1. ЛЕКЦИЯ. Понятие о математическом нейроне

Биологический прототип

Развитие искусственных нейронных сетей вдохновляется биологией. То есть, рассматривая сетевые конфигурации и алгоритмы, исследователи применяют термины, заимствованные из принципов организации мозговой деятельности. Но на этом аналогия заканчивается. Наши знания о работе мозга столь ограничены, что мало бы нашлось точно доказанных закономерностей для тех, кто пожелал бы руководствоваться ими. Поэтому разработчикам сетей приходится выходить за пределы современных биологических знаний в поисках структур, способных выполнять полезные функции. Во многих случаях это приводит к необходимости отказа от биологического правдоподобия, мозг становится просто метафорой, и создаются сети, невозможные в живой материи или требующие неправдоподобно больших допущений об анатомии и функционировании мозга.

Несмотря на то, что связь с биологией слаба и зачастую несущественна, искусственные нейронные сети продолжают сравнивать с мозгом. Их функционирование часто имеет внешнее сходство с человеческим познанием, поэтому трудно избежать этой аналогии. К сожалению, такие сравнения неплодотворны и создают неоправданные ожидания, неизбежно ведущие к разочарованию.

Нервная система человека, построенная из элементов, называемых нейронами, имеет ошеломляющую сложность. Около 10^{11} нейронов участвуют в примерно 10^{15} передающих связях, имеющих длину метр и более. Каждый нейрон обладает многими свойствами, общими с другими органами тела, но ему присущи абсолютно уникальные способности: принимать, обрабатывать и передавать электрохимические сигналы по нервным путям, которые образуют коммуникационную систему мозга.

Нейро́н (от др.-греч. – волокно, нерв) — структурно-функциональная единица нервной системы, представляющая собой электрически возбудимую клетку, которая обрабатывает, хранит и передает информацию посредством электрических и химических сигналов (рис.1).

Аксон – это нейрит (длинный цилиндрический отросток нервной клетки), по которому нервные импульсы идут от тела клетки (сомы) к другим нервным клеткам.

Дендрит – разветвлённый отросток нейрона, который получает информацию через химические (или электрические) синапсы от аксонов других

нейронов и передаёт её через электрический сигнал телу нейрона, из которого вырастает.

Си́напс (греч. — соединение, связь) — место контакта между двумя нейронами.

Рис.1. Биологический нейрон

Дендриты идут от тела нервной клетки к другим нейронам, где они принимают сигналы в точках соединения, называемых синапсами. Принятые синапсом входные сигналы передаются к телу нейрона. Здесь они суммируются, причем одни входы стремятся возбудить нейрон, другие — воспрепятствовать его возбуждению.

Когда суммарное возбуждение в теле нейрона превышает некоторый порог, нейрон возбуждается, посылая по аксону сигнал другим нейронам. У этой основной функциональной схемы много усложнений и исключений, тем не менее, большинство искусственных нейронных сетей моделируют лишь эти простые свойства.

Проведя аналогию с биологическим можно построить искусственный нейрон (рис. 2).

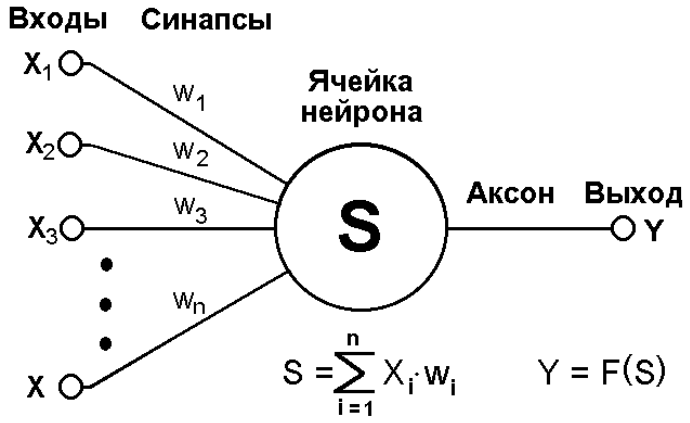

Рис.2. Искусственный (математический) нейрон

Наиболее часто нейронные сети используются для решения следующих задач:

Классификация образов – указание принадлежности входного образа,

представленного вектором признаков, одному или нескольким предварительно определенным классам.

Кластеризация – классификация образов при отсутствии обучающей выборки с метками классов.

Прогнозирование – предсказание значения $y(t_{n+1})$ при известной последовательности $y(t_1), y(t_2), \dots, y(t_n)$.

Оптимизация – нахождение решения, удовлетворяющего системе ограничений и максимизирующим или минимизирующим целевую функцию.

Память, адресуемая по содержанию (ассоциативная память) –память, доступная при указании заданного содержания.

Управление – расчет такого входного воздействия на систему, при котором она следует по желаемой траектории.

Математический нейрон Мак-Каллока – Питтса.

Первой работой, которая заложила теоретический фундамент для создания интеллектуальных устройств, моделирующих человеческий мозг на самом низшем—структурном—уровне, принято считать опубликованную в 1943 г. статью Уоррена Мак-Каллока и Вальтера Питтса «*Идеи логических вычислений в нервной деятельности*». Они предложили математическую модель нейрона мозга человека, назвав ее ***математическим*** или ***модельным нейроном***.

Авторы математического нейрона предложили изображать нейрон в виде кружочка со стрелочками (рис.3).

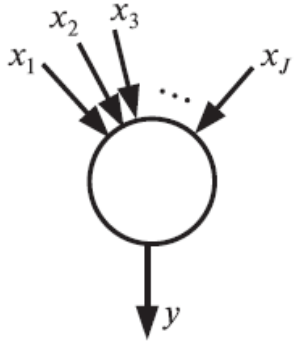

Рис.3. Одно из изображений искусственного нейрона

Стрелки означают входы и выход нейрона. Через входы математический нейрон принимает входные сигналы $\mathbf{x_1, x_2, \dots, x_j, \dots, x_J}$ и суммирует их, умножая каждый входной сигнал на некоторый весовой коэффициент $\mathbf{w_j}$:

$$S = \sum_{j=1}^{J} w_j x_j$$

После выполнения операции суммирования математический нейрон формирует выходной сигнал ***y*** согласно следующему правилу:

$$y = \begin{cases} 1, & если \quad S \geq \theta \\ 0, & если \quad S < \theta \end{cases}$$

где θ – порог чувствительности нейрона.

Таким образом, математический нейрон может существовать в двух состояниях. Если взвешенная сумма входных сигналов ***S*** меньше порога ***θ***, то его выходной сигнал ***y*** равен нулю. В этом случае говорят, что нейрон не возбуждён. Если же входные сигналы достаточно интенсивны и их взвешенная сумма достигает порога чувствительности ***θ***, то нейрон переходит в возбуждённое состояние, и на его выходе, согласно правилу, образуется сигнал ***y=1***.

Иногда встречается следующее схематичное изображение (рис.4), которое более точно отображает процессы происходящие в математическом нейроне.

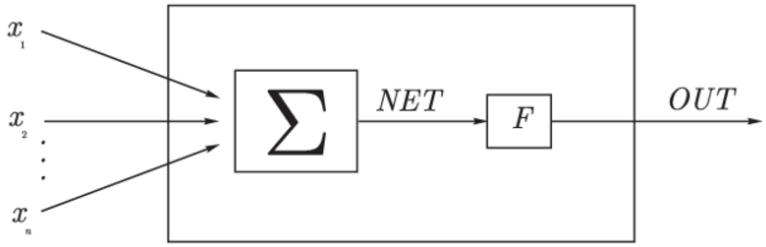

Рис.4. Схематичное изображение

Тогда входные сигналы (**$x_1, x_2, \ldots, x_n$**), можно в совокупности обозначить вектором ***X***, а веса **($w_1, w_2, \ldots, w_n$)** совокупностью **W.** Выход из суммирующего блока обозначается ***NET***.

$$NET = S = XW$$

Сигнал ***NET*** далее преобразуется активационной функцией ***F*** и дает выходной нейронный сигнал ***OUT***. Активационная функция может быть обычной линейной функцией

$$OUT = F(NET)$$

где ***F*** – константа, пороговой функцией

$$OUT = \begin{cases} 1, & \text{если } NET > T; \\ 0, & \text{если } NET \leqslant T \end{cases}$$

где ***T*** – некоторая постоянная пороговая величина, или же функция, более точно моделирующая нелинейную передаточную характеристику биологического нейрона и предоставляющей нейронной сети большие возможности.

1. Активационная (пороговая) функция

Активационная функция нейрона определяет нелинейное преобразование, осуществляемое нейроном и вычисляющая выходной сигнал формального нейрона. Выбор активационной функции определяется

спецификой поставленной задачи либо ограничениями, накладываемыми некоторыми алгоритмами обучения.

Существует множество видов активационных функций, но более всего распространены следующие три:

1. Пороговая функция.

Другое название — ***функция Хевисайда***. Представляет собой перепад. До тех пор пока взвешенный сигнал на входе нейрона не достигает некоторого уровня ***T*** – сигнал на выходе равен нулю. Как только сигнал ***NET*** на входе ***F*** превышает указанный уровень выходной сигнал ***OUT*** скачкообразно изменяется на единицу (рис.5).

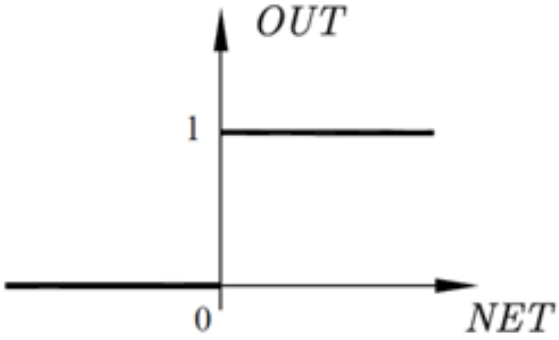

Рис.5. Пороговая функция

Математическую запись функции я вам уже приводил:

$$OUT = \begin{cases} 1, \text{если } NET \geq T \\ 0, \text{если } NET < \text{T} \end{cases}$$

Таким образом, математический нейрон представляет собой пороговый элемент с несколькими входами и одним выходом. Каждый математический нейрон имеет свое определенное значение порога чувствительности ***T***.

