Algorithm 1 STEEPEST DESCENT WITH INACCURATE LINE SEARCH

Input: P training patterns (instances, data);

Output: Connection weights and activation levels of the units;

begin

Set counter $t = 0$;

Initialize connection weights and threshold w_{ik} randomly;

Choose an initial learning rate, $\alpha^{(0)}$;

repeat

    for $p = 1, 2, \ldots, P$ **do**

        for $i = 1, 2, \ldots, I$ **do**

            $o_{ip} = \sum_{k=0}^{K} w_{ik}^{(t)} x_{kp}$; * $x_{0p} = -1$, $w_{i0} = \theta_i$ *\

            $O_{ip} = g_i(o_{ip})$;

            $\delta_{ip} = -\frac{\partial E(\mathbf{w})}{\partial O_{ip}} g_i'(o_{ip})$;

        end for

    end for

    for i=1,2,..., I **do**

        for k=0,1,..., K **do**

            $\Delta_{ik}^{(t)} = \alpha^{(t)} \sum_{p=1}^{P} \delta_{ip} x_{kp}$; * batch mode *\

            $w_{ik}^{(t+1)} = w_{ik}^{(t)} + \Delta_{ik}^{(t)}$;

        end for

    end for

    $\alpha^{(t+1)} = \eta \alpha^{(t)}$; * $0 < \eta < 1$*\

    $t \longleftarrow t + 1$,

until $\alpha^{(t)} < 0$

Output $\mathbf{w}^{(t)}$

end

Algorithm 2 STOCHASTIC ONLINE GRADIENT DESCENT

Input: P training patterns (instances, data);

Output: Connection weights and activation levels of the units;

begin
Set counter $t = 0$;
Initialize connection weights and threshold w_{ik} randomly;
Choose an initial learning rate, $\alpha^{(0)}$;
repeat
    Put patterns in random order;
    for $p = 1, 2, \ldots, P$ **do**
        for $i = 1, 2, \ldots, I$ **do**
            $o_{ip} = \sum_{k=0}^{K} w_{ik}^{(t)} x_{kp}$; * $x_{0p} = -1$, $w_{i0} = \theta_i$ *\
            $O_{ip} = g_i(o_{ip})$;
            $\delta_{ip} = -\frac{\partial E(\mathbf{w})}{\partial O_{ip}} g_i'(o_{ip})$;
        end for
        for i=1,2,. . . , I **do**
            for k=0,1,. . . , K **do**
                $\Delta_{ikp}^{(t)} = \alpha^{(t)} \delta_{ip} x_{kp}$;
                $w_{ik}^{(t+1)} = w_{ik}^{(t)} + \Delta_{ikp}^{(t)}$; * on-line mode *\
            end for
        end for
    end for
    $\alpha^{(t+1)} = \eta \alpha^{(t)}$; * $0 < \eta < 1$*\
    $t \longleftarrow t + 1$;
until $\alpha^{(t)} < 0$;
Output $\mathbf{w}_i^{(t)}$ $i = 1, 2, \ldots, I$;
end

Algorithm 3 STEEPEST DESCENT WITH INACCURATE LINE SEARCH

Input: P training patterns (instances, data);
Output: Connection weights $\mathbf{w}^*, \mathbf{W}^*$, training error $E(\mathbf{w}^*, \mathbf{W}^*)$;

begin
 Set counter $t = 0$;
 Initialize the weights and threshold $w_{jk}^{(0)}$ and $W_{ij}^{(0)}$ randomly;
 Choose an initial learning rate, $\alpha^{(0)}$;
 repeat
  for p=1,2,...,P **do**
   * from input to hidden *\
   for j=1,2,...,J **do**
    $h_{jp} = \sum_{k=0}^{K} w_{jk}^{(t)} x_{kp}$; * $x_{0p} = -1$ *\
    $H_{jp} = \chi_j(h_{jp})$;
   end for
   * from hidden to output *\
   for i=1,2,..., I **do**
    $o_{ip} = \sum_{j=0}^{J} W_{ij}^{(t)} H_{jp}$; * $H_{0p} = -1$ *\
    $O_{ip} = \omega_i(o_{ip})$;
   end for
   for i=1,2,..., I **do**
    $\delta_{O_{ip}} = -\frac{\partial E(\mathbf{w})}{\partial O_{ip}} w_i'(o_{ip})$;
   end for
   for j=1,2,..., J **do**
    $\delta_{H_{jp}} = \chi_i'(h_{jp}) \sum_{i=1}^{I} W_{ij}^{(t)} \delta_{O_{ip}}$;
   end for
  end for
  * from output to hidden *\
  for i=1,2, ..., I **do**
   for j=0,1,..., J **do**
    $\Delta W_{ij}^{(t)} = \alpha^{(t)} \sum_{p=1}^{P} \delta_{O_{ip}} H_{jp}$; * batch mode *\
    $W_{ij}^{(t+1)} = W_{ij}^{(t)} + \Delta W_{ij}^{(t)}$;
   end for
  end for
  * from hidden to input *\
  for j=1,2, ..., J **do**
   for k=0,1,..., K **do**
    $\Delta w_{jk}^{(t)} = \alpha^{(t)} \sum_{p=1}^{P} \delta_{H_{jp}} x_{kp}$; * batch mode *\
    $w_{jk}^{(t+1)} = w_{jk}^{(t)} + \Delta w_{jk}^{(t)}$;
   end for
  end for
  $\alpha^{(t+1)} = \eta \alpha^{(t)}$; * $0 < \eta < 1$ *\
  $t \longleftarrow t + 1$;
 until $\alpha^{(t)} < 0$;
 Output $\mathbf{w}^{(t)}$, $\mathbf{W}^{(t)}$ and $E(\mathbf{w}^{(t)}, \mathbf{W}^{(t)})$;
end

