МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ РОССИЙСКОЙ ФЕДЕРАЦИИ
ИНСТИТУТ НЕФТИ И ГАЗА ФЕДЕРАЛЬНОГО ГОСУДАРСТВЕННОГО БЮДЖЕТНОГО ОБРАЗОВАТЕЛЬНОГО УЧРЕЖДЕНИЯ ВЫСШЕГО ОБРАЗОВАНИЯ
«УФИМСКИЙ ГОСУДАРСТВЕННЫЙ НЕФТЯНОЙ ТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ»
(ФИЛИАЛ В Г. ОКТЯБРЬСКОМ)

Кафедра «Информационные технологии, математика и естественные науки»

Курс ДОП «Искусственный интеллект и нейронные сети»

Учебно-методическое пособие для практических и самостоятельных работ

Уфа
УНПЦ «Издательство УГНТУ»
2023

Учебно-методическое пособие предназначено для обучающихся по курсу ДОП «Искусственный интеллект и нейронные сети», а также может быть использовано и студентами других специальностей, изучающими данный курс.

Составители: Тынчеров К.Т., проф. кафедры ИТМЕН, д-р техн. наук ИНГ УГНТУ в г. Октябрьском
Селиванова М.В. ассистент кафедры РРНГМ ИНГ УГНТУ в г. Октябрьском
Шагалеева Е.Г., спец-т по УМР кафедры ИТМЕН ИНГ УГНТУ в г. Октябрьском

Рецензент: Хузина Л.Б. Член-корреспондент РАЕН, заведующий кафедрой «Бурение нефтяных и газовых скважин» ГБОУ ВО «Альметьевский государственный нефтяной институт», доктор технических наук, доцент

Оглавление

ПРАКТИКУМ
Практическая работа №1.
Математический нейрон

Теоретический материал к практической работе №1
Время выполнения работы 1 час (45 мин.)

Теоретический материал к практической работе №1

Математический нейрон был предложен американскими учеными Уорреном Мак-Каллоком и Вальтером Питтсем в 1943г.

Математический нейрон – это математическая модель биологического нейрона мозга. Его изображают в виде кружочка со стрелками, обозначающими входы и выход. На рис 1.1 математический нейрон имеет J входов и один выход.

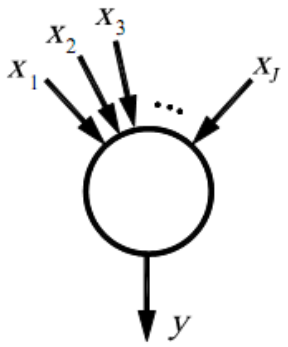


Рис. 1.1. Математический нейрон Мак-Каллока – Питтса

Через входы математический нейрон принимает входные сигналы x_j, которые суммирует, умножая каждый входной сигнал на некоторый весовой коэффициент w_j:

$$S = \sum_{j=1}^{J} w_j \cdot x_j. \qquad (1.1)$$

Затем математический нейрон формирует свой выходной сигнал согласно правилу:

$$y = \begin{cases} 1, \text{ если } S \geq \theta; \\ 0, \text{ если } S < \theta. \end{cases} \qquad (1.2)$$

в котором величину θ называют порогом 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 возбужден. Если же входные сигналы достаточно интенсивны и их взвешенная сумма достигает порога чувствительности θ, то нейрон переходит в возбужденное состояние, и на его выходе, согласно формуле (1.2), образуется сигнал $y = 1$. Весовые коэффициенты w_j имитируют электропроводность нервных волокон – силу синаптических связей между нейронами. Чем эти силы выше, тем больше вероятность перехода нейрона в возбужденное состояние. С

другой стороны, вероятность перехода нейрона в возбужденное состояния повышается при уменьшении порога чувствительности θ.

Логическая функция (1.2) называется активационной функцией нейрона. Ее графическое изображение имеет вид, представленный на рис 1.2. За этот вид ее иногда называют «функцией-ступенькой».

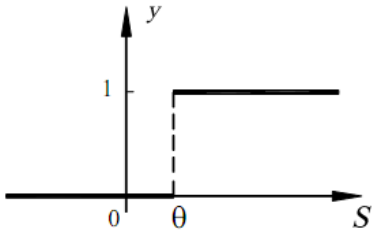


Рис. 1.2. Активационная функция нейрона: «ступенька»

С помощью математического нейрона можно моделировать различные логические функции, например, функцию логического умножения «И» («AND»), функцию логического сложения «ИЛИ» («OR») и функцию логического отрицания «НЕТ» («NOT»). Таблицы истинности этих логических функций приведены на рис.1.3.

x_1	x_2	y
0	0	0
0	1	0
1	0	0
1	1	1
«И»		

x_1	x_2	y
0	0	0
0	1	1
1	0	1
1	1	1
«ИЛИ»		

x	y
0	1
1	0
«НЕТ»	

Рис. 1.3. Таблицы истинности логических функций

С помощью этих таблиц и формул (1.1)-(1.2) нетрудно убедиться, что математический нейрон, имеющий два входа с единичными силами синаптических связей $w_1 = w_2 = 1$, моделирует функцию логического умножения «И» при $\theta=2$, этот же нейрон моделирует функцию логического сложения «ИЛИ» при задании $\theta = 1$. Математический нейрон с одним входом моделирует функцию «НЕТ» при задании $w = -1$ и $\theta = 0$.

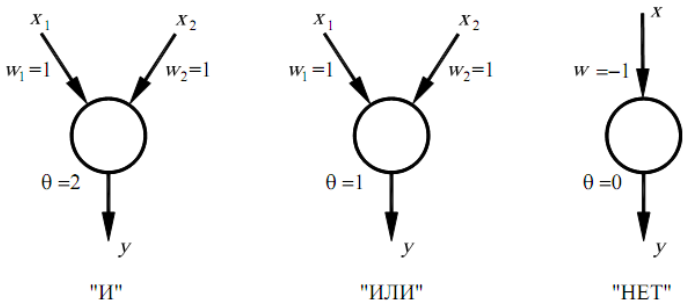


Рис.1.4. Математические нейроны, моделирующие логические функции

Однако существуют логические функции, которые невозможно моделировать с помощью математического нейрона Мак-Каллока – Питтса. Такой логической функцией является Исключающее ИЛИ», таблица истинности которой приведена на рис. 1.5.

x_1	x_2	y
0	0	0
0	1	1
1	0	1
1	1	0

Рис. 1.5. Таблица истинности функции «Исключающее ИЛИ»

Задачи, которые подобно проблеме «Исключающего ИЛИ» с помощью однослойного персептрона решены быть не могут, называют линейно неразделимыми задачами. В свое время ученые потратили немало сил и средств, пытаясь решить такие задачи, ошибочно полагая, что причина их неудач состоит в недостаточной мощности существующих компьютеров и в недостаточном количестве совершенных попыток.

По-видимому, нечто подобное может случиться и с Вами при выполнении практической работы №1. Подобрав значения синаптических весов w_1 , w_2 и порога θ, Вы успешно справляетесь с моделированием логических функций «И» и «ИЛИ», тогда как попытки моделирования функции «Исключающее ИЛИ» к успеху не приводят. Объяснению этого явления и преодолению проблемы «Исключающего ИЛИ» будет посвящена Практическая работа №5.

В заключение отметим, что в современной литературе иногда вместо понятия порога чувствительности нейрона θ используют термин нейронное смещения b, которое отличается от порога θ только знаком: $b = -\theta$. Если величину b добавить к сумме (4.1):

$$S = \sum_{j=1}^{J} w_j \cdot x_j + b, \tag{1.3}$$

то пороговая активационная функция нейрона примет вид:

$$y = \begin{cases} 1, \text{ если } S \geq \theta; \\ 0, \text{ если } S < \theta. \end{cases} \tag{1.4}$$

Графическое представление этой активационной функции приведено на рис. 1.6, а.

