COSC343: Artificial Intelligence

Lecture 13 : Deep learning

Lech Szymanski

Dept. of Computer Science, University of Otago

In today's lecture

- Hierarchical representation
- Deep learning
- Convolutional Neural Networks
- AlphaGo

Network with many hidden layers

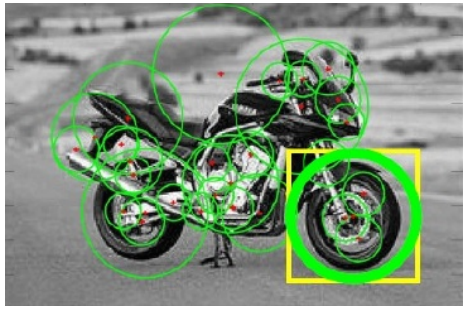
While it's possible to encode any function in a network with one hidden layer, it's often easier to encode complex functions in **deep networks** with many hidden layers.

The intuition:

- The first layer learns simple patterns...
- The next layer learns patterns of these patterns...and so on.

Visual object classification is a type of application where this *hierarchical feature* detection is very effective.

The hierarchical structure of object classification



- A motorbike has wheels, a seat, an engine...
- A wheel has a tyre, an axle, spokes...
- A tyre has treads, an axle has shiny bits...
- ...
- Lines, dots...
- Pixels

Deep learning

In the past couple of years, deep learning has become a dominant machine learning device for 'big', 'hard' learning tasks.

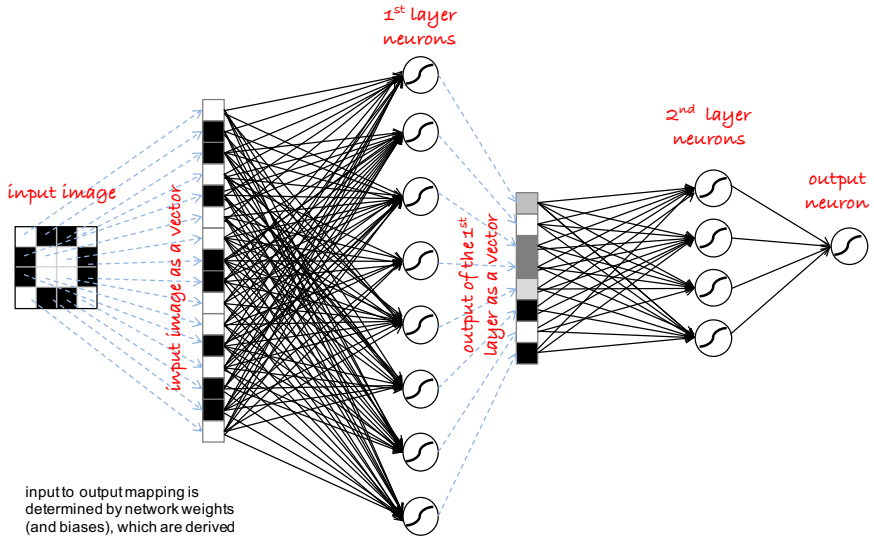
Deep learning consists of two different approaches to training artificial neural networks:

- Deep Belief Nets
  (invented by Geoffrey Hinton)

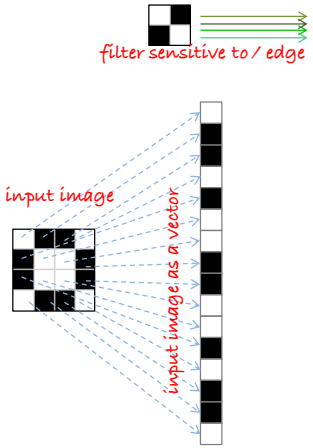
- Convolutional Neural Networks
  (invented by Yann LeCun)

we will only cover Convolutional Neural Networks

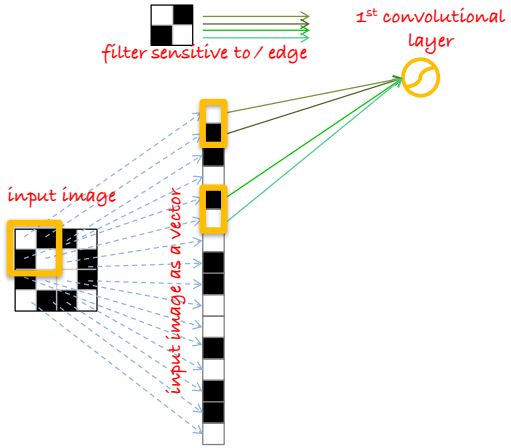
Processing images in a feed forward neural network



