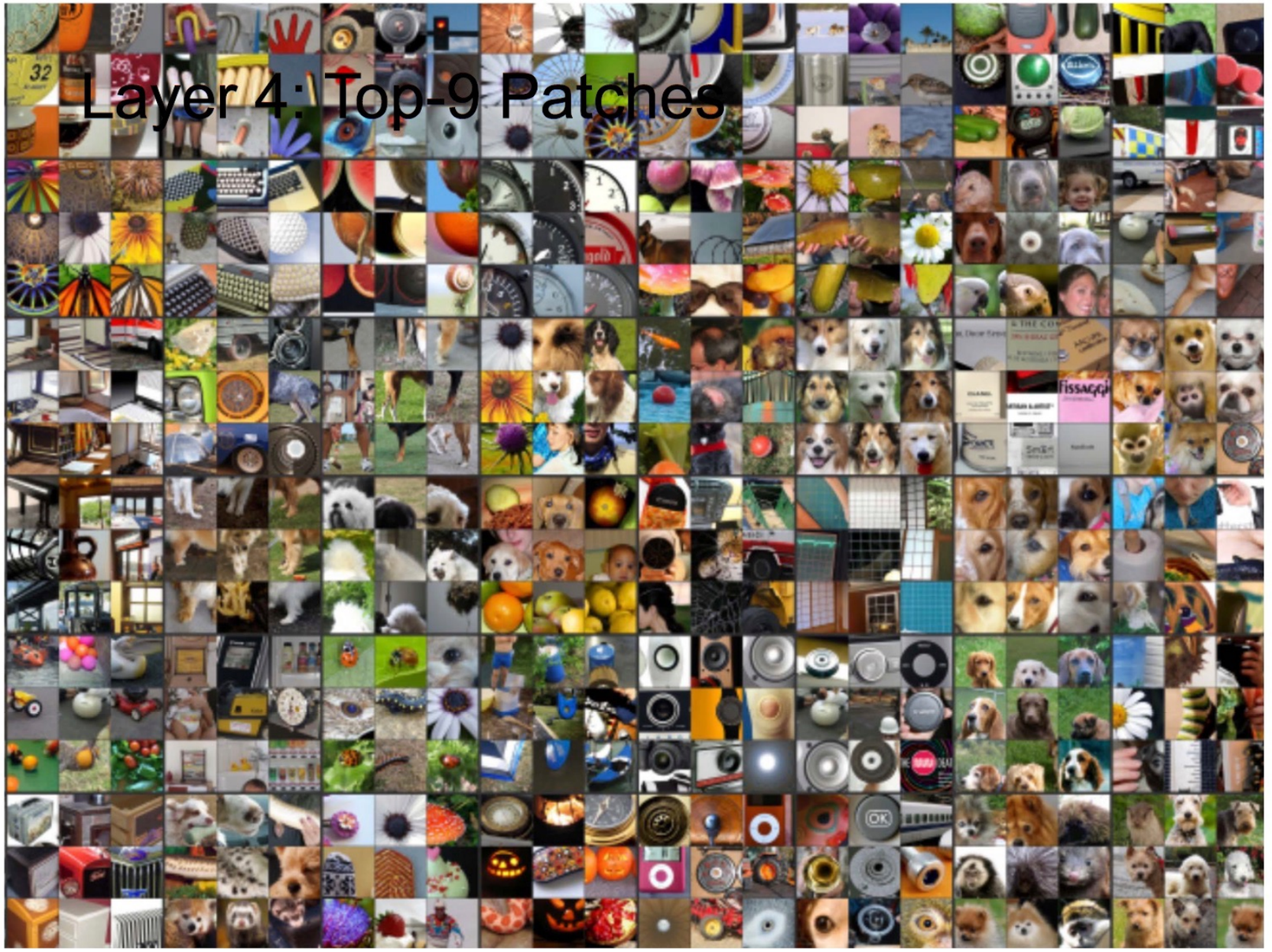
Filter Visualization

Deconv on Top Image Patches - Layer 3

Matt Zeiler

Filter Visualization

Top Image Patches - Layer 4



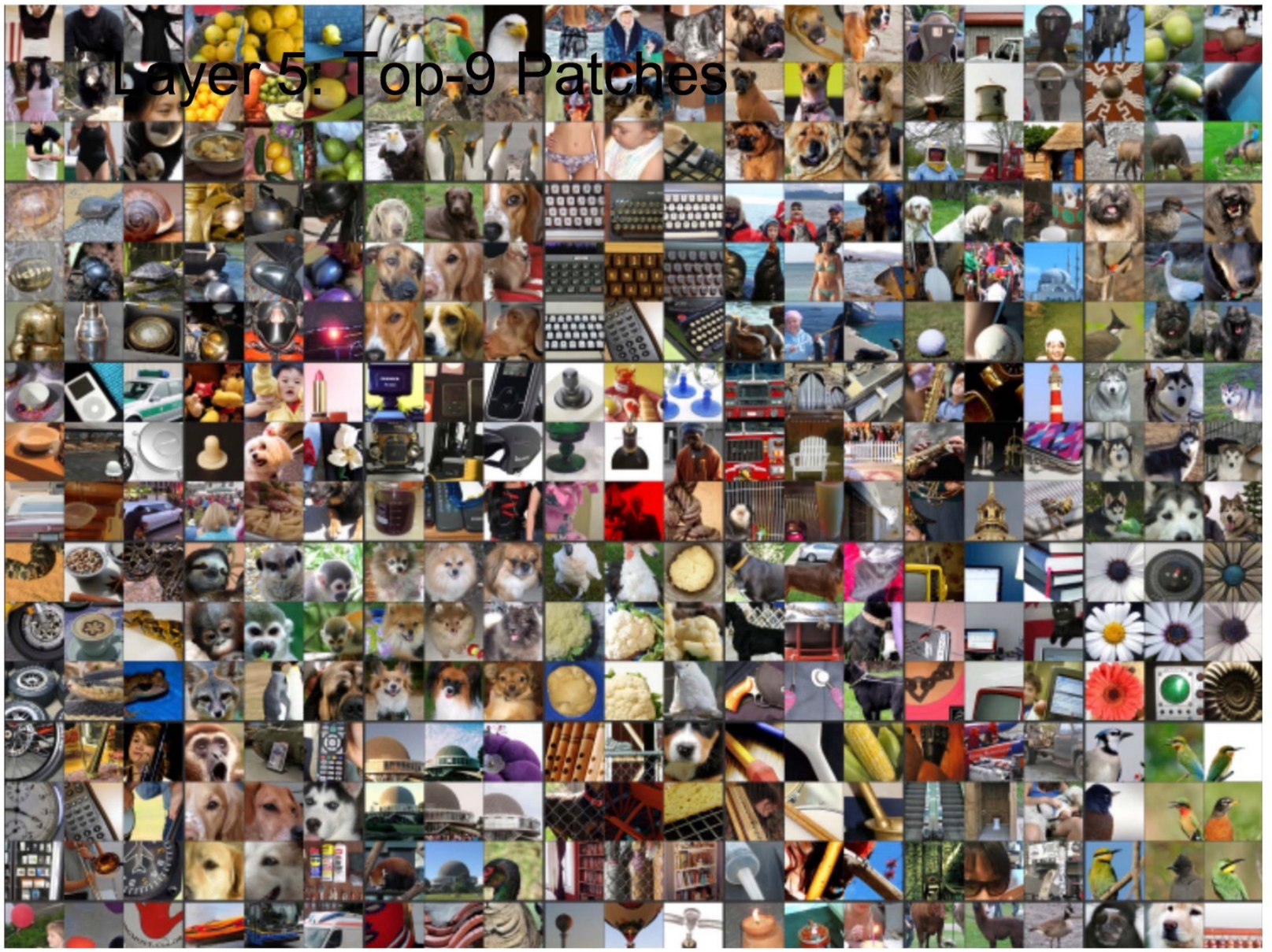
Filter Visualization

Deconv on Top Image Patches - Layer 4

