Object Recognition and Deep Convolutional Networks

Steven Mills

AI and Society

Object Recognition

Object recognition

- Given an image, assign a label
- E.g.: Bus, Dog, Koala, Man
- People are pretty good at this
  - Still some ambiguity
  - Animal vs. Dog vs. Labrador
  - Multiple objects in a scene
- Computers are not good at this
  - They are getting better quickly
  - Deep convolutional networks



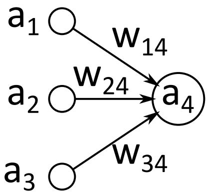

Neural Network Basics

Artificial Neural Networks

- Inspired by biology
- Simple units (neurons) and connections
- Neurons have activation values, a_i
- Connections have weights, w_{ij}
- Neurons compute functions like

$$a_j = \sum_i w_{ij} a_i$$

- Training by learning the weights

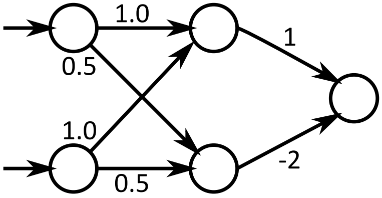


$$a_4 = w_{14} a_1 + w_{24} a_2 + w_{34} a_3$$

Deep Networks

Some tasks require layers of neurons

- ‘Hidden’ layers between input/output
- Needed when problems are not *linearly separable*
- Simple example: XOR
- Deep networks – many hidden layers
- Inspired by real brains
- Early layers compute basic features
- Later layers recognise objects etc.



XOR network – neurons are binary and activate if their input is ≥ 1

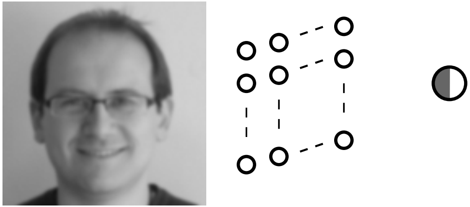
Neural Networks and Images

Our input will be an image

- One neruon per pixel?
  $\Longrightarrow$ Lots of neurons
  $\Longrightarrow$ Lots of weights to learn
- Fully connected networks not practical

Example: Face detection

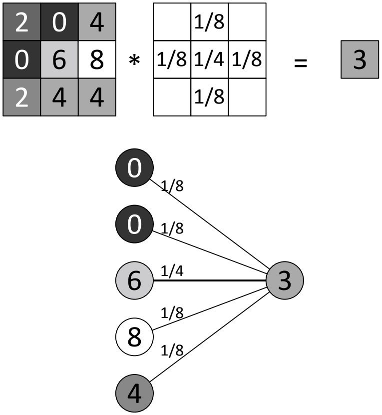
- Input layer a 100×100 image
- Output layer one neuron (face/not)
- Suppose one 20×20 unit hidden layer
- 4,000,000 weights to learn



Convolutional Neural Networks (ConvNets)

Convolution at each pixel

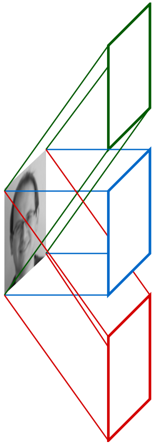
- Convolution is based on a small local neighbourhood
- The same kernel is applied at each point in the image
- One neuron per pixel
- Only local connections
- Weights shared between hidden neurons
- 100×100 pixel image, 3×3 kernel
  - $98 \times 98 = 9,604$ hidden neurons
  - $9604 \times 9 = 8,6436$ connections
  - Only 9 weights to learn



Convolutional Neural Networks (ConvNets)

We can apply multiple convolutions

- Several layers of hidden neurons
- Each neuron is connected to a local neighbourhood of pixels
- This is its area of support, region of interest, or receptive field
- The weights for each neuron in a hidden layer are the same
- Each layer learns a low-level feature
- May need quite a few of these



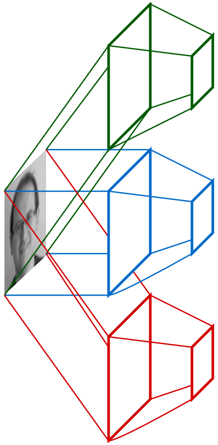
Convolutional Neural Networks (ConvNets)

The individual layers are quite large

- Training them is expensive
- There is a lot of redundant information
- They operate at a very fine scale

