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.

3 Binary Threshold Neurons McCullochPitts

Used in Perceptrons.

Question 3. *Also called Linear Threshold Neurons?*

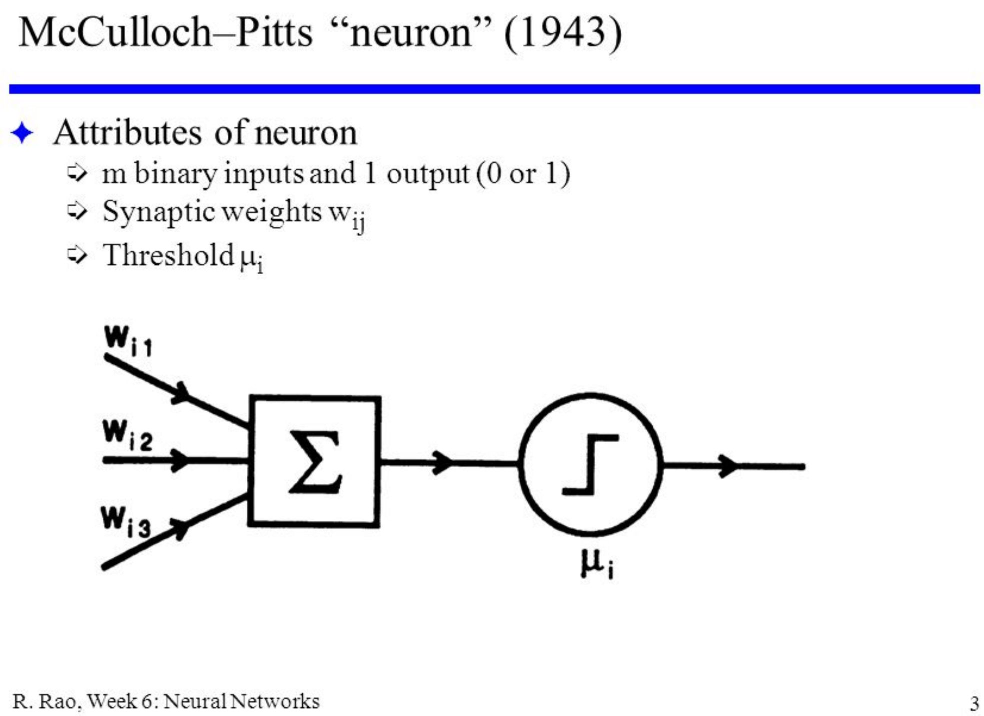
Definition 4. *First compute $z = w^T x$, then output*

$$y = \begin{cases} 1 & \text{if } z \geq 0 \\ 0 & \text{otherwise,} \end{cases}$$

representing "all VS none" activation.

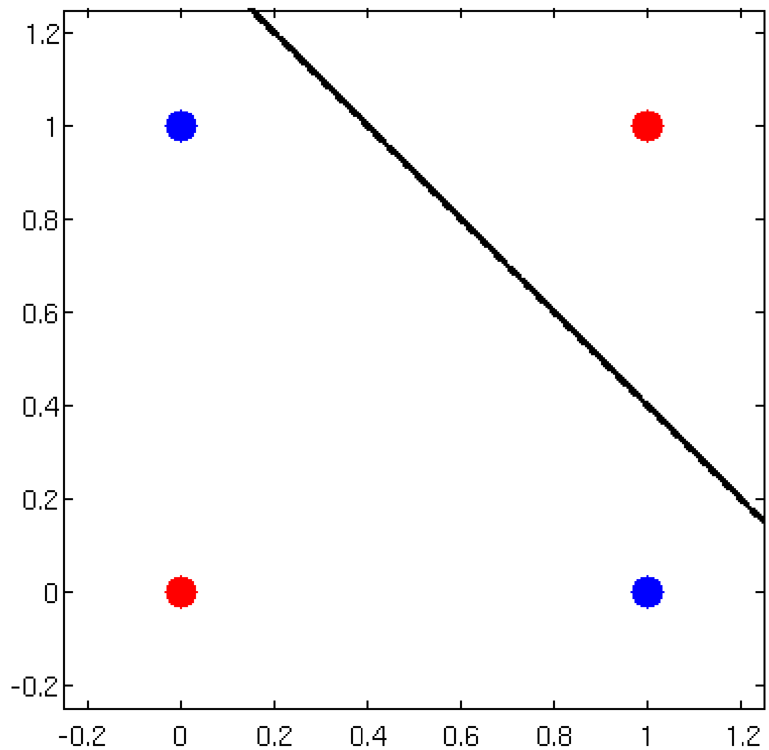
Note 5. Here we implicitly added the threshold as a bias unit: $w = (b, w_1, w_2, \ldots)$.

Remark 6. The function $y(z)$ is also called the Heaviside / unit step function.