Matt Zeiler

Filter Visualization

Top Image Patches - Layer 5



Filter Visualization

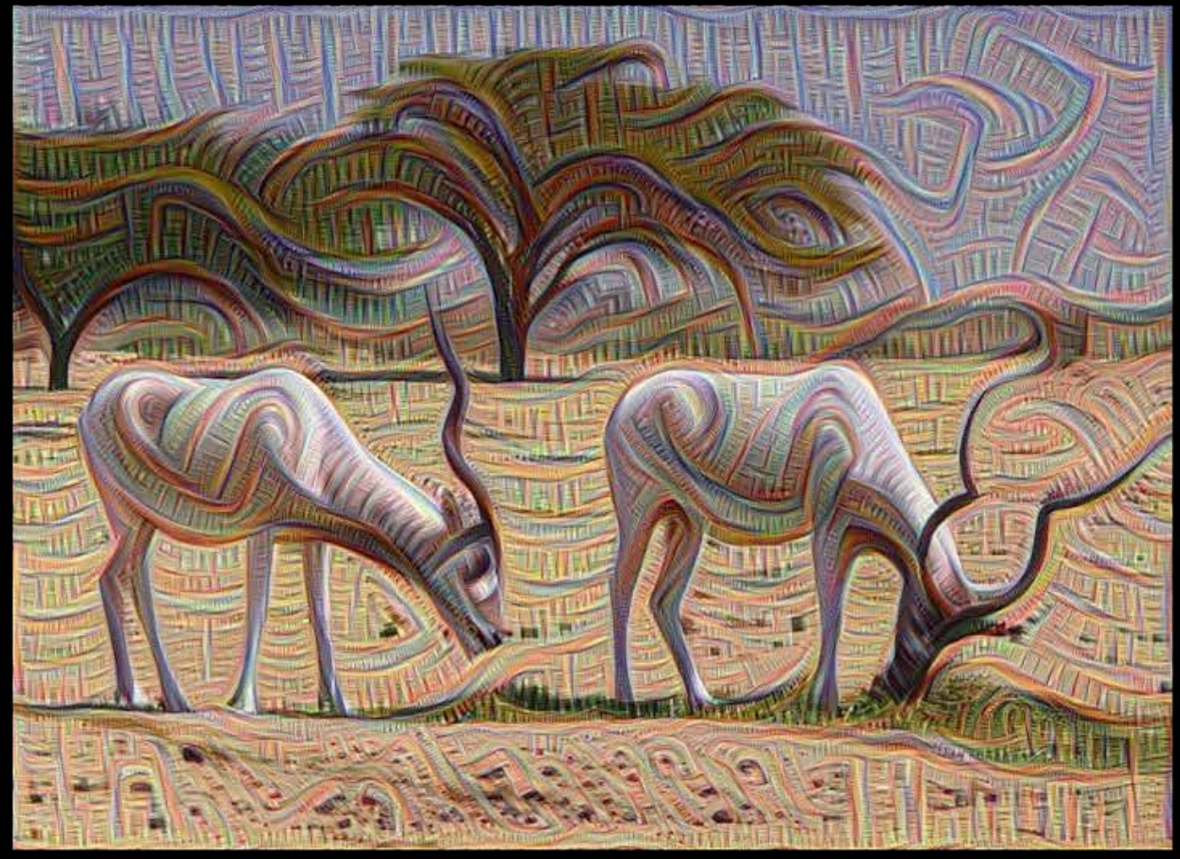
Deconv on Top Image Patches - Layer 5

Deep Dream

Related to visualizing filters through gradient ascent.

