Neural Networks Notes

Nhan Trong

July 7, 2016

> Pipey. *Looks like you want to compress a movie file, can I help? You know with Pied Piper's revolutionary neural network optimized sharded data distribution system, it's just six clicks away, follow meeee!*
>
> Silicon Valley

Part I
Different Types of Neurons and Learning

Question 1. *How many other neurons does a neuron talk to? Do they change neighbours?*

Note 2. "Goal of unsupervised learning: provides a compact, low-dimensional representation of the input," like Pied Piper's compression algorithm using neural networks!

1 Keywords

Fruit flies, MNIST, TIMIT, linear, binary threshold, rectified / linear threshold, logistic, stochastic binary neurons, supervised, unsupervised, reinforcement learning.

Part II
Neural Network Architectures

2 Keywords

Feed forward, recurrent, symmetrically connected neural network, perceptrons, convexity condition.

Part III
Perceptron Learning Algorithm

TODO 1. Proof of why perceptron learning works is very sketchy! Need more details.

Part IV
Linear Neuron Learning Algorithm

Definition 3. *Given a training case x and a weight vector w, the neuron's estimate h of the desired output is*

$$h(x, w) = \sum_i w_i x_i = w^T x.$$

Define the cost function J to be the squared difference error

$$J = \sum_x (h - y)^2.$$

Finally the goal of learning is to minimize J:

$$\min_w J.$$

Actually that was using notation from Andrew Ng's class. In this course it is:

Definition 4. *Given a training case x_n and a weight vector w, the neuron's estimate y_n of the desired output is*

$$y_n = \sum_i w_i x_{ni} = w^T x_n.$$

Define the cost function E_n to be the squared difference error

$$E_n = \frac{1}{2}(t_n - y_n)^2,$$

where t_n is the target output, i.e. the "ground-truth", and define the total error to be

$$E = \sum_n E_n.$$

Finally the goal of learning is to minimize E:

$$\min_w E.$$

3 Delta Rule: Learning by Gradient Descent

The error partials are

$$\frac{\partial E}{\partial w_i} = \frac{1}{2}\sum_n \frac{dE_n}{dy_n}\frac{\partial y_n}{\partial w_i} = -\sum_n (t_n - y_n)x_{ni}.$$

The Delta Rule / Gradient Descent says that we should change w_i in the opposite direction as the change in error along w_i, give or take a learning rate α:

$$\Delta w_i = -\alpha\frac{\partial E}{\partial w_i} = \sum_n \alpha(t_n - y_n)x_{ni},$$

i.e. α tells us how much to change, and the negative sign tells us which direction to go, namely the opposite direction. E.g. if $\frac{\partial E}{\partial w_i} > 0$, that means the error goes up as w_i increases, so we want to decrease w_i to make it go down, and vice versa: always go in the opposite direction as $\frac{\partial E}{\partial w_i}$, hence the negative sign.

4 Keywords

Linear neurons / linear filters, iterative / programmatic VS analytic / mathematical approach, Delta Rule / Gradient Descent, batch VS online.

References

[1] Geoffrey E. Hinton's Neural Networks video lectures.

[2] http://www.cs.toronto.edu/~rgrosse/csc321/