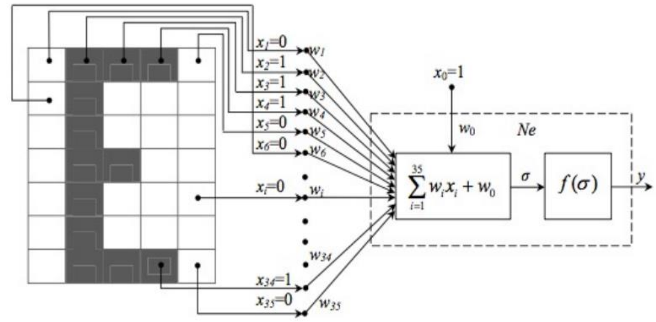
Элементарный перцептрон



- Рассмотрим принцип действия персептрона на примере классификации букв русского алфавита на гласные и согласные. Данный персептрон называется элементарным, поскольку использует только один нейрон МакКаллока-Питтса (Ne).
- Задача персептрона заключается в том, чтобы формировать выходной сигнал y, равный единице, если на вход поступает гласная буква, и нулю, если согласная.
- Для того, чтобы персептрон решал требуемую задачу, он должен пройти режим обучения
- Суть режима обучения заключается в настройке весов w_i и w_0 на совокупность входных образов решаемой задачи

Режим обучения перцептрона

- Обучающий набор данных для персептрона должен состоять из образцов представления знаний, которым предлагается его обучить, т.е. из букв русского алфавита.
- В процессе обучения персептрону предъявляются эти буквы и анализируется его реакция y.
- Если, например, на вход персептрона поступает буква «А», а выходной сигнал y случайно оказался равным единице, означающей, что буква гласная, то корректировать веса не нужно.
- Однако если выход неправилен и y равен нулю, то следует увеличить веса тех активных входов, которые способствуют возбуждению персептрона.

Алгоритм обучения Хебба

Алгоритм обучения Хебба представляет собой следующую последовательность шагов:

- Шаг 1. [Инициализация]. Всем весам персептрона присваиваются некоторые малые случайные значения из диапазона [-0,1;+0,1].
- Шаг 2. На вход персептрона подается текущий входной вектор $X[t]=\{x_1[t], x_2[t], K, x_{35}[t]\}$ и вычисляется выход персептрона y.
- Шаг 3. Если выход правильный, то перейти к шагу 2.
- Шаг 4. [Первое правило Д. Хебба]. Если выход неправильный и равен нулю, то увеличить веса активных входов, например, в соответствии с формулами: $w_i[t+1]= w_i[t]+x_i[t]$;
  $w_0[t+1]= w_0[t]+x_0$.
- Шаг 5. [Второе правило Д. Хебба]. Если выход неправильный и равен единице, то уменьшить веса активных входов, например, в соответствии с формулами: $w_i[t+1]= w_i[t]-x_i[t]$;
  $w_0[t+1]= w_0[t]-x_0$.
- Шаг 6. Осуществляется переход на шаг 2 с новым входным вектором X[t+1] или процесс обучения завершается

Однослойный перцептрон

Развитие идеи элементарного персептрона привело к появлению однослойного персептрона и созданию алгоритма его обучения, основанного на дельта-правиле Уидроу-Хоффа.

В отличие от элементарного персептрона данная ИНС имеет 10 нейронов, организованных таким образом, чтобы каждой цифре соответствовал свой нейрон.

- Выход первого нейрона y_1 должен быть равен единице, если персептрону предъявляется цифра «1» и нулю для выходов всех остальных нейронов.

- Выход y2 должен быть равен единице, если персептрону показывается цифра «2», при этом остальные выходы нейронов должны быть равны нулю.
- И так далее до цифры «0».