Пример расчета с порогом.

Авторы математического нейрона в своей статье также показали, что с помощью математического нейрона с помощью пороговой функции можно моделировать различные логические функции, например функцию логического умножения ***«И» («AND» конъюнкция),*** функцию логического сложения ***«ИЛИ» («OR»)*** и функцию логического отрицания ***«НЕТ» («NOT»)***.

Таблицы истинности логических функций.

И (∧)		
x_1	x_2	$x_1 \wedge x_2$
0	**0**	**0**
0	**1**	**0**
1	**0**	**0**
1	**1**	**1**

ИЛИ (∨)		
x_1	x_2	$x_1 \vee x_2$
0	**0**	**0**
0	**1**	**1**
1	**0**	**1**
1	**1**	**1**

НЕТ(¬)	
x	$\neg x$
0	**1**
1	**0**

Представим математические нейроны (рис.6)

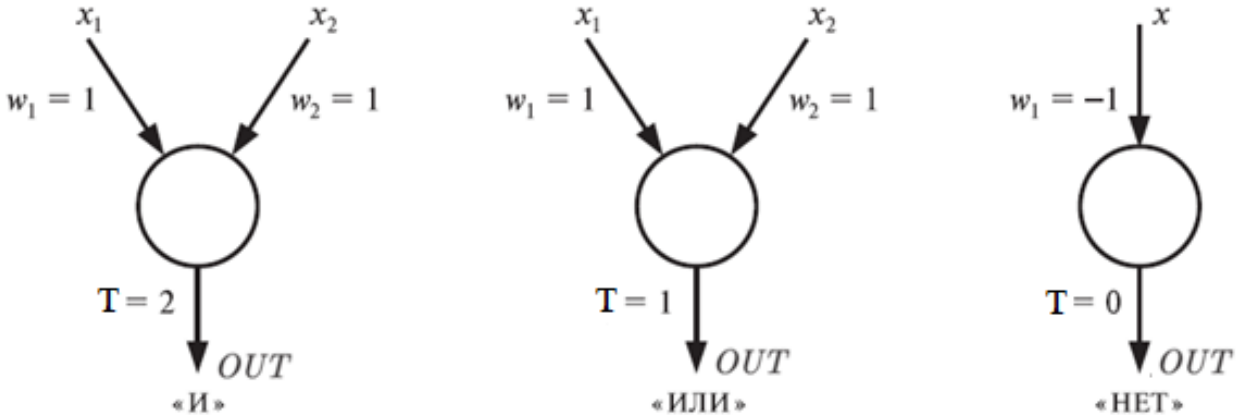

Рис.6. Математические нейроны для «И», «ИЛИ» и «НЕТ»

Рассчитываем ***логическое «И»***. Сигнал ***NET*** может быть :

При $x_1 = 0, x_2 = 0 \rightarrow x_1 * w_1 + x_2 * w_2 = 0*1 + 0*1 = NET_1 = 0$

При $x_1 = 0, x_2 = 1 \rightarrow x_1 * w_1 + x_2 * w_2 = 0*1 + 1*1 = NET_2 = 1$

При $x_1 = 1, x_2 = 0 \rightarrow x_1 * w_1 + x_2 * w_2 = 1*1 + 0*1 = NET_3 = 1$

При $x_1 = 1, x_2 = 1 \rightarrow x_1 * w_1 + x_2 * w_2 = 1*1 + 1*1 = NET_4 = 2$

Устанавливаем пороговую функцию

$$OUT = \begin{cases} 1, \text{если } NET \geq 2 \\ 0, \text{если } NET < 2 \end{cases}$$

Тогда на выходе: $OUT_1 = 0, OUT_2 = 0, OUT_3 = 0, OUT_4 = 1$

Рассчитываем ***логическое «ИЛИ».*** Сигналы ***NET*** может такие же как у *логического «И»,* но пороговая иная.

$$OUT = \begin{cases} 1, \text{если } NET \geq 1 \\ 0, \text{если } NET < 1 \end{cases}$$

Тогда на выходе $OUT_1 = 0, OUT_2 = 1, OUT_3 = 1, OUT_4 = 1$

Рассчитываем ***логическое «НЕТ».*** Тогда сигнал ***NET*** может быть :

При $x = 0 \rightarrow x * w = 0*(-1) = NET_1 = 0$

При $x = 1 \rightarrow x * w = 1*(-1) = NET_2 = -1$

Устанавливаем пороговую функцию для логического отрицания.

$$OUT = \begin{cases} 1, \text{если } NET \geq 0 \\ 0, \text{если } NET < 0 \end{cases}$$

Тогда на выходе: $OUT_1 = 1, OUT_2 = 0$

Таким образом, если рассматривать пороговую функцию, то в ней происходит смещение на величину ***T*** (рис.7).

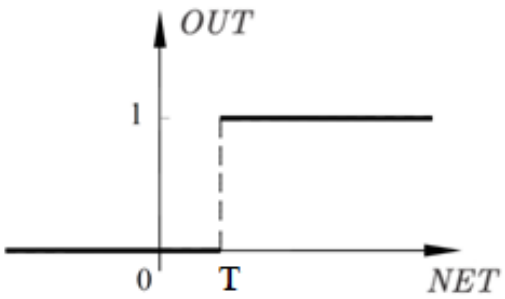

Рис.7. Пороговая функция со смещением

Пороговая функция может иметь и другую интерпретацию (рис.8).

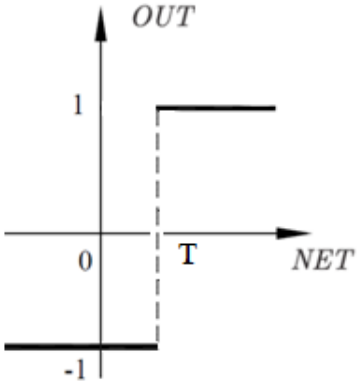

Рис.8. Пороговая функция симметричный вид

Математическую запись функции будет выглядеть:

$$OUT = \begin{cases} 1, \text{если } NET \geq T \\ -1, \text{если } NET < T \end{cases}$$

Поэтому для упрощения расчета в современном исполнении нейрона вводят понятия смещения ***b,*** которое отличается от ***T*** только знаком: ***b*** = - ***T***. При этом нейронное смещение ***b*** можно рассматривать как вес w_0 некоторого дополнительного входного сигнала $x_0 = 1$, величина которого всегда равна единице.

$$\mathbf{NET} = \mathbf{S} = \sum_{j=1}^{J} w_j x_j + w_0 x_0$$

Таким образом, дополнительный вход x_0 и соответствующий ему вес w_0 используются для ***инициализации нейрона***. Под инициализацией подразумевается смещение активационной функции нейрона по горизонтальной оси, то есть формирование порога чувствительности нейрона.

Тогда пороговая активационная функция нейрона примет вид (рис.9):

$$OUT = \begin{cases} 1, \text{если } NET \geq 0 \\ 0, \text{если } NET < 0 \end{cases}$$

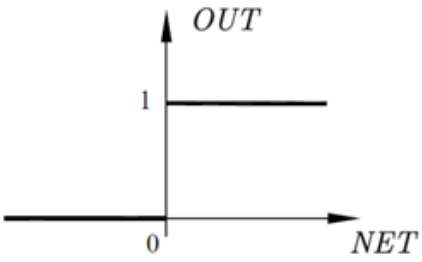

Рис.9. Пороговая функция для смещения

Или более симметричный вид (рис.10)

$$OUT = \begin{cases} 1, \text{если } NET \geq 0 \\ -1, \text{если } NET < 0 \end{cases}$$

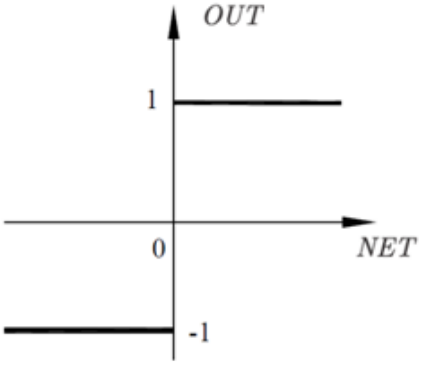

Рис. 10. Симметричный вид для смещения

Пример расчета со смещением.

