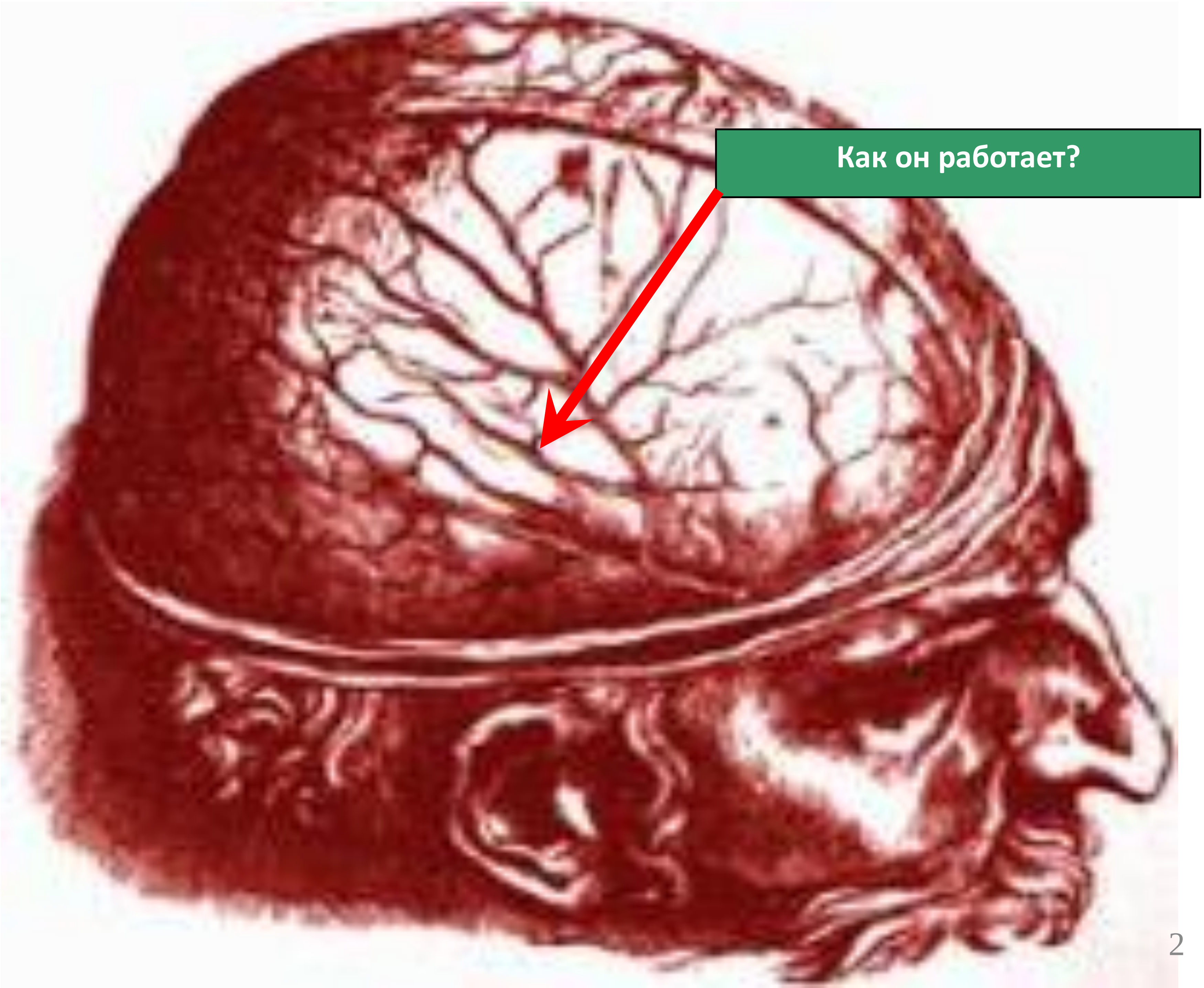
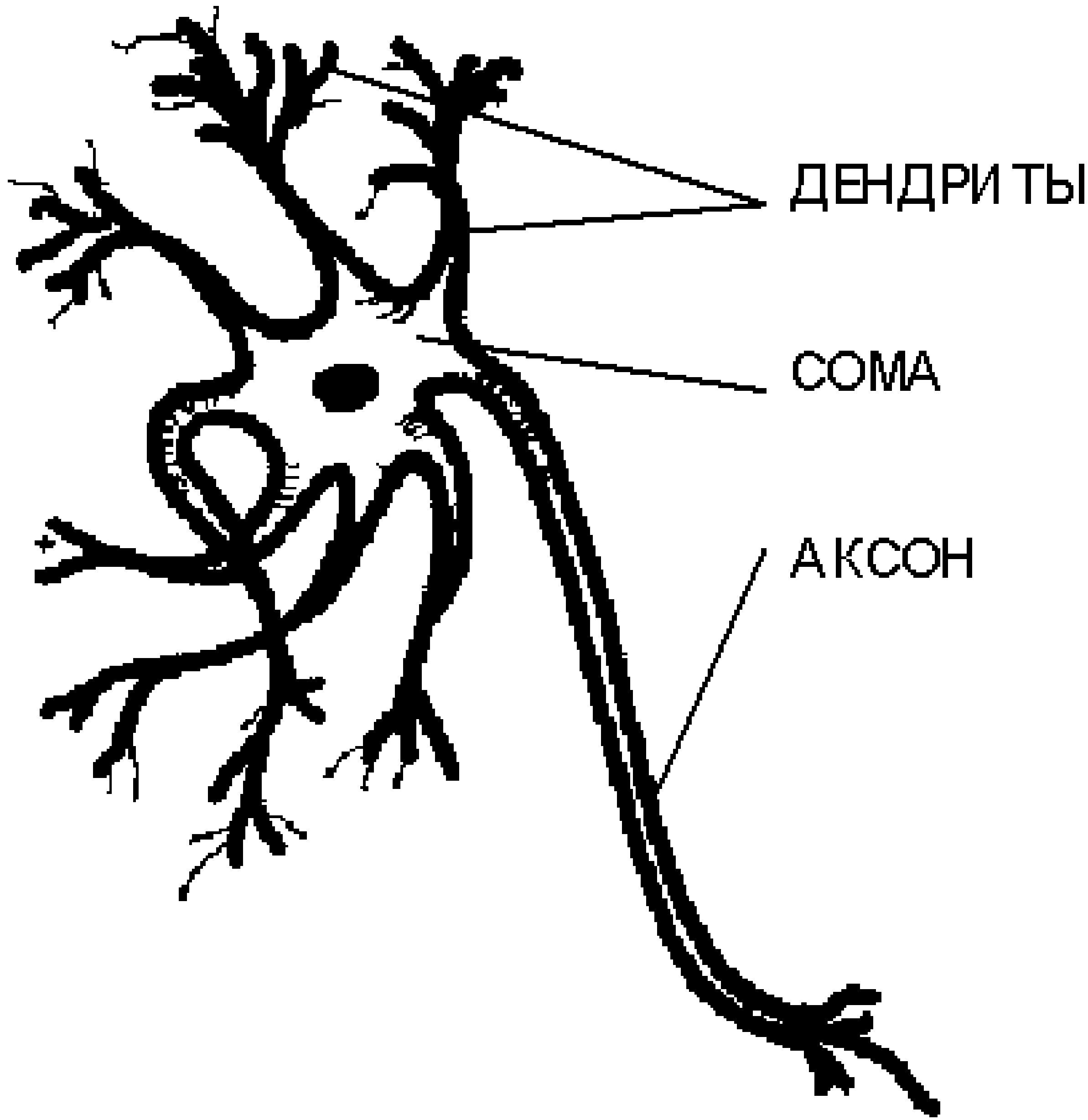
Нейронные сети
Введение


Как он работает?

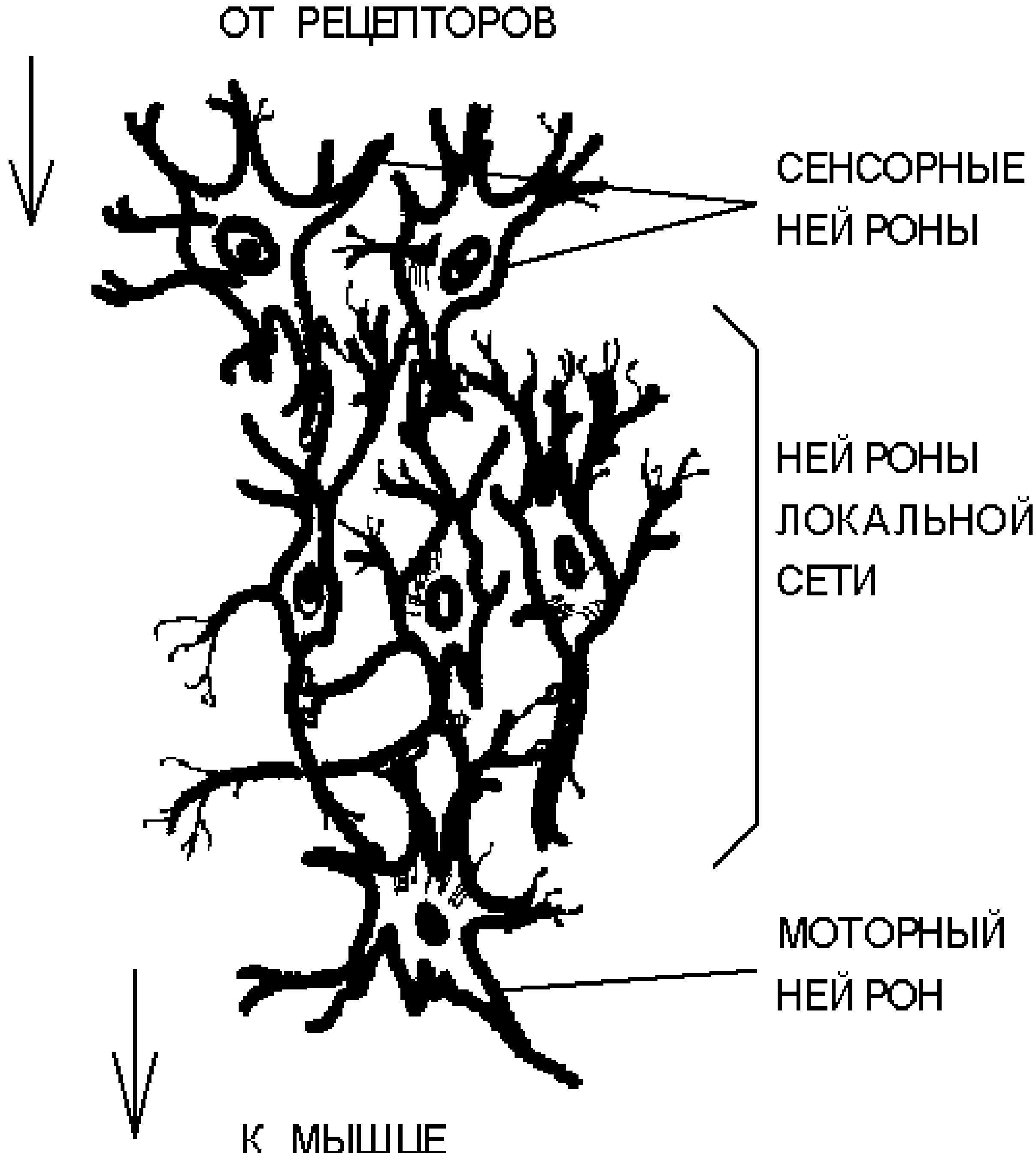
Современная биология:

