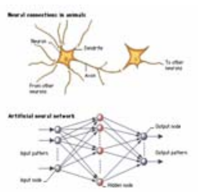
COMPETITIVE LEARNING



Agustin Rodriguez
axr8892@cs.rit.edu

Muneeb Mahmood
mam4810@cs.rit.edu

Outline of presentation

- Background of Neural Networks
- Introduction of Competitive Learning Networks
- Demos of Competitive Learning
- Similarities / Differences between HL and CL
- Applications of CL
- References
- Questions

Background

- What's the drive behind NN?
- What parameters are calibrated?
- What are the two ways that NN can learn ?
- All in a nutshell.

What's the drive behind NN?

- For simplicity sakes, we want to make optimal decisions under uncertainty, by calibrating parameters based on available resources.
- This gives the perception of an "intelligent" system that can adapt to change by utilizing only the available resources.

All in a nutshell

- We are exploring different ways of extending the ideas behind the linear Processing Element (PE), or ADALINE (for adaptive linear element) by introducing new biological concepts, such as those described in Competitive Learning and Hebbian Learning, which allow us to make optimal decisions under uncertainty.

What parameters are calibrated?

Calibrate the adaptive parameters:

- Input data = X1, X2, X3, . . . Xd .
- Output data = Y1, Y2, Y3 . . . Yd .
- Step size (Learning Rate) = ŋ (0 < ŋ < 1).
- Weights = w .
- NOTE: ŋ is defined by the designer. ŋ controls how large the update is at each step. ŋ is typically selected large, and then decreased progressively.

Learning Types

- **Supervised** - learn from the outputs by usage of the feed back error (for example, ADALINE) to change the weights. Such systems require a training set, since a desired response is used to guide the learning process.

- **Unsupervised/self-organized** - learn from the inputs by applying internal rules (For example, **CL & HL)** to change the weights. CL networks are online – learn sample by sample.

Competitive Learning

- Unsupervised learning.
- One neuron wins over all others.
- Only the winning neuron (or its close associates) learn.
- **Hard Learning** – weight of only the winner is updated.
- **Soft Learning** – weight of winner and close associates is updated

Competitive Learning

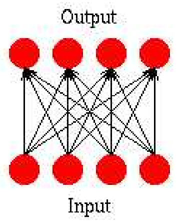
- A simple competitive network is basically composed of two networks:

  1) Hamming net
  2) Maxnet

  Each of them specializes in a different function.

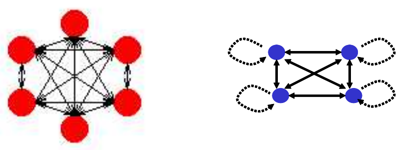
The Hamming net

- The Hamming net measures how much the input vector resembles the weight vector of each perceptron.



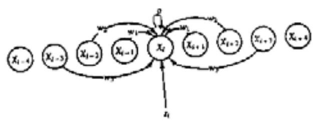
The Maxnet

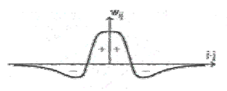
- The Maxnet finds the perceptron with the maximum value.



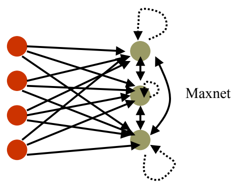
Maxnet - Mexican Hat Output

- Close neighbors: (w > 0)
- Not so close neighbors: (w < 0)
- Distance neighbors: (w = 0)



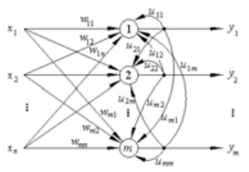


Simple Competitive Network



Simple Competitive Network

- How does it work ?



Input and Weight Vectors

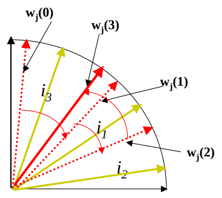
- Inner product / Dot product
  |W||X| Cos θ
- Euclidean distance

$$\sqrt{\sum_{i=1}^{n}(I_i - w_{ki})^2}$$

Competitive learning rule

- Wi*(n+1) = Wi*(n) + Ŋ(x(n) - Wi*(n))
- i* - processing element that wins the competition.
- n – next weight or input.
- W – weight.
- X – input.
- Ŋ – learning rate.