Еще более симметричный вид, представленный на рис. 1.6, б, активационная функция нейрона приобретает при использовании формулы:

$$y = \begin{cases} 1, \text{ если } S \geq \theta; \\ -1, \text{ если } S < \theta. \end{cases} \tag{1.5}$$

а) б)

Рис.1.6. Пороговые активационные функции нейрона, заданные формулами:
а – (1.4); *б* – (1.5)

В формуле (1.3) нейронное смещение b можно рассматривать как вес w_0 некоторого дополнительного входного сигнала x_0 , величина которого всегда равна единице:

$$S = \sum_{j=1}^{J} w_j \cdot x_j + w_0 x_0 = \sum_{j=0}^{J} w_j \cdot x_j. \qquad (1.6)$$

Нейрон с дополнительным входом x_0 изображен на рис. 1.7.

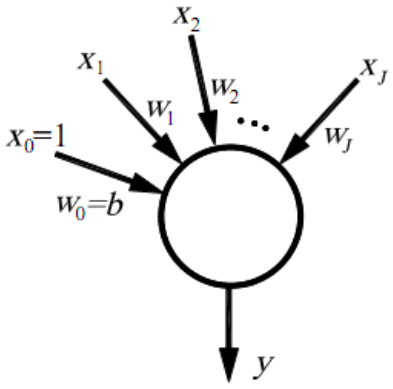
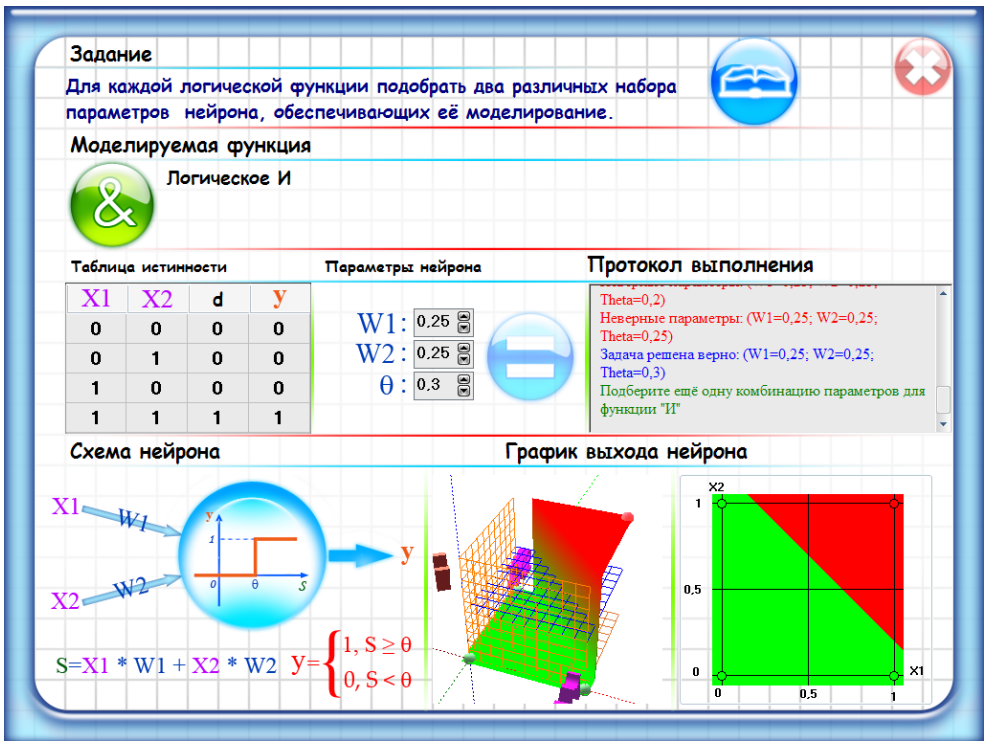


Рис. 1.7. Нейронное смещение b интерпретируется как вес дополнительного входа w_0, сигнал которого x_0 всегда равен 1

Запустите программу «Лабораторный практикум ИИ» и приступите к исследованию.



Задание этой практической работы состоит в том, чтобы путем подбора синаптических весов и порога чувствительности математического нейрона заставить его моделировать логические функции: «И», «ИЛИ» и др. Работа выполняется в интерактивном режиме практически без помощи преподавателя. Читая сообщения, появляющиеся в «Протоколе выполнения», студенты сами пытаются выполнить все задания. Преподавателю рекомендуется объявить конкурс – кто первый справится с работой, и не мешать студентам соревноваться.

В распоряжении студентов имеется теоретический материал (открывается путем нажатия на кнопку в виде развернутой книги), таблицы истинности логических функций (слева по центру), схема математического нейрона с формулами его работы (слева внизу), графическое изображение работы нейрона (снизу по центру и справа).

После выполнения исследований необходимо оформить отчет (форма выложена в личном кабинете студента), защитить практическую работу у преподавателя.

Практическая работа №2
Классификация чисел

Теоретический материал к практической работе №2
Время выполнения работы 1 час (45 мин.)

Теоретический материал к практической работе №2

Как мы уже знаем из пройденного материала, американские ученые У.Мак-Каллок и В. Питтс предложили математическую модель нейрона мозга человека, назвав ее математическим нейроном. Так же как и биологический нейрон мозга, математический нейрон имеет несколько входов и один выход. Кроме того, он может существовать в возбужденном и невозбужденном состояниях, причем переход в возбужденное состояние зависит от величины поступающих к нему сигналов и сил синаптических связей. Таким образом, математический нейрон весьма правдоподобно имитирует структуру и свойства своего прототипа – биологического нейрона мозга. На этом основании У.МакКаллок и В.Питтс высказали весьма смелое предположение, которое впоследствии легло в основу современной нейроинформатики. Они предположили, что если математические нейроны связать между собой проводниками электрического тока, имитирующими нервные волокна, то такой искусственный мозг будет способен решать интеллектуальные задачи, подобно тому, как это делает естественный человеческий мозг.

Идея Мак-Каллока–Питтса была воплощена в жизнь в 1958 г. американским ученым Фрэнком Розенблаттом, также считающимся основателем нейроинформатики. Сначала он 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.

Затем, спустя два года, Ф.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.

Разберем принцип действия персептрона на примере несколько более простой задачи, чем задача распознавания букв. На рисунке 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. На матрицу фотоэлементов накладывается карточка с изображением цифры, например «4», как изображено на рисунке. Если на какой-либо фотоэлемент попадает фрагмент цифры, то этот фотоэлемент вырабатывает сигнал в виде единицы, в противном случае – ноль.

На рисунке первый фотоэлемент выдает сигнал $x_1 = 0$, второй фотоэлемент – $x_2 = 1$ и т.д.

Согласно формулам

$$S = \sum_{j=1}^{J} w_j \cdot x_j. \tag{1.1}$$

$$y = \begin{cases} 1, \text{ если } S \geq \theta; \\ 0, \text{ если } S < \theta. \end{cases} \tag{1.2}$$

математический нейрон выполняет суммирование входных сигналов x_j, помноженных на синаптические веса w_j , после чего результат суммирования S сравнивается с порогом чувствительности θ и вырабатывается выходной сигнал y.

Первоначальные значения синаптических весов w_j и порога чувствительности θ Ф.Розенблатт задавал датчиком случайных числе, 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 ноль, если число было нечетным.