- Клетка - элементарный процессор, способный к простейшей обработке информации
- Нейрон - элемент клеточной структуры мозга
- Нейрон осуществляет прием и передачу информации в виде импульсов нервной активности
- Природа импульсов - электрохимическая


ДЕНДРИ ТЫ
СОМА
А КСОН

Интересные данные

- Тело клетки имеет размер 3 - 100 микрон
- Гигантский аксон кальмара имеет толщину 1 миллиметр и длину несколько метров
- Потенциал, превышающий 50 мВ изменяет проводимость мембраны аксона
- Общее число нейронов в ЦНС человека порядка 100.000.000.000
- Каждая клетка связана в среднем с 10.000 других нейронов
- Совокупность в объеме 1 мм*3 - независимая локальная сеть


ОТ РЕЦЕПТОРОВ
СЕНСОРНЫЕ НЕЙ РОНЫ
НЕЙ РОНЫ ЛОКАЛЬНОЙ СЕТИ
МОТОРНЫЙ НЕЙ РОН
К МЫШЦЕ

Нервная ткань:

- Лишена регенерации
- Её нейроны способны формировать новые отростки и синаптические контакты
- Развитие нейронных ответвлений сопровождается конкуренцией за синаптические участки
- Специфическая изменчивость нейронных сетей лежит в основе их способности к обучению

Создание первых ИНС

Первые шаги в области искусственных нейронных сетей были сделаны В. Мак-Калахом и В. Питсом, которые показали в 1943 г., что с помощью пороговых нейронных элементов можно реализовать исчисление логических функций для распознавания образов.

В 1949 г. Дональдом Хеббом было предложено правило обучения, ставшее основой для обучения ряда сетей, а в начале шестидесятых годов Ф. Розенблатт исследовал модель нейронной сети, названной им персептроном.

Исследование ИНС

Анализ однослойных персептронов, проведенный М. Минским и С. Пайпертом в 1969 г., показал присущие им ограничения, связанные с невозможностью представления «исключающего или» такими сетями, что сыграло негативную роль для дальнейшего развития исследований в области нейронных сетей.

Возрождение ИНС

В восьмидесятые годы возрождается интерес к искусственным нейронным сетям в связи с разработкой методов обучения многослойных сетей.

Джон Хопфилд исследовал устойчивость сетей с обратными связями и в 1982 г. предложил их использовать для решения задач оптимизации. В это же время Тео Кохонен предложил и исследовал самоорганизующиеся сети, а **метод обратного распространения ошибки** стал мощным средством обучения нейронных сетей.

ИСКУССТВЕННЫЕ НЕЙРОННЫЕ СЕТИ

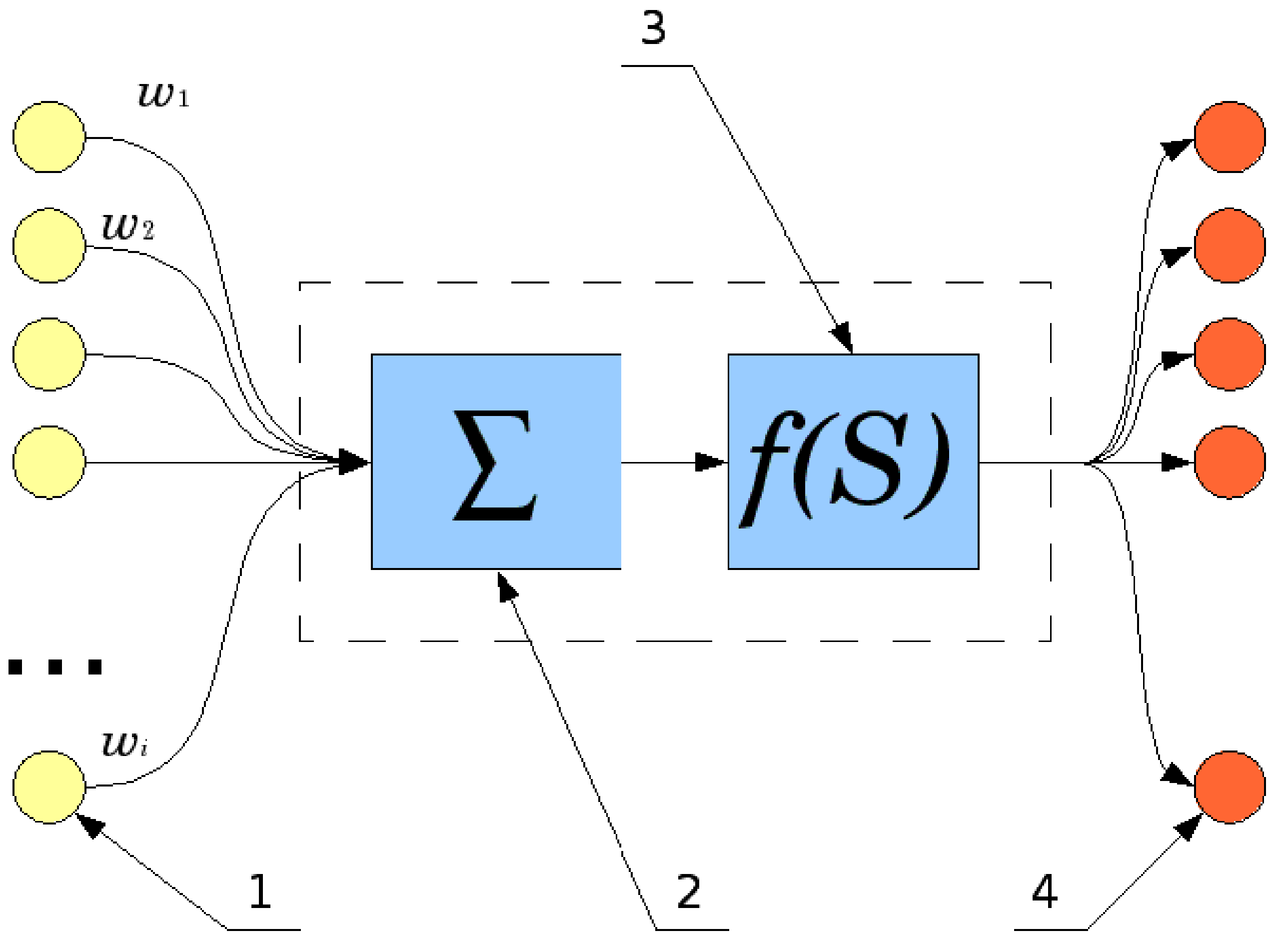
Основным элементом искусственной нейронной сети является нейронный элемент или формальный нейрон, осуществляющий **операцию нелинейного преобразования суммы произведений входных сигналов на весовые коэффициенты.**

НЕЙРОННЫЙ ЭЛЕМЕНТ

Связи, по которым выходные сигналы одних нейронов поступают на входы других, часто называют синапсами по аналогии со связями между биологическими нейронами. Каждая связь характеризуется своим весом. Связи с положительным весом называются **возбуждающими**, а с отрицательным — **тормозящими**.

Нейрон имеет один выход, часто называемый аксоном по аналогии с биологическим прототипом. С единственного выхода нейрона сигнал может поступать на произвольное число входов других нейронов. Схема искусственного нейрона приведена далее.

Схема искусственного нейрона



Элементы искусственного нейрона

1- нейроны, выходные сигналы которых поступают на вход ($\mathbf{x_i}$), $\mathbf{w_i}$ — веса входных сигналов

2- сумматор входных сигналов, умноженных на их весовые коэффициенты;

3- вычислитель передаточной функции (**функции активации**);

4- нейроны, на входы которых подается выходной сигнал данного нейрона.

Нейрон имеет один выход, часто называемый аксоном по аналогии с биологическим прототипом.

С единственного выхода нейрона сигнал может поступать на произвольное число входов других нейронов.

Уровень возбуждения нейронного элемента равен или в векторном виде $S=X\cdot W$. Взвешенная сумма S представляет собой скалярное произведение вектора весов на входной вектор:

$$S = \sum_{i=1}^{n} w_i \cdot x_i \cdot = |w| \cdot |x| \cdot \cos \alpha’$$

где $|w|$, $|x|$ – длины векторов W и X соответственно,

а α – угол между этими векторами.

В большинстве случаев **функции активации** является монотонно возрастающей и имеет область значений [−1, 1] или [0, 1], однако существуют исключения. Искусственный нейрон полностью характеризуется своей передаточной функцией.

Использование различных передаточных функций позволяет вносить нелинейность в работу нейрона и в целом нейронной сети.

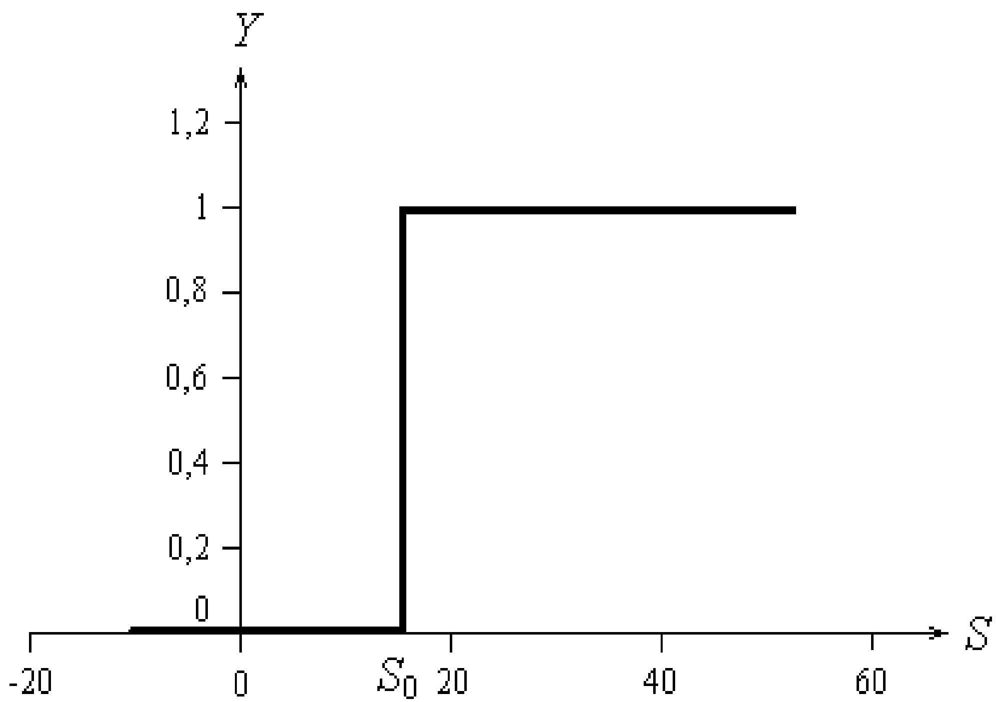
ФУНКЦИИ АКТИВАЦИИ НЕЙРОННЫХ ЭЛЕМЕНТОВ

Наиболее распространенными функциями активации, нелинейными усилительными характеристиками нейронного элемента или передаточными функциями являются следующие: **пороговая, сигнум, логистическая, гиперболический тангенс, линейная, радиальная базисная** и др.

