ImageNet Classification with Deep Convolutional Neural Networks

Puneeth Chaganti

8th Aug, 2015

Who uses them?

Some recent popular uses of Neural Networks

- A picture is worth a thousand (coherent) words
- DeepMind Masters Atari Games (video)
- NVIDIA DRIVE
- Deep Dream – Generator
- Speech Recognition

Abstract of the paper

- Trained a large CNN to classify ImageNet images into 1000 classes
- Error rates

top-1	37.5%
top-5	17.0%

- 60 million parameters and 650,000 neurons
- five convolutional layers, some of which are followed by max-pooling layers, and three fully-connected layers with a final 1000-way softmax.
- non-saturating neurons
- **very efficient GPU implementation of the convolution operation**

- recently-developed regularization method called **dropout**
- ILSVRC-2012 results

top-5	15.3%
next best entry	26.2%

Handwriting Recognition

`http://neuralnetworksanddeeplearning.com/images/mnist_100_digits.png`

- It is challenging to do
- Imagine doing it by trying to describe stuff...
- Learn by examples!

Perceptron - a simple artificial neuron

`http://neuralnetworksanddeeplearning.com/images/tikz0.png`

$$\text{output} \quad = \quad \begin{cases} 0 & \text{if } \sum_j w_j x_j \leq \ \text{threshold} \\ 1 & \text{if } \sum_j w_j x_j > \ \text{threshold} \end{cases} \tag{1}$$

- Binary inputs
- Binary output
- Weigh up evidence, and make a decision!

$$\text{output} = \begin{cases} 0 & \text{if } w \cdot x + b \leq 0 \\ 1 & \text{if } w \cdot x + b > 0 \end{cases} \tag{2}$$

NAND Perceptron

`http://neuralnetworksanddeeplearning.com/images/tikz2.png`

`http://neuralnetworksanddeeplearning.com/images/tikz3.png`

So a perceptron is just a fancy NAND gate?

No! Enter learning algorithms!

Problem with training a network of Perceptrons

`http://neuralnetworksanddeeplearning.com/images/tikz8.png`

Sigmoid Neuron

`http://neuralnetworksanddeeplearning.com/images/tikz9.png`

- Real inputs in [0, 1]
- Real outputs in [0, 1]

`https://upload.wikimedia.org/wikipedia/commons/thumb/b/bc/Logistisch.svg/850px-Logistisch.svg.png`

$$\sigma(z) \equiv \frac{1}{1+e^{-z}}. \tag{3}$$

Non-linear functions for activation

Sigmoid activation ~ binary activation, when w.x + b is -inf or +inf.

The actual function doesn't matter, as much as the continuity of it.

- f(x) = $(1 + e^{-x})^{-1}$
- f(x) = tanh(x)
- f(x) = max(0, x) [ReLU]

Neural networks

`http://neuralnetworksanddeeplearning.com/images/tikz11.png`

- input, output, hidden layers (one or more)
- each layer has multiple "units"
- each connection has a weight
- the weights are adjusted based on the error in the output using an algorithm known as back propagation.

Hyper-parameters

- The structure of a Neural Network is decided based on the task at hand
- Weights are learned by back-propogation, but not the structure of the network.
- Hyperparameters:
  - the number of hidden layers (blue nodes above)
  - number of units per layer
  - number of connections per unit
  - learning rate (we'll talk about it!)
- Hyperparameters are chosen in a variety of ways - is a big deal!

Classifying handwritten digits

`http://neuralnetworksanddeeplearning.com/images/tikz12.png`

Why 10 outputs?

`http://neuralnetworksanddeeplearning.com/images/tikz13.png`

Learning with Gradient descent

Cost or Loss function

$$C(w,b) \equiv \frac{1}{2n} \sum_x \|y(x) - a\|^2. \tag{4}$$

Minimizing the cost

`http://neuralnetworksanddeeplearning.com/images/valley.png`

$$\Delta C \approx \frac{\partial C}{\partial v_1} \Delta v_1 + \frac{\partial C}{\partial v_2} \Delta v_2. \tag{5}$$

$$\nabla C \equiv \left(\frac{\partial C}{\partial v_1}, \frac{\partial C}{\partial v_2} \right)^T. \tag{6}$$

$$\Delta C \approx \nabla C \cdot \Delta v. \tag{7}$$

How to choose Δv to make ΔC negative?

Learning rate

$$\Delta v = -\eta \nabla C, \tag{8}$$

$$\Delta C \approx -\eta \nabla C \cdot \nabla C = -\eta \|\nabla C\|^2 \tag{9}$$

$$v \rightarrow v' = v - \eta \nabla C. \tag{10}$$

Learning rate!

Stochastic Gradient descent

- Used to speed up the learning
- $C = \frac{1}{n} \sum_x C_x$
- $\nabla C = \frac{1}{n} \sum_x \nabla C_x$
- mini-batch

$$\frac{\sum_{j=1}^{m} \nabla C_{X_j}}{m} \approx \frac{\sum_x \nabla C_x}{n} = \nabla C, \tag{11}$$

- training epoch

Backward propagation

`http://neuralnetworksanddeeplearning.com/chap2.html`

Towards Deep Learning

`http://neuralnetworksanddeeplearning.com/images/tikz14.png`

References

- A Deep Learning Tutorial: From Perceptrons to Deep Networks
- CS231n: Convolutional Neural Networks for Visual Recognition
- Neural Networks and Deep Learning
- Visualizing and Understanding Deep Neural Networks by Matt Zeiler
- Image Scaling at Flipboard