# Project: Object Recognition in Photographs

Changning Shou, Jun Yong Go

# Agenda

- Simple CNN for object recognition
- Deeper CNN
- Even deeper CNN
- Experimenting with data augmentation
- Other Techniques

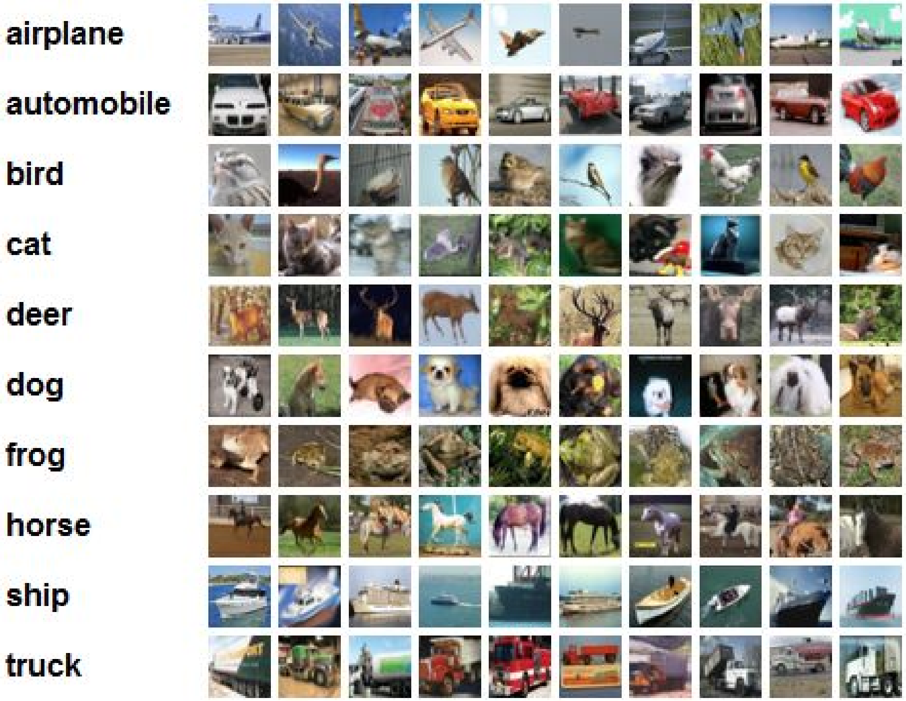
# Setup

- Nvidia GTX 1060
- CNN
  - Better than ANN’s
  - Translational invariance
    - Number of pooling layers and filter size are secondary to data augmentation (Source)
  - Faster due to parameter sharing
    - Each unit in a feature map shares the same weight matrix (Source)
- CIFAR-10 Dataset
- Keras

# Photograph Object Recognition Dataset

CIFAR-10 Dataset

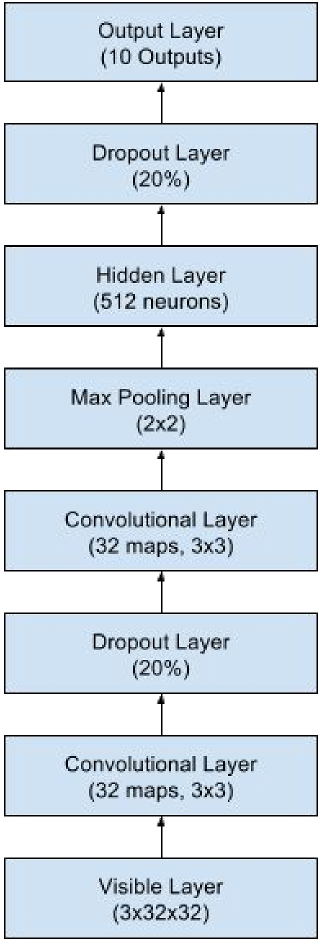
- 60,000 photos divided into 10 classes
  - 50,000 / 10,000
- 32 x 32 pixel images
- Performances:
  - 94% Human performance
  - 96% State-of-the-art results




Source

# Simple CNN: Architecture

- Pattern:
  - Convolutional
  - Dropout
  - Convolutional
  - Max Pooling




Source

# Simple CNN: Parameters

- Seed: 7
- Momentum: 0.9
- Learning Rate: 0.01
- Epochs: 25
- Batch Size: 32
- Logarithmic loss function

# 71%

Simple CNN Accuracy

# Deep CNN: Architecture

- Same pattern as in simple CNN
- Pattern repeated 3 times with 32, 64, and 128 feature maps
  - Increasing number of feature maps with decreasing size given the max pooling layers
  - Additional and larger Dense layers at the output end to better translate the large number of feature maps to class values

# Deep CNN: Parameters

- Seed: 7
- Momentum: 0.9
- Learning Rate: 0.01
- Epochs: 25
- Batch Size: 64
- Logarithmic loss function

# 77%

Deep CNN Accuracy

~15 min to run

# Deeper CNN - Feature Maps: Architecture

- More feature maps closer to input
    - Deep CNN ➔ 6 convolutional layers
    - 32, 32, 64, 64, 128, 128 ➔ 64, 64, 64, 128, 128, 128

# Deeper CNN - Feature Maps: Parameters

- Seed: 7
- Momentum: 0.9
- Learning Rate: 0.01
- Epochs: 25
- Batch Size: 64
- Logarithmic loss function