Пороговая бинарная функция

Для пороговой бинарной функции нейронный элемент остается неактивным до достижения входом порогового значения S_0.

$$Y(s) = \begin{cases} 0, & S \le S_0 \\ 1, & S > S_0 \end{cases}$$



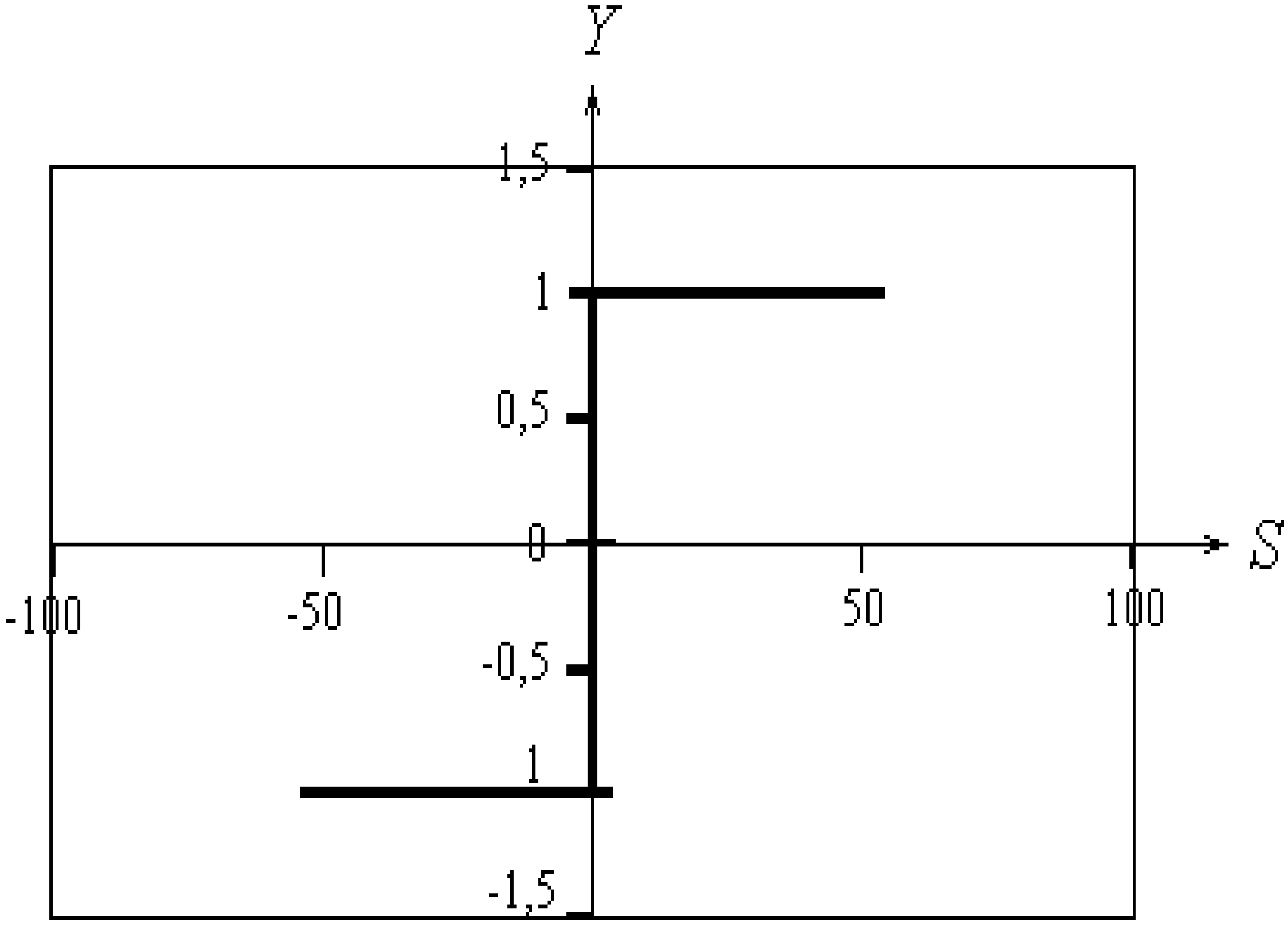
Сигнум

Если S_0=0 , то бинарная пороговая функция называется единичной функцией активации с жестким ограничением (hardlim(S)).

Сигнум, или модифицированная пороговая функция, для которой значение S_0=0 задается уравнением

$$Y(s) = \begin{cases} -1, S < S_0 \\ 0, S = 0 \\ 1, S > S_0 \end{cases}$$

Сигнум



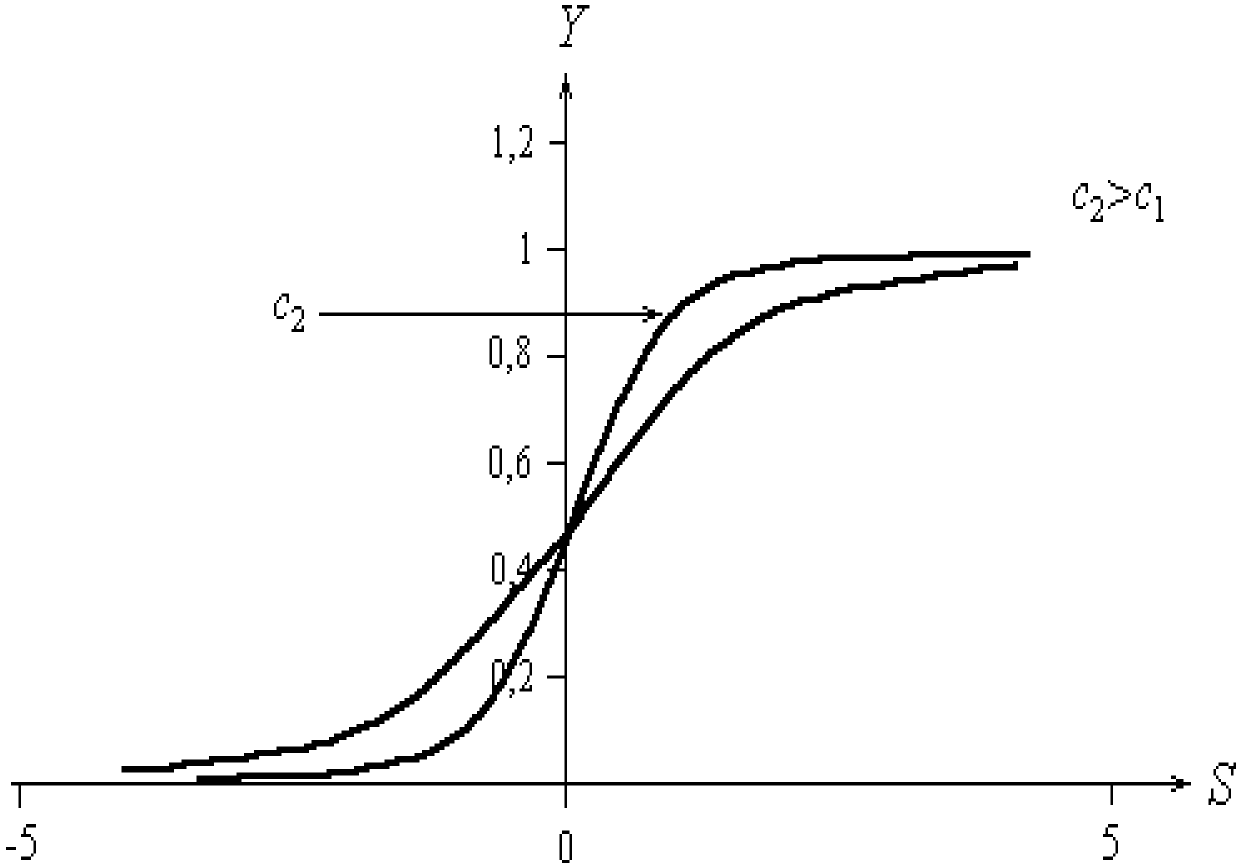
Сигмоидная логистическая функция

Сигмоидная логистическая функция (S-образная, имеющая две горизонтальные асимптоты и одну точку перегиба) является возрастающей сжимающей функцией, значения которой принадлежат интервалу (0; 1)

$$Y(s) = \frac{1}{1 + \exp(-c \cdot s)} \quad ,$$

где $c > 0$ – коэффициент, характеризующий крутизну логистической функции, усиливающей слабые сигналы (logsig(S)).

Сигмоидная логистическая функция



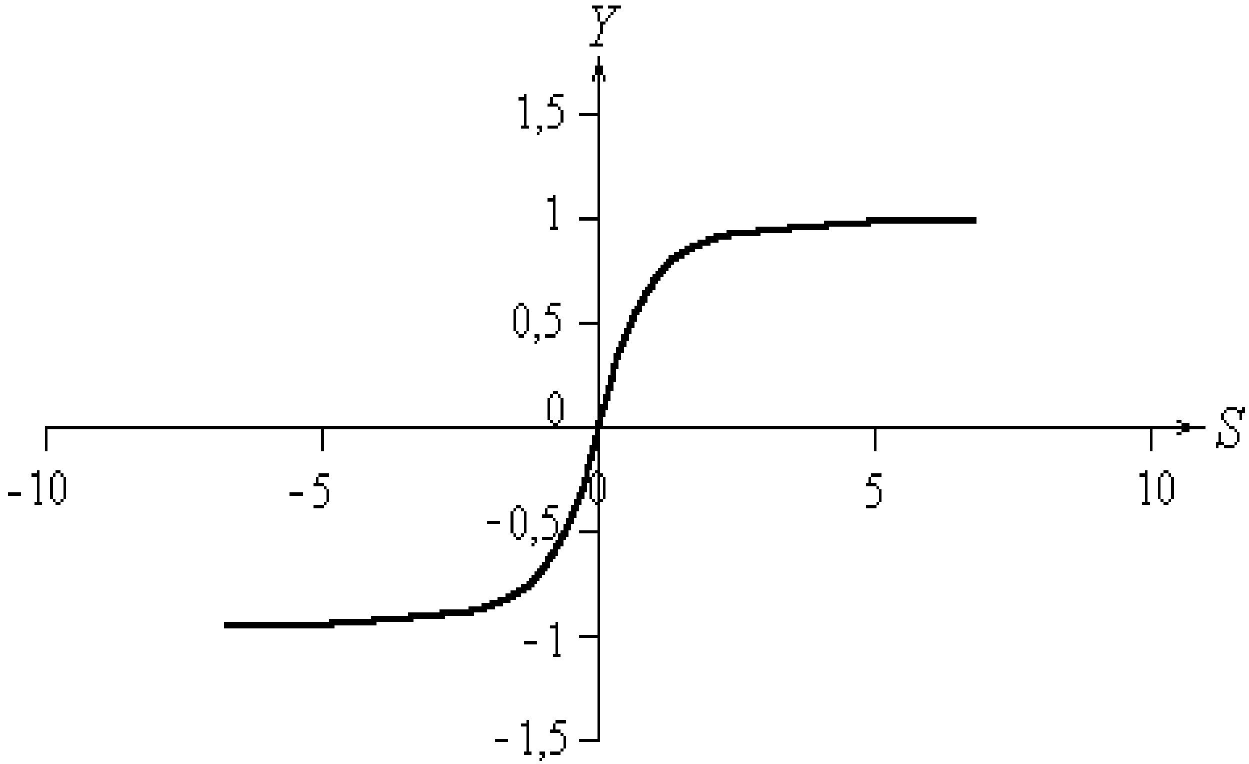
Биполярная логистическая функция

Биполярная логистическая функция уравнение которой

$$Y(s) = \frac{2}{1 + \exp(-c \cdot s)} - 1,$$

принимает значения в диапазоне (-1; 1).