Enforce ‘continuity prior’: produced image must have statistics similar to natural images

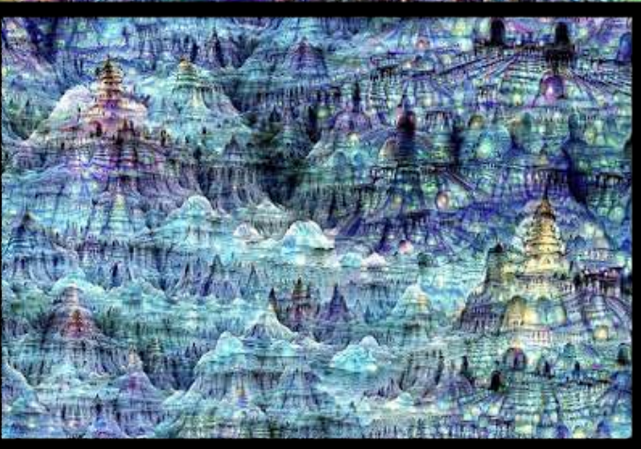
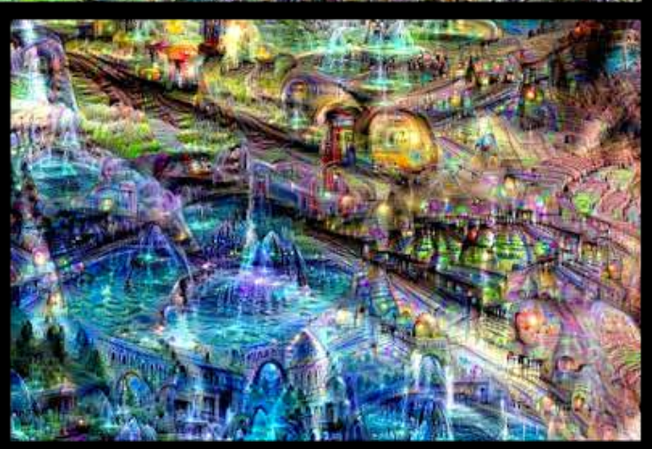
Start from an image, either noise or an actual image. Randomly enhance various filters throughout the network.