Архитектура и алгоритмы обучения НС

Однослойный перцептрон

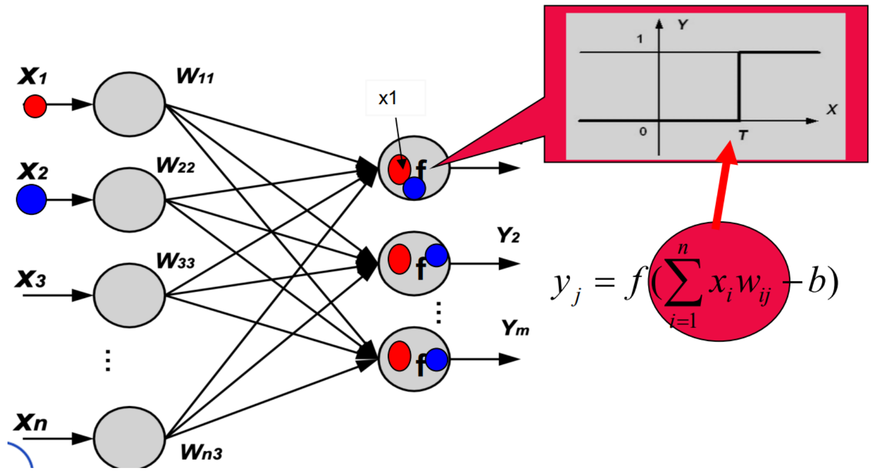
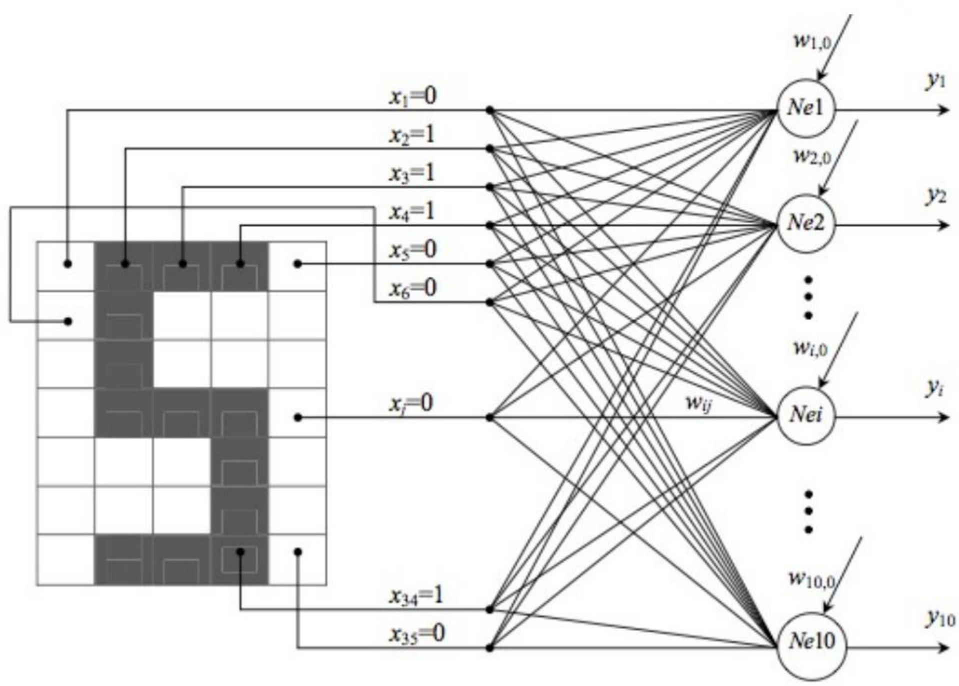
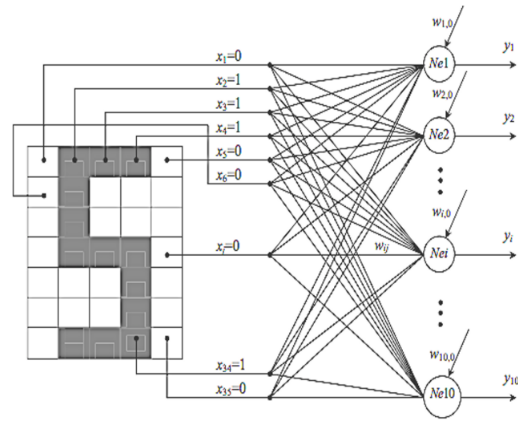


Схема однослойного перцептрона, предназначенная для распознавания цифр



Алгоритм обучения однослойного персептрона с помощью дельта-правила выглядит следующим образом:



- **Шаг 1. [Инициализация].** Всем весам персептрона w_{ij} и $w_{i,0}$ (i=1;10,j=1;35)присваиваются небольшие случайные значения из диапазона [-0,1; +0,1].

- **Шаг 2.** На вход персептрона подается очередной входной вектор $X[t]=\{x_1[t],x_2[t],\ldots,x_{35}[t]\}$, где t – номер итерации.

- Каждый из 10 нейронов выполняет взвешенное суммирование входных сигналов $\sigma_i[t]=\sum_{j=1}^{35} w_{ij}[t]+w_{i,0}[t]$

  и вырабатывает выходной сигнал $y_i[t]=\begin{cases}1, & если\ \sigma_i[t]\geq 0;\\ 0, & если\ \sigma_i[t]<0.\end{cases}$

Шаг 3. Для каждого нейрона определяется ошибка $\beta_i[t]=(d_i[t]-y_i[t])$, где $d_i[t]$ – требуемое значение выхода *i-го* нейрона, а $y_i[t]$ – полученное на шаге 2 значение *i-го* выхода.