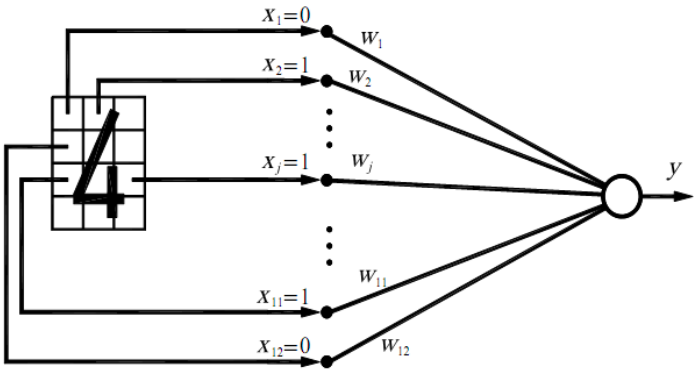


Рис. 1.9. Персептрон, классифицирующий числа на четные и нечетные

Эта задачу Ф.Розенблатт решил путем поочередного накладывания на фотоэлементы карточек и обучения персептрона, заключающегося в корректировке 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. В данном случае увеличению подлежали w_2, w_{11} и др.

Следуя этой идее, можно сформулировать итерационный алгоритм корректировки синаптических весов, обеспечивающий обучение персептрона в нужном направлении.

Шаг 1. Датчиком случайных чисел всем синаптическим весам w_j (j =1, 2, ..., 12) и порогу чувствительности нейрона θ присвоить некоторые малые случайные значения.

Шаг 2. Предъявить персептрону какую-либо цифру. Системой фотоэлементов вырабатывается входной вектор x_j (j =1,2 ,..., 12).

Шаг 3. Нейрон выполняет взвешенное суммирование входных сигналов

$$S = \sum_{j=1}^{12} w_j \cdot x_j.$$

и вырабатывает выходной сигнал y =1, если $S \geq \theta$, или $y = 0$, если $S < \theta$.

Шаг 4,*а*. Если выходной сигнал правильный, то перейти на шаг 2.

Шаг 4,*б*. Если выходной сигнал неправильный и равен нулю, то увеличить веса активных входов, добавить каждому j -му синаптическому весу величину j -го входного сигнала:

$$w_j(t+1) = w_j(t) + x_j.$$

Тогда, если вход был неактивен, т.е. x_j =0, то j -й синаптический вес не изменится. Если же вход был активен, т.е. x_j=1, то j -й синаптический вес будет увеличен на 1.

Здесь и далее буква t означает номер итерации, которые в искусственном интеллекте называют эпохами; $w_j(1 + t)$ – новое значение (на новой эпохе) j -го синаптического веса; $w_j(t)$ – его старое значение (на предыдущей эпохе).

Шаг 4,*в*. Если выходной сигнал неправильный и равен единице, то уменьшить веса активных входов, например с помощью аналогичной формулы:

$$w_j(t+1) = w_j(t) - x_j.$$

Шаг 5. Перейти на шаг 2 или завершить процесс обучения.

В приведенном здесь алгоритме шаг 4,*б* называют первым правилом Хебба, а шаг 4, *в* – вторым правилом Хебба в честь канадского ученого физиолога Д.О.Хебба, предложившего этот алгоритм в 1949г. Отметим, что правила Хебба удивительным образом напоминают процесс обучения ребенка или школьника методом «*поощрения – наказания*» (или дрессировки животного методом «кнута и пряника»). Как и в случаях с ребенком, обучаемом этим методом, 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.

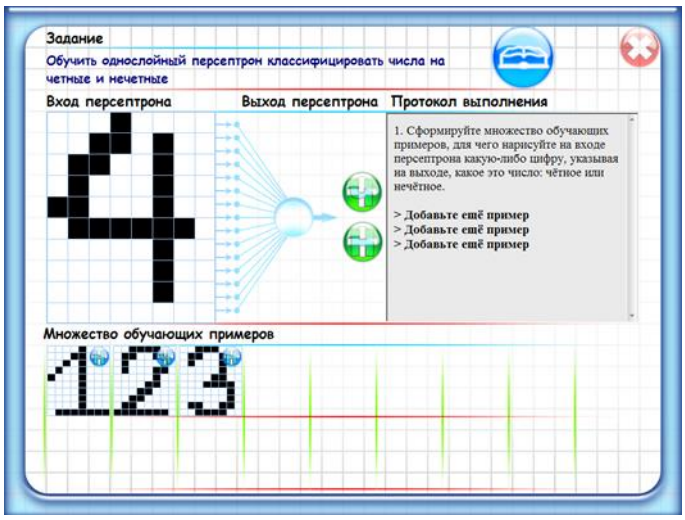
Естественно возникает вопрос, всегда ли алгоритм обучения персептрона приводит к желаемому результату. Ответ на этот вопрос дает теорема сходимости персептрона:

Если существует множество значений весов, которые обеспечивают конкретное различение образов, то в конечном итоге алгоритм обучения

персептрона приводит либо к этому множеству, либо к эквивалентному ему множеству, такому, что данное различение образов будет достигнуто.

В настоящее время считается, что по числу выполненных доказательств теорема сходимости персептрона занимает первое место в мире. Ранее самой доказанной в мире теоремой считалась теорема Пифагора.

Запустите программу и приступайте к выполнению лабораторных исследований.



Задание состоит в том, чтобы обучить персептрон классифицировать цифры на четные и нечетные.

Читая «Протокол выполнения», студенты рисуют цифры на табло «Вход персептрона» и, отмечая их четность или нечетность, формируют множество обучающих примеров, которое постепенно располагается в нижней части рабочего окна. Путем нажатия кнопки «Обучить» (она появляется по ходу выполнения работы) студенты запускают процесс обучения персептрона, наблюдают за его графическим отображением, а затем убеждаются в правильности работы обученного персептрона.

Как и прежде, перед началом занятия преподавателю рекомендуется выдать теоретический материал и, после запуска лабораторных работ, напоминать студентам о необходимости внимательного чтения и выполнения всех пунктов «Протокола выполнения».

После выполнения исследований необходимо оформить отчет (форма выложена в личном кабинете студента), защитить практическую работу у преподавателя.

Практическая работа №3
Распознавание печатных букв

Теоретический материал к практической работе №3
Время выполнения работы 1 час (45 мин.)

Теоретический материал к практической работе №3

Рассмотренный в предыдущей практической работе алгоритм обучения персептрона можно представить в более общей форме. Если за d обозначить требуемый выходной сигнал (от слов *desire response*, что в переводе с английского означает – желаемый отклик), то на каждой эпохе обучения можно рассчитывать разницу между требуемым ответом персептрона d и реальным значением y , вычисляемым на его выходе:

$$\varepsilon=(d - y). \qquad (1.7)$$

Тогда:

- случай $\varepsilon = 0$ соответствует шагу 4,*а*;
- случай $\varepsilon > 0$ соответствует шагу 4,*б;*
- случай $\varepsilon < 0$ соответствует шагу 4,*в*.

Идея алгоритма обучения персептрона с помощью правил Хебба сохранится, если итерационный процесс корректировки весов вести по формулам:

$$w_j(t+1) = w_j(t) + \Delta w_j, \qquad (1.8)$$

$$\Delta w_j = \varepsilon x_j, \qquad (1.9)$$

в которых $w_j(t)$ и $w_j(t+1)$– старое и новое значения весовых коэффициентов персептрона, j – номер входного сигнала.

Кроме того, можно получить аналогичную итерационную формулу для подстройки нейронного смещения b, если учесть, что его можно интерпретировать как вес w_0 дополнительного входа x_0 , значение которого всегда равно 1 (см. теоретический материал к практической работе №1):

$$w_0(t+1) = w_0(t) + \Delta w_0, \qquad (1.10)$$

$$\Delta w_0 = \varepsilon. \qquad (1.11)$$

В итерационные формулы полезно ввести коэффициент скорости обучения η, с помощью которого можно управлять величиной коррекции синаптических весов и нейронного смещения:

$$\Delta w_j = \eta \varepsilon x_j, \qquad (1.12)$$

$$\Delta w_0 = \eta \varepsilon. \qquad (1.13)$$

При слишком больших значениях коэффициента η обычно теряется устойчивость процесса обучения, тогда как при слишком малых – повышаются временные затраты. На практике коэффициент скорости обучения η обычно задают в пределах от 0,05 до 1.