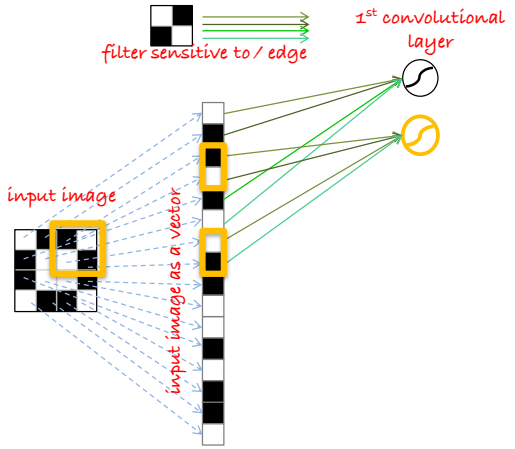
Processing images in a convolutional neural network



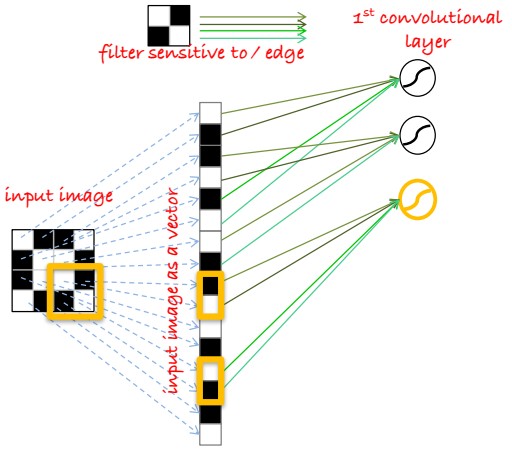
Processing images in a convolutional neural network



Processing images in a convolutional neural network

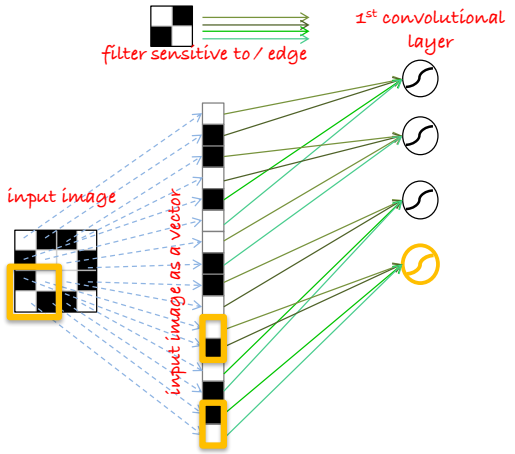


Processing images in a convolutional neural network



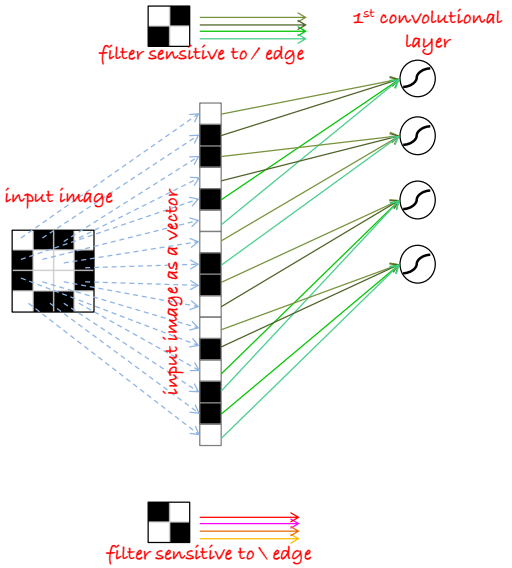
Lech Szymanski (Otago) COSC343 Lecture 13

Processing images in a convolutional neural network



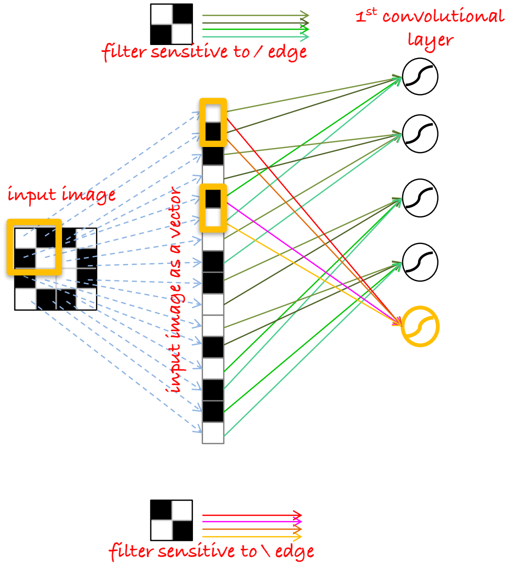
Lech Szymanski (Otago) COSC343 Lecture 13