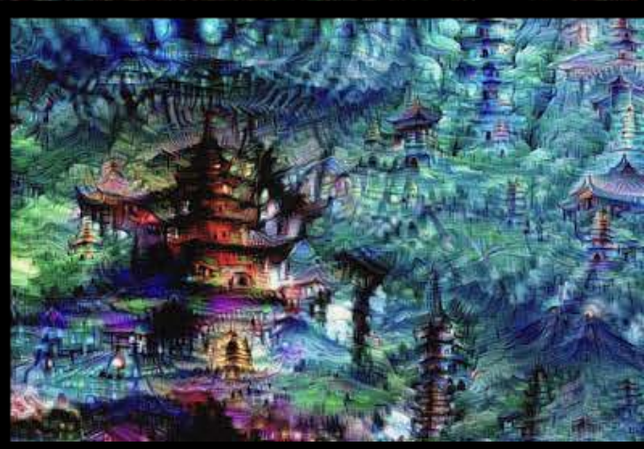


https://research.googleblog.com/2015/06/inceptionism-going-deeper-into-neural.html

Deep Dream





https://research.googleblog.com/2015/06/inceptionism-going-deeper-into-neural.html

Neural Style Transfer

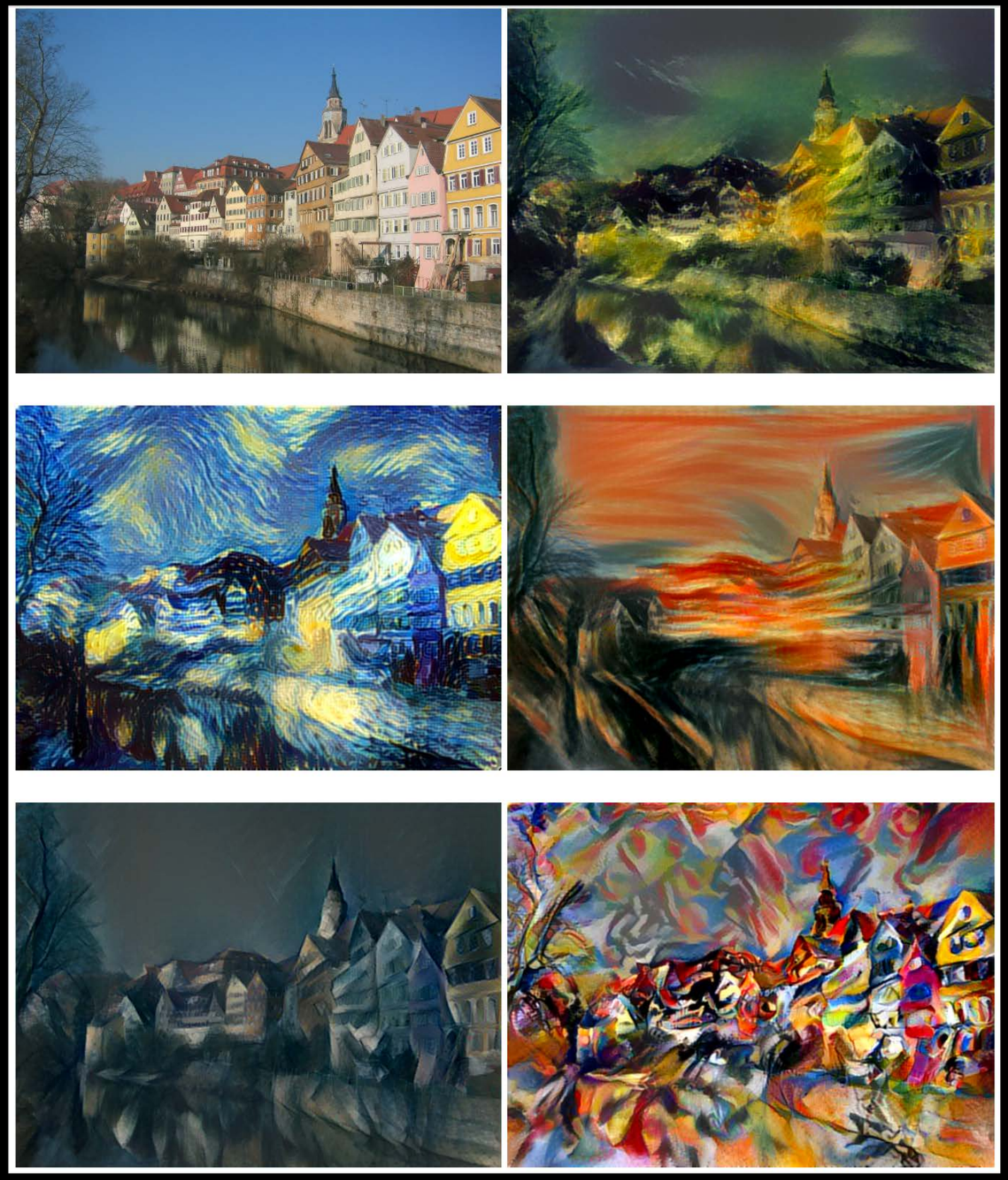
Capture high level statistics of one image, i.e. stylistic essence.