Рассчитываем ***логическое «И»*** (рис.11)

Установим $x_0 = 1$, а $w_0 = -3/2$

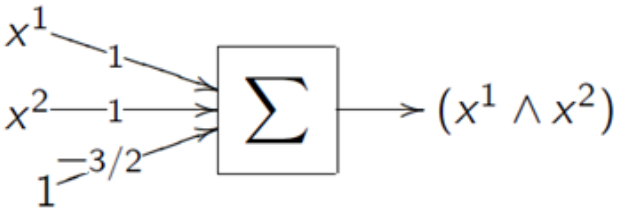

Рис. 11. Математический нейрон для логического «И»

При $x_1=0$, $x_2=0 \rightarrow x_1*w_1 + x_2*w_2 + x_o* w_0 = 0*1 + 0*1 - 1*3/2 = NET_1 = -1,5$

При $x_1=0$, $x_2=1 \rightarrow x_1*w_1 + x_2*w_2 + x_o* w_0 = 0*1 + 1*1 - 1*3/2 = NET_2 = -0,5$

При $x_1=1$, $x_2=0 \rightarrow x_1*w_1 + x_2*w_2 + x_o* w_0 = 1*1 + 0*1 - 1*3/2 = NET_3 = -0,5$

При $x_1=1$, $x_2=1 \rightarrow x_1*w_1 + x_2*w_2 + x_o* w_0 = 1*1 + 1*1 - 1*3/2 = NET_4 = 0,5$

Знаем, что пороговая функция для смещения равна

$$OUT = \begin{cases} 1, \text{если } NET > 0 \\ 0, \text{если } NET \leq 0 \end{cases}$$

Тогда на выходе: $OUT_1 = 0, OUT_2 = 0, OUT_3 = 0, OUT_4 = 1$

Пороговая функция делит входное векторное пространство на две части гиперплоскостью $(x_1 + x_2 = 1,5)$, классифицируя входные векторы как

относящиеся к первому классу, если выходной сигнал ***OUT > 0*** или ко второму классу, если выходной сигнал ***OUT ≤ 0*** (рис. 12)**.**

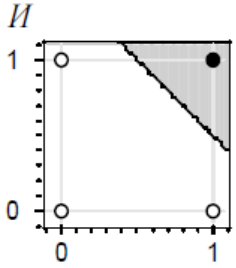

Рис. 12. Гиперплоскость для логического «И»

Рассчитываем ***логическое «ИЛИ»*** (рис.13)

Установим $x_0 = 1$, а $w_0 = -1/2$

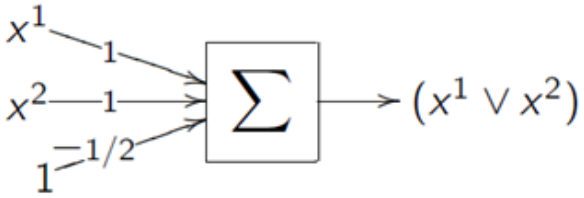

Рис. 13. Математический нейрон для логического «ИЛИ»

При $x_1=0, x_2=0 \rightarrow x_1*w_1 + x_2*w_2 + x_o* w_0 = 0*1 + 0*1 - 1*1/2 = NET_1 = -0,5$

При $x_1=0, x_2=1 \rightarrow x_1*w_1 + x_2*w_2 + x_o* w_0 = 0*1 + 1*1 - 1*1/2 = NET_2 = 0,5$

При $x_1=1, x_2=0 \rightarrow x_1*w_1 + x_2*w_2 + x_o* w_0 = 1*1 + 0*1 - 1*1/2 = NET_3 = 0,5$

При $x_1=1, x_2=1 \rightarrow x_1*w_1 + x_2*w_2 + x_o* w_0 = 1*1 + 1*1 - 1*1/2 = NET_4 = 1,5$

Тогда согласно активационной функции на выходе получаем $OUT_1 = 0$, $OUT_2 = 1$, $OUT_3 = 1$, $OUT_4 = 1$

Графическая интерпретация будет следующей при гиперплоскости $(x_1 + x_2 = 0,5)$ (рис.14)

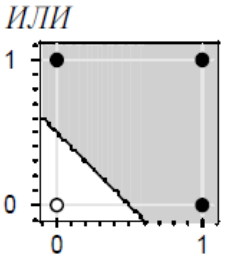

Рис. 14. Гиперплоскость для логического «И»

Аналогично рассчитываем логическое отрицание при $x_0 = 1$, а $w_0 = 1/2$ (рис.15)

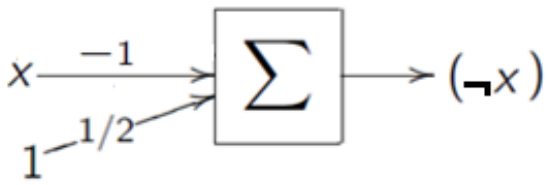

Рис. 15. Математический нейрон для логического «НЕТ»

При $x = 0 \rightarrow x^* w + x_0^* w_0 = 0^*(-1) + 1^* 1/2 = NET_1 = 1/2$

При $x = 1 \rightarrow x^* w + x_0^* w_0 = 1^*(-1) + 1^* 1/2 = NET_2 = -1/2$

Тогда на выходе: $\boldsymbol{OUT_1 = 1, OUT_2 = 0}$

Графическая интерпретация будет следующей при гиперплоскости ***(x = 0,5)*** (рис.16)

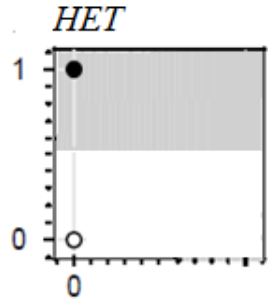

Рис. 16. Гиперплоскость для логического «НЕТ»

Сигмоидальная функция или сигмоид

Если блок ***F*** сужает диапазон изменения величины ***NET*** так, что при любых значениях ***NET*** значения ***OUT*** принадлежат некоторому конечному интервалу, то ***F*** называется «сжимающей» функцией. В качестве «сжимающей» функции часто используется ***"сигмоидальная"*** (S-образная) функция, показанная. Часто под сигмоидой понимают логистическую функцию (рис.17).

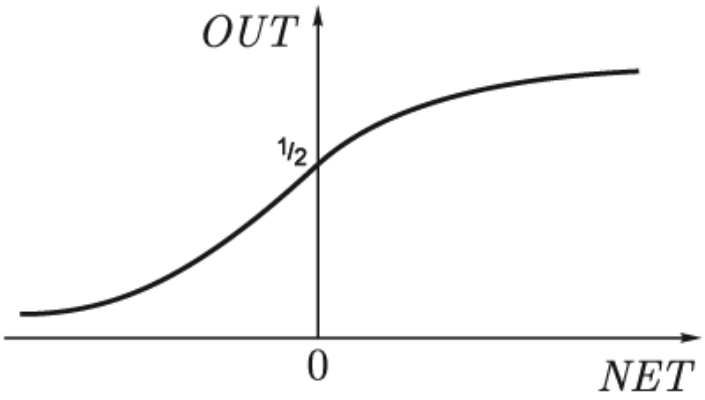

Рис. 17. Сигмоид

Эта функция математически выражается как

$$F(x) = {}^{1}/_{(1+e^{-x})}$$

Таким образом

$$OUT = {}^{1}/_{(1+e^{-NET})}$$

Одна из причин, по которой сигмоид используется в нейронных сетях, это простое выражение его производной через саму функцию (которое и позволило существенно сократить вычислительную сложность *метода обратного распространения ошибки, сделав его применимым на практике*).(рис. 18)

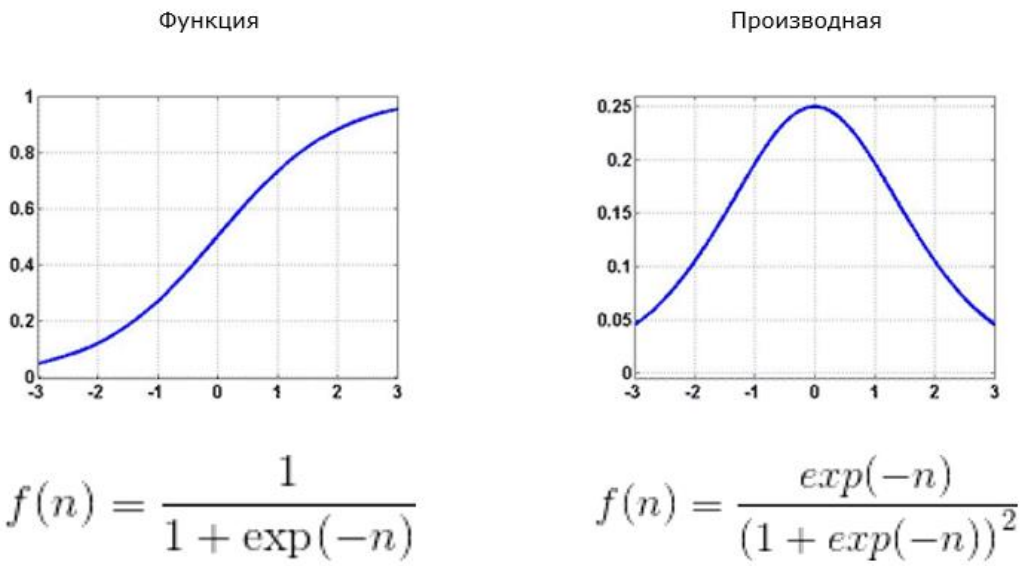

Рис. 18. Функция и производная

где exp = e^n (экспонента)

Тогда

$$\frac{\partial OUT}{\partial NET} = OUT(1 - OUT)$$

Его дополнительное преимущество состоит в автоматическом контроле усиления. Для слабых сигналов (т.е. когда ***OUT*** близко к нулю) кривая вход-выход имеет сильный наклон, дающий большое усиление. Когда величина сигнала становится больше, усиление падает. Таким образом, большие сигналы воспринимаются сетью без насыщения, а слабые сигналы проходят по сети без чрезмерного ослабления. Другими словами центральная область логистической функции, имеющая большой коэффициент усиления, решает проблему обработки слабых сигналов, в то время как области с падающим усилением на положительном и отрицательном концах подходят для больших возбуждений.

Гиперболический тангенс

Другой широко используемой активационной функцией является *гиперболический тангенс* (рис. 19). По форме она сходна с логистической функцией и часто используется биологами в качестве математической модели активации нервной клетки. В качестве активационной функции искусственной нейронной сети она записывается следующим образом:

$$OUT = th(x).$$

Подобно логистической функции гиперболический тангенс является ***S-образной функцией***, но он симметричен относительно начала координат, и в точке ***NET = 0*** значение выходного сигнала ***OUT*** равно нулю. В отличие от

логистической функции, гиперболический тангенс принимает значения различных знаков, и это его свойство применяется для целого ряда сетей.