Processing images in a convolutional neural network

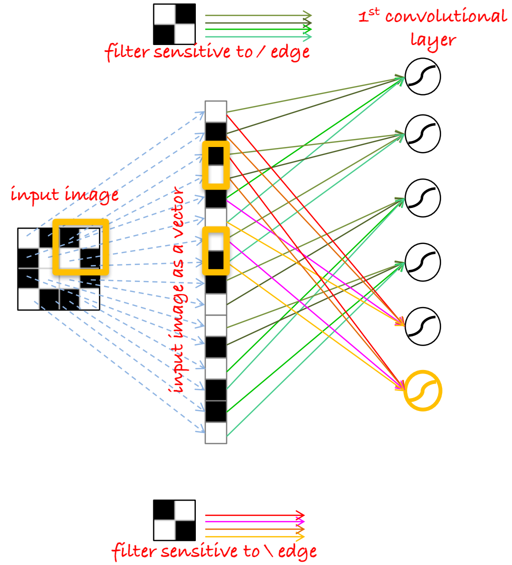


Lech Szymanski (Otago) COSC343 Lecture 13

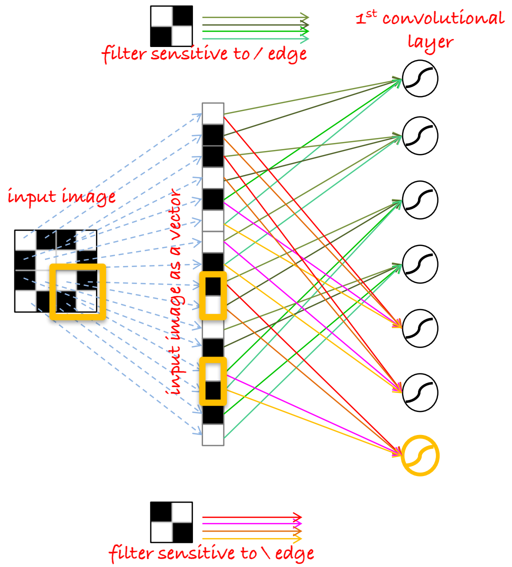
Processing images in a convolutional neural network



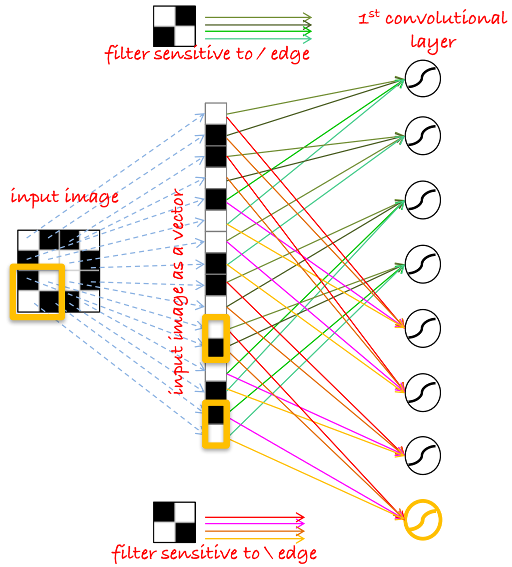
Processing images in a convolutional neural network



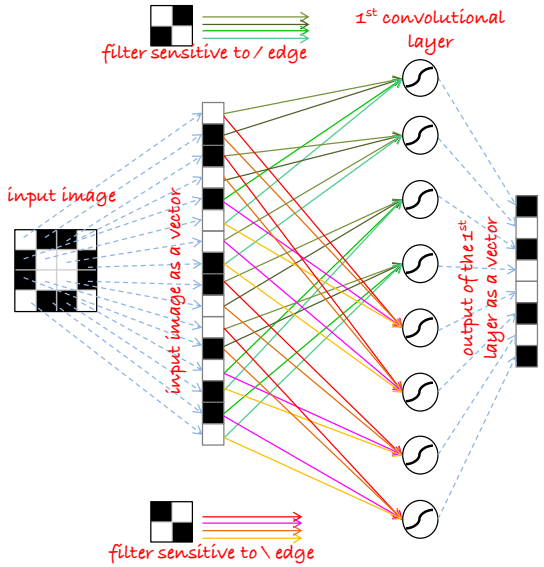
Processing images in a convolutional neural network



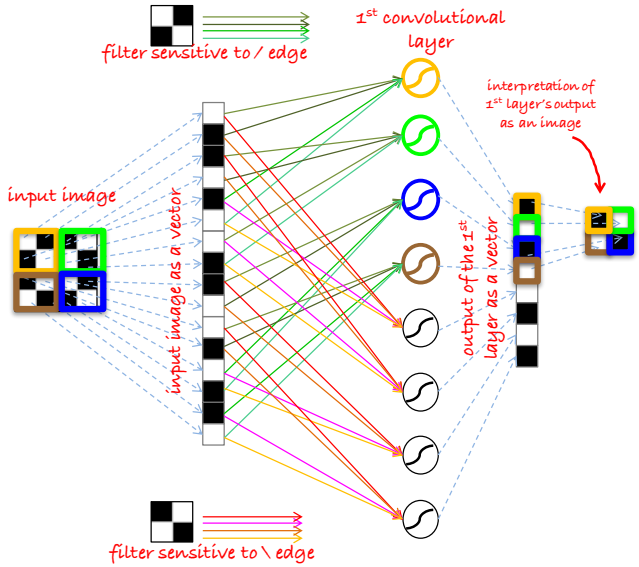
Processing images in a convolutional neural network