Алгоритм обучения персептрона с использованием этих формул известен под названием *дельта-правила*.

Дальнейшее развитие идеи персептрона и алгоритмов обучения связано с усложнением его структуры и развитием функциональных свойств. На рисунке представлена схема персептрона, предназначенного для распознавания букв русского алфавита. В отличие от предыдущей схемы, такой персептрон имеет 33 выходных нейрона: каждой букве алфавита соответствует свой выходной нейрон. Полагается, что сигнал первого выходного нейрона y_1 должен быть равен единице, если персептрону предъявлена буква «А», и нулю для всех остальных букв. Выход второго нейрона y_2 должен быть равен единице, если персептрону предъявлена буква «Б», и нулю во всех остальных случаях. И так далее до буквы «Я».

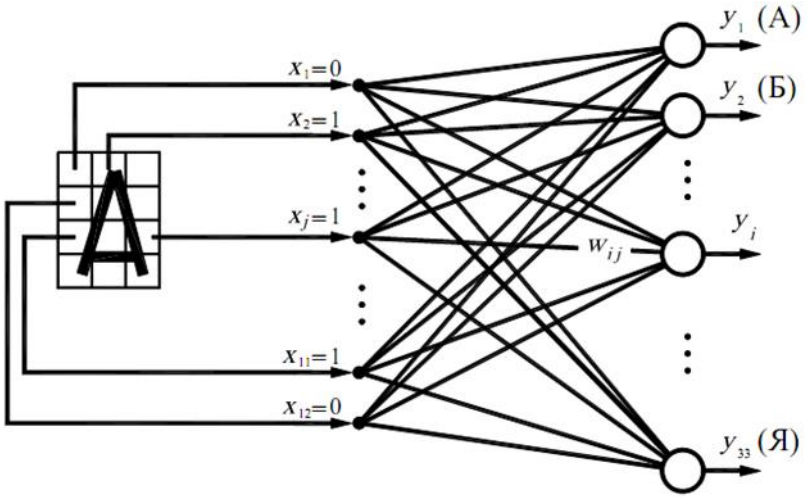


Рис. 1.10. Персептрон, предназначенный для распознавания букв русского алфавита

Алгоритм обучения данного персептрона выглядит следующим образом.

Шаг 1. Датчиком случайных чисел всем весовым коэффициентам w_{ij} и нейронным смещениям w_{i0} (i =1, ..., 33 , j= 1,… ,12) присваиваются некоторые малые случайные значения.

Шаг 2. Персептрону предъявляется какая-либо буква алфавита, системой фотоэлементов вырабатывается входной вектор x_j (j =1,2 ,... 12). Сигналы дополнительных нейронных входов присваиваются единичными: x_0=1 .

Шаг 3. Каждый нейрон выполняет взвешенное суммирование входных сигналов

$$S_i = \sum_{j=0}^{12} w_{ij} x_j$$

и вырабатывает выходной сигнал y_i =1, если $S_i \geq 0$; $y_i = 0$, если $S_i < 0$.

$$\varepsilon_i = d_i - y_i,,$$

где d_i – вектор правильных (желаемых) ответов персептрона, например, для буквы «А» d_1 =1, d_2 = 0,…, d_{33}= 0 и т.д.

Шаг 5. Производится корректировка весовых коэффициентов и нейронных смещений:

$$w_{ij}(t+1) = w_{ij}(t) + \Delta w_{ij}; \qquad \Delta w_{ij} = \eta \varepsilon_i x_j;$$
$$w_{i0}(t+1) = w_{i0}(t) + \Delta w_{i0}; \qquad \Delta w_{i0} = \eta \varepsilon_i,$$

где t – номер итерации (эпохи).

Шаг 6. Повторение шагов 2 – 5 необходимое количество раз.

Заметим, что в этом алгоритме формулы для корректировки нейронных смещений $w_{i\,0}$ можно не писать, т.к. они будут выполняться автоматически, если цикл по индексу j начинать не от единицы, а от нуля.

Как уже отмечалось ранее, первый действующий персептрон был создан в 1958-1961 гг. Он был предназначен для распознавания букв латинского алфавита. Буквы, отпечатанные на карточках, поочередно накладывали на табло фотоэлементов и осуществляли процесс обучения персептрона согласно приведенному здесь алгоритму. После выполнения достаточно большого количества итераций персептрон научился безошибочно распознавать все буквы, участвовавшие в обучении. Таким образом, была подтверждена гипотеза о том, что компьютер, построенный по образу и подобию человеческого мозга, будет способен решать интеллектуальные задач и, в частности – решать задачу распознавания образов.

Но это было не все. Помимо того, что персептрон научился распознавать знакомые образы, т.е. те образы, которые демонстрировались ему в процессе обучения, он успешно справлялся с распознаванием образов, которые «видел» впервые. Выяснилось, что персептрон оказался способным распознавать буквы, отпечатанные с небольшими искажениями и даже другим шрифтом, если шрифт не слишком сильно отличался от используемого при обучении персептрона.

Свойство мозга узнавать образы, которые ему встретились впервые, называется *свойством обобщения*. Это свойство было унаследовано персептроном непосредственно от его прототипа – мозга. Оно было унаследовано благодаря тому, что персептрон является адекватной моделью мозга, удачно отражающей как его структурные, так и функциональные качества. Именно свойство обобщения впоследствии позволило применить персептрон для решения широчайшего круга практических задач, недоступных для традиционных методов. Именно благодаря этому свойству нейронные сети стали эффективнейшим инструментом научных исследований и практических приложений. Именно благодаря этому свойству нейросетевые и нейрокомпьютерные технологии заняли то лидирующее положение, которое они занимают в настоящее время.

Практическая работа №4
Распознавание печатных и рукописных букв

Теоретический материал к практической работе №4
Время выполнения работы 1 час (45 мин.)

Теоретический материал к практической работе №4

При выполнении предыдущей практической работы мы убедились, что персептрон научился распознавать не только буквы, на которых его обучали, но и буквы, которых в обучающем множестве не было, если они не слишком отличались от букв обучающего множества. Свойство распознавать новые образы, которые персептрон никогда «не видел», мы назвали свойством *обобщения*.

Дальнейшее развитие идеи персептрона было связано с попытками расширить круг его применения и усовершенствовать алгоритм обучения. Существенное развитие персептрона было сделано американскими учеными *Б.Уидроу* и *М.Е. Хоффом*, которые вместо изученной на первой практической работе ступенчатой активационной функции ввели непрерывную нелинейную функцию активации

$$y = \frac{1}{1+e^{-S}},, \qquad (1.14)$$

график которой изображен на рис. 4.11.

Эту функцию назвали сигмоидой, из-за того, что ее графическое изображение напоминает латинскую букву «*S*». Другое название сигмоиды – логистическая функция.

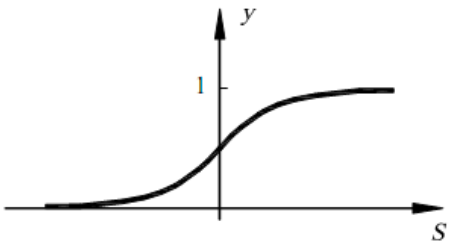


Рис. 1.11. Сигмоидная активационная функция $y = f_\sigma(S)$

Подобно обычной пороговой функции активации, сигмоида отображает точки области определения $(-\infty, +\infty)$ в значения из интервала (0, +1).