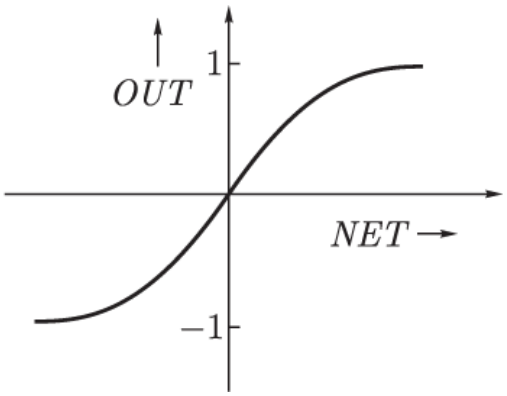

Рис. 19. Функция и производная

Линейная функция активации

Линейная функция представляет собой прямую линию и пропорциональна входу (то есть взвешенной сумме на этом нейроне) (рис. 20).

$$A = cx$$

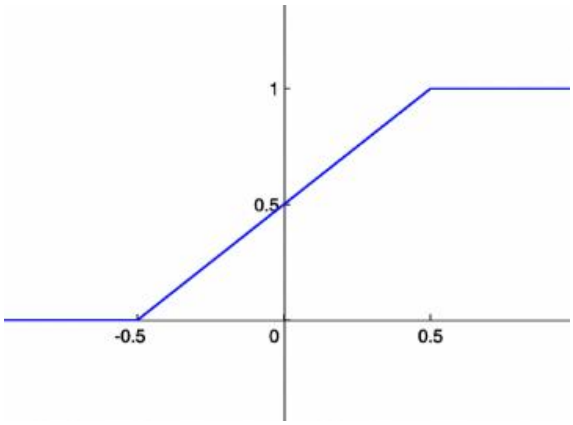

Рис. 20. Линейная функция

Такой выбор активационной функции позволяет получать спектр значений, а не только бинарный ответ. Можно соединить несколько нейронов вместе и, если более одного нейрона активировано, решение принимается на основе применения операции max (или softmax).

Однако (вы знакомы с методом градиентного спуска), и можете заметить, что для этой функции производная равна постоянной.

Производная от $A = cx$ по x равна c. Это означает, что градиент никак не связан с X. Градиент является постоянным вектором, а спуск производится по постоянному градиенту. Если производится ошибочное предсказание, то изменения, сделанные обратным распространением ошибки, тоже постоянны и не зависят от изменения на входе.

При этом каждый слой активируется линейной функцией. Значение с этой функции идет в следующий слой в качестве входа, второй слой считает взвешенную сумму на своих входах и, в свою очередь, включает нейроны в зависимости от другой линейной активационной функции.

Не имеет значения, сколько слоев мы имеем. Если все они по своей природе линейные, то финальная функция активации в последнем слое будет просто линейной функцией от входов на первом слое. Это означает, что два слоя (или N слоев) могут быть заменены одним слоем. Мы потеряли возможность делать наборы из слоев.

Полулинейный элемент

Активационная функция ReLu (англ. Rectified linear unit - Полулинейный элемент или линейный выпрямитель) имеет вид (рис.21)

$$A(x) = max(0, x)$$

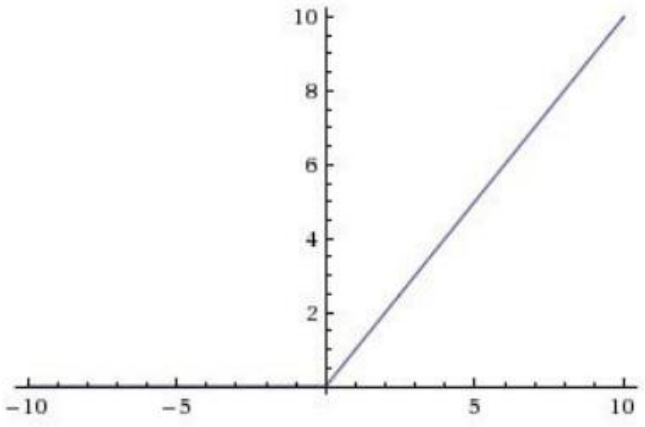

Рис. 21. Полулинейный элемент

ReLu возвращает значение x, если x положительно, и 0 в противном случае.На первый взгляд кажется, что ReLu имеет все те же проблемы, что и линейная функция, так как ReLu линейна в первом квадранте. Но на самом деле, ReLu нелинейна по своей природе. (На самом деле, такая функция является хорошим аппроксиматором, так как любая функция может быть аппроксимирована комбинацией ReLu). Область допустимых значений ReLu — [0,inf), то есть активация может “взорваться”.

Разреженность активации. Представим большую нейронную сеть с множеством нейронов. Использование сигмоиды или гиперболического тангенса будет влечь за собой активацию всех нейронов аналоговым способом. Это означает, что почти все активации должны быть обработаны для описания выхода сети. Другими словами, активация плотная, а это затратно. В идеале мы хотим, чтобы некоторые нейроны не были активированы, это сделало бы активации разреженными и эффективными.

ReLu позволяет это сделать. Представим сеть со случайно инициализированными весами (или нормализированными), в которой примерно 50% активаций равны 0 из-за характеристик ReLu (возвращает 0 для отрицательных значений x). В такой сети включается меньшее количество нейронов (разреженная активация), а сама сеть становится легче. Отлично, кажется, что ReLu подходит нам по всем параметрам. Но ничто не безупречно, в том числе и ReLu.

Из-за того, что часть ReLu представляет из себя горизонтальную линию (для отрицательных значений X), градиент на этой части равен 0. Из-за равенства нулю градиента, веса не будут корректироваться во время спуска. Это означает, что пребывающие в таком состоянии нейроны не будут реагировать на изменения в ошибке/входных данных (просто потому, что градиент равен нулю, ничего не будет меняться). Такое явление называется проблемой умирающего ReLu. Из-за этой проблемы некоторые нейроны просто выключатся и не будут отвечать, делая значительную часть нейросети пассивной. Однако существуют вариации ReLu, которые помогают эту проблему избежать.

ReLu менее требовательно к вычислительным ресурсам, чем гиперболический тангенс или сигмоида, так как производит более простые математические операции. Поэтому имеет смысл использовать ReLu при создании глубоких нейронных сетей.

Однослойный перцептрон

По архитектуре связей ИНС могут быть сгруппированы в два класса: сети прямого распространения, в которых графы не имеют петель, и рекуррентные сети, или сети с обратными связями (рис. 1).

Рис.1. Классификация нейронных сетей.

Если ИНС не имеет обратных связей, то она называется ***статической***, а если обратные связи существуют, то сеть ***динамическая*** (рекуррентная)

Перцептрон, или персептрон *(англ. perceptron от лат. perceptio —* ***восприятие****; нем. Perzeptron) — математическая или компьютерная модель восприятия информации мозгом человека*

Сеть РБФ (радиально-базисных функций) – искусственная нейронная сеть, которая использует радиальные базисные функции как функции активации.

АРТ (Адаптивная резонансная теория, сети адаптивного резонанса) – разновидность искусственных нейронных сетей основанная на теории адаптивного резонанса.

Рассмотрим наиболее простые модели нейронных сетей: *однослойный и многослойный персептрон****.***

Однослойные искусственные нейронные сети

Хотя один нейрон и способен выполнять простейшие процедуры распознавания, но для серьезных нейронных вычислений необходимо соединять нейроны в сети. Простейшая сеть состоит из группы нейронов, образующих слой, как показано в правой части рисунка. Отметим, что круги слева *X* служат лишь для распределения входных сигналов. Они не выполняют каких-либо вычислений и поэтому не будут считаться слоем (рис. 2).

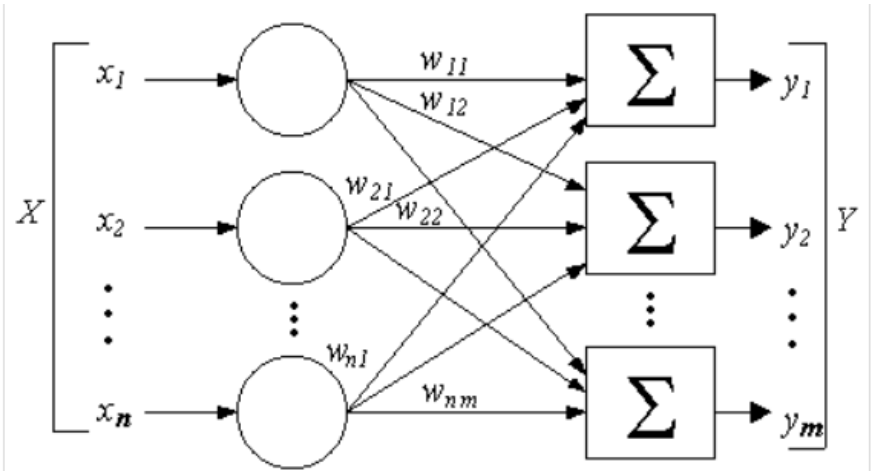

Рис.2. Однослойный перцептрон

Для большей наглядности обозначим их кругами, чтобы отличать их от вычисляющих нейронов, обозначенных квадратами. Каждый элемент из множества входов ***X*** отдельным весом соединен с каждым искусственным нейроном. А каждый нейрон выдает взвешенную сумму входов в сеть. В искусственных сетях многие соединения могут отсутствовать, но здесь они показаны все для демонстрации общей картины. Могут существовать также соединения между выходами и входами элементов в слое.