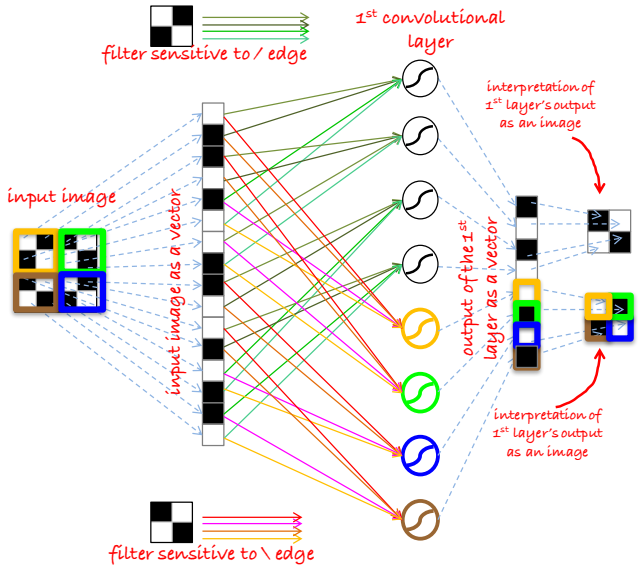
Processing images in a convolutional neural network



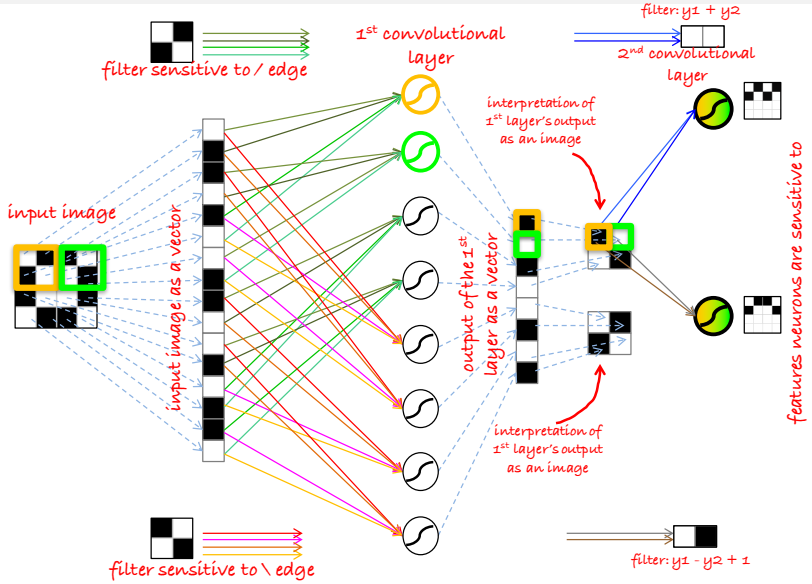
Processing images in a convolutional neural network



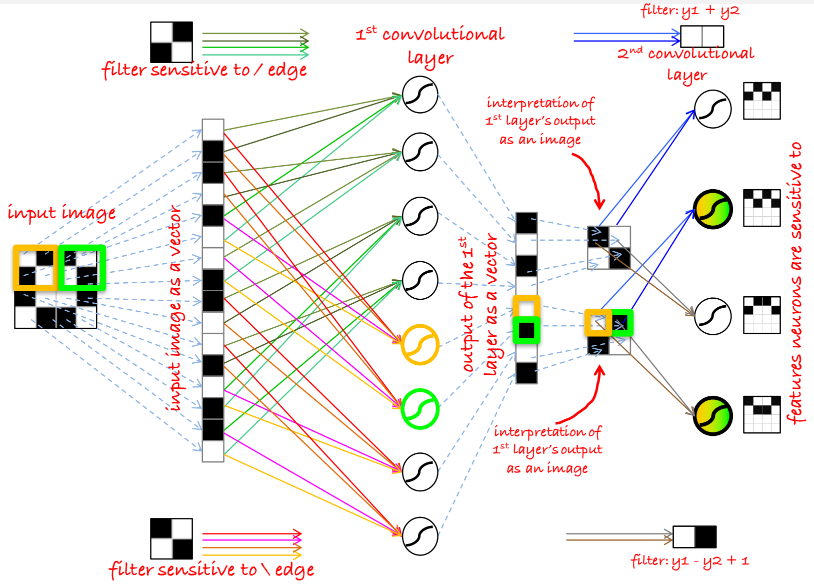
Processing images in a convolutional neural network