Pooling groups pixels together

- Typically takes a 2×2 patch
- These patches do not overlap
- Returns a simple function of the values
- Max-pooling is commonly used
- Reduces size of layers by 4

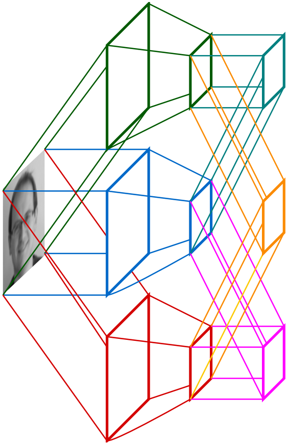


Convolutional Neural Networks (ConvNets)

Later layers also use convolution

- Their input is one or more earlier layers
- Locally connected to input layers
- This combines different features at the same image location
- These can learn higher-level features
- Layers of convolution and pooling build up more and more complex behaviour

Build up layers of convolution and pooling



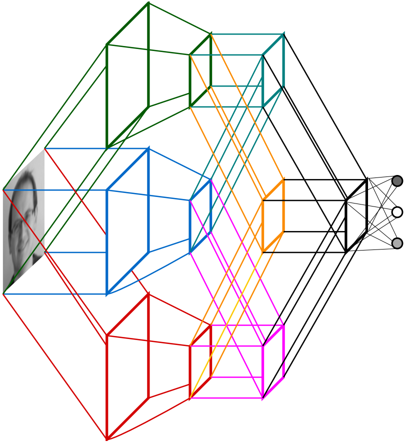
Convolutional Neural Networks (ConvNets)

Eventually we need to output something

- This might be a binary classification
- More often several output classes
- Activation represents confidence

These last layers are often fully connected

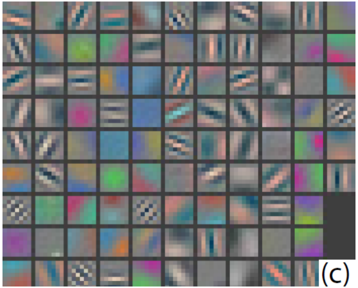
- Bring together multiple features
- Combine information across the image
- Size of layers usually fairly small



Visualising ConvNets

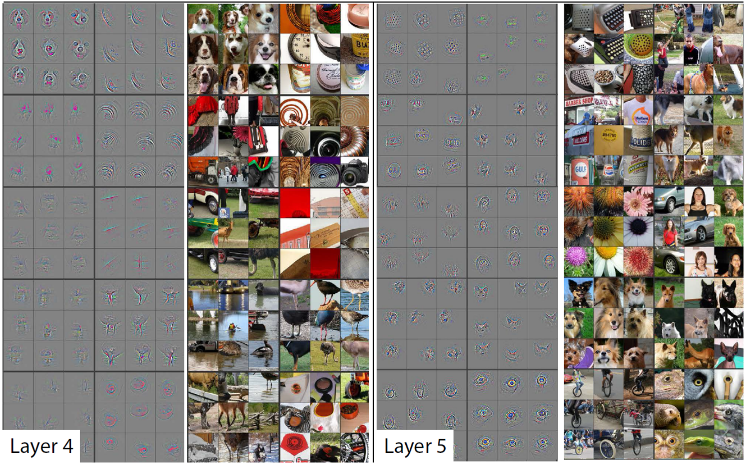
We can visualise each ConvNet

- Work backwards from the weights
- Make a characteristic image that gives the greatest response
- The first layer looks like *Gabor filters*
- Colour gradients also common
- Higher layers respond to more complex patterns



Layer 1 features, from Zeiler and Fergus, *Visualizing and Understanding Convolutional Networks*, ECCV 2014

Visualising ConvNets

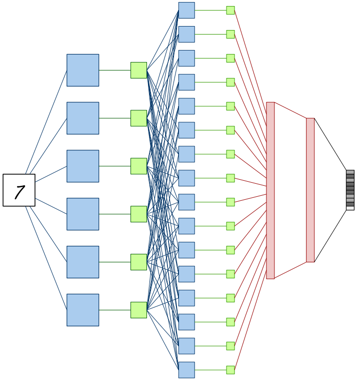


Layer 4 and 5 responses, from Zeiler and Fergus, *Visualizing and Understanding Convolutional Networks*, ECCV 2014