Биполярная логистическая функция

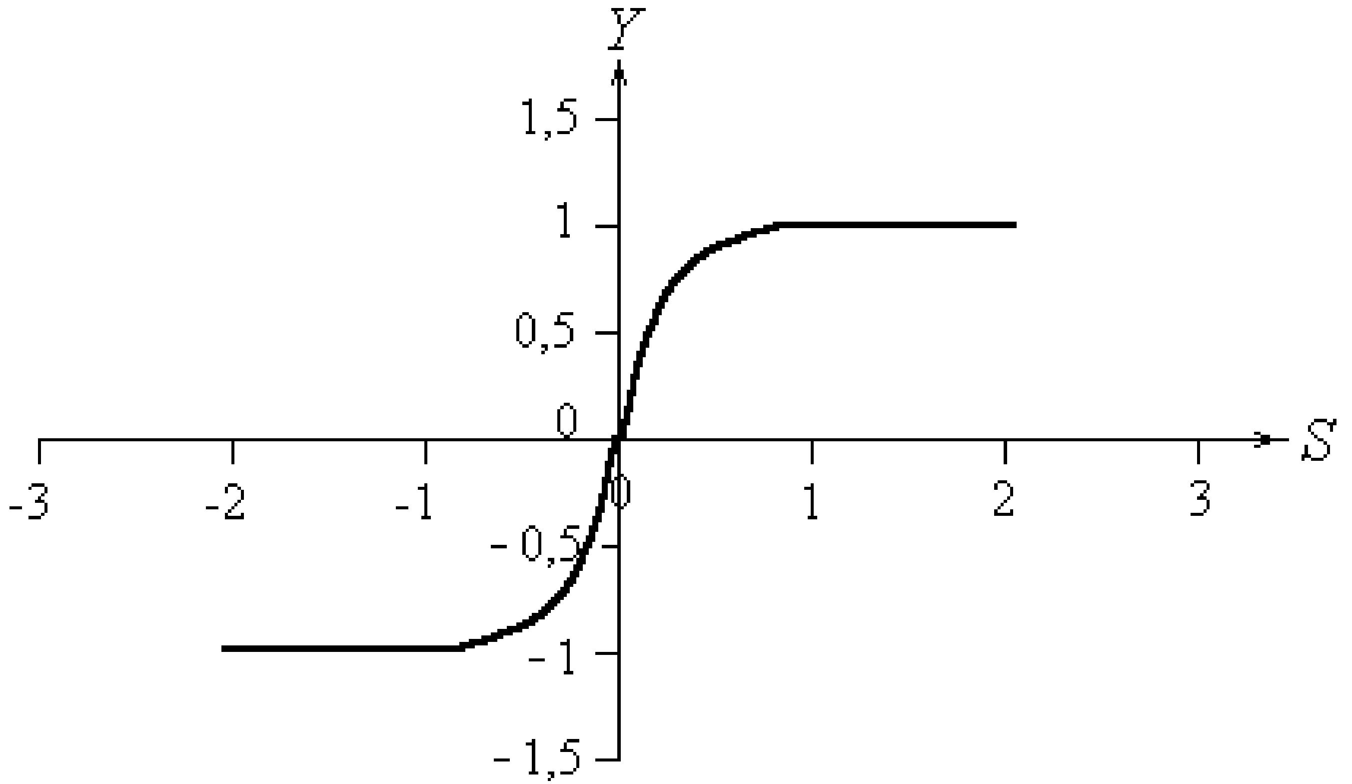


Гиперболический тангенс

аналогичен биполярной логистической функции без смещения и является симметричной функцией (tansig(S)):

$$Y(s) = \frac{\exp(c \cdot s) - \exp(-c \cdot s)}{\exp(c \cdot s) + \exp(-c \cdot s)}$$

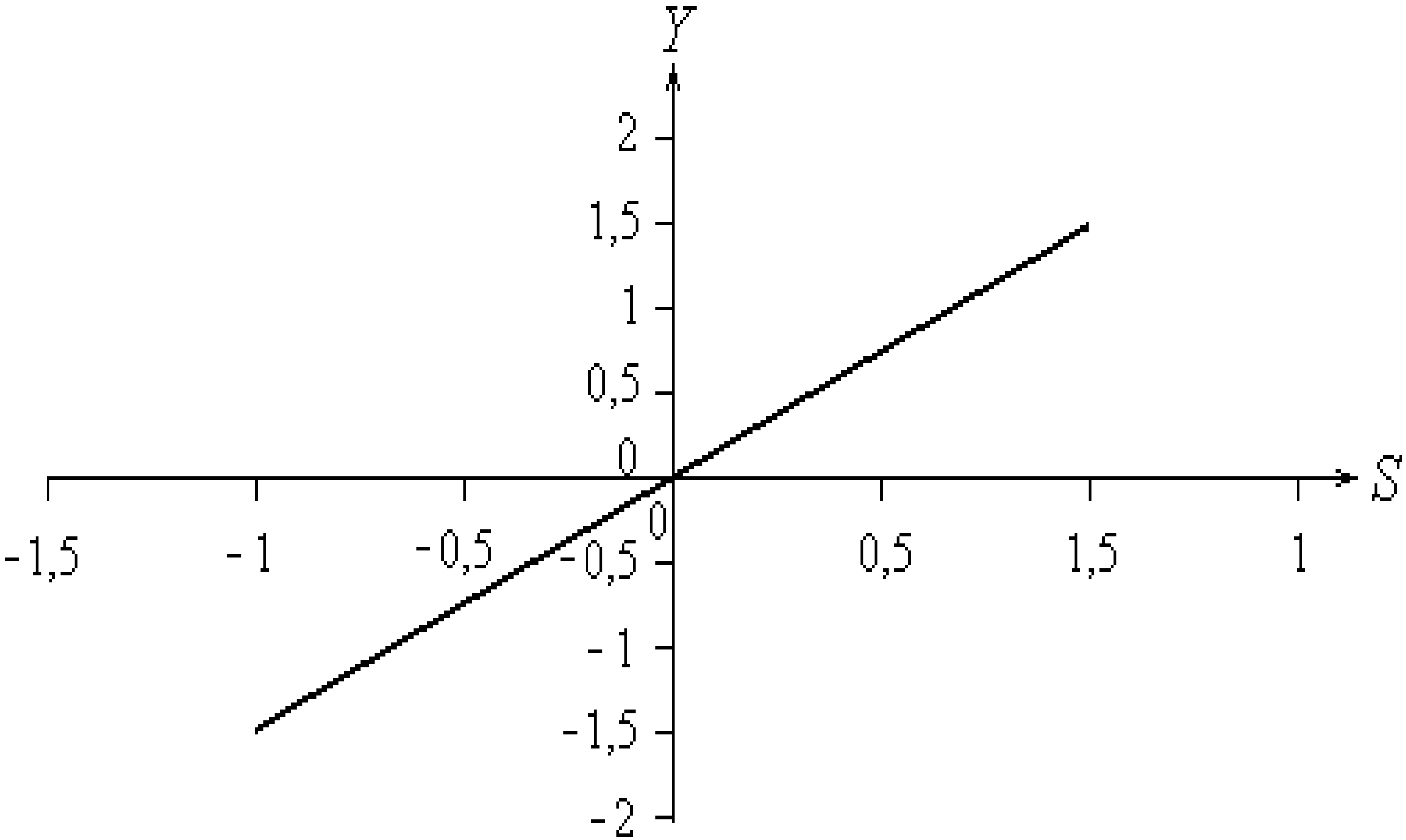
Гиперболический тангенс



Линейная функция

Линейная функция активации, уравнение которой $Y(s) = k \cdot s$, где k – угловой коэффициент наклона прямой, представлена далее (purelin(S)).