Шаг 4. [Дельта-правило]. Производится модификация весовых коэффициентов персептрона в соответствии с формулами:

$w_{ij}[t+1]=w_{ij}[t]+\Delta w_{ij}[t]$; $\Delta w_{ij}[t]=\alpha\cdot\beta_i[t]\cdot x_j[t]$;

$w_{i,0}[t+1]=w_{i,0}[t]+\Delta w_{i,0}[t]$; $\Delta w_{i,0}[t]=\alpha\cdot\beta_i[t]$,

где α – коэффициент скорости обучения с помощью которого можно

управлять величиной коррекции весов Δ ($0<\alpha\leq 1$).

Алгоритм обучения однослойного перцептрона

__Шаг 1.__ Инициализация синаптического веса и смещения:

Значение синаптического веса $w_{ij}(0)$ ($1\leq i\leq n$) и смещения нейрона b устанавливаются равными некоторым малым случайным числам.

Обозначение: $w_j(t)$ – вес связи от j-го элемента входного сигнала к нейрону в момент времени t.

__Шаг 2.__ Подача к сети нового входного и желательно выходного сигналов:

Входной сигнал $x=(x_1,x_2\ldots x_n)$ подается к нейрону вместе с желательным выходным сигналом d.

__Шаг 3.__ Вычисление выходного сигнала нейрона:

$$y(t)=f\left(\sum_{i=1}^{n} w_i(t)x_i(t)-b\right)$$

Шаг 4. Адаптация (настройка) значений веса:

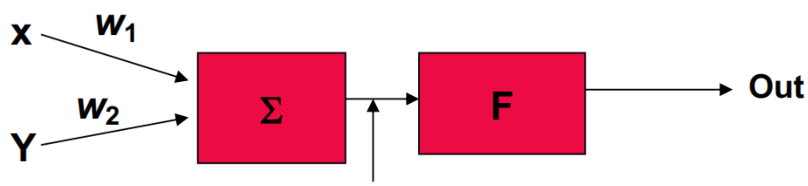
$$w_i(t+1) = w_i(t) + r[d(t) - y(t)]x_i(t), 1 \le i \le n$$

Где
$$d(t) = \begin{cases} +1, & если\ класс\ A \\ -1, & если\ класс\ B \end{cases}$$

r – шаг обучения (меньше, чем 1),
d(*t*) – желательный выходной сигнал.

Шаг 5. Переход к шагу 2.

Проблема линейной неразделимости однослойного перцептрона

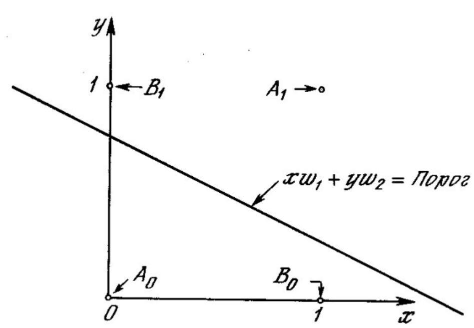


NET = $xw_1 + yw_2$

F- пороговая функция (если сумма NET меньше 0.5 – выход 0)

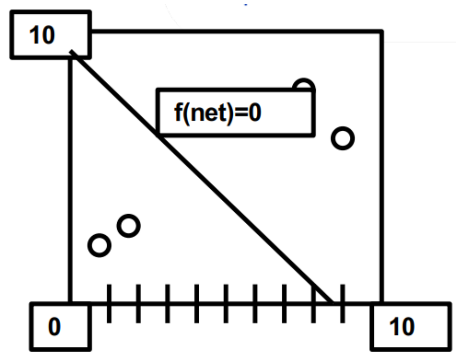
Таблица истинности для функции ИСКЛЮЧАЮЩЕЕ ИЛИ

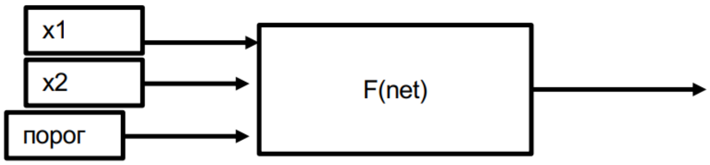
Точки	Значения *x*	Значения *y*	Требуемый выход
A_0	0	0	0
B_0	1	0	1
B_1	0	1	1
A_1	1	1	0



Пример решения задачи однослойным перцептроном

x1	x2	y
1.0	1.0	1
9.4	6.4	-1
2.5	2.1	1
8.0	7.7	-1



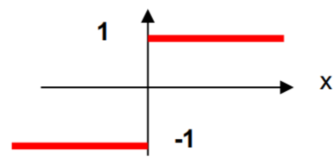


$F(net) = f(w_1x_1+w_2x_2+w_3x_3)$,

где $f(x)=sign(x)$

Если f(x)=+1 , то x принадлежит одному классу,

если f(x)= -1 – другому.