Example: LeNet5

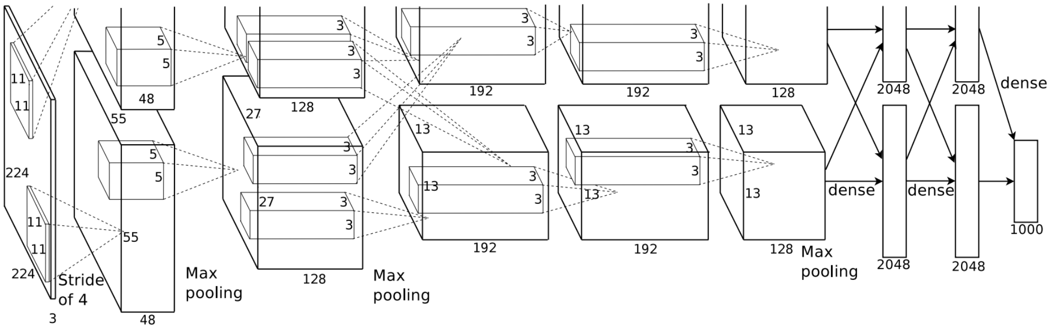
Recognising hand-written digits

1. Input is a 32×32 image
2. Six 28×28 convolutional maps with 5×5 connections
3. Non-linear pooling to 14×14
4. 16 10×10 convolutional maps with 5×5 connections
5. Non-linear pooling to 5×5
6. Complete connections to 120 neurons
7. Complete connections to 84 neurons
8. Gaussian connections to 10 output neurons



LeNet-5 Architecture, Adapted from LeCun et al., *Gradient-Based Learning Applied to Document Recognition*, Proc. IEEE 1998

Example: ImageNet Classifier

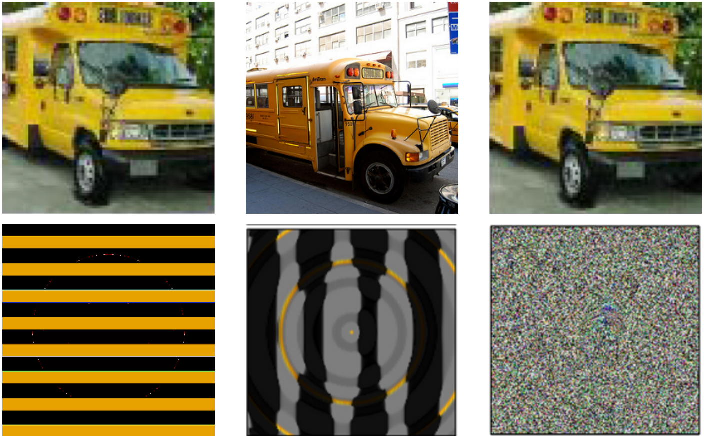


CNN Architecture, from Krizhevsky et al., *ImageNet Classification with Deep Convolutional Neural Networks*, Adv. in Neural Information Processing Systems 2012

Problems with Deep Networks

- Designing and training these networks is not easy
  - How many layers? How many filters? How large are the filters? Connectivity? What sort of pooling?
  - Need very large sets of labelled training data
  - Needs *lots* of computation to learn the weights
- We don't have any rules to guide us here
  - Some people become good at making design decisions
  - They can't adequately explain what they are doing
- Can a machine learn to design deep networks?

Problems with Deep Networks – Spot the School Bus



Szegedy et al., *Intriguing Properties of Neural Networks*, ICML 2014

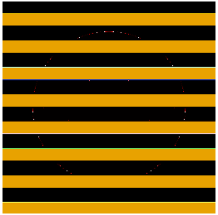
Nguyen et al., *Deep Neural Networks are Easily Fooled: High Confidence Predictions for Unrecognisable Images*, CVPR 2015

Problems with Deep Networks – Spot the School Bus

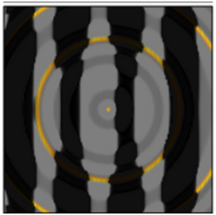
School bus

(No data)

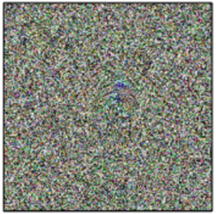
Not a bus



School bus



King penguin



Peacock