Практически сигмоида обеспечивает непрерывную аппроксимацию классической пороговой функции.

Для сигмоиды приняли обозначение $y = f_\sigma(S)$. Персептроны с сигмоидными активационными функциями с одним выходом назвали *адалайн*, с несколькими выходами – *мадалайн* (от английских слов *ADAptive LInear NEuron* и *Ma ny ADALINE*).

Появление персептронов с непрерывными активационными функциями обусловило появление новых подходов к их обучению. Б. Уидроу и М.Е. Хофф предложили минимизировать квадратичную ошибку, определяемую формулой:

$$\varepsilon = \frac{1}{2}\sum_{i=1}^{I}\left(d_i - y_i\right)^2, \qquad (1.15)$$

в которой d_i – требуемый (желаемый) выход i -го нейрона, а y_i – тот, который получился в результате вычислений персептрона.

Рассмотрим алгоритм коррекции весовых коэффициентов персептрона, имеющего J входов и I выходов (рис. 1.12).

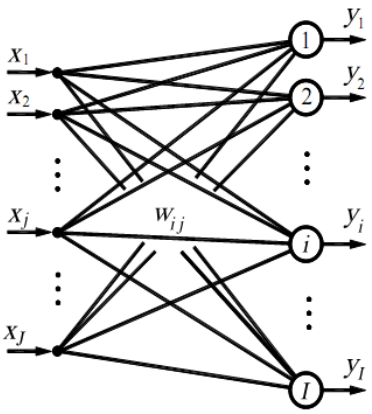


Рис. 1.12. Персептрон с J входами и I выходами.

Квадратичная ошибка обучения персептрона ε зависит от того, какими являются весовые коэффициенты w_{ij}. Другими словами ε является функцией от весовых коэффициентов w_{ij}: ε= ε (w_{ij}). В школьных курсах обычно изучаются функции только от одного аргумента: y= y (x) , которые на координатной плоскости x, y изображаются, как известно, в виде кривых линий. Если функция z зависит от двух аргументов: z = z (y, x), то она изображается в трехмерной системе координат x, y, z в виде поверхности. Функция ошибка персептрона ε=ε (w_{ij}) зависит от большого количества аргументов w_{ij} , поэтому для ее графического представления требуется многомерная система координат, которую мы в нашем трехмерном мире представить себе не можем. В этой многомерной системе координат функция ε=ε (w_{ij}) изображается в виде многомерной поверхности, называемой *гиперповерхностью*.

Чтобы хоть как-то представить себе гиперповерхность, предположим, что все аргументы w_{ij} имеют постоянные значения за исключением двух, например w_{11} и w_{12}, которые являются переменными. Тогда в трехмерной системе координат w_{11} и w_{12}, ε гиперповерхность будет иметь вид фигуры, напоминающей параболоид, которую назовем псевдопарабалоидом (рис. 1.13). Процесс обучения персептрона теперь можно представить как отыскание такого сочетания весовых коэффициентов w_{ij}, которому соответствует самая нижняя точка гиперпсевдопараболоида. Задачи подобного рода называются *оптимизационными*. Говорят, что оптимизационная задача состоит в

минимизации функции $\varepsilon=\varepsilon(w_{ij})$ в многомерном пространстве параметров $\varepsilon=\varepsilon(w_{ij})$.

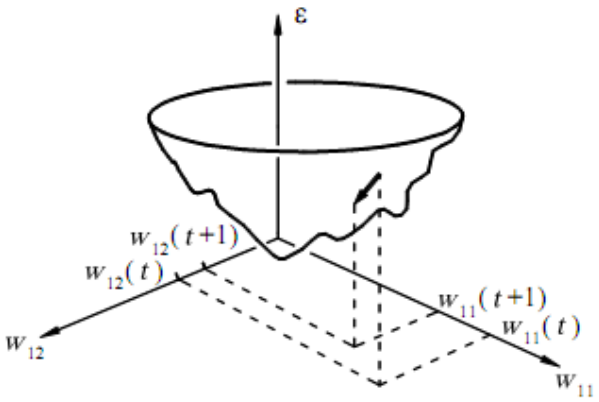


Рис. 1.13. Графическое изображение функции-ошибки персептрона $\varepsilon=\varepsilon(w_{ij})$

Таким образом, если раньше говорили, что персептрон обучают методом «поощрения – наказания», то теперь стали говорить, что задача обучения персептрона – это задача оптимизации (минимизации) персептронной ошибки (погрешности).

Существует множество методов решения оптимизационных задач. Наиболее простым методом является перебор весовых коэффициентов w_{ij} с последующими вычислениями и сравнениями между собой соответствующих этим коэффициентам значений функции ε. Более эффективен *метод градиентного спуска*, согласно которому изменение (коррекция) каждого весового коэффициента Δw_{ij} производится в сторону, противоположную градиенту функции ε. Градиент функции является очень важным математическим понятием. Здесь мы не будем на нем останавливаться, а только укажем, что градиент функции $\varepsilon=\varepsilon(w_{ij})$ представляет собой вектор, проекциями которого на оси координат являются производные от функции ε по этим координатам (их обозначают $\partial\varepsilon/\partial w_{ij}$), и что градиент функции всегда направлен в сторону ее наибольшего возрастания. Поскольку наша задача состоит в отыскании минимума функции $\varepsilon=\varepsilon(w_{ij})$, то нам надо опускаться по поверхности ошибок, что обеспечивается движением в сторону, противоположную градиенту этой функции. Отсюда и упомянутое выше название – метод градиентного спуска.

Движение в сторону, противоположную градиенту (т.е. противоположную направлению возрастания функции), будет осуществляться, если на каждой итерации к координатам текущей точки w_{ij} мы будем добавлять величину, прямо пропорциональную производной по координате w_{ij}, взятую с противоположным знаком:

$$\Delta w_{ij} = -\eta \frac{\partial \varepsilon}{\partial w_{ij}}, \qquad (1.16)$$

где η – некоторый коэффициент, обычно задаваемый в пределах от 0,05 до 1, и называемый, как и раньше, *коэффициентом скорости обучения*.

Обратите внимание, что согласно формуле (1.3) мы движемся не только в сторону убывания функции, но и со скоростью, прямо пропорциональной скорости убывания (крутизне) функции, т.к. делаем шаг Δw_{ij}, пропорциональный производной, взятой со знаком минус.

Квадратичная ошибка ε является сложной функцией, зависящей от выходных сигналов персептрона y_i, которые, в свою очередь, зависят от w_{ij} , т.е. $\varepsilon=\varepsilon(y_i(w_{ij}))$. По правилу дифференцирования сложной функции

$$\frac{\partial \varepsilon}{\partial w_{ij}} = \frac{\partial \varepsilon}{\partial y_i}\frac{\partial y_i}{\partial w_{ij}}. \qquad (1.17)$$

Выходные сигналы нейронов y_i вычисляются с помощью сигмоидных активационных функций $y_i=f_\sigma(S_i)$, аргументом которых являются суммы

$$S_i = \sum_{j=1}^{J} w_{ij} x_j.$$

Следовательно,

$$\frac{\partial y_i}{\partial w_{ij}} = \frac{\partial f_в(S_i)}{\partial S_i}\frac{\partial S_i}{\partial w_{ij}} = f_\sigma'(S_i)x_j. \qquad (1.18)$$

Кроме того, если продифференцировать (1.2) по y_n , где $n\in[1,I]$, то

$$\frac{\partial \varepsilon}{\partial y_n} = -(d_n - y_n),$$

значит

$$\frac{\partial \varepsilon}{\partial y_i} = -(d_i - y_i). \qquad (1.19)$$

Подставив (1.18) и (1.19) в (1.17) и затем полученное выражение в (1.16), окончательно будем иметь

$$\Delta w_{ij} = -\eta\left(-(d_i - y_i) f_\sigma'(S_i)x_j\right) = \eta(d_i - y_i) f_\sigma'(S_i)x_j. \qquad (1.20)$$

