Back-Propagation

The diagram illustrates a neural network with three layers: Input, Layer 0, and Layer m + 1. The input layer takes in the inputs $x_1, x_2, ..., x_n$. The output layer computes the outputs $Y_1^{(m+1)}, Y_2^{(m+1)}, ..., Y_p^{(m+1)}$. Each neuron in the hidden layers has connections from the input layer and feeds into the output layer. The neurons in Layer 0 include Neuron #1, Neuron #2, ..., Neuron #k, and in Layer m+1, they are Neuron #1, Neuron #2, ..., Neuron #p. The diagram shows how the inputs are propagated through the network and how the output is computed.
• Train a BP Neural Net to recognize the following XOR function (Bipolar/Unipolar):

<table>
<thead>
<tr>
<th>$X_1$</th>
<th>$X_2$</th>
<th>Desire O/P</th>
</tr>
</thead>
<tbody>
<tr>
<td>-1(0)</td>
<td>-1(0)</td>
<td>-1(0)</td>
</tr>
<tr>
<td>-1(0)</td>
<td>+1</td>
<td>+1</td>
</tr>
<tr>
<td>+1</td>
<td>-1(0)</td>
<td>+1</td>
</tr>
<tr>
<td>+1</td>
<td>+1</td>
<td>-1(0)</td>
</tr>
</tbody>
</table>
BP-Example
Random values were chosen between -0.5 and +0.5

The initial weights to the hidden layer are:

<table>
<thead>
<tr>
<th>Weight from</th>
<th>Weight from</th>
<th>Weight from</th>
<th>Weight from</th>
<th>Bias to the four hidden units</th>
</tr>
</thead>
<tbody>
<tr>
<td>-0.3378</td>
<td>0.2771</td>
<td>0.2859</td>
<td>-0.3329</td>
<td></td>
</tr>
<tr>
<td>0.1970</td>
<td>0.3191</td>
<td>-0.1448</td>
<td>0.3594</td>
<td>Weights from the first input unit</td>
</tr>
<tr>
<td>0.3099</td>
<td>0.1904</td>
<td>-0.0347</td>
<td>-0.4861</td>
<td>Weights from the second input unit</td>
</tr>
</tbody>
</table>
BP-Example

Random values were chosen between -0.5 and +0.5

The initial weights from the hidden units to the output unit are:

<table>
<thead>
<tr>
<th>Weight</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>-0.1401</td>
<td>Bias on the output unit</td>
</tr>
<tr>
<td>0.4919</td>
<td>Weight from the first hidden unit</td>
</tr>
<tr>
<td>-0.2913</td>
<td>Weight from the second hidden unit</td>
</tr>
<tr>
<td>-0.3979</td>
<td>Weight from the third hidden unit</td>
</tr>
<tr>
<td>0.3581</td>
<td>Weight from the fourth hidden unit</td>
</tr>
</tbody>
</table>
BP-Example

Binary representation (0,1) with unipolar sigmoid

>3000 epochs
BP-Example

Bipolar representation (-1,1) with bipolar sigmoid

$>380$ epochs
BP-Example

Modified Bipolar representation with bipolar sigmoid but targets of +0.8 or -0.8

>260 epochs
BP-Example

Nguyen-Widrow weight Initialization
The weights were scaled with $\beta = 0.7(4)^{1/2} = 1.4$

The epochs required were as follows:

<table>
<thead>
<tr>
<th></th>
<th>Random</th>
<th>Nguyen-Widrow</th>
</tr>
</thead>
<tbody>
<tr>
<td>Binary XOR</td>
<td>2,891</td>
<td>1,935</td>
</tr>
<tr>
<td>Bipolar XOR</td>
<td>387</td>
<td>224</td>
</tr>
<tr>
<td>Modified bipolar XOR</td>
<td>264</td>
<td>127</td>
</tr>
</tbody>
</table>
BP- Improvements

• Second order derivatives (Parker, 1982)

• Dynamic range modification (Stronetta and Huberman, 1987)

\[ F(x) = -\frac{1}{2} + \frac{1}{1 + e^{-x}} \]

• Meta Learning (Jacobs, 1987; Hagiwara, 1990)

• Selective updates (Huang and Huang, 1990)
• Use of Momentum weight change
  (Rumelhart, 1986)

\[ \Delta w_{kmi}(t+1) = \eta \cdot \delta_{km} \cdot x'_i(t) + \alpha \Delta w_{kmi}(t) \]

• Exponential smoothing (Sejnowski and Rosenberg, 1987)

\[ \Delta w_{kmi}(t+1) = (1-\alpha) \cdot \delta_{km} \cdot x'_i(t) + \alpha \Delta w_{kmi}(t) \]
BP- Improvements (Cont.)