Удобно считать веса элементами матрицы ***W***. Матрица имеет ***m*** строк и ***n***

столбцов, где m – число входов, а n – число нейронов.

$$W = \begin{pmatrix} w_{11} & w_{12} & - & w_{1m} \\ w_{21} & w_{22} & - & w_{2m} \\ - & - & - & - \\ w_{n1} & w_{n2} & - & w_{nm} \end{pmatrix}$$

Например, w_{23} – это вес, связывающий второй вход с третьим нейроном. Таким образом, вычисление выходного вектора N, компонентами которого являются выходы OUT нейронов, сводится к матричному умножению $N = XW$, где N и X – векторы-строки.

Алгоритм обучения однослойного персептрона

Способность искусственных нейронных сетей к обучению является их наиболее интригующим свойством. Подобно биологическим системам, которые они моделируют, эти нейронные сети сами совершенствуют себя в результате попыток создать лучшую модель поведения.

Используя критерий линейной разделимости, можно решить, способна ли однослойная нейронная сеть реализовывать требуемую функцию. Даже в том случае, когда ответ положительный, это принесет мало пользы, если у нас нет способа найти нужные значения для весов и порогов. Чтобы сеть представляла практическую ценность, нужен систематический метод (алгоритм) для вычисления этих значений. Ф.Розенблатт создал такой метод в своем алгоритме обучения персептрона и доказал: персептрон может быть обучен всему, что он может реализовывать.

Обучение – это процесс, в котором свободные параметры нейронной сети настраиваются посредством моделирования среды, в которую эта сеть встроена. Тип обучения определяется способом подстройки этих параметров.

Это определение процесса обучения нейронной сети предполагает следующую последовательность событий:

- В нейронную сеть поступают стимулы из внешней среды.
- В результате первого пункта изменяются свободные параметры нейронной сети.
- После изменения внутренней структуры нейронная сеть отвечает на возбуждения уже иным образом.

Вышеуказанный список четких правил решения проблемы обучения нейронной сети называется алгоритмом обучения. Несложно догадаться, что не существует универсального алгоритма обучения, подходящего для всех архитектур нейронных сетей. Существует лишь набор средств, представленный множеством алгоритмов обучения, каждый из которых имеет свои достоинства. Алгоритмы обучения отличаются друг от друга способом настройки

синаптических весов нейронов. Еще одной отличительной характеристикой является способ связи обучаемой нейронной сети с внешним миром. В этом контексте говорят о парадигме обучения, связанной с моделью окружающей среды, в которой функционирует данная нейронная сеть.

Существуют два концептуальных подхода к обучению нейронных сетей: ***обучение без учителя***, ***обучение с учителем*** и ***обучение с подкреплением.***

Обучение нейронной сети с учителем *(supervised learning)* предполагает, что для каждого входного вектора из обучающего множества существует требуемое значение выходного вектора, называемого целевым. Эти вектора образуют обучающую пару. Веса сети изменяют до тех пор, пока для каждого входного вектора не будет получен приемлемый уровень отклонения выходного вектора от целевого.

Обучение нейронной сети без учителя *(unsupervised learning)* является намного более правдоподобной моделью обучения с точки зрения биологических корней искусственных нейронных сетей. Обучающее множество состоит лишь из входных векторов. Алгоритм обучения нейронной сети подстраивает веса сети так, чтобы получались согласованные выходные векторы, т.е. чтобы предъявление достаточно близких входных векторов давало одинаковые выходы.

Обучение с частичным привлечением учителя или полуавтоматическое обучение, или частичное обучение (англ. Semi-supervised learning) – способ машинного обучения, разновидность обучения с учителем, которое также использует неразмеченные данные для тренировки – обычно небольшое количество размеченных данных и большое количество неразмеченных данных. Полуавтоматическое обучение занимает промежуточную позицию между обучением без учителя (без привлечения каких-либо размеченных данных для тренировки) и обучением с учителем (с привлечением лишь размеченных данных). Многие исследователи машинного обучения обнаружили, что неразмеченные данные, при использовании в сочетании с небольшим количеством размеченных данных, могут значительно улучшить точность обучения.

Обучение с подкреплением представляет собой метод машинного обучения на основе обучения интеллектуального агента, который действует во внешней среде. Предполагается, что в каждый дискретный момент времени программируемый агент находится в определенном состоянии s. В зависимости от этого состояния у агента есть на выбор несколько возможных действий. Агент выбирает некоторое действие, после чего оказывается в новом состоянии и

получает определенное подкрепление, которое зависит от предыдущего состояния и выбранного действия. Предполагается, что агенту нужно увеличивать сумму своих подкреплений. В обучении с подкреплением ключевым моментом является функции $Q(s, a)$, которая является субъективной оценкой действия а в состоянии s. На основании этой функции агент принимает решение. В тоже время в результате игры агент постоянно модифицирует эту функции, что и является обучением.

Правила обучения Хебба

В 1949 году канадский физиолог и психолог Хебб высказал идеи о характере соединения нейронов мозга и их взаимодействии. Он первым предположил, что обучение заключается в первую очередь в изменениях силы синаптических связей. Теория Хебба считается типичным случаем самообучения, при котором испытуемая система спонтанно обучается выполнять поставленную задачу без вмешательства со стороны экспериментатора.

Постулат обучения Хебба является самым старым и самым известным среди всех правил обучения. При ведем цитату из книги Хебба:

Если аксон клетки A находится на достаточно близком расстоянии от клетки B и постоянно и периодически участвует в ее возбуждении, то наблюдается процесс метаболических изменений в одном или обоих нейронах, выражающийся в том, что эффективность нейрона A как одного из возбудителей нейрона B возрастает.

Хебб предложил положить это наблюдение в основу процесса ассоциативного обучения (на клеточном уровне). По его мнению, это должно было привести к постоянной модификации шаблона активности, пространственно-распределенного «ансамбля нервных клеток».

Это утверждение было сделано в нейробиологическом контексте, но его можно перефразировать в следующее правило, состоящее из двух частей:

1) если два нейрона по обе стороны синапса активизируются одновременно, т.е. синхронно, то прочность соединения возрастает;

2) если два нейрона по обе стороны синапса активизируются асинхронно, то такой синапс ослабляется, или вообще отмирает.

Функционирующий таким образом синапс называется синапсом Хебба (обратите внимание, что исходное правило Хебба не содержало второй части последнего утверждения). Если быть более точным, то синапс Хебба использует зависящий от времени, в высшей степени локальный механизм взаимодействия, изменяющий эффективность синаптического соединения в зависимости от

корреляции *(статическая зависимость двух и более величин)* между предсинаптической и постсинаптической активностью. Из этого определения можно вывести следующие четыре свойства (ключевых механизма), характеризующие синапс Хебба.

1. *Зависимость от времени.* Синапс Хебба зависит от точного времени возникновения предсинаптического и постсинаптического сигналов.

2. *Локальность.* По своей природе синапс является узлом передачи данных, в котором информационные сигналы (представляющие текущую активность предсинаптических и постсинаптических элементов) находятся в пространственно-временной близости. Эта локальная информация используется синапсом Хебба для выполнения локальных синаптических модификаций, характерных для данного входного сигнала.

3. *Интерактивность*. Изменения в синапсе Хебба определяются сигналами на обоих его концах. Это значит, что форма обучения Хебба зависит от степени взаимодействия предсинаптического и постсинаптического сигналов (предсказание нельзя построить на основе только одного из этих сигналов). Заметим, что такая зависимость или интерактивность может носить детерминированный или статистический характер.

4. *Корреляция.* Одна из интерпретаций постулата обучения Хебба состоит в том, что условием изменения эффективности синаптической связи является зависимость между предсинаптическим и постсинаптическим сигналами. В соответствии с этой интерпретацией для модификации синапса необходимо обеспечить одновременность предсинаптического и постсинаптического сигналов.

Другая интерпретация постулата обучения Хебба использует характерные для синапса Хебба механизмы взаимодействия в статистических терминах. В частности, синаптические изменения определяются корреляцией предсинаптического и постсинаптического сигналов во времени.

Первое правило Хебба – Если сигнал персептрона неверен и равен нулю, то необходимо увеличить веса тех входов, на которые была подана единица.
Вытекает ***первый тип ошибки***: на выходе персептрона ***0***, правильный ответ ***1***.
Для того, чтобы персептрон выдавал правильный ответ необходимо, чтобы скалярное произведение стало больше. Поскольку переменные принимают значения 0 или 1, увеличение суммы может быть достигнуто за счет увеличения весов. Однако нет смысла увеличивать веса при переменных, которые равны нулю. Увеличиваем веса только при тех, которые равны 1. Для закрепления единичных сигналов с весов, следует провести ту же процедуру и на всех

остальных слоях.

Второе правило Хебба — Если сигнал персептрона неверен и равен единице, то необходимо уменьшить веса тех входов, на которые была подана единица.

Отсюда вытекает второй тип ошибки: на выходе персептрона ***1***, правильный ответ ***0***. Для уменьшения скалярного произведения в правой части, необходимо уменьшить веса связей при тех переменных, которые равны 1. Необходимо также провести эту процедуру для всех активных нейронов предыдущих слоев

Систематизируя определения, можно выделить следующие модели модификации синаптических связей: *хеббовскую*, антихеббовскую и нехеббовскую. Следуя этой схеме, можно сказать, что синаптическая связь по модели Хебба усиливается при положительной корреляции предсинаптических и постсинаптических сигналов и ослабляется в противном случае. И, наоборот, согласно антихеббовской модели, синаптическая связь ослабляется при положительной корреляции предсинаптическоro и постсинаптическоro сигналов и усиливается в противном случае. Однако в обеих моделях изменение синаптической эффективности основано на зависящем от времени в высшей степени локальном и интерактивном по своей природе механизме. В этом контексте антихеббовская модель все равно остается хеббовской по природе, но не по функциональности. Нехеббовская модель вообще не связана с механизмом модификации, предложенным Хеббом.