F(x) – кусочно-линейная биполярная пороговая функция

Инициализируем веса **w** случайным образом – [0.75, 0.5, -0.6]

Расчет выход первого образа

$$F(net)^1=f(0.75*1+0.5*1-0.6*1)=f(0.65)=1$$

Т.к. значение $F(net)^1$ корректно настройка весовых коэффициентов не нужна. Значит $W^2=W^1$.

Для второго примера

$$F(net)^2=f(0.75*9.4+0.5*6.4-0.6*1)=f(9.65)=1$$

На выходе ожидалось – 1, следовательно необходимо подстроить веса сети.

Примем r = 0.2 .

$$W^3=W^2+0.2(-1-1)\,x^2 = \begin{bmatrix} 0.75 \\ 0.5 \\ -0.6 \end{bmatrix} - 0.4 \begin{bmatrix} 9.4 \\ 6.4 \\ 1.0 \end{bmatrix} = \begin{bmatrix} -3.01 \\ -2.06 \\ -1.00 \end{bmatrix}$$

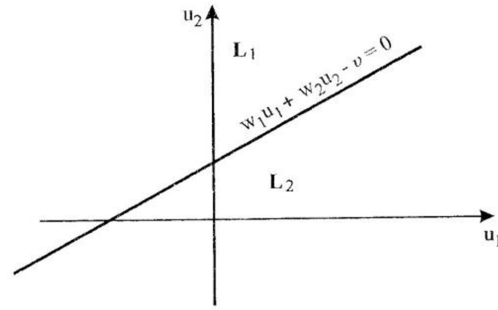
Вычисляем выход сети для третьего примера с учетом новых весов

$F(net)^3 = f(-3.01*2.5 - 2.06*2.1-1.0*1) = f(-12.84) = -1$

Снова подстраиваем веса сети

$$W^4 = W^3 + 0.2(-1-(-1))\,x^3 = \begin{bmatrix} -3.01 \\ -2.06 \\ -1 \end{bmatrix} + 0.4 \begin{bmatrix} 2.5 \\ 2.1 \\ 1.0 \end{bmatrix} = \begin{bmatrix} -2.01 \\ -1.22 \\ -0.60 \end{bmatrix}$$