• Accelerating the BP algorithm (Kothari, Klinkhachorn, and Nutter, 1991)
  – Gradual increase in learning accuracy
• Without incorporating the disadvantage of increased network size, more complex neurons or otherwise violating the parallel structure of computation
Gradual increase in learning accuracy

- Temporal instability
- Absence of true direction of descent

```c
Void Acc_BackProp (Struct Network *N, struct Train_Set *T)
{
    Assume_coarse_error (\(\varepsilon\))
    while (\(\varepsilon < \) Eventual_Accuracy) {
        while (not_all_trained) {
            Present_Next_Pattern;
            while (!Trained)
                Train_Pattern;
        }
        Increase_Accuracy (\(\varepsilon -= \) Step);
    }
}
```
Training with gradual increase in accuracy

Direction of descent suggested by examplar 1

Direction of descent suggested by examplar 2

Direction of descent suggested by examplar 3

... 

Direction of descent suggested by examplar M

Direction of steepest descent
Minimization of the error for a 4 bit 1's complementor
(Graph has been curtailed to show detail)
Error VS Training Passes

Minimization of the error for a 3-to-8 Decoder
Error VS Training Passes

Minimization of the error for the Xor problem

BP
BPGA
BP+Mom
BPGA+Mom
Minimization of the error for simple shape recognizer
Minimization of the error for a 3 bit rotate register
## Error VS Training Passes

<table>
<thead>
<tr>
<th>Problem</th>
<th>BP</th>
<th>BPGIA</th>
<th>BP+Mom.</th>
<th>BPGIA+Mom.</th>
</tr>
</thead>
<tbody>
<tr>
<td>1’s complement (4×8×4)</td>
<td>9.7 (134922)</td>
<td>6.6 (92567)</td>
<td>2.2 (25574)</td>
<td>1.0 (11863)</td>
</tr>
<tr>
<td>3 to 8 decoder (3×8×8)</td>
<td>5.4 (347634)</td>
<td>4.2 (268833)</td>
<td>1.1 (61366)</td>
<td>1.0 (53796)</td>
</tr>
<tr>
<td>Exor (2×2×1)</td>
<td>4.5 (211093)</td>
<td>1.8 (88207)</td>
<td>2.5 (107337)</td>
<td>1.0 (45916)</td>
</tr>
<tr>
<td>Rotate register (3×6×3)</td>
<td>4.3 (72477)</td>
<td>2.0 (33909)</td>
<td>1.1 (15929)</td>
<td>1.0 (14987)</td>
</tr>
<tr>
<td>Differentiation between a square, circle and triangle (16×20×1)</td>
<td>2.3 (71253)</td>
<td>1.3 (33909)</td>
<td>6.11 (145363)</td>
<td>1.0 (25163)</td>
</tr>
</tbody>
</table>
Training with gradual increase in accuracy

• On an average, doubles the convergence rate of back propagation or the back propagation algorithm utilizing a momentum weight change without requiring additional or more complex neurons
Example of training set

Judith Dayhoff, Neural Network Architectures: An Introduction, Van Nostrand Reinhold
Example of training set

Judith Dayhoff, Neural Network Architectures: An Introduction, Van Nostrand Reinhold
Example of training set

Judith Dayhoff, Neural Network Architectures: An Introduction, Van Nostrand Reinhold
Example: Feature detector

- Judith Dayhoff, Neural Network Architectures: An Introduction, Van Nostrand Reinhold
Example: Feature detector

Judith Dayhoff, Neural Network Architectures: An Introduction, Van Nostrand Reinhold
#Epochs VS #Hidden units

Judith Dayhoff, Neural Network Architectures: An Introduction, Van Nostrand Reinhold
Thought on BP

• Activation Function:
  – Any differentiable function (function to be continuous)
  – Sigmoid function is widely used

Simon Haykin, Neural Networks: A comprehensive Foundation, Prentice Hall
Thought on BP

• Rate of Learning:
  – BP algorithm provides an “approximation” to the trajectory in weight space computed by the method of steepest descent.
  – The smaller $\eta$ value, the smaller the changes to the weights, and the smoother will be the trajectory in weight space: slower rate of learning!
  – To speed up the learning rate, larger $\eta$ value may resulting large changes in weights thus the network may become unstable (i.e., Oscillatory)
  – Rumelhart(1986) introduced a “Momentum term” to avoid instability.
Thought on BP

• Sequential Training
  – Highly popular
    • The algorithm is simple to implement
    • It provides effective solutions to large and difficult problems.

Simon Haykin, Neural Networks: A comprehensive Foundation, Prentice Hall
Thought on BP

• Stopping Criteria
  – Kramer and Sangiovanni-Vincentelli (1989)

  “The BP algorithm is considered to have converged when the Euclidean norm of the gradient vector reaches a sufficiently small gradient threshold”

  “The BP algorithm is considered to have converged when the absolute rate of change in the average squared error per epoch is sufficiently small”

Simon Haykin, Neural Networks: A comprehensive Foundation, Prentice Hall
BP Critique

- Very Robust
- Slow learning
- Network paralysis
- Local Minima
- Temporal instability

Simon Haykin, Neural Networks: A comprehensive Foundation, Prentice Hall
Summary of BP training

• Initialize network with small bipolar random weight values (recommend <0.3)

• Use small learning rate (recommend < 0.3)

• Train network as per algorithm

• Stop when Each pattern is trained to satisfaction (recommend 0.1)
Summary of BP training

• Unbounded input; Bounded output

• Classifiers

• Multi-output network

Single output network with normalization