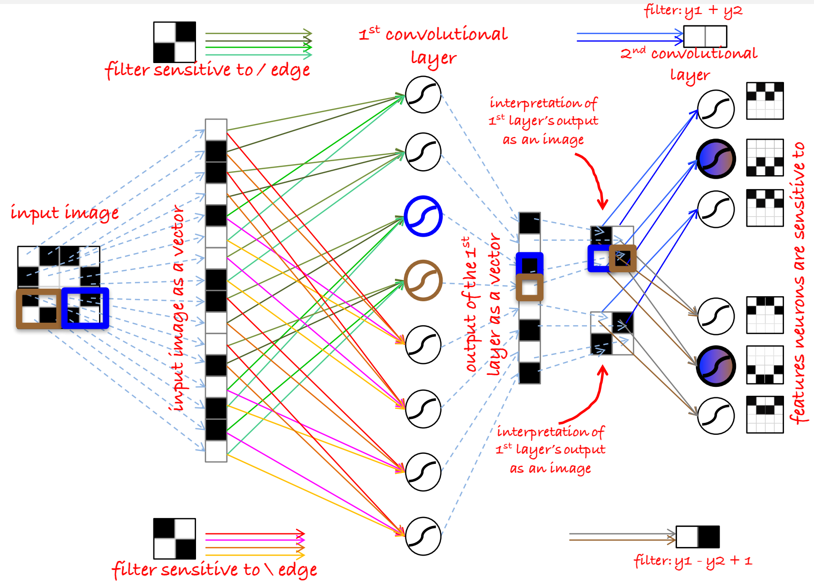
Processing images in a convolutional neural network



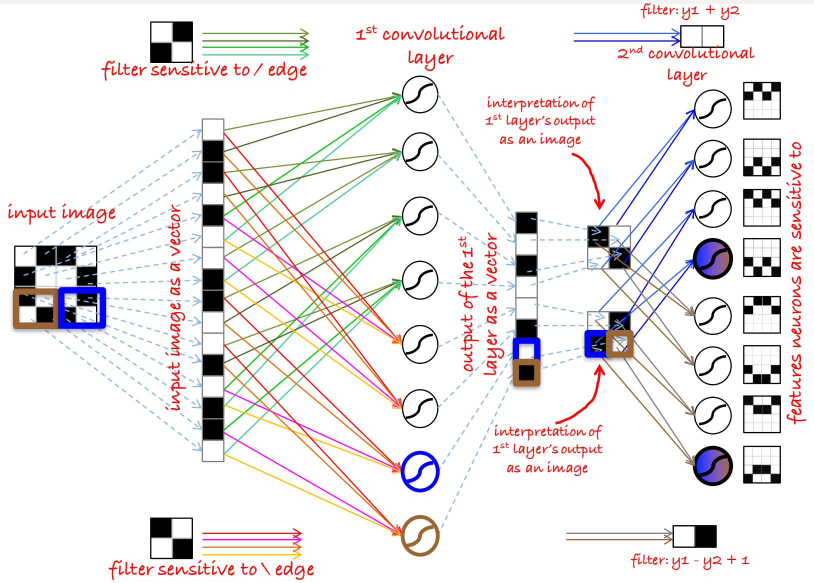
Processing images in a convolutional neural network



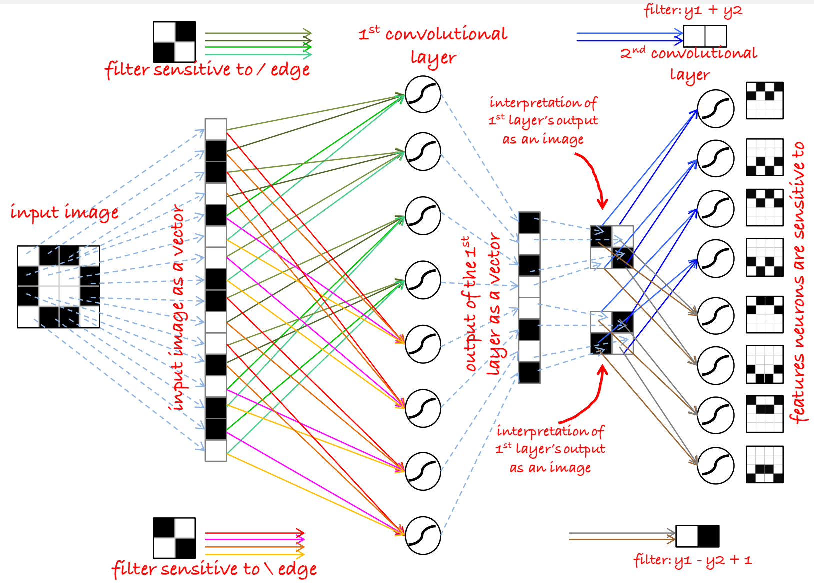
Processing images in a convolutional neural network



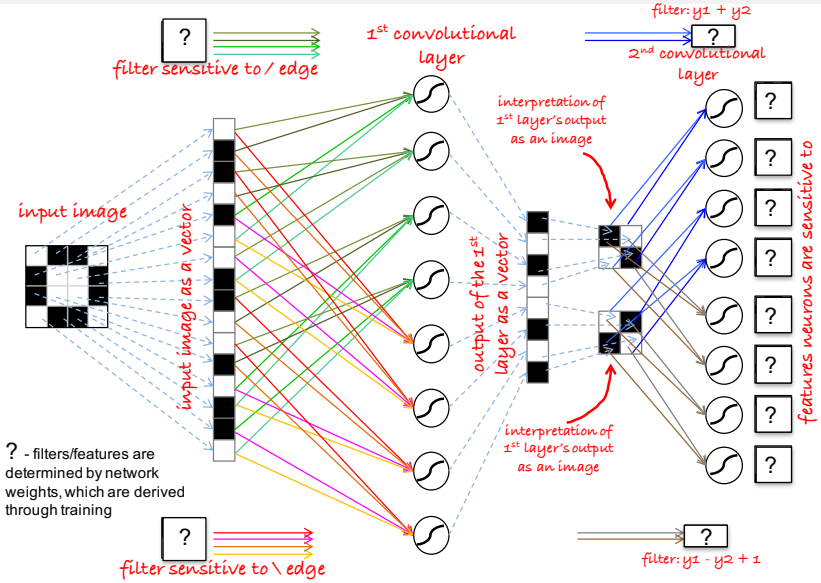
Processing images in a convolutional neural network