Линейная сеть

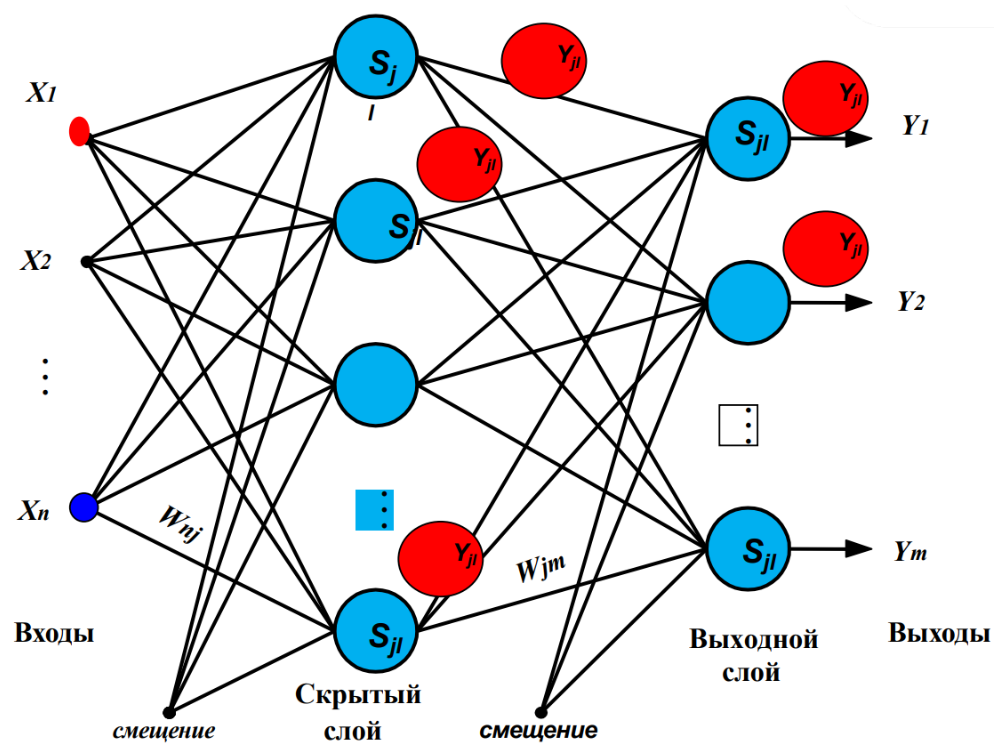


Линейная модель сети – это сеть без промежуточных слоев, которая в выходном слое содержит только линейные элементы.

Во время работы сеть фактически умножает вектор входов на матрицу весов, а затем к полученному вектору прибавляет вектор смещения (вектор пороговых значений).

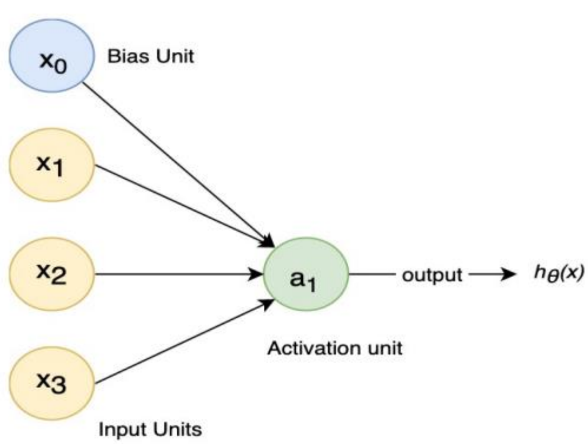
Линейная сеть является хорошей точкой отсчета для оценки качества построенных Вами нейронных сетей. Может оказаться так, что задачу, считавшуюся очень сложной, можно успешно не только нейронной сетью, но и простым линейным методом. Если же в задаче не так много обучающих данных, то, вероятно, просто нет оснований использовать более сложные модели.

Многослойная полносвязная НС или многослойный перцептрон (МП)



Модель нейрона (логическая единица)

Модель одного нейронного блока



$$x_0 = 1$$

$$x = \begin{bmatrix} x_o \\ x_1 \\ x_2 \\ x_3 \end{bmatrix}$$

Вес:

$$\theta = \begin{bmatrix} \theta_0 \\ \theta_1 \\ \theta_2 \\ \theta_3 \end{bmatrix}$$

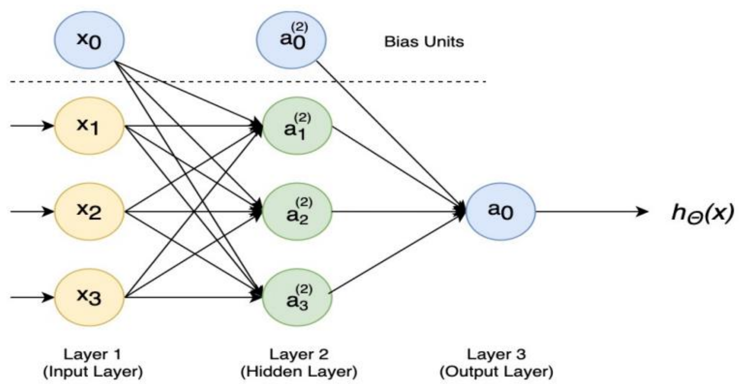
Работа многослойного перцептрона

$$S_{jl} = \Sigma\, w_{ijl} * x_{ijl},\ Y_{jl} = F\,(S_{ji} - b_{\,jl}),\ X_{ij(l+1)} = Y_{il},$$

где i – номер входа,
j – номер нейрона в слое,
l – номер слоя,
x_{ijl} – i-й входной сигнал j-го нейрона в слое l,
w_{ijl} – весовой коэффициент i-го входа нейрона номер j в слое l,
S_{jl} – сигнал S_j – го нейрона в слое l,
Y_{jl} – выходной сигнал нейрона,
b_{jl} – пороговый уровень нейрона j в слое l.

Введем обозначения

w_{ijl} – вектор-столбец веса для всех входов нейрона j в слое l,
W_l – матрица веса всех нейронов слоя l.



$a_i^{(j)}$ - «активация» i-го агрегата в j-м слое .

$\Theta^{(j)}$ - матрица весов, управляющая отображением функции из слоя j в слой $j + 1$. Например , для первого слоя: $\Theta^{(1)} \in R^{3x4}$.