Линейная функция



Радиально-базисная функция

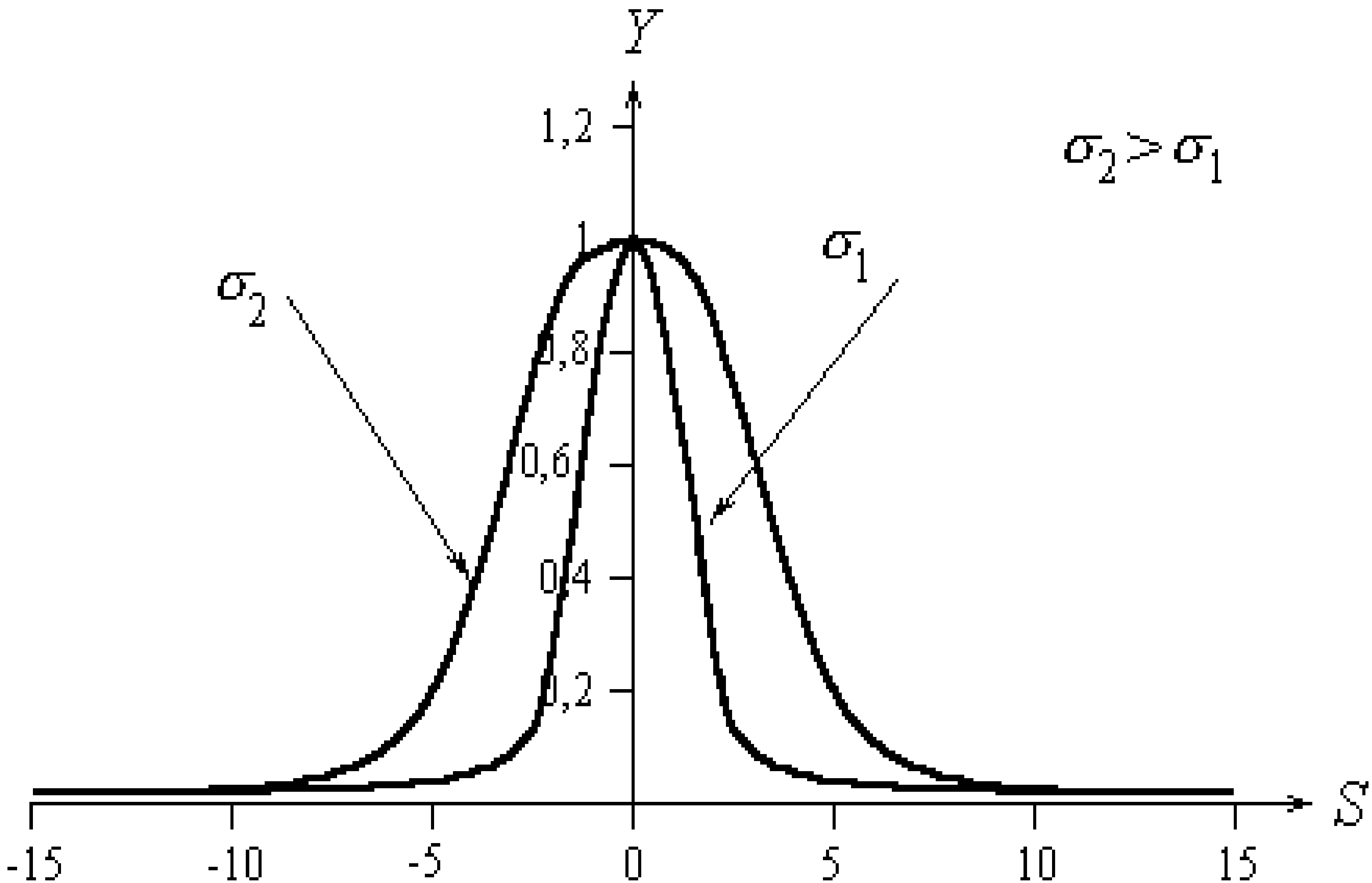
Радиально-базисная функция активации (radbas(S)) характеризуется функцией Гаусса для нормального закона распределения, в соответствии с которой:

$$Y(s) = \exp\left(\frac{-s^2}{2 \cdot \sigma^2}\right) ,$$

где σ – среднеквадратичное отклонение, характеризующее крутизну радиально-базисной функции . Величина s определяется в соответствии с евклидовым расстоянием между входным и весовым векторами:

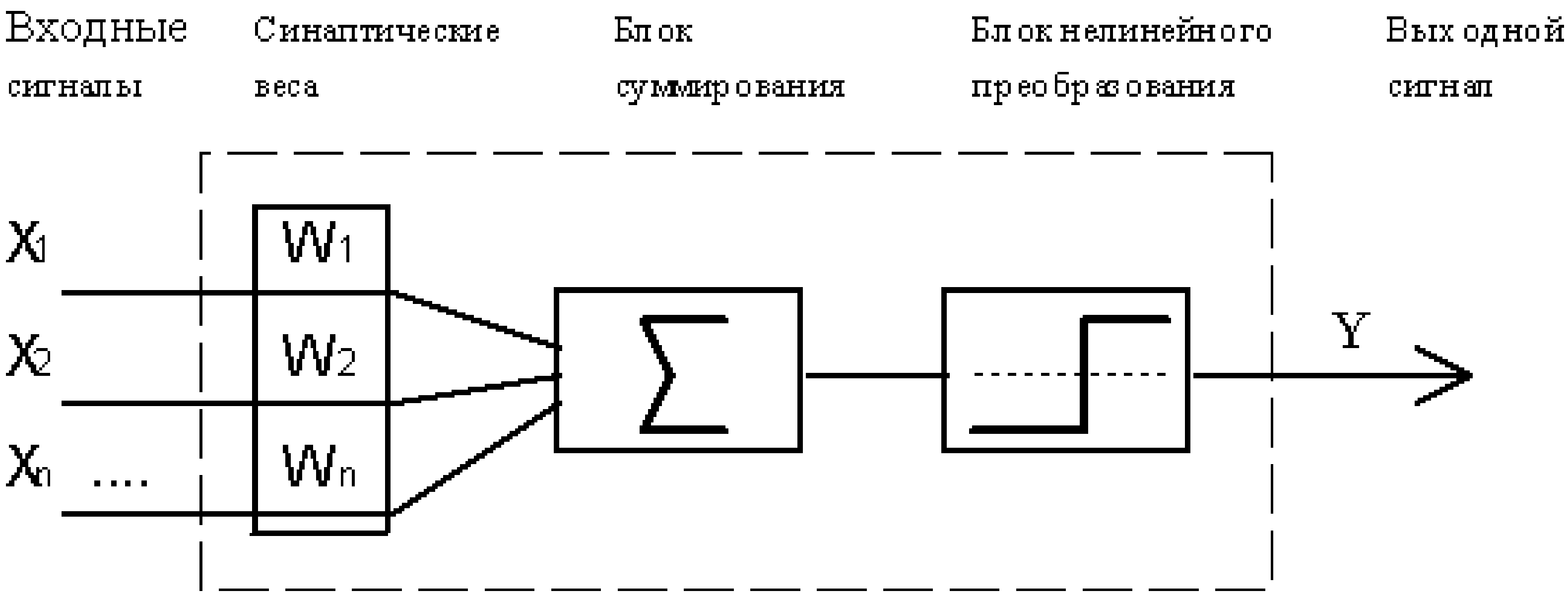
$$S^2 = |X - W|^2 = \left(\sum_{i=1}^{n}(x_i - w_i)^2\right) .$$

Радиально-базисная функция



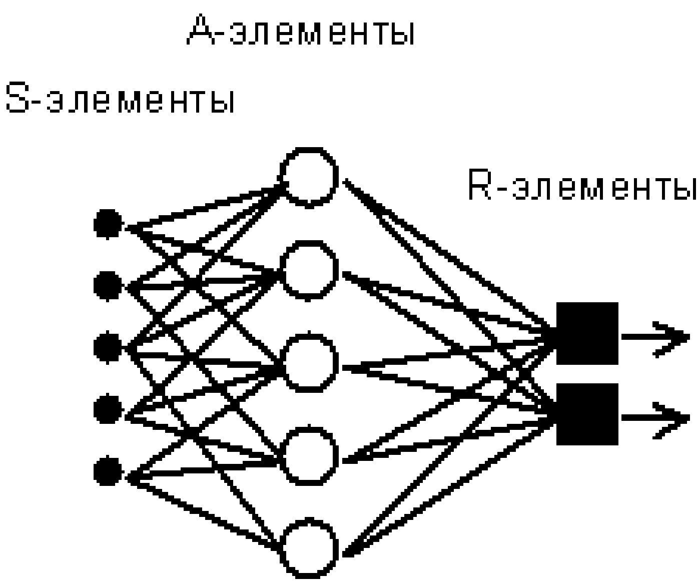
Многомерные радиальные распределения позволяют производить многомерный анализ путем сведения его к анализу одномерных симметричных распределений, таких как многомерное нормальное распределение или равномерное в шаре с центром в начале координат