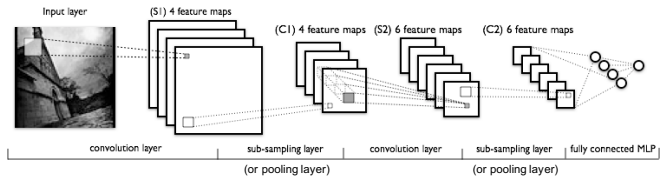
Processing images in a convolutional neural network



Processing images in a convolutional neural network



Convolutional neural network (convNet) parameters



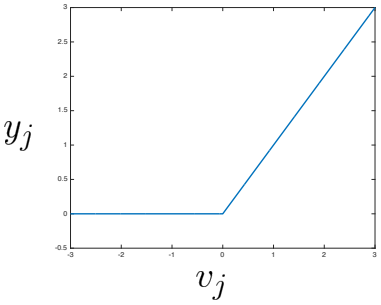
- The entire model is trained using backpropagation:
  - Typically with the softmax function at the output (for classification)
  - The pooling layer has no trainable parameters, but it can pass error "blame" backward
  - Convolutional layers just needs some additional averaging of the updates across given filter's weights
  - Rectifier linear unit (ReLU) activation function is typically used in convolutional layers
  - A sigmoid function is typically used in the fully connected MLP

ReLU activation function

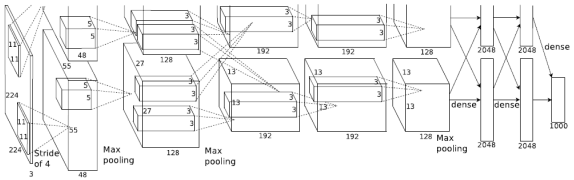
Rectifier Linear Unity activation function

$$y_j = f_{\text{relu}}(v_j) = \begin{cases} v_j & v_j > 0 \\ 0 & \text{otherwise} \end{cases}$$

$$\frac{dy_j}{dv_j} = \begin{cases} 1 & v_j > 0 \\ 0 & \text{otherwise} \end{cases}$$

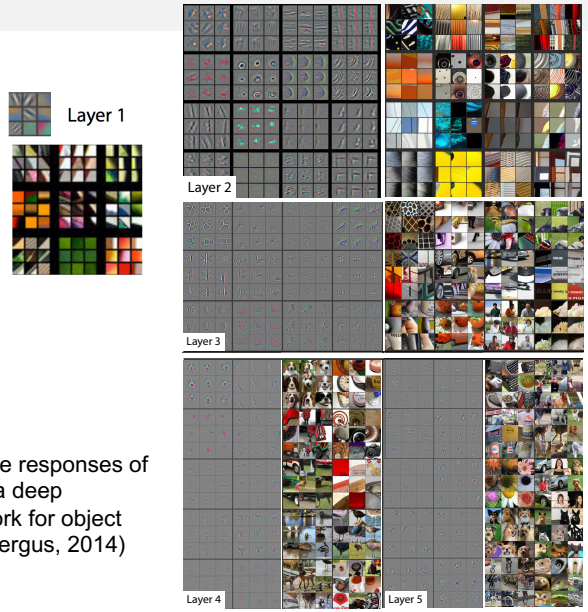


convNet network architecture



- As is typical for neural networks, the training cannot determine the architecture
- In a convNet there are many more things to set, because layers are not fully connected
- For each convolutional layer, the user must specify
  - Number of filters
  - Size of the filters
  - The step size (how a given filter is shifted over the image map)
  - The padding (whether filters extend beyond the edge of the image)
- For each pooling layer, the user must specify
  - The size of the pooling window (the subsampling ratio)
  - The type of pooling (max or average)
- The user must decide on the number of convolutional+pooling layers and the architecture of the fully connected MLP that follows

convNet features

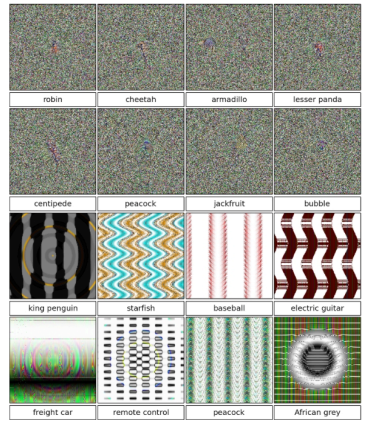


Here's a visualisation of the responses of units in different layers of a deep convolutional neural network for object classification (Zeiler and Fergus, 2014)

convNet problems

convNets do throw away location information:

