WIKIPEDIA

Generalized Hebbian algorithm

The **generalized Hebbian algorithm** (**GHA**), also known in the literature as **Sanger's rule**, is a linear feedforward neural network model for unsupervised learning with applications primarily in principal components analysis. First defined in 1989,[1] it is similar to Oja's rule in its formulation and stability, except it can be applied to networks with multiple outputs. The name originates because of the similarity between the algorithm and a hypothesis made by Donald Hebb[2] about the way in which synaptic strengths in the brain are modified in response to experience, i.e., that changes are proportional to the correlation between the firing of pre- and post-synaptic neurons.[3]

Contents

Theory

The GHA combines Oja's rule with the Gram-Schmidt process to produce a learning rule of the form

$$\Delta w_{ij} ~=~ \eta \left(y_i x_j - y_i \sum_{k=1}^{i} w_{kj} y_k \right),$$ [4]

where w_{ij} defines the synaptic weight or connection strength between the jth input and ith output neurons, x and y are the input and output vectors, respectively, and η is the *learning rate* parameter.

Derivation

In matrix form, Oja's rule can be written

$$\frac{\mathrm{d}w(t)}{\mathrm{d}t} ~=~ w(t)Q - \mathrm{diag}[w(t)Qw(t)^{\mathrm{T}}]w(t),$$

and the Gram-Schmidt algorithm is

$$\Delta w(t) ~=~ -\mathrm{lower}[w(t)w(t)^{\mathrm{T}}]w(t),$$

where $w(t)$ is any matrix, in this case representing synaptic weights, $Q = \eta\, \mathbf{x}\, \mathbf{x}^{\mathrm{T}}$ is the autocorrelation matrix, simply the outer product of inputs, diag is the function that diagonalizes a matrix, and lower is the function that sets all matrix elements on or above the diagonal equal to 0. We can combine these equations to get our original rule in matrix form,

$$\Delta w(t) = \eta(t) \left(\mathbf{y}(t)\mathbf{x}(t)^{\mathrm{T}} - \mathrm{LT}[\mathbf{y}(t)\mathbf{y}(t)^{\mathrm{T}}]w(t)\right),$$

where the function LT sets all matrix elements above the diagonal equal to 0, and note that our output $\mathbf{y}(t) = w(t)\,\mathbf{x}(t)$ is a linear neuron.[1]

Stability and PCA

[5] [6]

Applications

The GHA is used in applications where a self-organizing map is necessary, or where a feature or principal components analysis can be used. Examples of such cases include artificial intelligence and speech and image processing.

Its importance comes from the fact that learning is a single-layer process—that is, a synaptic weight changes only depending on the response of the inputs and outputs of that layer, thus avoiding the multi-layer dependence associated with the backpropagation algorithm. It also has a simple and predictable trade-off between learning speed and accuracy of convergence as set by the learning rate parameter η.[5]

See also

- Hebbian learning
- Factor analysis
- Contrastive Hebbian learning
- Oja's rule

References

1. Sanger, Terence D. (1989). "Optimal unsupervised learning in a single-layer linear feedforward neural network" (http://courses.cs.washington.edu/courses/cse528/09sp/sanger_pca_nn.pdf) (PDF). *Neural Networks*. **2** (6): 459–473. CiteSeerX 10.1.1.128.6893 (https://citeseerx.ist.psu.edu/viewdoc/summary?doi=10.1.1.128.6893). doi:10.1016/0893-6080(89)90044-0 (https://doi.org/10.1016%2F0893-6080%2889%2990044-0). Retrieved 2007-11-24.
2. Hebb, D.O. (1949). *The Organization of Behavior* (https://books.google.com/books?id=uyV5AgAAQBAJ). New York: Wiley & Sons. ISBN 9781135631918.
3. Hertz, John; Anders Krough; Richard G. Palmer (1991). *Introduction to the Theory of Neural Computation*. Redwood City, CA: Addison-Wesley Publishing Company. ISBN 978-0201515602.
4. Gorrell, Genevieve (2006), "Generalized Hebbian Algorithm for Incremental Singular Value Decomposition in Natural Language Processing.", *EACL*, CiteSeerX 10.1.1.102.2084 (https://citeseerx.ist.psu.edu/viewdoc/summary?doi=10.1.1.102.2084)
5. Haykin, Simon (1998). *Neural Networks: A Comprehensive Foundation* (2 ed.). Prentice Hall. ISBN 978-0-13-273350-2.
6. Oja, Erkki (November 1982). "Simplified neuron model as a principal component analyzer". *Journal of Mathematical Biology*. **15** (3): 267–273. doi:10.1007/BF00275687 (https://doi.org/

10.1007%2FBF00275687). PMID 7153672 (https://pubmed.ncbi.nlm.nih.gov/7153672). S2CID 16577977 (https://api.semanticscholar.org/CorpusID:16577977). BF00275687.