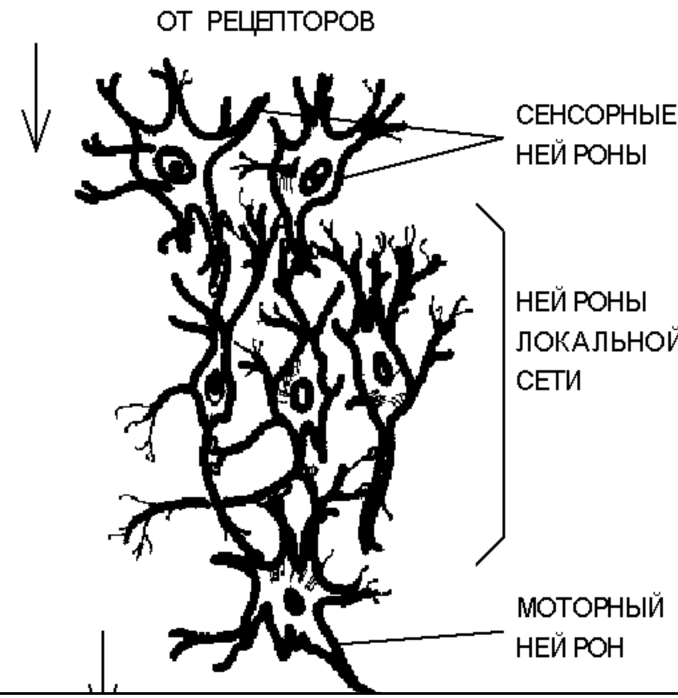
Формальный нейрон



$$net = \sum_{i=1}^{n} W_i x_i$$

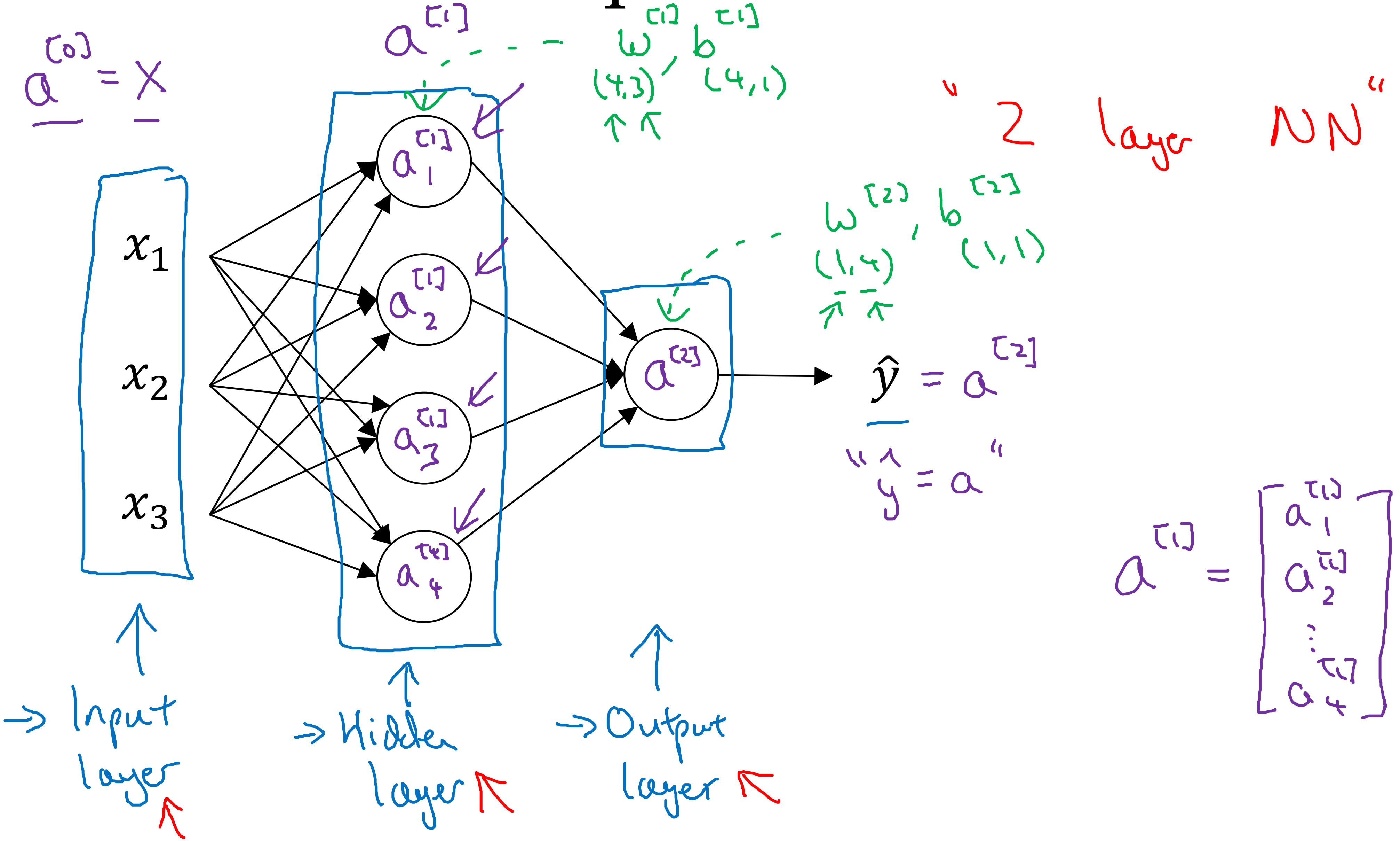
Перцептрон Розенблата



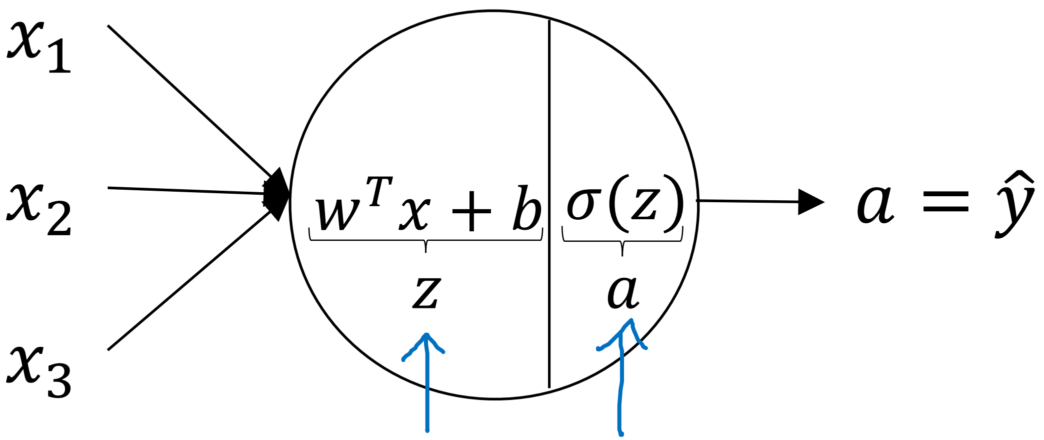


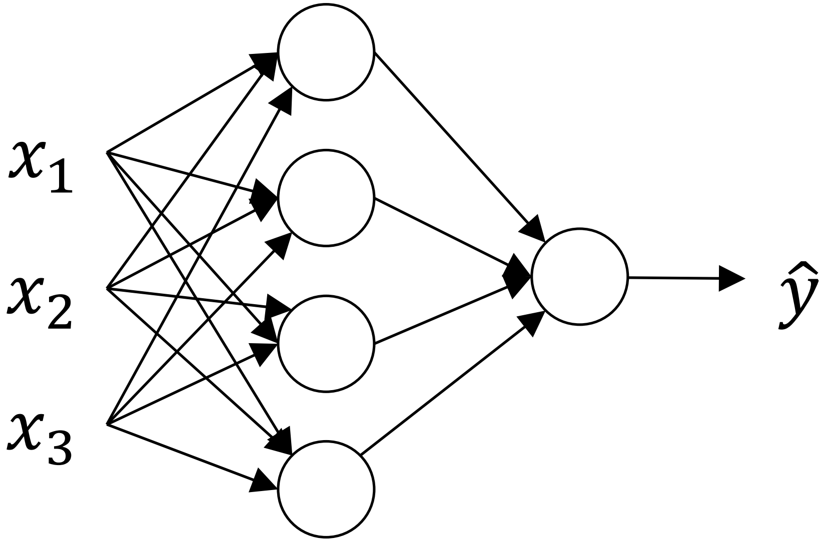
Розенблат: нейронная сеть рассмотренной архитектуры будет способна к воспроизведению *любой* логической функции.
(неверное предположение)