Это выражение получено для нейронов с активационными функциями любого вида. Если $f_\sigma(S_i)$ – сигмоида, заданная формулой (4.14), то

$$f_\sigma'(S_i) = \left(\left(1+e^{-S_i}\right)^{-1}\right)' = f_\sigma(S_i)\left(1 - f(S_i)\right). \qquad (1.21)$$

Подставив это выражение в (4.20), получим:

$$\Delta w_{ij} = \eta(d_i - y_i) f_\sigma(S_i)\left(1 - f(S_i)\right)x_j = \eta(d_i - y_i) y_i(1 - y_i)x_j. \qquad (1.22)$$

Итак, мы получили итерационную формулу для обучения персептрона

$$w_{ij}(t+1) = w_{ij}(t) + \Delta w_{ij}, \qquad (1.23)$$

где

$$\Delta w_{ij} = \eta \delta_i x_j, \qquad (1.24)$$

$$\delta_i = y_i(1 - y_i)(d_i - y_i). \qquad (1.25)$$

Введенную здесь с помощью формулы (1.25) величину δ_i в дальнейшем будем называть нейронной ошибкой. Алгоритм (1.23-1.25) называют обобщенным *дельта-правилом*. Его преимущество по сравнению с обычным дельта-правилом состоит в более быстрой сходимости и в возможности более точной обработки входных и выходных непрерывных сигналов, т.е. в расширении круга решаемых персептронами задач.

Итак, ведение сигмоидной функции активации вместо функции ступеньки и появление нового алгоритма обучения – обобщенного дельта-правила, расширило область применения персептрона. Теперь он может оперировать не только с бинарными (типа «ноль» и «единица»), но и с непрерывными (аналоговыми) выходными сигналами.

Эти работы отличаются от предыдущей тем, что студенты обучают персептрон распознавать буквы русского алфавита.

Выполняя пункты «Протокола выполнения» студенты убеждаются, что персептрон может не только распознавать буквы, которые были в обучающем множестве примеров, но и буквы, которые персептрон «увидел» впервые.

Студенты должны выполнять задание до тех пор, пока в «Протоколе выполнения» не появится сообщение «Работа выполнена успешно».

После выполнения исследований необходимо оформить отчет (форма выложена в личном кабинете студента), защитить практическую работу у преподавателя.

Практическая работа №5
Двухслойный персептрон

Теоретический материал к практической работе №5
Время выполнения работы 1 час (45 мин.)

1. Ограниченность однослойного персептрона. Как уже отмечалось ранее, Ф. Розенблатту удалось обучить свой персептрон распознавать буквы алфавита. Это был колоссальный успех: Электронное устройство, созданное по образу и подобию человеческого мозга, обученное подобно человеку, успешно моделировало интеллектуальные функции человека. Это был успех в познании самой природы человеческого мышления. Мозг начал раскрывать свои тайны. Появилась возможность исследовать мозг методами моделирования, не прибегая к сложнейшим антигуманным и мало что дающим натурным экспериментам. Это была сенсация, приковавшая к себе внимание мыслящих людей всего мира. Казалось, что ключ к интеллекту был найден и полное воспроизведение человеческого мозга и всех его функций – всего лишь вопрос времени. Писателям-фантастам, ученым, инженерам, бизнесменам, политикам виделись самые радужные перспективы практического применения идей искусственного интеллекта.

Правительство Соединенных Штатов Америки выделило крупные субсидии на развитие нового перспективного научного направления. Класс решаемых нейросетями задач расширялся. Но по мере расширения фронта научных исследований появлялись трудности. Неожиданно оказалось, что многие новые задачи персептрон решить не мог. Причем эти новые задачи, внешне ничем не отличались от тех, с которыми персептрон успешно справлялся ранее. Возникла необходимость объяснения парадоксов, глубокого анализа и создания теоретической базы нейроинформатики.

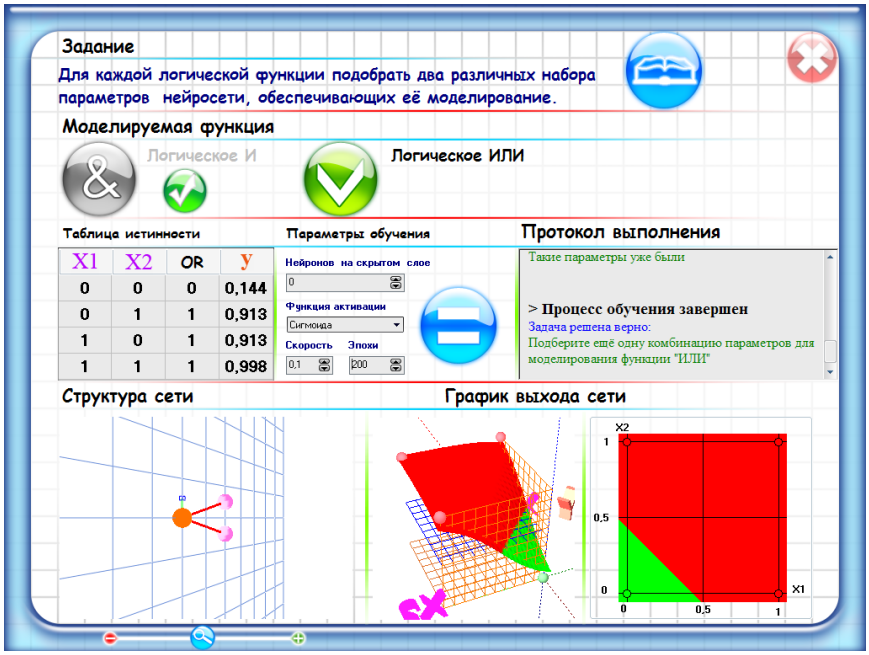
Следующий период истории искусственного интеллекта начался с появления в 1969 г. книги двух известных американских математиков М. Минского и С. Пайперт а «Персептроны». Авторы этой книги математически строго доказали, что использовавшиеся в то время однослойные персептроны в принципе не способны решать многие простые задачи. Одну из таких задач, вошедшую в историю нейроинформатики под названием проблемы «Исключающего ИЛИ», мы рассмотрим подробно.

«Исключающее ИЛИ» – это логическая функция двух аргументов, каждый из которых может иметь значение «истинно» либо «ложно». Сама она принимает значение «истинно», когда только один из аргументов имеет значение «истинно». Во всех остальных случаях эта функция принимает значение «ложно». Если закодировать значение «истинно» единицей, а значение «ложно» – нулем, то требуемое соответствие между аргументами x_1 , и x_2 самой функцией y можно представить в виде табл. 4.1, называемой таблицей истинности логической функции.

Таблица 1.1

x_1	x_2	y
0	0	0
0	1	1
1	0	1
1	1	0

Задача состоит в том, чтобы научиться моделировать функцию «Исключающее ИЛИ» с помощью однонейронного персептрона с двумя входами x_1 , и x_2 и одним выходом y (рис. 1.14). При выполнении практической работы №1 Вы уже пытались решить эту задачу путем подбора значений синаптических весов w_1, w_2 и порога θ, однако сделать это Вам не удалось. Вам не удалось это сделать, хотя в других случаях, при моделировании логических функций «И» и «ИЛИ», у Вас проблем не возникало.



Внешне функции «И», «ИЛИ» и «Исключающее ИЛИ» мало чем отличаются друг от друга, и Вам было не понятно, почему Ваш однонейронный персептрон успешно справлялся с моделированием двух первых функций, а с моделированием третьей функции он справиться не мог.