И так по существу, Хэбб предположил, что синаптическое соединение двух нейронов усиливается, если оба эти нейрона возбуждены. Это можно представить как усиление синапса в соответствии с корреляцией уровней возбужденных нейронов, соединяемых данным синапсом. Поэтому алгоритм обучения Хэбба иногда называется корреляционным алгоритмом.

Пусть имеются два нейрона ***i*** и ***j***, между которыми существует сила связи w_{ij} (рис. 3)

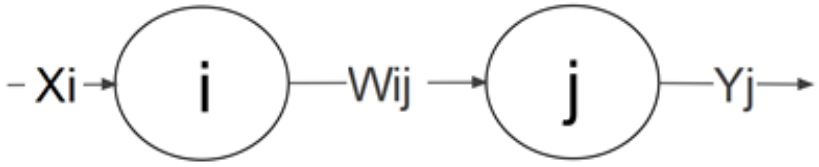

Рис.3. Синаптическая связь между нейронами

Идея алгоритма выражается следующим равенством:

$$w_{ij}(t+1) = w_{ij}(t) + \eta NET_i NET_j \qquad (1)$$

где $w_{ij}(t)$ – сила синапса от нейрона ***i*** к нейрону ***j*** в момент времени ***t***; ***NETi*** – уровень возбуждения пресинаптического нейрона; ***NETj*** – уровень возбуждения постсинаптического нейрона; η – положительная константа, определяющая скорость обучения.

Выражение (1) ясно подчеркивает корреляционную природу синапса Хебба. Ее иногда называют правилом умножения активности.

Правила Хебба могут применяться при обучении «с учителем» и «без учителя». При обучении с учителем выходные сигналы сравниваются с эталонными значениями. При использовании алгоритма без учителя: реальные сигналы с выхода нейронной сети используются для кластеризации входных образов.

Алгоритм обучения по правилу Хебба

Алгоритм обучения по правилу Хебба сводится к следующей последовательности действий:

1. Инициализация весовых коэффициентов и порогов случайными значениями, близкими к нулю (чтобы сеть сразу не могла войти в насыщение).
2. Подача на вход НС очередного входного образа.
3. Вычисление значения выхода.
4. Если значение выхода не совпадает с эталонным значением, то происходит модификация коэффициентов в соответствии с формулами при скорости обучения η=1.

$$w_{11}(t+1) = w_{11}(t) + x_1 y_1$$

$$w_{21}(t+1) = w_{21}(t) + x_2 y_1$$

$$T(t+1) = T(t) - y_1$$

В противном случае осуществляется переход к пункту 5.

5. Если не все вектора из обучающей выборки были поданы на вход НС, то происходит переход к пункту 2. Иначе, переход к пункту 6.
6. Останов.

Примеры. Требуется реализовать обучение нейронной сети по правилу Хебба с учителем для операции «***логическое И***», используя биполярную кодировку двоичных сигналов (1, -1). Скорость обучения η=1.

Представим простейший нейрон (рис. 4).

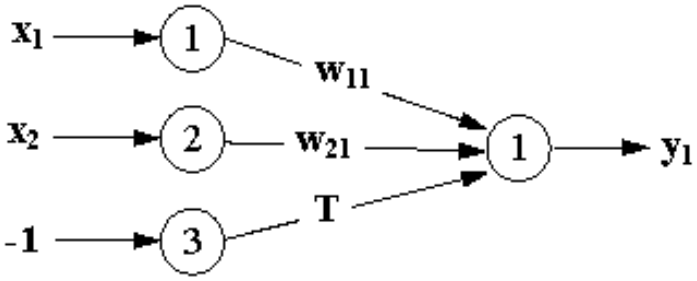

Рис. 4. Пример операции «логическое И»

Так как входное значение, подаваемое на дополнительный нейрон, равно -1, то взвешенная сумма определяется как:

$$S = w_{11}x_1 + w_{21}x_2 - T$$

$$T = w_0x_0$$

При функции активации $OUT = \begin{cases} 1, \text{если } NET > 0 \\ -1, \text{если } NET \leq 0 \end{cases}$

Имеем следующую обучающую выборку

X_1	X_2	У
1	**1**	**1**
1	**-1**	**-1**
-1	**1**	**-1**
-1	**-1**	**-1**

*В таблице **«-1»** – ложь, **«1»** – истина.*

Итак, рассмотрим процесс обучения:

Общее количество входных образов, подаваемых на нейронную сеть, ***L=4***.

Пусть $w_{11} = 0$, $w_{21} = 0$, $T = 0$ при $k = 0$ (число итераций).

При $x_1=1$, $x_2=1 \rightarrow x_1*w_1 + x_2*w_2 - T = 1*0 + 1*0 - 0 = NET_1 = 0 \rightarrow OUT_1 = -1$

По первому образу нет соответствия $\mathbf{OUT_1 = -1}$, а должно быть **y = 1.**

Производим модификацию коэффициентов. Подаем на вход последовательно четыре образа.

Для *первого образа* $k = 1$, $x_1 = 1$, $x_2 = 1$, $y = 1$

$w_{11}(k+1) = w_{11}(k) + x_1 y;$

$w_{11}(1) = w_{11}(0) + x_1 y = 0 + (1)*(1) = \mathbf{1}$

$w_{21}(k+1) = w_{21}(k) + x_2 y;$

$w_{21}(1) = w_{21}(0) + x_2 y = 0 + (1)*(1) = \mathbf{1}$

$T(k+1) = T(k) - y$

$T(1) = T(0) - y = 0 - (1) = \mathbf{-1}$

Проверяем, на соответствия.

При $x_1=1$, $x_2=1 \rightarrow x_1*w_1 + x_2*w_2 - T = 1*1 + 1*1 + 1 = NET_1 = 3 \rightarrow OUT_1 = 1$

При $x_1=1$, $x_2=-1 \rightarrow x_1*w_1 + x_2*w_2 - T = 1*1 - 1*1 + 1 = NET_2 = 1 \rightarrow OUT_2 = 1$

По второму образу нет соответствия $\mathbf{OUT_2 = 1}$, а должно быть $\mathbf{y = -1}$

Переходим ко ***второму образу***. $\boldsymbol{k = 2, x_1 = 1, x_2 = -1, y = -1}$

$w_{11}(2) = w_{11}(1) + x_1 y = 1 + 1*(-1) = \mathbf{0}$

$w_{21}(2) = w_{21}(1) + x_2 y = 1 + (-1)*(-1) = \mathbf{2}$

$T(2) = T(1) - y = -1 - (-1) = \mathbf{0}$

Проверяем, на соответствия.

При $x_1=1, x_2=1 \rightarrow x_1*w_1 + x_2*w_2 - T = 1*0 + 1*2 - 0 = NET_1 = 2 \rightarrow OUT_1 = 1$

При $x_1=1, x_2=-1 \rightarrow x_1*w_1 + x_2*w_2 - T = 1*0 - 1*2 - 0 = NET_2 = -2 \rightarrow OUT_2 = -1$

При $x_1=-1, x_2=1 \rightarrow x_1*w_1 + x_2*w_2 - T = -1*0 + 1*2 - 0 = NET_3 = 2 \rightarrow OUT_3 = 1$

По третьему образу нет соответствия $\mathbf{OUT_3 = 1}$ **,** а должно быть $\mathbf{y = -1.}$

Следующий т***ретий образ . $k = 3, x_1 = -1, x_2 = 1, y = -1$***

$w_{11}(3) = w_{11}(2) + x_1 y = 0 + (-1)*(-1) = \mathbf{1}$

$w_{21}(3) = w_{21}(2) + x_2 y = 2 + 1*(-1) = \mathbf{1}$

$T(3) = T(2) - y = 0 - (-1) = \mathbf{1}$

Проверяем, на соответствия.

При $x_1=1, x_2=1 \rightarrow x_1*w_1 + x_2*w_2 - T = 1*1 + 1*1 - 1 = NET_1 = 1 \rightarrow OUT_1 = 1$

При $x_1=1, x_2=-1 \rightarrow x_1*w_1 + x_2*w_2 - T = 1*1 - 1*1 - 1 = NET_2 = -1 \rightarrow OUT_2 = -1$

При $x_1=-1, x_2=1 \rightarrow x_1*w_1 + x_2*w_2 - T = -1*1 + 1*1 - 1 = NET_2 = -1 \rightarrow OUT_3 = -1$

При $x_1=-1, x_2=-1 \rightarrow x_1*w_1 + x_2*w_2 - T = -1*1 - 1*1 - 1 = NET_2 = -3 \rightarrow OUT_3 = -1$

Соответствие по всем образам. Обучение прекращается при подаче на вход нейронной сети всех входных образов. В итоге, мы обучили нейронную сеть (подобрали весовые коэффициенты) решать задачу «***логического И***».

Теперь пусть требуется реализовать обучение нейронной сети по правилу Хебба с учителем для операции «***логическое ИЛИ***», используя также биполярную кодировку двоичных сигналов (1, -1). Скорость обучения η=1.

Представим простейший нейрон (рис. 5).