Neural Network Representation



Neural Network Representation

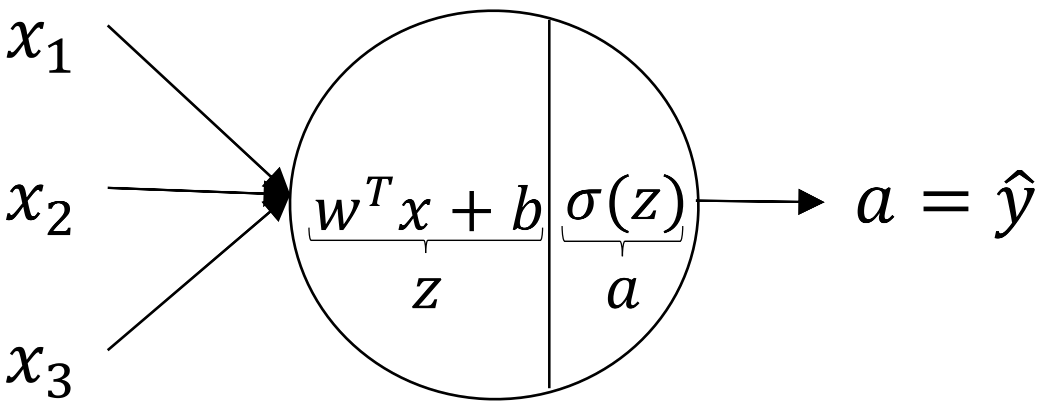




$$z = w^T x + b$$

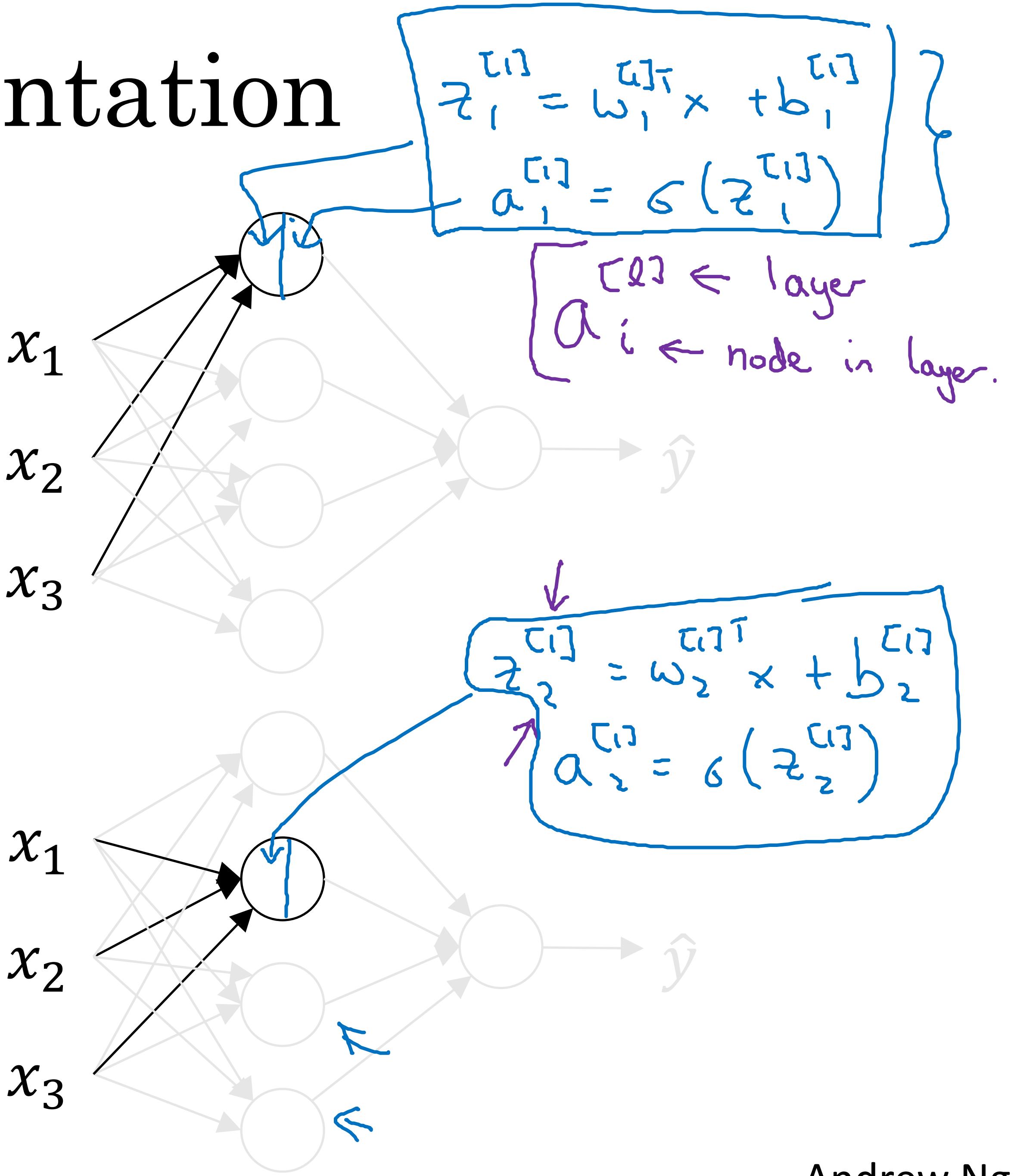
$$a = \sigma(z)$$

Neural Network Representation

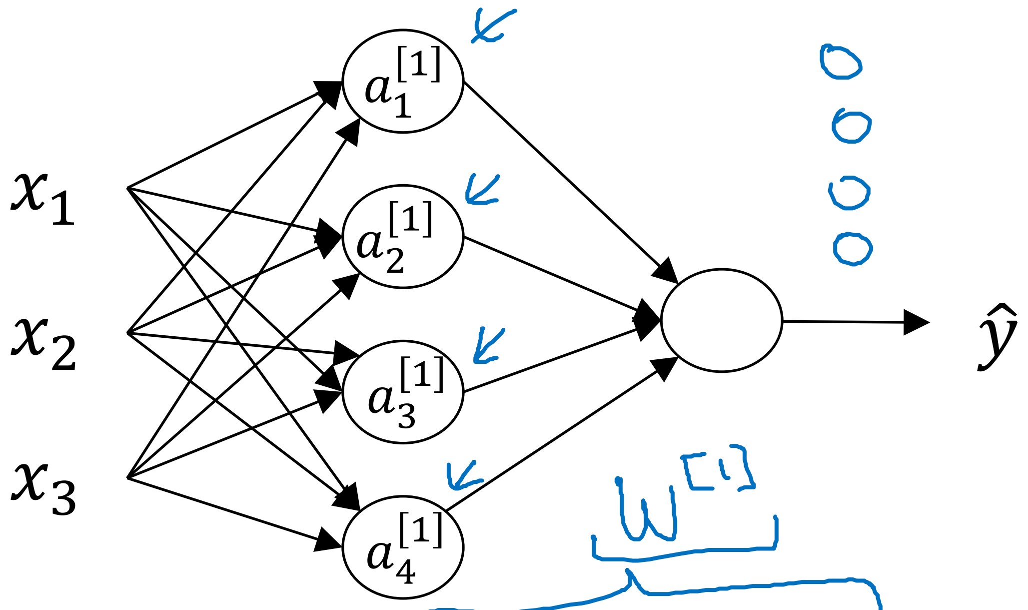


$$z = w^T x + b$$

$$a = \sigma(z)$$



Neural Network Representation



$(w_1^{[1]})^T x$ $\qquad a^{[1]}$

$$z_1^{[1]} = w_1^{[1]T}x + b_1^{[1]}, \quad a_1^{[1]} = \sigma(z_1^{[1]})$$

$$z_2^{[1]} = w_2^{[1]T}x + b_2^{[1]}, \quad a_2^{[1]} = \sigma(z_2^{[1]})$$

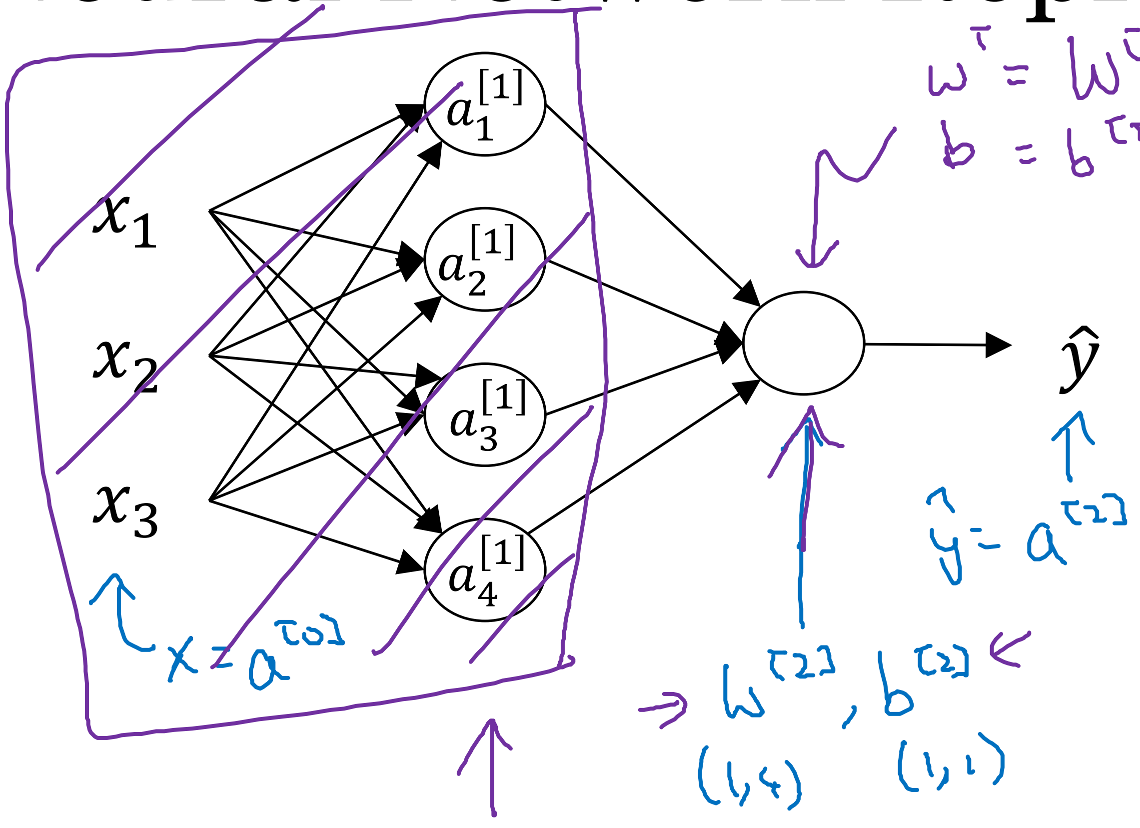
$$z_3^{[1]} = w_3^{[1]T}x + b_3^{[1]}, \quad a_3^{[1]} = \sigma(z_3^{[1]})$$

$$z_4^{[1]} = w_4^{[1]T}x + b_4^{[1]}, \quad a_4^{[1]} = \sigma(z_4^{[1]})$$

$$z^{[1]} = \overbrace{\begin{bmatrix} — w_1^{[1]T} — \\ — w_2^{[1]T} — \\ — w_3^{[1]T} — \\ — w_4^{[1]T} — \end{bmatrix}}^{W^{[1]}}_{(4,3)} \begin{bmatrix} x_1 \\ x_2 \\ x_3 \end{bmatrix} + \underbrace{\begin{bmatrix} b_1^{[1]} \\ b_2^{[1]} \\ b_3^{[1]} \\ b_4^{[1]} \end{bmatrix}}_{b^{[1]}} \ (4,1) = \begin{bmatrix} w_1^{[1]T}x + b_1^{[1]} \\ w_2^{[1]T}x + b_2^{[1]} \\ w_3^{[1]T}x + b_3^{[1]} \\ w_4^{[1]T}x + b_4^{[1]} \end{bmatrix} = \begin{bmatrix} z_1^{[1]} \\ z_2^{[1]} \\ z_3^{[1]} \\ z_4^{[1]} \end{bmatrix}$$

$$a^{[1]} = \begin{bmatrix} a_1^{[1]} \\ \vdots \\ a_4^{[1]} \end{bmatrix} = \sigma(z^{[1]})$$

Neural Network Representation learning



$z = w^T x + b$

$\hat{y} = a = \sigma(z)$

Given input x:

$$z^{[1]} = W^{[1]}\cancel{x}\,a^{[0]} + b^{[1]}$$

(4,1) (4,3) (3,1) (4,1)

$$a^{[1]} = \sigma(z^{[1]})$$

(4,1) (4,1)

$$z^{[2]} = W^{[2]}a^{[1]} + b^{[2]}$$

(1,1) (1,4) (4,1) (1,1)

$$a^{[2]} = \sigma(z^{[2]})$$

(1,1) (1,1)

Обучение сети

- Обучить нейронную сеть это значит, сообщить ей, чего от нее добиваются.
- Показав ребенку изображение буквы и получив неверный ответ, ему сообщается тот, который хотят получить.
- Ребенок запоминает этот пример с верным ответом и в его памяти происходят изменения в нужном направлении.

Обучение перцептрона

Начальные значения весов всех нейронов полагаются случайными.

$$W(t = 0)$$

Сети предъявляется входной образ x^{α}, в результате формируется выходной образ.

$$\tilde{y}^{\alpha} \neq y^{\alpha}$$

Обучение перцептрона

Вычисляется вектор ошибки, делаемой сетью на выходе.

$$\delta^{\alpha} = \left(y^{\alpha} - \boldsymbol{y}^{\alpha}\right)$$

Идея: изменение вектора весовых коэффициентов в области малых ошибок должно быть пропорционально ошибке на выходе.

Обучение перцептрона

Вектор весов модифицируется по следующей формуле:

$$W(t+\Delta t) = W(t) + \eta x^{\alpha} \cdot \left(\delta^{\alpha}\right)^{T}$$

$0 < \eta < 1$ - темп обучения.

Параметры

- Обучение проводится для всех обучающих векторов.
- Один цикл предъявления всей выборки называется **эпохой.**
- Обучение завершается по истечении нескольких эпох, когда вектор весов перестанет значимо меняться.

Возможности применения

Теорема о полноте:

Любая непрерывная функция может быть приближена функциями, вычисляемыми нейронными сетями.

Нейронные сети являются универсальными структурами, позволяющими реализовать любой алгоритм!

Этапы построения сети

- Выбор архитектуры сети
  - Число входов
  - Функции активации
  - Как соединить нейроны
  - Что взять за вход, что за выход
- Подбор весов (обучение сети)
- Построить вручную
- Воспользоваться пакетом нейросетевого моделирования

- Согласно методу наименьших квадратов, минимизируемой целевой функцией ошибки НС является величина:
- $E(w)=\frac{1}{2}\sum_{j,p}(y_{j,p}^{(N)}-d_{j,p})^2$ (1)
- где $y_{j,p}^{(N)}$ – реальное выходное состояние нейрона j выходного слоя N нейронной сети при подаче на ее входы p-го образа; djp – идеальное (желаемое) выходное состояние этого нейрона.
- Суммирование ведется по всем нейронам выходного слоя и по всем обрабатываемым сетью образам. Минимизация ведется методом градиентного спуска, что означает подстройку весовых коэффициентов следующим образом:
- $\Delta w_{ij}^{(n)} = -\eta\cdot\frac{\partial E}{\partial w_{ij}}$ (2)
- Здесь wij – весовой коэффициент синаптической связи, соединяющей i-ый нейрон слоя n-1 с j-ым нейроном слоя n, h – коэффициент скорости обучения, 0<h <1.
- Как показано в [2],
- $\frac{\partial E}{\partial w_{ij}}=\frac{\partial E}{\partial y_j}\cdot\frac{dy_j}{ds_j}\cdot\frac{\partial s_j}{\partial w_{ij}}$ (3)

- Здесь под yj, как и раньше, подразумевается выход нейрона j, а под sj – взвешенная сумма его входных сигналов, то есть аргумент активационной функции. Так как множитель dyj/dsj является производной этой функции по ее аргументу, из этого следует, что производная активационной функция должна быть определена на всей оси абсцисс. В связи с этим функция единичного скачка и прочие активационные функции с неоднородностями не подходят для рассматриваемых НС. В них применяются такие гладкие функции, как гиперболический тангенс или классический сигмоид с экспонентой. В случае гиперболического тангенса
- $$\frac{dy}{ds} = 1 - s^2 \qquad (4)$$
- Третий множитель ¶ sj/¶ wij, очевидно, равен выходу нейрона предыдущего слоя yi(n-1).
- Что касается первого множителя в (3), он легко раскладывается следующим образом[2]:
- $$\frac{\partial E}{\partial y_j} = \sum_k \frac{\partial E}{\partial y_k} \cdot \frac{dy_k}{ds_k} \cdot \frac{\partial s_k}{\partial y_j} = \sum_k \frac{\partial E}{\partial y_k} \cdot \frac{dy_k}{ds_k} w_{jk}^{(n+1)} \qquad (5)$$
- Здесь суммирование по k выполняется среди нейронов слоя n+1.
- Введя новую переменную
- $$\delta_j^{(n)} = \frac{\partial E}{\partial y_j} \cdot \frac{dy_j}{ds_j} \qquad (6)$$