4 Limitations of the Binary Threshold Neuron

Proposition 7. *A single binary threshold neuron cannot learn the XOR function, because geometrically its truth table represented on a plane is not linearly separable.*



4.1 Group Invariance Theorem

Proposition 8. *Perceptrons can't learn patterns if they're subject to transformations that form a group, e.g. translations with wrap-around.*

Question 9. *Details?*

5 Keywords

Data space, weight space, Group Invariance Theorem.

Part IV

Linear Neuron Learning Algorithm

Definition 10. *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:

$$\min_w E.$$

6 Delta Rule: Learning by Gradient Descent

The error partials are

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

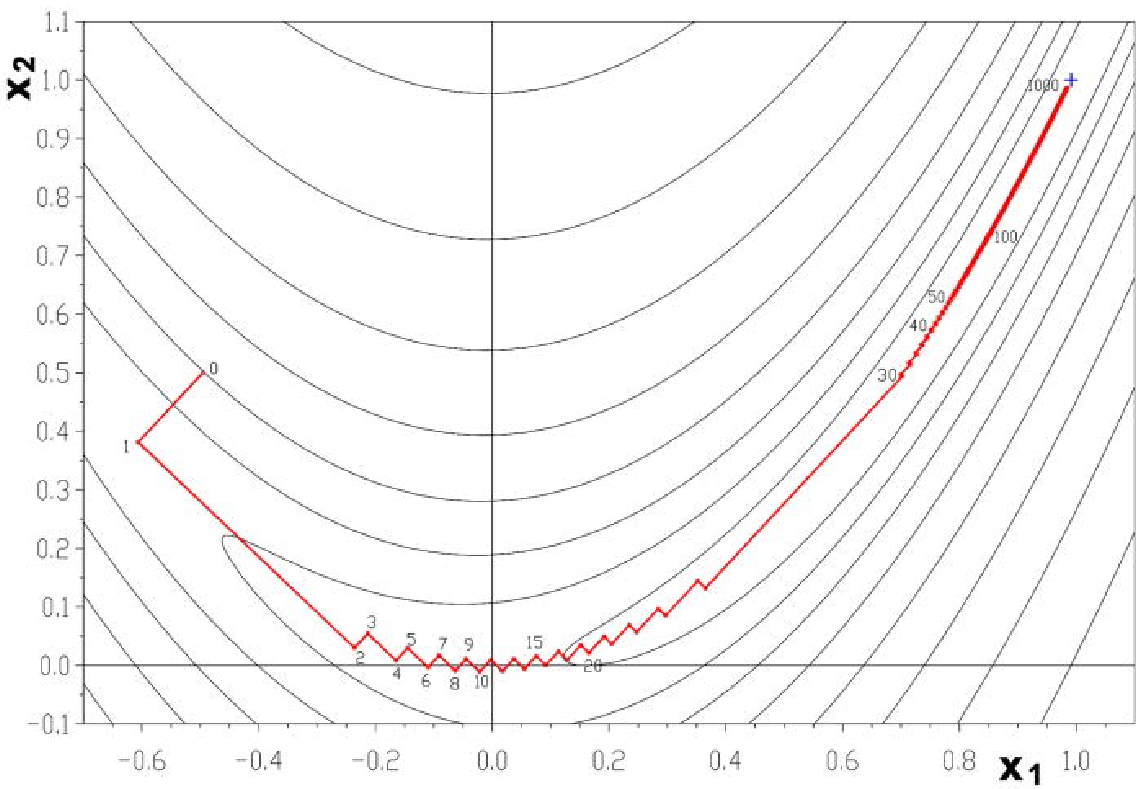
The Delta Rule / Gradient Descent says that we should change w_i in the opposite direction as the change in error along w_i, give or take a learning rate α:

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

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.

7 Error Surface of a Linear Neuron

Question 11. *IIRC feature normalization should help with slow learning due to unscaled data? What about pathological cases like this, called the Rosenbrock Valley?*



8 Keywords

Linear neurons / linear filters, iterative / computational VS analytic / mathematical approach, Delta Rule / Gradient Descent, batch VS online, error surface, extended weight space, Rosenbrock function.

Part V
Logistic Neurons

9 Learning Rule

Definition 12. *The estimator for a logistic neuron is given by*

$$y = \frac{1}{1 + e^{-z}}$$

where $z = w^T x$. The function $y(z)$ is also known as a logistic / sigmoid function, and z is sometimes called the logit. As before, the error is the squared difference

$$E = \frac{1}{2}\sum_n (t_n - y_n)^2.$$

Proposition 13. *The estimator derivatives are*

$$\frac{\partial y}{\partial w_i} = \frac{dy}{dz}\frac{\partial z}{\partial w_i} = y(1-y)x_i,$$

and so the error derivatives are

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

10 Learning with Hidden Units

11 Backpropagation Algorithm

12 Keywords

Sigmoid function, logit, Backpropagation Algorithm.

References

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

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