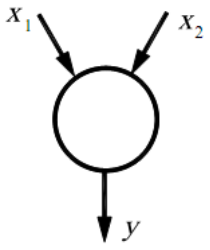


Рис. 1.14. Однонейронный персептрон с двумя входами и одним выходом.

Для объяснения этого парадокса американскими математиками М.Минским и С. Пайпертом была предложена геометрическая интерпретация, состоящая в следующем. Они предложили изобразить на координатной плоскости x_1, и x_2 все возможные комбинации входных сигналов в виде четырех точек: A, B, C, D, как показано на рис. 1.15. Точка A имеет координаты x_1=0, x_2 = 1; точка B имеет координаты x_1 = 0 , x_2 =1 и т.д. согласно таб. 1.2.

Таблица 1.2- Таблица истинности логической функции «Исключающее ИЛИ», дополненная точками A, B, C, D.

Точки	x_1	x_2	x_3
A	0	0	0
B	0	1	1
C	1	0	1
D	1	1	0

Тогда в точке A выход персептрона y должен быть равен нулю, в точке B – единице, в точке C – единице и в точке D – нулю.

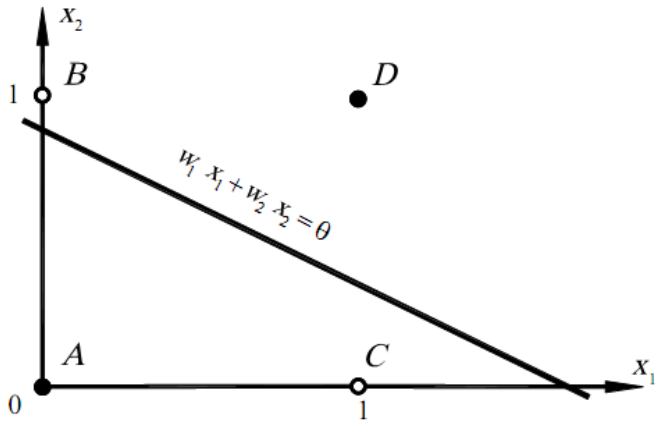


Рис. 1.15. Графическая интерпретация к объяснению проблемы «Исключающего ИЛИ»

Как нам известно (см. например практическую работу №1), математический нейрон Мак-Каллока – Питтса, изображенный на рис. 1.26, осуществляет преобразование

$$S = w_1 \cdot x_1 + w_2 \cdot x_2; \tag{1.26}$$

$$y = \begin{cases} 1, \text{ если } S \geq \theta; \\ 0, \text{ если } S < \theta. \end{cases} \tag{1.27}$$

Заменим в уравнении (1.26) S на θ :

$$w_1 \cdot x_1 + w_2 \cdot x_2 = \theta.. \tag{1.28}$$

Если в этом уравнении величины x_1 , и x_2 считать переменными, а θ , w_1 и $_2$ – константами, то на координатной плоскости x_1, x_2 рассматриваемое уравнение изобразится в виде прямой линии, положение и наклон которой определяются

значениями коэффициентов w_1 и w_2, и порога θ . Для всех точек плоскости x_1, x_2, лежащих на этой линии, выполняется равенство $\theta = S$ и поэтому, согласно формуле (1.26), выход персептрона равен единице.

Для точек, лежащих выше указанной линии сумма $w_1 x_1 + w_2 x_2$ больше чем θ , и поэтому по формулам (1.26)-(1.27) выход персептрона также равен единице, а для точек, лежащих ниже этой линии, сумма $w_1 x_1 + w_2 x_2$ меньше чем θ , и выход персептрона равен нулю. Поэтому линию, изображающую уравнение (1.28), называют *пороговой прямой*.

А теперь посмотрим на табл. 1.2. Согласно этой таблице в точках A и D выход персептрона должен быть нулевым, а в точках B и C – единичным. Но для этого надо расположить пороговую прямую так, чтобы точки A и D лежали ниже этой линии, а точки B и C – выше, что невозможно. Это значит, что, сколько бы персептрон ни обучали, какие бы значения ни придавали его синаптическим весам и порогу, персептрон в принципе не способен воспроизвести соотношение между входами и выходом, требуемое таблицей истинности функции «Исключающее ИЛИ».

Помимо проблемы «Исключающего ИЛИ» М.Минский и С.Пайперт привели ряд других задач, в которых точки, изображающие входные сигналы, не могут быть разделены пороговой прямой (в многомерных случаях – плоскостью, гиперплоскостью). Такие задачи получили название *линейно неразделимых*.

После выхода в свет книги М.Минского и С.Пайперта «Персептроны» всем стало ясно, что активно предпринимавшиеся в то время попытки обучать персептроны решению многих задач, которые, как оказалось, относятся к классу линейно неразделимых, с самого начала были обречены на провал.

Это была пустая трата времени, сил и финансовых ресурсов.

Выводы

Однонейронный персептрон в принципе не позволяет моделировать логическую функцию «Исключающее ИЛИ» и решать другие линейно неразделимые задачи.

2. Решение проблемы «Исключающего ИЛИ» Появление книги М. Минского и С.Пайперта «Персептроны» вызвало шок в научном мире. Строгие математические доказательства М.Минского и С.Пайперта были неуязвимы. Всеобщий энтузиазм сменился не менее всеобщим пессимизмом. В газетах стали появляться критические статьи с сообщениями о том, что ученые мужи в своих исследованиях зашли в тупик, впустую израсходовав огромные государственные деньги. Правительство США немедленно прекратило финансирование нейропроектов и приступило к поискам виновных. Бизнесмены, потерявшие надежду вернуть вложенные капиталы, отвернулись от ученых и нейроинформатика была предана забвению, длившемуся более 20 лет.

Тем не менее, работы в области нейросетевых и нейрокомпьютерных технологий продолжались отдельными энтузиастами. Работы продолжались в засекреченных научно-исследовательских институтах Советского Союза, отделенного в то время от Запада «железным занавесом». Не имея информации

о настроениях зарубежных коллег, советские ученые спокойно продолжали заниматься захватившей их умы темой и к началу 80-х гг. удивили мир ракетами и самолетами, управлявшимися компьютерами нового поколения – нейрокомпьютерами. Советские компьютеры, в отличие от американских, стойко переносили довольно серьезные повреждения, продолжая работать в сложных условиях, что было особенно важно для объектов военного назначения. Выявилось еще одно свойство нейрокомпьюторов, унаследованное от мозга – свойство живучести.

Советским ученым С.О. Мкртчаном было показано, что с помощью многослойных персептронов может быть смоделирована любая логическая функция, если только известна ее логическая формула. Более того, им был разработан специальный математический аппарат, позволяющий конструировать такие персептроны. Оказалось, что проблема «Исключающего ИЛИ», явившаяся камнем преткновения для однонейронного персептрона, может быть разрешена с помощью нейронной сети, состоящей из трех нейронов – технейронного персептрона, изображенного на рис. 1.16.

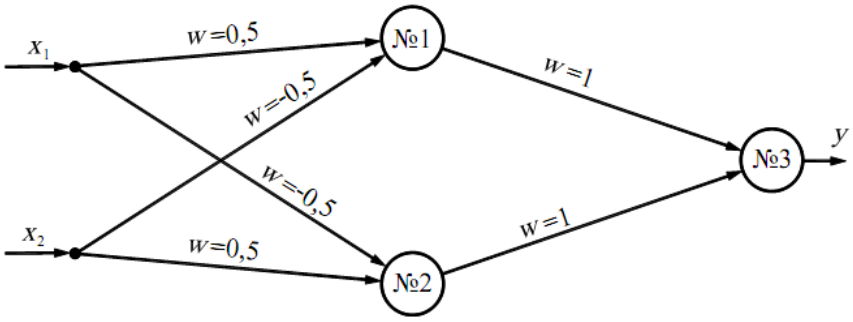


Рис. 1.16. Нейронная сеть, моделирующая функцию «Исключающее ИЛИ»

Работа этого персептрона происходит по следующему алгоритму.