L - общее количество слоев в сети (в нашем примере 3).

s_l - количество единиц (не считая единицы смещения) в слое l .

K - количество выходных единиц (1 в нашем примере, но может быть любым действительным числом для мультиклассовой классификации).

Ошибка сети

Суммарная квадратичная ошибка

$$E = \frac{1}{2}\sum_S \sum_j (d_j^S - y_j^S)^2$$

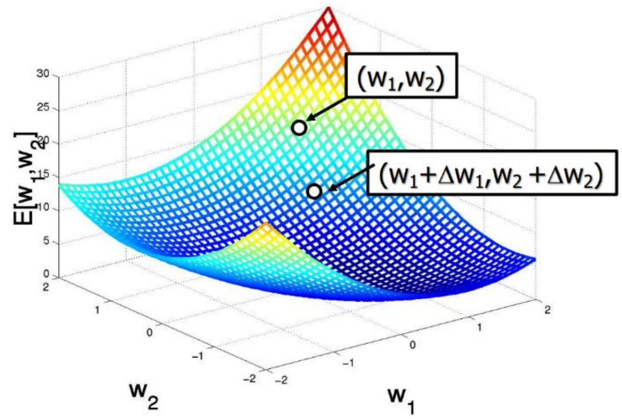
Средняя относительная ошибка

$$\sigma = \frac{1}{SN_o}\sum_S \sum_j \left(\frac{|\, d_j^S - y_j^S \,| + 1}{|\, d_j^S \,| + 1} - 1 \right) 100\%$$

Алгоритм обучения МП - обратное распространение ошибки

Алгоритм обратного распространения ошибки — это итерационный градиентный алгоритм, который используется с целью минимизации среднеквадратичного отклонения текущего выхода многослойного перцептрона и желаемого выхода

Шаги работы многослойного перцептрона



Шаг 1. Инициализация веса и смещений.
Веса $w_{ij}(k)$ и смещения $w_{i0}(k)$ во всех слоях задаются случайным образом как малые величины, например, в интервале от -1 до +1.

Шаг 2. Подача нового входного вектора *X* и соответствующего желаемого выходного вектора *D*.

Шаг 3. Прямой проход – расчет фактического выхода.
Вычисление выхода $Y_i(k)$ для i-го узла в k-м скрытом слое, k=1,…,K и
Y_i в выходном слое:

$$Y_i^{(k)} = f_\delta\Big(w_{i0}^{(k)} + \sum_{j=1}^{H_{k-1}} w_{ij}^{(k)} Y_j^{(k-1)}\Big)$$

где $Y_j^{(0)}=X_j$

$$Y_i = f_\delta\Big(w_{i0} + \sum_{j=1}^{H_k} w_{ij}^{(K+1)} Y_j^{(K)}\Big)$$

Шаг 4. Обратный проход – адаптация веса и порогов. Использование рекурсивного алгоритма, который начинает работать на выходных узлах и возвращается к первому скрытому слою:

$$w_{ij}^{(k)}(t+1) = w_{ij}^{(k)}(t) + \eta\, \delta_i^{(k)} Y_j^{(k-1)} \qquad (k=1,\dots,K+1).$$

Для $K=K+1$ член $\delta_i^{(k)}$, который описывает ошибку, известен:

$$\delta_i^{(K+1)} = (D_i - Y_i)\, Y_i(1-Y_i)$$

и его можно рекурсивно рассчитать для всех других случаев:

$$\delta_i^{(k)} = Y_i^{(k)} \textstyle\sum_j \delta_j^{(k+1)} w_{ij}^{(k+1)}, \quad (k=1,\dots,K).$$

Отметим.

$Y_i^{(k)}(1-Y_i^{(k)})$ – производная функции активации $\quad Y = \dfrac{1}{1+e^{-\alpha X}}$

Параметр обучения η обычно выбирается в интервале от 0 до +1.

Шаг 5. Возврат к шагу 2.