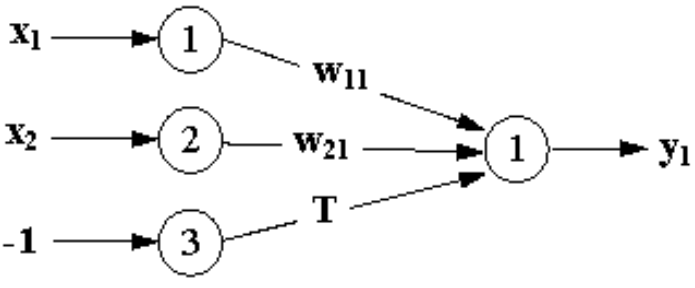

Рис. 5. Пример операции «логическое ИЛИ»

Так как входное значение, подаваемое на дополнительный нейрон, равно -1, то взвешенная сумма определяется как:

$$S = w_{11}x_1 + w_{21}x_2 - T$$

$$T = w_0 x_0$$

При функции активации $OUT = \begin{cases} 1, \text{если } NET > 0 \\ -1, \text{если } NET \leq 0 \end{cases}$

Имеем следующую обучающую выборку

X_1	X_2	У
-1	**-1**	**-1**
1	**-1**	**1**
-1	**1**	**1**
1	**1**	**1**

*В таблице **«-1»** – ложь, **«1»** – истина.*

Итак, рассмотрим процесс обучения:

Общее количество входных образов, подаваемых на нейронную сеть, ***L=4***.

Пусть $w_{11} = 0$, $w_{21} = 0$, $T = 0$ при $k = 0$ (число итераций).

При x_1=-1, x_2=-1→$x_1*w_1 + x_2*w_2 – T$ = -1*0 - 1*0 – 0=NET_1= 0→ OUT_1 = -1

При x_1=1, x_2=-1→$x_1*w_1 + x_2*w_2 – T$ = 1*0 - 1*0 – 0=NET_1= 0→ OUT_1 = -1

По второму образу нет соответствия $\mathbf{OUT_2 = -1}$, а должно быть **y = 1.**

Производим модификацию коэффициентов. Подаем на вход последовательно четыре образа.

Для *первого образа k = 1, x_1 =- 1, x_2 = -1, y = -1*

$w_{11}(k+1) = w_{11}(k) + x_1 y$;

$w_{11}(1) = w_{11}(0) + x_1 y = 0 + (-1)*(-1) = \mathbf{1}$

$w_{21}(k+1) = w_{21}(k) + x_2 y$;

$w_{21}(1) = w_{21}(0) + x_2 y = 0 + (-1)*(-1) = \mathbf{1}$

$T(k+1) = T(k) - y$

$T(1) = T(0) – y = 0 - (-1) = \mathbf{1}$

Проверяем, на соответствия.

При x_1=-1, x_2= -1→$x_1*w_1 + x_2*w_2 – T$ =-1*1 - 1*1 - 1=NET_1= -3 → OUT_1 = -1

При x_1=1, x_2= -1→$x_1*w_1 + x_2*w_2 – T$ =1*1 - 1*1 - 1=NET_2= -1 → OUT_2 = -1

По второму образу нет соответствия $\mathbf{OUT_2 = -1}$, а должно быть **y = 1.**

Переходим ко ***второму образу. k = 2, x_1 = 1, x_2 = -1, y = -1***

$w_{11}(2) = w_{11}(1) + x_1 y = 1 + 1*(1) = \mathbf{2}$

$w_{21}(2) = w_{21}(1) + x_2 y = 1 + (-1)*(1) = \mathbf{0}$

$T(2) = T(1) – y = 1 - (1) = \mathbf{0}$

Проверяем, на соответствия.

При x_1=-1, x_2=-1→$x_1*w_1 + x_2*w_2 – T$=-1*2 - 1*0 – 0 =NET_1= -2 → OUT_1 = -1

При x_1=1, x_2= -1→$x_1*w_1 + x_2*w_2 – T$ =1*2 - 1*0 - 0=NET_2= 2→ OUT_2 = 1

При x_1=-1, x_2= 1→$x_1*w_1 + x_2*w_2 – T$ =-1*2 + 1*0 - 0=NET_3= -2→ OUT_3 = -1

По третьему образу нет соответствия $\mathbf{OUT_3 = -1}$ **,** а должно быть **y = 1.**

Следующий т***ретий образ . k = 3, x_1 = -1, x_2 = 1, y = -1***

$w_{11}(3) = w_{11}(2) + x_1 y = 2 + (-1)*(1) =$ **1**

$w_{21}(3) = w_{21}(2) + x_2 y = 0 + 1*(1) =$ **1**

$T(3) = T(2) – y = 0 - (1) =$ **-1**

Проверяем, на соответствия.

При x_1=-1, x_2= -1→$x_1*w_1 + x_2*w_2 – T$ =-1*1 - 1*1 + 1=NET_1= -1→OUT_1= -1

При x_1=1, x_2= -1→$x_1*w_1 + x_2*w_2 – T$ =1*1 - 1*1 + 1=NET_2= 1→ OUT_2 = 1

При x_1=-1, x_2= 1→$x_1*w_1 + x_2*w_2 – T$ =-1*1 + 1*1 + 1=NET_2= 1→OUT_3= 1

При x_1=1, x_2=1→$x_1*w_1 + x_2*w_2 – T$=1*1 + 1*1 + 1 =NET_1= 3 → OUT_1 = 1

Соответствие по всем образам. Обучение прекращается при подаче на вход нейронной сети всех входных образов. В итоге, мы обучили нейронную сеть (подобрали весовые коэффициенты) решать задачу «логического ИЛИ».

Однако зададимся вопросом: можно классифицировать логические функции? Да, и к тому же эта задача отлично проиллюстрирует такую классификацию.

Классификация.

Задача классификации представляет собой задачу отнесения образца к одному из нескольких попарно не пересекающихся множеств.

Прежде всего, нужно определить уровень сложности системы. В реальных задачах часто возникает ситуация, когда количество образцов ограничено, что осложняет определение сложности задачи. Возможно выделить три основных уровня сложности. Первый (самый простой) – когда классы можно разделить прямыми линиями (или гиперплоскостями, если пространство входов имеет размерность больше двух) – так называемая линейная разделимость. Во втором случае классы невозможно разделить линиями (плоскостями), но их возможно отделить с помощью более сложного деления – нелинейная разделимость. В третьем случае классы пересекаются и можно говорить только о вероятностной разделимости (рис. 5).

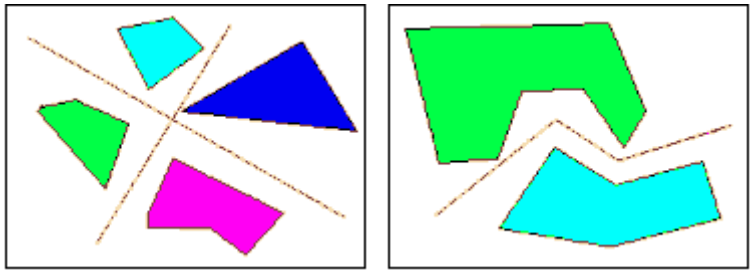

Рис.5. Линейно и нелинейно-разделимые классы

В идеальном варианте после предварительной обработки мы должны получить линейно разделимую задачу, так как после этого значительно упрощается построение классификатора. К сожалению, при решении реальных

задач мы имеем ограниченное количество образцов, на основании которых и производится построение классификатора. При этом мы не можем провести такую предобработку данных, при которой будет достигнута линейная разделимость образцов.

Разделяющая линия для операции «логическое И» будет $1x_1 + 1x_2 - 1 = 0 \rightarrow x_1 + x_2 = 1$ и для операции «логическое ИЛИ» $1x_1 + 1x_2 + 1 = 0 \rightarrow x_1 + x_2 = -1$ (рис.6)

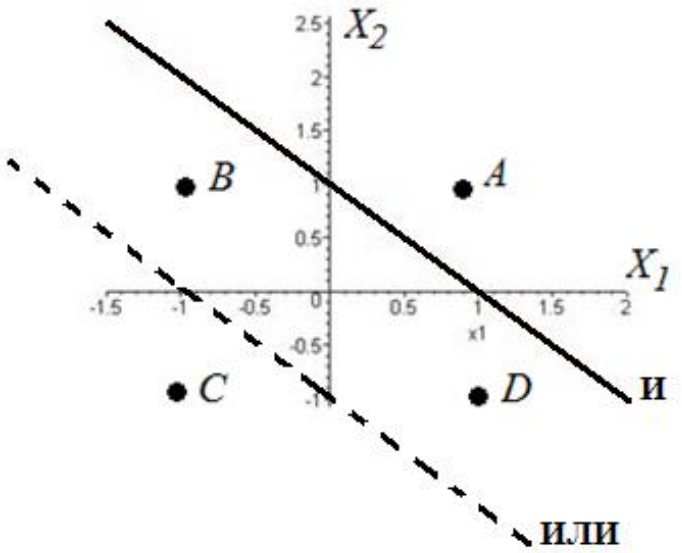

Рис.6. Разделяющие линии для операции «логическое И» и «логическое ИЛИ»

Таким образом, отделяются нули от единиц.

В методе Хэбба обучение является исключительно локальным явлением, охватывающим только два нейрона и соединяющий их синапс; не требуется глобальной системы обратной связи для развития нейронных образований.

Последующее использование метода Хэбба для обучения нейронных сетей привело к большим успехам, но наряду с этим показало ограниченность метода; некоторые образы просто не могут использоваться для обучения этим методом. В результате появилось большое количество расширений и нововведений, большинство из которых в значительной степени основано на работе Хэбба.

Персептрон Розенблатта

В 1958 г. Фрэнк Розенблат создал компьютерную программу для IBM-794, эмулирующую деятельность математических нейронов. Это была первая нейронная сеть или сокращенно—нейросеть. Она была названа персептроном от английского слова perception—осознание.

Персептрон (Perceptron) — простейший вид нейронных сетей. В основе лежит математическая модель восприятия информации мозгом, состоящая из

сенсоров, ассоциативных и реагирующих элементов.

Затем, спустя два года, Розенблатт смонтировал электронное устройство, в котором функции математических нейронов выполняли отдельные электросхемы, работающие на электронных лампах. Это был первый нейрокомпьютер, который успешно решал сложнейшую интеллектуальную задачу—распознавал буквы латинского алфавита, изображенные на карточках, подносимых к его считывающему устройству –электронному глазу.