- мы получим рекурсивную формулу для расчетов величин d j(n) слоя n из величин d k(n+1) более старшего слоя n+1.
- $$\delta_j^{(n)} = \left[\sum_k \delta_k^{(n+1)} \cdot w_{jk}^{(n+1)}\right] \cdot \frac{dy_j}{ds_j} \quad (7)$$
- Для выходного же слоя
- $$\delta_l^{(N)} = (y_l^{(N)} - d_l) \cdot \frac{dy_l}{ds_l} \quad (8)$$
- Теперь мы можем записать (2) в раскрытом виде:
- $$\Delta w_{ij}^{(n)} = -\eta \cdot \delta_j^{(n)} \cdot y_i^{(n-1)} \quad (9)$$
- Иногда для придания процессу коррекции весов некоторой инерционности, сглаживающей резкие скачки при перемещении по поверхности целевой функции, (9) дополняется значением изменения веса на предыдущей итерации
- $$\Delta w_{ij}^{(n)}(t) = -\eta \cdot (\mu \cdot \Delta w_{ij}^{(n)}(t-1) + (1-\mu) \cdot \delta_j^{(n)} \cdot y_i^{(n-1)}) \quad (10)$$
- где m – коэффициент инерционности, t – номер текущей итерации.
- Таким образом, полный алгоритм обучения НС с помощью процедуры обратного распространения строится так:

- 1. Подать на входы сети один из возможных образов и в режиме обычного функционирования НС, когда сигналы распространяются от входов к выходам, рассчитать значения последних. Напомним, что
- $$s_j^{(n)} = \sum_{i=0}^{M} y_i^{(n-1)} \cdot w_{ij}^{(n)} \quad (11)$$
- где M – число нейронов в слое n-1 с учетом нейрона с постоянным выходным состоянием +1, задающего смещение; yi(n-1)=xij(n) – i-ый вход нейрона j слоя n.
- yj(n) = f(sj(n)), где f() – сигмоид (12)
- yq(0)=Iq, (13)
- где Iq – q-ая компонента вектора входного образа.
- 2. Рассчитать d (N) для выходного слоя по формуле (8).
- Рассчитать по формуле (9) или (10) изменения весов D w(N) слоя N.
- 3. Рассчитать по формулам (7) и (9) (или (7) и (10)) соответственно d (n) и D w(n) для всех остальных слоев, n=N-1,...1.
- 4. Скорректировать все веса в НС
- $$w_{ij}^{(n)}(t) = w_{ij}^{(n)}(t-1) + \Delta w_{ij}^{(n)}(t) \quad (14)$$
- 5. Если ошибка сети существенна, перейти на шаг 1. В противном случае – конец.

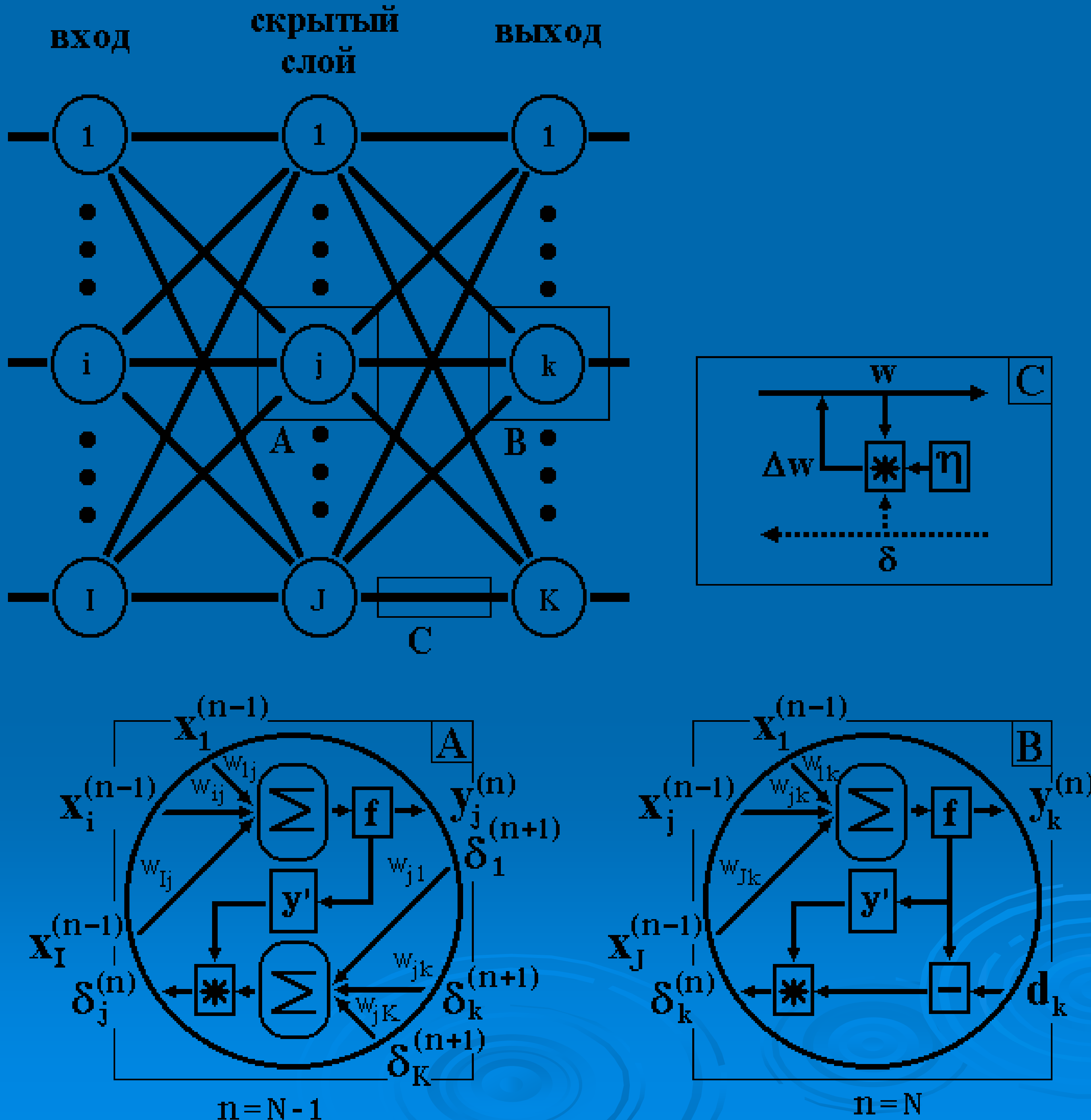


Рис. 2.

- Сети на шаге 1 попеременно в случайном порядке предъявляются все тренировочные образы, чтобы сеть, образно говоря, не забывала одни по мере запоминания других. Алгоритм иллюстрируется Рис. 2.
- Из выражения (9) следует, что когда выходное значение yi(n-1) стремится к нулю, эффективность обучения заметно снижается. При двоичных входных векторах в среднем половина весовых коэффициентов не будет корректироваться[3], поэтому область возможных значений выходов нейронов [0,1] желательно сдвинуть в пределы [-0.5,+0.5], что достигается простыми модификациями логистических функций. Например, сигмоид с экспонентой преобразуется к виду
- $f(x) = -0.5 + \dfrac{1}{1+e^{-\alpha \cdot x}}$ (15)
- Теперь коснемся вопроса емкости НС, то есть числа образов, предъявляемых на ее входы, которые она способна научиться распознавать. Для сетей с числом слоев больше двух, он остается открытым. Как показано в [4], для НС с двумя слоями, то есть выходным и одним скрытым слоем, детерминистская емкость сети Cd оценивается так:
- Nw/Ny<Cd<Nw/Ny? log(Nw/Ny) (16)
- где Nw – число подстраиваемых весов, Ny – число нейронов в выходном слое.

- Следует отметить, что данное выражение получено с учетом некоторых ограничений. Во-первых, число входов Nx и нейронов в скрытом слое Nh должно удовлетворять неравенству Nx+Nh>Ny. Во-вторых, Nw/Ny>1000. Однако вышеприведенная оценка выполнялась для сетей с активационными функциями нейронов в виде порога, а емкость сетей с гладкими активационными функциями, например – (15), обычно больше. Кроме того, фигурирующее в названии емкости прилагательное "детерминистский" означает, что полученная оценка емкости подходит абсолютно для всех возможных входных образов, которые могут быть представлены Nx входами. В действительности распределение входных образов, как правило, обладает некоторой регулярностью, что позволяет НС проводить обобщение и, таким образом, увеличивать реальную емкость. Так как распределение образов, в общем случае, заранее не известно, мы можем говорить о такой емкости только предположительно, но обычно она раза в два превышает емкость детерминистскую.
- В продолжение разговора о емкости НС логично затронуть вопрос о требуемой мощности выходного слоя сети, выполняющего окончательную классификацию образов. Дело в том, что для разделения множества входных образов, например, по двум классам достаточно всего одного выхода. При этом каждый логический уровень – "1" и "0" – будет обозначать отдельный класс. Однако результаты работы сети, организованной таким образом, можно сказать – "под завязку", – не очень надежны. Для повышения достоверности классификации желательно ввести избыточность путем выделения каждому классу одного нейрона в выходном слое или, что еще лучше, нескольких, каждый из которых обучается определять принадлежность образа к классу со своей степенью достоверности, например: высокой, средней и низкой. Такие НС позволяют проводить классификацию входных образов, объединенных в нечеткие множества. Это свойство приближает подобные НС к условиям реальной жизни.

- Рассматриваемая НС имеет несколько "узких мест". Во-первых, в процессе обучения может возникнуть ситуация, когда большие положительные или отрицательные значения весовых коэффициентов сместят рабочую точку на сигмоидах многих нейронов в область насыщения. Малые величины производной от логистической функции приведут в соответствие с (7) и (8) к остановке обучения, что парализует НС. Во-вторых, применение метода градиентного спуска не гарантирует, что будет найден глобальный, а не локальный минимум целевой функции. Эта проблема связана еще с одной, а именно – с выбором величины скорости обучения. Доказательство сходимости обучения в процессе обратного распространения основано на производных, то есть приращения весов и, следовательно, скорость обучения должны быть бесконечно малыми, однако в этом случае обучение будет происходить неприемлемо медленно. С другой стороны, слишком большие коррекции весов могут привести к постоянной неустойчивости процесса обучения. Поэтому в качестве h обычно выбирается число меньше 1, но не очень маленькое, например, 0.1, и оно, вообще говоря, может постепенно уменьшаться в процессе обучения. Кроме того, для исключения случайных попаданий в локальные минимумы иногда, после того как значения весовых коэффициентов застабилизируются, h кратковременно сильно увеличивают, чтобы начать градиентный спуск из новой точки. Если повторение этой процедуры несколько раз приведет алгоритм в одно и то же состояние НС, можно более или менее уверенно сказать, что найден глобальный максимум, а не какой-то другой.
- Существует и иной метод исключения локальных минимумов, а заодно и паралича НС, заключающийся в применении стохастических НС, но о них лучше поговорить отдельно.