Demo1 – Weight update



Demo2 – Matching stored patterns

- http://www.psychology.mcmaster.ca/4i03/demos/competitive1-demo.html
- http://www.neuroinformatik.ruhr-uni-bochum.de/ini/VDM/research/gsn/DemoGNG/GNG.html

Hebbian vs. Competitive

Hebbian

- The weight between two neurons increases if the two neurons activate simultaneously and reduces if they activate separately.
- Nodes which tend to be either both positive or negative at the same time result in strong positive weights while those which tend to be opposite result in strong negative weights.
- **Weight update:** Δ Wij = Ŋ Xj Yi
- **Learning Rule:** W(n+1) = W(n) + Ŋ X(n) Y(n)
- n – iteration number, X – input, Y – output,
  Wij – Common weight b/w I and j processing elements

Competitive

- **Learning rule:** Wi*(n+1) = Wi*(n) + Ŋ(x(n) - Wi*(n))

Hebbian vs. Competitive

Similarities:

- Both use unsupervised methods.
- Both can be used for data representation.
- Both express a strong connection to biological systems. Basically, they work with principles of associative memories. Such memories are similar to human memory, since the memory is contained in the interconnection weights (pattern activity). In lesser words, it is enough to input the data to get the recall. <u>For instance, to remember the bird that reputedly puts its head in the sand, the description may be adequate to retrieve the name "ostrich" and a visual image of the bird -- comparable to the associative memory retrieval.</u>

Hebbian vs. Competitive

Differences:

- H networks are used to extract information globally from the input space.
- H networks requires all weights to be updated at each epoch.
- H networks implement associative memory while C networks are selectors – only one can win!
- C networks are used to clusters similar inputs.
- C networks compete for resources.
- C networks, only the winner's weight is updated each epoch.
- Note: epoch – one complete presentation of the input data to the network being trained.

Hebbian Learning

Limitations

- The weights continue to increase & don't reduce.
- Unlike LMS (Least Mean Square) or backpropagation, it is unstable producing very large positive or negative weights.
- Unable to learn certain patterns – fails because updating weights cannot be sensitive to other connections and instability issues.
- You don't know what it has learnt.

Note:

- Oja's Rule / Instar rule are much more stable
- Δ W = n(yx – y2w) – Oja's weight update
- ΔW = n(yx – yw) - Instar weight update

Competitive Learning

Advantages

- Similar to Instar weight update and hence stable unlike Hebbian Learning
- Won't result in excessive positive or negative weights
- Ideal when we have to analyze raw data of which we have no prior knowledge.
- Idea of completion can be applied to different types of networks (Hebbian, Kohonen networks, etc)
- Can be used for Vector quantization.
- Can be used for clustering – competitive rule allows a single layer network to group data that lies in a neighborhood of the input space

Competitive Learning

Limitations

- If a PE weight vector is far away from any of the data clusters, it may never win competition.
- You are not sure about what it has learnt.
- If used for clustering, number of clusters have to be decided in advance.
- Inner product takes account of both direction and magnitude of vector. If vectors are not normalized, may select wrong weight vector after competition.

Criteria for competition

- Hamming distance – based on inner product
- Euclidean distance $\sqrt{\sum_{i=1}^{n}(I_i - w_{ki})^2}$

Note:

- You can simply change the criteria of competition to overcome some of the limitations (i.e. incases where the a criteria doesn't work well with a specific data set, simply change it to a more appropriate one)

Applications of Competitive Networks

- Vector quantization
- Analyze raw data
- Bibliographic classification
- Image browsing systems
- Medical diagnosis (visualization of data and
- Speech recognition (this Kohonen used initially)
- Data compression
- Separating sound sources
- Environmental modeling
- Feature extraction
- Dimensionality reduction
- Optimization

References

[1] Jose, Neil, Curt; "Neural and Adaptive Systems", John Wiley and Sons Inc.
[2] http://www.cs.hmc.edu/claremont /keller/152-slides/265.html
[3] http://neuron.eng.wayne.edu/tarek /MITbook/t_contents.html
[4] http://www-cse.stanford.edu/classes /sophomore-college/projects-00/neural-networks/Architecture/competitive.html
[5] http://www.idi.ntnu.no/~keithd/classes/advai/lectures/compnet.ppt#264,7,Simple Competitive Learning
[6] http://www.cs.umbc.edu/~ypeng/F04NN/lecture-notes/NN-Ch5-1.ppt#296,7,
[7] http://www.soc.staffs.ac.uk/mo3/SOM/UnsupLearn.ppt
[8] http://en.wikipedia.org/wiki/Neural_networks
[9] http://neuron-ai.tuke.sk/NCS/VOL1/P3_html/node14.html

Questions / Answers

- Agustin Rodriguez
  axr8892@cs.rit.edu

- Muneeb Mahmood
  mam4810@cs.rit.edu