В самом общем своем виде (как его описывал Розенблатт) он представляет систему из элементов трех разных типов: сенсоров, ассоциативных элементов и реагирующих элементов (рис. 7).

S-нейроны (сенсорные) – предназначены для формирования входных сигналов; выполняют чисто распределительную функцию. Каждый сенсорный нейрон связан с одним или несколькими нейронами следующего слоя (ассоциативными нейронами), при этом каждый ассоциативный нейрон может быть связан с несколькими S-нейронами.

A-нейроны (ассоциативные) – предназначены для непосредственной обработки входных сигналов. Выходы ассоциативных нейронов соединены с входами нейронов третьего слоя.

R-нейроны (эффекторные) – предназначены для передачи сигналов к другим нейронам или сетям. Нейроны этого слоя имеют несколько входов (дендритов) и один выход (аксон), который возбуждается, если суммарная величина входных сигналов превосходит порог срабатывания (функция активации нейронов – пороговая).

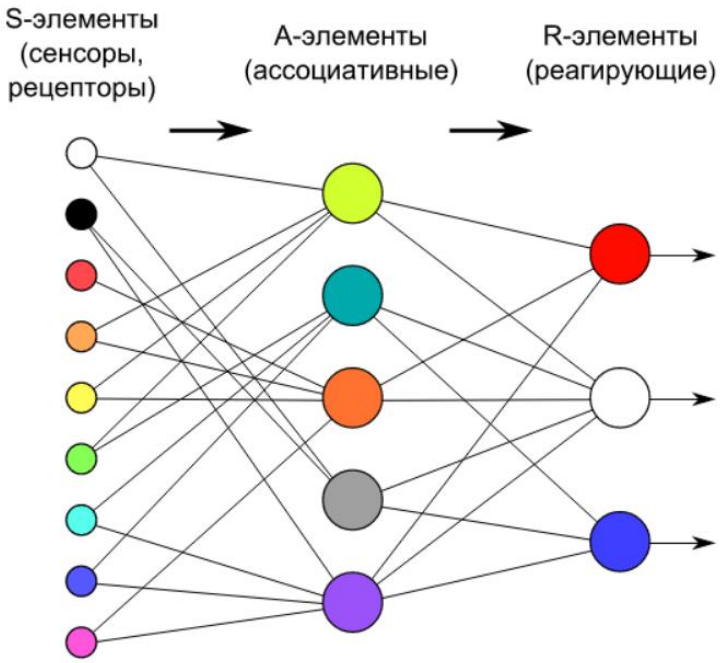

Рис.7. Персептрон Розенблатта

Рассмотрим принцип работы персептрона.

Первыми в работу включаются S-элементы. Они могут находиться либо в состоянии покоя (сигнал равен 0), либо в состоянии возбуждения (сигнал равен 1). Далее сигналы от S-элементов передаются A-элементам по так называемым S-A связям. Эти связи могут иметь веса, равные только -1, 0 или 1. Затем сигналы от сенсорных элементов, прошедших по S-A связям попадают в A-элементы, которые еще называют ассоциативными элементами. Стоит заметить, что одному A-элементу может соответствовать несколько S-элементов. Если сигналы, поступившие на A-элемент, в совокупности превышают некоторый его порог θ, то этот A-элемент возбуждается и выдает сигнал, равный 1. В противном случае (сигнал от S-элементов не превысил порога A-элемента), генерируется нулевой сигнал.

Почему A-элементы назвали ассоциативными? Дело в том, что A-элементы являются агрегаторами сигналов от сенсорных элементов. Например, у нас есть группа сенсоров, каждый из которых распознает кусок буквы «Д» на исследуемой картинке. Однако только их совокупность (то есть когда несколько сенсоров выдали сигнал, равный 1) может возбудить A-элемент целиком. На другие буквы A-элемент не реагирует, только на букву «Д». То есть он ассоциируется с буквой «Д». Отсюда и такое название.

Разберем принцип действия персептрона на примере решения конкретной задачи. На рис. 8 приведен один из простейших вариантов исполнения персептрона, предназначенного для классификации чисел на четные и нечетные. Представим себе матрицу из 12 фотоэлементов, расположенных в виде четырех горизонтальных рядов по три фотоэлемента в каждом ряду.

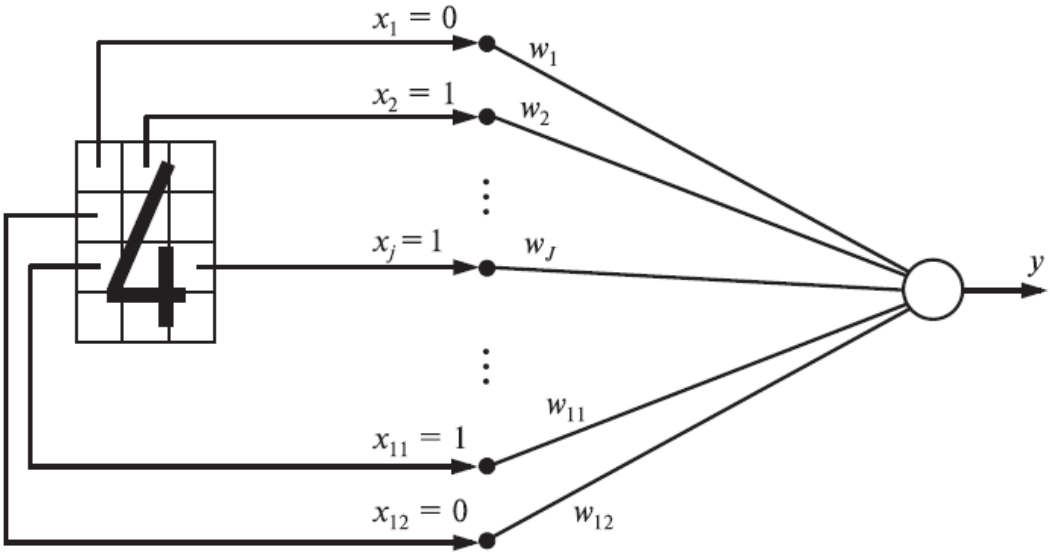

Рис. 8. Персептрон, классифицирующий числа на четные и нечетные

На матрицу фотоэлементов накладывется карточка с изображением

цифры, например, «4». Если на какой-либо фотоэлемент попадает фрагмент цифры, то этот фотоэлемент вырабатывает сигнал в виде единицы, в противном случае - нуль. На первый фотоэлемент не попал фрагмент цифры, и поэтому его сигнал $x_1 = 0$; на второй фотоэлемент попал фрагмент цифры, и поэтому он вырабатывает сигнал $x_2 = 1$ и т. д.

Так, каждая цифра получает соответствующий бинарный код. Пример для цифры «5» приведен на рис. 9.

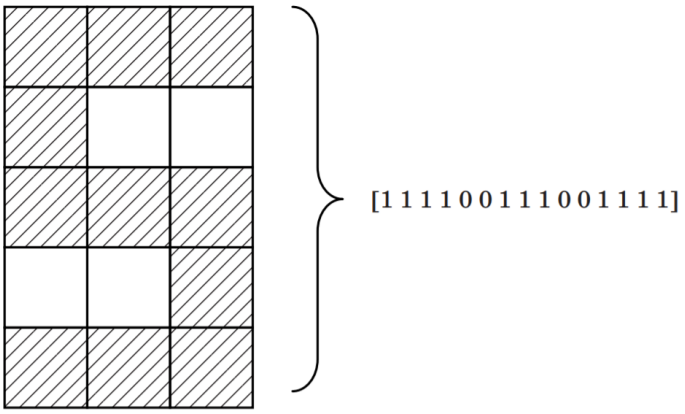

Рис. 9. Кодирование изображений цифр

Математический нейрон выполняет суммирование входных сигналов x_j , помноженных на синаптические веса w_j. Затем результат суммирования S сравнивается с порогом чувствительности T и вырабатывается выходной сигнал y. Первоначальные значения синаптических весов w_j. и порога чувствительности T Розенблатт задавал датчиком случайных чисел, поэтому на выходе персептрона случайным образом вырабатывался сигнал: либо 0, либо 1

Задача состояла в следующем. Требовалось подобрать значения синаптических весов w_j такими, чтобы выходной сигнал y принимал значение единица, если на карточке было изображено четное число, и нуль, если число было нечетным.

Эту задачу Ф. Розенблатт решил путем поочередного накладывания на фотоэлементы карточек и обучения персептрона, путем корректировки синаптических весов w_j. Если, например, на вход персептрона предъявлялась карточка с цифрой «4» и выходной сигнал y случайно оказывался равным единице, означающей четность, то корректировать синаптические веса было не нужно, так как реакция персептрона правильна. А если выходной сигнал оказался равным нулю, что неправильно, то следовало увеличить (поощрить) веса тех активных входов, которые способствовали возбуждению нейрона.

Отметим, что алгоритм обучения персептрона с помощью правил Хебба удивительным образом напоминают процесс обучения ребенка или студента

методом «поощрения—наказания» (или дрессировки животного методом «кнута и пряника»). Обратим внимание также на то, что первоначальные значения синаптических весов w_j задаются датчиком случайных чисел. Это соответствует тому, что при рождении человека или животного его мозг еще не накопил знаний, и поэтому силы синаптических связей w_j имеют какие-то случайные значения. Как и в случаях с ребенком, студентом и животным, обучаемом методом «поощрения-наказания», алгоритм обучения персептрона за конечное число попыток (их называют итерациями, или эпохами) может привести к цели— персептрон в конце концов усвоит необходимые знания, закодирует их в виде конкретных значений матрицы сил синаптических связей w_j и, таким образом, научится различать четные и нечетные числа.