Run gradient ascent on new image to match high level statistics of first image.

Can transfer high-level features between images.

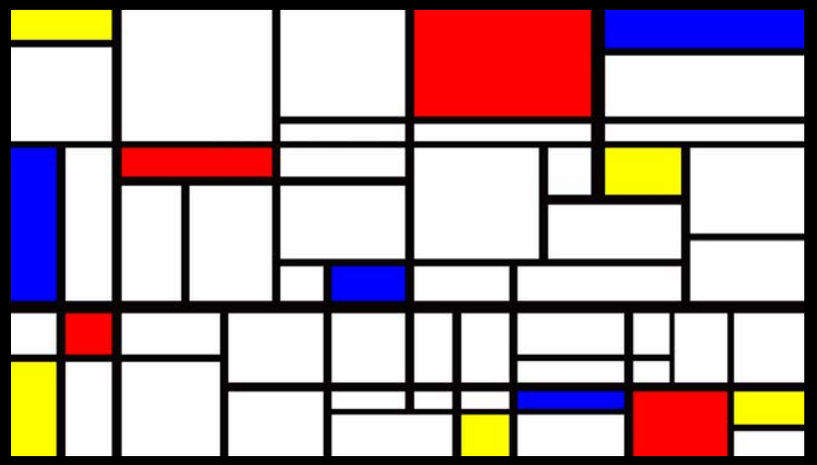
Gatys et al., 2015

Neural Style Transfer



Gatys et al., 2015

Neural Style Transfer



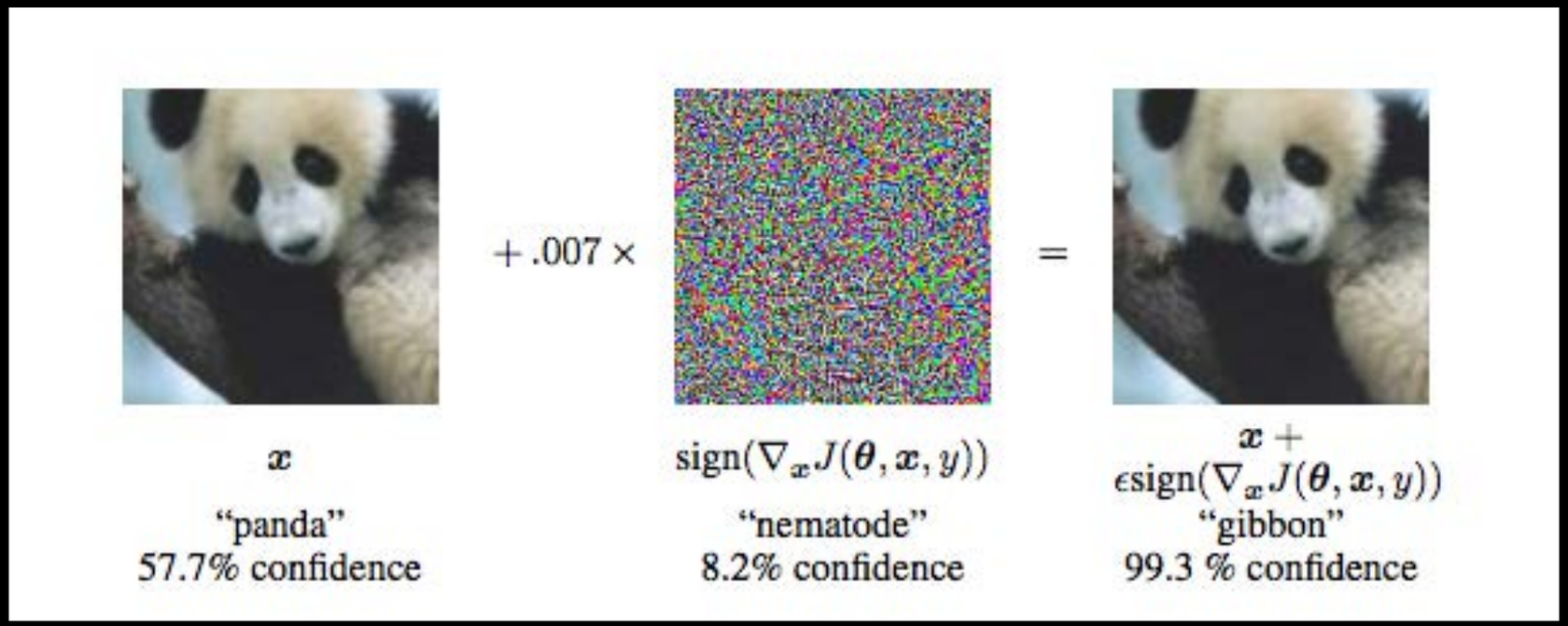
Open Problems

Unsupervised Learning. All training examples need labels, but this is unrealistic.

Limited Understanding/Reasoning. Great at picking out patterns, but no deeper understanding.

Low-Shot Learning. These networks need many training examples of each class. Do not do well with class imbalance.

Limited. How do we make better models?



References

Hubel, David H., and Torsten N. Wiesel. "Receptive fields and functional architecture of monkey striate cortex." The Journal of physiology 195.1 (1968): 215-243.

Freiwald, Winrich A., Doris Y. Tsao, and Margaret S. Livingstone. "A face feature space in the macaque temporal lobe." Nature neuroscience 12.9 (2009): 1187-1196.

Tsao, Doris Y., Sebastian Moeller, and Winrich A. Freiwald. "Comparing face patch systems in macaques and humans." Proceedings of the National Academy of Sciences 105.49 (2008): 19514-19519.

LeCun, Yann, et al. "Backpropagation applied to handwritten zip code recognition." Neural computation 1.4 (1989): 541-551.

Krizhevsky, Alex, Ilya Sutskever, and Geoffrey E. Hinton. "Imagenet classification with deep convolutional neural networks." Advances in neural information processing systems. 2012.

Nair, Vinod, and Geoffrey E. Hinton. "Rectified linear units improve restricted boltzmann machines." Proceedings of the 27th International Conference on Machine Learning (ICML-10). 2010.