- As long as a combination of features is found, the actual order of those features might be ignored
- As a result, it's quite easy to "fool" convNet into bad classification (Nguyen, Yosinski, Clune, 2015)
- As any other learning model, they perform well when test data is not too different from the training data...
- ...and so state of the art deep learning network are trained on HUGE training sets (utilising parallel nature of neural networks to speed up the computation with lots of GPUs



Example: how AlphaGo works

- **AlphaGo versus Lee Sedol** was a five-game Go match between South Korean professional Go player Lee Sedol and AlphaGo, a computer program developed by Google DeepMind.
- Played in Seoul, South Korea between 9 and 15 March 2016.
- AlphaGo won 4 out of 5 games.
- AlphaGo's victory was a major milestone in artificial intelligence research.
- Go had previously been regarded as a hard problem in machine learning that was expected to be out of reach for the technology of the time.

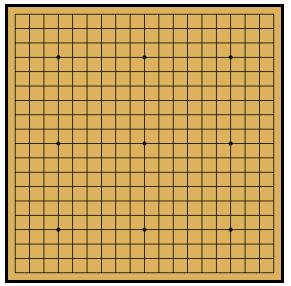
AlphaGo relies heavily on a Convolutional Neural Network

Example: how Alpha Go works

Silver et al., "Mastering the game of Go with deep neural networks and tree search", *Nature* 529, 484–489 (28 January 2016) doi:10.1038/nature16961.

Game of Go:

- Two players alternately place black and white stones on the vacant intersections of a board with a 19×19 grid of lines
- Objective is to surround a larger total area of the board with one's stones than the opponent.
- Perfect information game
- $2x10^{170}$ possible board states
- Average game has 150 moves, with an average of about 250 choices per move

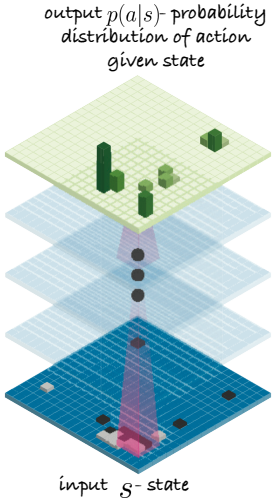


Katpatuka (Wikipedia)

Example: how Alpha Go works

Policy model:

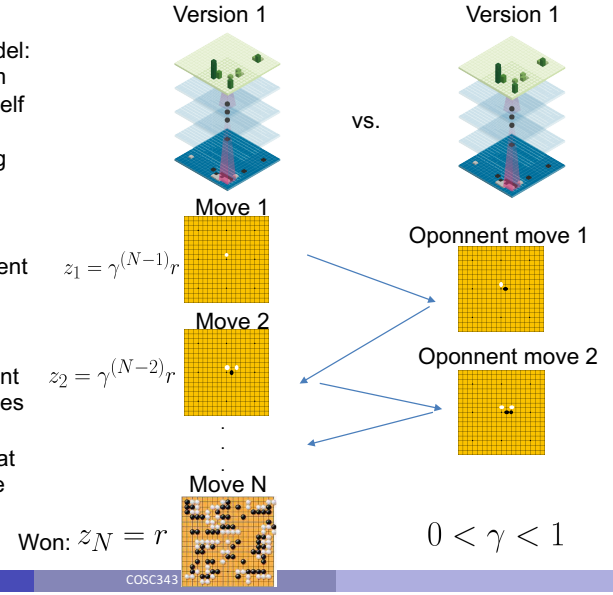
- Its job is to predict next action to take (next move) given the current state
- A convolutional neural network
  - Input: the board state as an image
  - Output: an image of the board with probability distribution of placing a stone at a given position
  - Softmax activity at the output
  - 13 layers (convolutional + pooling)
  - Trained on 30 million moves done by experts (input is state, the desired output is a move made by an expert)
  - 56% correct (predicting the expert's move) after training



Example: how Alpha Go works

Improving the Policy model:

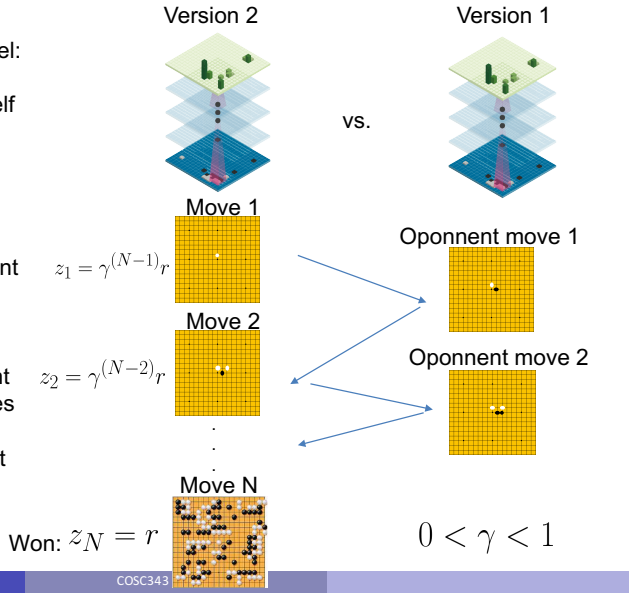
- Play against a random previous version of itself
- Policy gradient reinforcement learning
  - Save state of the network at every move
  - Reward/punishment given once game won/lost
  - Propagate reward/punishment back through states and change the parameters so that expectation of the reward goes up