Производные других видов активационных функций

Биполярная сигмоидная $y = \dfrac{1}{1+e^{-S}} - 0.5$ → **$Y' = 1/2\,(1-y^2)$**

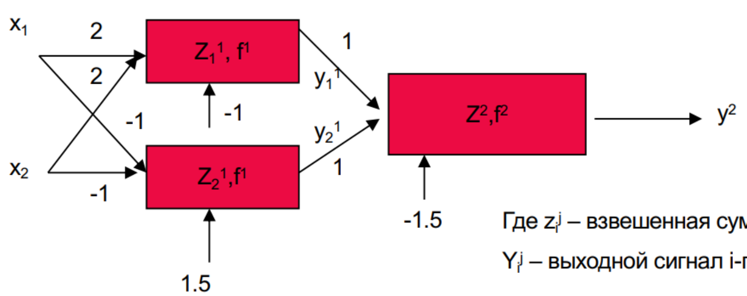
Гиперболический тангенс $Y = th(S)$ → **$Y' = (1-y^2)$**

Логарифмическая $y = \ln\,(S+[s^2+1]^{1/2})$ → **$Y' = 1/\,(S^2+1)^{1/2}$**

В общем виде алгоритм обучения МП по методу обратного распространения ошибки:

- Инициализация весов случайными значениями
- Последовательно или случайно берется образ из обучающей выборки, для которого сначала производится прямой ход вычисления выходных значений, а затем обратный ход коррекции весов
- Вычисляется текущая ошибка E и сравнивается с требуемой E_{min}, если $E < E_{min}$, то обучение прекращается, иначе повтор со 2 шага.

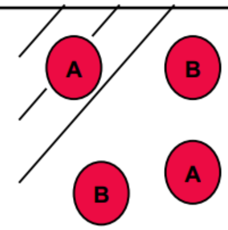
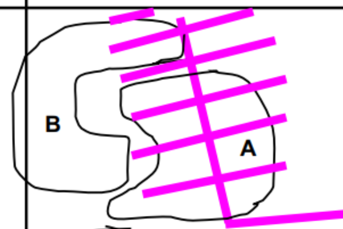
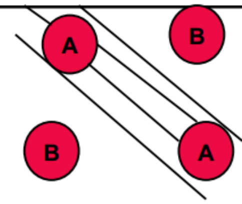
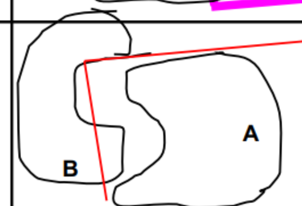
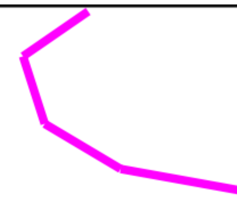
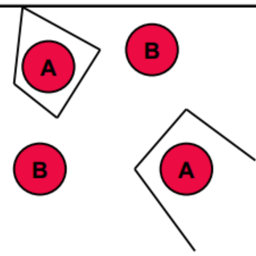
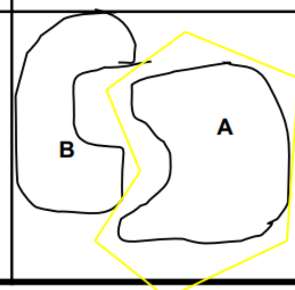
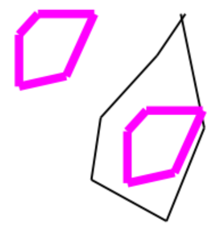
Пример – одна из возможных реализаций функции «исключающее ИЛИ»



Где z_i^j – взвешенная сумма входов i-го нейрона j-го слоя;

Y_i^j – выходной сигнал i-го нейрона j-го слоя

x_1	x_2	y_1^1	y_2^1	y^2
0	0	2*0+2 *0-1=0	(-1) *0+(-1) *0+1.5=1	1 *0+1*0-1.5=0
0	1	2 *0+2 *1-1=1	(-1) *0+(-1) *0+1.5=1	1 *1+1*1-1.5=1
1	0	2 *1+2 *0-1=1	(-1) *1+(-1) *0+1.5)=1	1 *1+1*1-1.5=1
1	1	2 *1+2 *1-1=1	(-1) *1+(-1)*1+1.5=0	1 *0+1*0-1.5=0

Структура персептрона	Область решения	Исключающее ИЛИ	Произвольно расположенные области	Области наиболее общей формы
Однослойная	Полуплоскость, ограниченная гиперплоскостью			
Двухслойная	Выпуклые открытые или закрытые области			
Трехслойная	Произвольной сложности			

Дельта правило обучения многослойного перцептрона

<u>Дельта правило используется только с непрерывными, дифференцируемыми функциями</u> <u>в режиме обучения "с учителем".</u>

Начальный вес любой. Коррекция выполняется пропорционально величине производной с данной координаты. Производная берется от функции активации. Подстройка веса j для нейрона i по формуле:

$$\Delta w_{ij} = \eta\, [d_i - f(\mathrm{net}_i)]\, f'(\mathrm{net}_i)\, x_j, \text{ где } j = 1,2,\ldots,n;$$

$\eta > 0$ – коэффициент обучения подбирается эвристично для данной предметной области.