# 80%

Deeper CNN Accuracy - More Feature Maps

# Deeper CNN - Less Pooling: Architecture

- Less aggressive pooling
- 2x2 ➔ 3x3

# Deeper CNN - Less Pooling: Parameters

- Seed: 7
- Momentum: 0.9
- Learning Rate: 0.01
- Epochs: 25
- Batch Size: 64
- Logarithmic loss function

# 77%

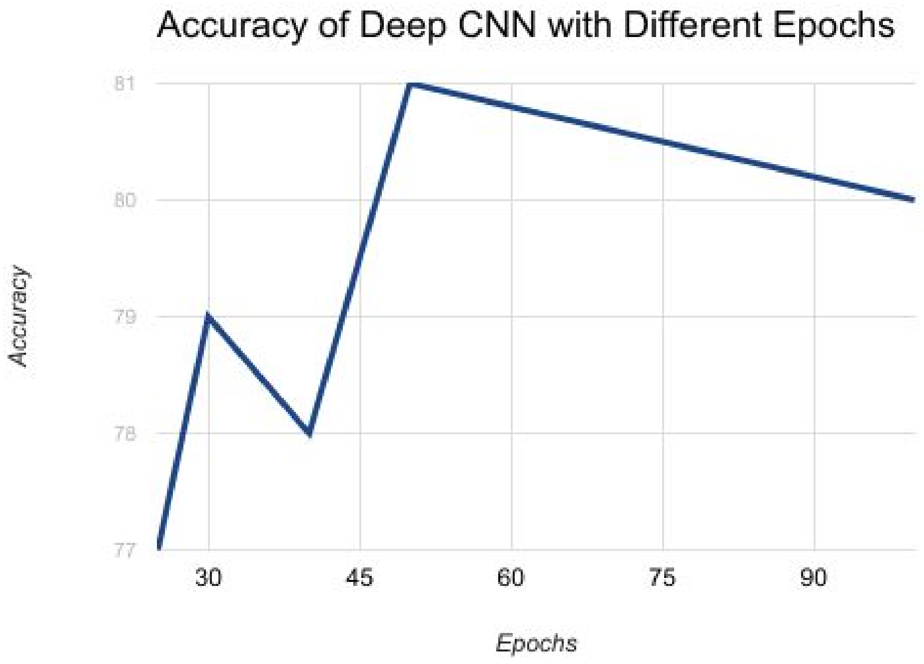
Deeper CNN Accuracy - Less Aggressive Pooling

# 82%

Deeper CNN Accuracy -
More Features and Less Aggressive Pooling

# Experiment: Epochs

- Deep CNN architecture
- Epochs: 25, 30, 40, 50, 100


Accuracy of Deep CNN with Different Epochs
81
80
79
78
77
Accuracy
30
45
60
75
90
Epochs

# Experiment: Data Augmentation

- Rotation: 180 degrees
- Flipping: horizontal and vertical

# 59%

Deeper CNN Accuracy - Rotations/Flips - 25 Epochs

# 66%

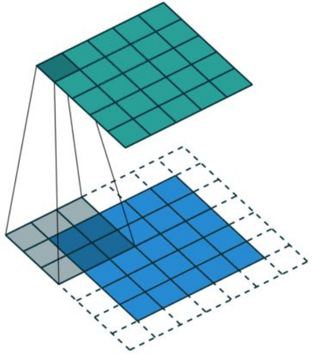
Deeper CNN Accuracy - Rotations/Flips - 100 Epochs

# Other Techniques

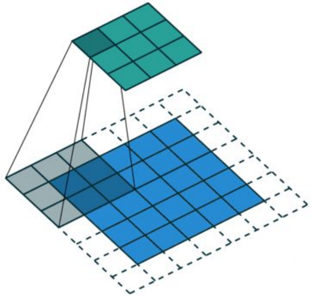
- Fractional Max Pooling - Benjamin Graham
  - Pooling through overlapping squares or disjoint collection of rectangles
    - Other additions: translations, rotations, reflections, stretching...
  - Idea is to reduce spatial size of image by a factor of $\alpha$ ($1 < \alpha < 2$)
    - Alternative to (2 x 2) max pooling, stochastic pooling, which reduce size of hidden layers by factor of two
    - Intuition: More pooling layers and more opportunity to view and recognize features in the image
  - Introducing randomness to pooling
    - Contrast to stochastic pooling, which has randomness in its pooling
    - Random vs. Pseudorandom pooling
  - Performance: 96.53% with data augmentation

# Other Techniques

- The All Convolutional Net - Jost Tobias Springenberg , Alexey Dosovitskiy, Thomas Brox, Martin Riedmiller
  - Replace max pooling with convolutional layer with increased stride (strided convolution / deconvolutional layer / transposed convolutional layer)
    - Homogeneous network consisting of convolutional layers
  - Results give evidence to effectiveness of small convolutional layers and raises questions about the necessity of pooling in CNNs
    - Pooling layer is just a feature-wise convolution that uses average, p-norm, max function
  - Increasing stride of previous convolutional layer vs. Adding a convolutional layer with corresponding stride
  - Performance: 92% without data augmentation



Zero-padding, Stride = 1



Zero-padding, Stride = 2

Source