Algorithm 4 BACKPROPAGATION

Input: P training patterns (instances, data);
Output: Connection weights $\mathbf{w}^*, \mathbf{W}^*$, training error $E(\mathbf{w}^*, \mathbf{W}^*)$;

begin
Set counter $t = 0$;
Initialize the weights and threshold $w_{jk}^{(0)}$ and $W_{ij}^{(0)}$ randomly;
Choose an initial learning rate, $\alpha^{(0)}$;
repeat
  Put patterns in random order;
  for p=1,2,...,P **do**
    * from input to hidden *\
    for j=1,2,...,J **do**
      $h_{jp} = \sum_{k=0}^{K} w_{jk}^{(t)} x_{kp}$; * $x_{0p} = -1$ *\
      $H_{jp} = \chi_j(h_{jp})$;
    end for
    * from hidden to output *\
    for i=1,2,..., I **do**
      $o_{ip} = \sum_{j=0}^{J} W_{ij}^{(t)} H_{jp}$; * $H_{0p} = -1$ *\
      $O_{ip} = \omega_i(o_{ip})$;
    end for
    * from output to hidden *\
    for i=1,2,..., I **do**
      $\delta_{O_{ip}} = \delta_{ip} = -\frac{\partial E(\mathbf{w})}{\partial O_{ip}} w_i'(o_{ip})$;
    end for
    * from hidden to input *\
    for j=1,2,..., J **do**
      $\delta_{H_{jp}} = \chi_j'(h_{jp}) \sum_{i=1}^{I} W_{ij}^{(t)} \delta_{O_{ip}}$;
    end for
    * from output to hidden *\
    for i=1,2, ..., I **do**
      for j=0,1,..., J **do**
        $\Delta W_{ijp}^{(t)} = \alpha^{(t)} \delta_{O_{ip}} H_{jp}$; * on-line mode *\
        $W_{ij}^{(t+1)} = W_{ij}^{(t)} + \Delta W_{ijp}^{(t)}$;
      end for
    end for
    * from hidden to input *\
    for j=1,2, ..., J **do**
      for k=0,1,..., K **do**
        $\Delta w_{jkp}^{(t)} = \alpha^{(t)} \delta_{H_{jp}} x_{kp}$; * on-line mode *\
        $w_{jk}^{(t+1)} = w_{jk}^{(t)} + \Delta w_{jkp}^{(t)}$;
      end for
    end for
  end for
  $\alpha^{(t+1)} = \eta \alpha^{(t)}$; $0 < \eta < 1$*
  $t \longleftarrow t + 1$;
until $\alpha^{(t)} < 0$;
Output $\mathbf{w}^{(t)}$, $\mathbf{W}^{(t)}$ and $E(\mathbf{w}^{(t)}, \mathbf{W}^{(t)})$;
end

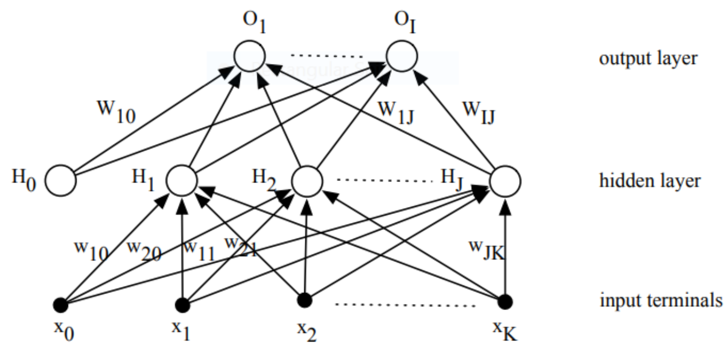

Figure 1: A two-layer perceptron

Table 1: Definitions for a two-layer perceptron

Symbol	Definition
O_i	output unit i
H_j	hidden unit j
x_k	input terminal k
W_{ij}	weight for the connection between hidden unit j and output unit i
w_{jk}	weight for the connection between input terminal k and hidden unit j
i	index for the output units, $i = 1, 2, \ldots, I$
j	index for the hidden units, $j = 0, 1, 2, \ldots, J$
k	index for the input terminals, $k = 0, 1, 2, \ldots, K$
p	index for the input patterns, $p = 1, 2, \ldots, P$
I	the number of output units
J	the number of hidden units
K	the number of input terminals
y_{ip}	target output of output unit i for pattern p
O_{ip}	actual output of output unit i for pattern p
o_{ip}	net input of output unit i for pattern p
H_{jp}	actual output of hidden unit j for pattern p
h_{jp}	net input of hidden unit j for pattern p
x_{kp}	input value of input terminal k for pattern p
$\omega_i(o)$	activation function for ouput unit i
$\chi_j(h)$	activation function for hidden unit j

Algorithm 5 HIDDEN_UNIT

Input: Test patterns (instances, data);
Output: The number of hidden units;

begin
Set counter $q = 1$;
Determine an initial number of hidden units, $J^{(q)}$;
Set $\overline{E}_T^{(0)}$ to a very large number;
repeat
    Run backpropagation algorithm on the training data with $J^{(q)}$;
    Find average test error $\overline{E}_T^{(q)}$ and variance $(s_T^{(q)})^2$ on the test set;
    Report $\overline{E}_T^{(q)}$ and $(\overline{s}_T^{(q)})^2$;
    Set $J^{(q+1)} \longleftarrow J^{(q)} + 1$;
    $q \longleftarrow q + 1$;
until $\overline{E}_T^{(q)} \geq \overline{E}_T^{(q-1)}$;
Output $J^{(q-1)}$ as the best number of hidden units;
end