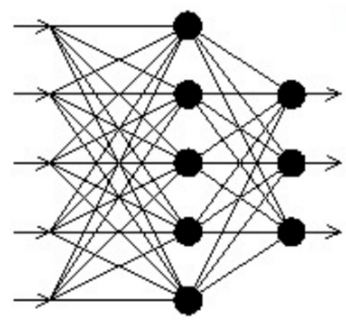
Ошибка при обучении на шаге k:

$$E_k = 1/2\, [d_i - f(\mathrm{net}_i)]^2, \text{ где } d_i \text{ – ожидаемый выход.}$$

Общая ошибка при обучении: $E = 1/2p \sum [d_i - f(\mathrm{net}_i)]^2$,

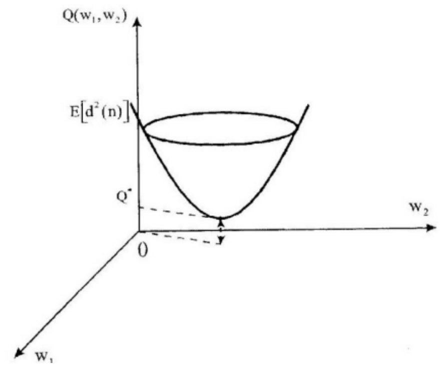
где p – число примеров в обучаемой выборке.

Обучение многослойного персептрона



Целью обучения является поиск таких значений весов и порогов сети, которые бы минимизировали ошибку прогноза, выдаваемого сетью.

Функции ошибок:
- сумма квадратов ошибок;
- среднеквадратическая ошибка.



Поверхности ошибок:
- параболоид;
- сложная форма (с множеством глобальных минимумов).

Алгоритм обратного распространения

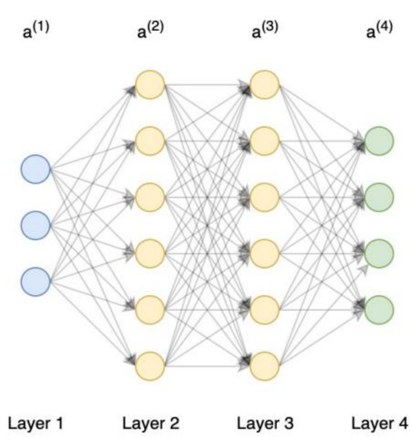
- вектор градиента указывает направление кратчайшего спуска
- величина шага берется пропорциональна «крутизне склона»
- в алгоритм вводится слагаемое импульса (или инерции)

Мультиклассовая классификация

Чтобы нейронная сеть работала с мультиклассовым уведомлением, мы можем использовать подход **One-vs-All** .

Допустим, мы хотим , чтобы наша сеть , чтобы отличить , если есть *пешеход* или *автомобиль* на *мотоцикл* или *грузовик* находится на изображении.

В этом случае выходной слой нашей сети будет иметь 4 единицы (входной слой будет намного больше, и в нем будут все пиксели изображения. Скажем, если все наши изображения будут 20x20 пикселей, то входной слой будет иметь 400 единиц каждое. из которых будет содержать черно-белый цвет соответствующей картинки).



$h_\Theta(x) \in R^4$

В этом случае мы ожидаем, что наша окончательная гипотеза будет иметь следующие значения:

$$h_\Theta(x) = \begin{bmatrix} 1 \\ 0 \\ 0 \\ 0 \end{bmatrix},\ for\ "pedestrian"$$

$$h_\Theta(x) = \begin{bmatrix} 0 \\ 1 \\ 0 \\ 0 \end{bmatrix},\ for\ "car"$$

$$h_\Theta(x) = \begin{bmatrix} 0 \\ 0 \\ 1 \\ 0 \end{bmatrix},\ for\ "motorcycle"$$

В этом случае для обучающей выборки:

$$\left\{ \left(x^{(1)},\ y^{(1)}\right),\ \left(x^{(2)},\ y^{(2)}\right),\ \dots,\ \left(x^{(m)},\ y^{(m)}\right) \right\}$$

Мы бы хотели иметь:

$$y^{(i)}\ one\ of \begin{bmatrix} 1 \\ 0 \\ 0 \\ 0 \end{bmatrix}, \begin{bmatrix} 0 \\ 1 \\ 0 \\ 0 \end{bmatrix}, \begin{bmatrix} 0 \\ 0 \\ 1 \\ 0 \end{bmatrix}, \begin{bmatrix} 0 \\ 0 \\ 0 \\ 1 \end{bmatrix}$$