Example: how Alpha Go works

Improving the Policy model:

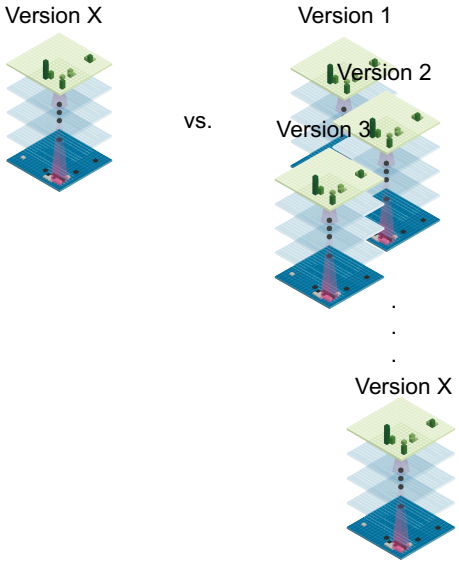
- Play against a random previous version of itself
- Policy gradient reinforcement learning
  - Save state of the network at every move
  - Reward/punishment given once game won/lost
  - Propagate reward/punishment back through states and change the parameters so that expectation of the reward goes up



Example: how Alpha Go works

Improving the Policy model:

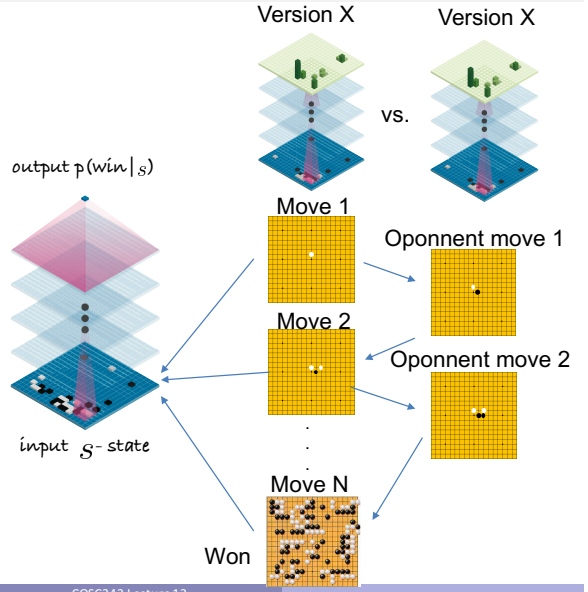
- Play against a random previous version of itself
- Policy gradient reinforcement learning
  - Save state of the network at every move
  - Reward/punishment given once game won/lost
  - Propagate reward/punishment back through states and change the parameters so that expectation of the reward goes up



Example: how Alpha Go works

Value function:

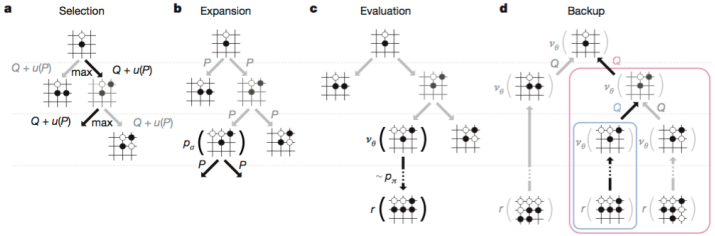
- Predict chance of winning given game played by strongest policy against itself
- A convolutional neural network:
  - Similar architecture as policy model
  - Input is the game state
  - Output is a single value – chance of winning from the current state
  - Trained using stochastic gradient descent minimizing MSE



Example: how Alpha Go works

When making a move in a real game, use a tree search algorithm to look many moves ahead…

- Use the policy network to generate possible own (and opponents optimal) moves
- Use the value network to estimate chances of winning for a given scenario



This is still a massive tree search, but computationally manageable, because…

- Considering only several actions, deemed best by the policy
- Evaluation of the value (chance of winning) for given action is fast

Summary

- Deep learning – just neural networks with many hidden layers and clever constraints on the architecture
- Convolutional neural networks
  - Hierarchical features
  - Set stacked, trainable filters that selectively seek higher abstraction features in further layers of the network
  - Today's state of the art machine learning for object recognition in images and speech processing

Reading for the lecture: No reading

Reading for next lecture: AIMA Section 4.1.4