Zeiler, Matthew D., and Rob Fergus. "Visualizing and understanding convolutional networks." European Conference on Computer Vision. Springer International Publishing, 2014.

Ioffe, Sergey, and Christian Szegedy. "Batch normalization: Accelerating deep network training by reducing internal covariate shift." arXiv preprint arXiv:1502.03167 (2015).

Srivastava, Nitish, et al. "Dropout: a simple way to prevent neural networks from overfitting." Journal of Machine Learning Research 15.1 (2014): 1929-1958.

He, Kaiming, et al. "Deep residual learning for image recognition." arXiv preprint arXiv:1512.03385 (2015).

He, Kaiming, et al. "Identity mappings in deep residual networks." arXiv preprint arXiv:1603.05027 (2016).

Deng, Jia, et al. "Imagenet: A large-scale hierarchical image database." Computer Vision and Pattern Recognition, 2009. CVPR 2009. IEEE Conference on. IEEE, 2009.

Simonyan, Karen, and Andrew Zisserman. "Very deep convolutional networks for large-scale image recognition." arXiv preprint arXiv:1409.1556 (2014).

Szegedy, Christian, et al. "Going deeper with convolutions." Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition. 2015.

Szegedy, Christian, et al. "Rethinking the inception architecture for computer vision." arXiv preprint arXiv:1512.00567 (2015).

Gatys, Leon A., Alexander S. Ecker, and Matthias Bethge. "A neural algorithm of artistic style." arXiv preprint arXiv:1508.06576 (2015).

Szegedy, Christian, et al. "Intriguing properties of neural networks." arXiv preprint arXiv:1312.6199 (2013).

Goodfellow, Ian J., Jonathon Shlens, and Christian Szegedy. "Explaining and harnessing adversarial examples." arXiv preprint arXiv:1412.6572 (2014).