Нейрон № 1:

$$S_1 = 0{,}5 \cdot x_1 + (-0{,}5) \cdot x_2;$$

$$y_1 = \begin{cases} 1, \text{ если } S_1 \geq \theta; \\ 0, \text{ если } S_1 < \theta. \end{cases}$$

Нейрон № 2:

$$S_2 = (-0{,}5) \cdot x_1 + 0{,}5 \cdot x_2;$$

$$y_2 = \begin{cases} 1, \text{ если } S_2 \geq \theta; \\ 0, \text{ если } S_2 < \theta. \end{cases}$$

Нейрон № 3:

$$S_3 = 1 \cdot y_1 + 1 \cdot y_2;$$

$$y_3 = \begin{cases} 1, \text{ если } S_3 \geq \theta; \\ 0, \text{ если } S_3 < \theta. \end{cases}$$

Задавшись значением порога θ =0,5 и заполнив с помощью этих формул табл. 1.3, легко убедиться, что трехнейронный персептрон успешно моделирует функцию «Исключающее ИЛИ».

Таблица 1.3 - Процесс формирования сигналов в трехнейронном персептроне

x_1	x_2	S_1	S_2	y_1	y_2	S_3	y_3	y
0	0	0	0	0	0	0	0	0
0	1	-0.5	0.5	0	1	1	1	1
1	0	0.5	-0.5	1	0	1	1	1
1	1	0	0	0	0	0	0	0

Впоследствии было показано, что и другие линейно неразделимые задачи, приведенные в книге М.Минского и С.Пайперта, могут быть решены с помощью нейросетей, содержащих один или несколько скрытых нейронных слоев, т.е. слоев нейронов, расположенных между входным и выходным слоями.

Многие исследователи понимали, что нужно создавать нейросети более сложной архитектуры, содержащие скрытые слои нейронов, но не представляли, как такие сети обучать. Правила Хебба и их обобщение – дельта-правило, годились только для корректировки синаптических весов нейронов выходного слоя, тогда как вопрос о настройке параметров скрытых нейронных слоев оставался открытым.

Вывод

Логическую функцию «Исключающее ИЛИ» может моделировать нейронная сеть, состоящая из трех нейронов, изображенная на рис. 1.16.

3. Алгоритм обратного распространения ошибки

Эффективный алгоритм обучения многослойных персептронов, открывший путь их широкому практическому применению, стал известен только в 1986 г. благодаря публикациям Д.Румельхарта, Г.Хилтона и Р.Вильямса. Идея этого алгоритма заключается в том, что ошибки нейронов выходного слоя $\varepsilon_i = d_i - y_i$ используются для вычисления ошибок нейронов, расположенных в скрытых слоях. Значения ошибок как бы распространяются от выходного слоя нейронов вовнутрь сети от последующих нейронных слоев к предыдущим. Отсюда название метода алгоритмом *обратного распространения ошибки* (back propagation).

Интересно отметить, что алгоритм обратного распространения ошибки был предложен на один год ранее в работах А.Паркера и А.Ле-Кана, изданных независимо одна от другой. Более того, еще в 1974 г. этот простой и изящный алгоритм был защищен П. Вербосом в его докторской диссертации.

Однако тогда он остался незамеченным, и только спустя более десяти лет был «переоткрыт» заново и получил всеобщее признание и применение. Остались незамеченными и работы советских ученых, еще раньше разрабатывавших подобные алгоритмы в своих засекреченных институтах и успешно применявших их при построении систем управления объектами военного назначения.

Рассмотрим идею алгоритма обратного распространения ошибки, попытавшись обобщить дельта-правило на случай обучения двухслойного персептрона, имеющего N входов, I выходов и скрытый слой из J нейронов (рис. 1.17). Этот персептрон на самом деле имеет три слоя, однако в литературе его называют двухслойным, поскольку нейроны входного слоя имеют всего один вход, не имеют синаптических весов и не выполняют суммирования входных сигналов, а лишь передают один единственный входной сигнал нейронам следующего слоя.

Алгоритм корректировки синаптических весов нейронов выходного слоя оставим таким же, как для однослойного персептрона (см. обобщенное дельта-правило: теоретический материал к практической работе №4), заменив x_i на y_i:

$$w_{ij}(t+1) = w_{ij}(t) + \Delta w_{ij}; \qquad (1.29)$$

$$\Delta w_{ij} = \eta \, \delta_i \, y_j; \qquad (1.30)$$

$$\delta_i = y_i (1 - y_i)(d_i - y_i). \qquad (1.31)$$

Синаптические веса нейронов скрытого слоя попытаемся корректировать с помощью все тех же формул, в которых индекс i заменим на j, а индекс j заменим на индекс n:

$$\Delta w_{in} = \eta \, \delta_i \, y_n; \qquad (1.32)$$

$$\delta_j = y_j (1 - y_j)(d_j - y_j). \qquad (1.33)$$

При использовании этих формул возникает вопрос о вычислении нейронной ошибки $(d_j - y_j)$, которая для скрытого слоя неизвестна. Идея авторов рассматриваемого алгоритма состояла в том, чтобы в качестве этой ошибки использовать суммарные нейронные ошибки с выходного слоя, помноженные на силы соответствующих синаптических связей, т.е.

$$\left(d_j - y_j\right) = \sum_{i=1}^{I} \delta_i w_{ij}. \qquad (1.34)$$

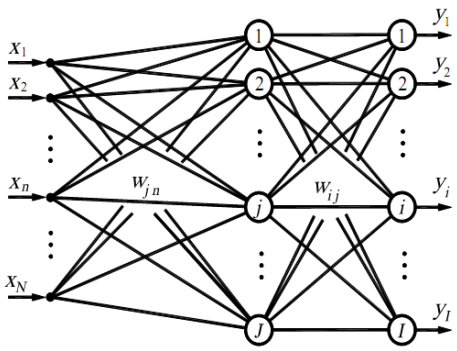


Рис. 1.17. Двухслойный персептрон с N входами, I выходами и скрытым слоем из J нейронов

Итак, для скрытого слоя окончательно имеем

$$\Delta w_{in} = \eta \, \delta_i \, x_n; \qquad (1.35)$$

$$\delta_j = y_j\left(1-y_j\right)\sum_{i=1}^{I}\delta_i w_{ij}. \qquad (1.36)$$

Используя эту идею, несложно расписать алгоритм обратного распространения ошибки для обучения персептрона, имеющего произвольное количество скрытых слоев. Однако прежде отметим, что мы будем использовать нейроны, имеющие сигмоидную активационную функцию (см. теоретический материал к практической работе №4), и выполняющие операцию суммирования с учетом нейронного смещения (см. теоретический материал к практической работе №1):

$$S_i = \sum_{j=1}^{J} w_{ij}x_j. \qquad (1.37)$$

Здесь w_{i0} – вес дополнительного входа i -го нейрона, имитирующий его смещение b_i , а $x_0 = 1$ – величина сигнала дополнительного входа.

Алгоритм обратного распространения ошибки распишем для многослойного персептрона, имеющего входной слой k =0, несколько скрытых слоев k= 1 ..., , 2, $K-1$ и выходной слой k =K (рис. 1.18).

Нейроны входного слоя не выполняют математических преобразований, а лишь передают входные сигналы нейронам первого слоя. Будем полагать, что каждый k -й слой содержит H_k нейронов. Таким образом, персептрон имеет $N = H_0$ входов и $M = H_K$ выходов. В алгоритме будем использовать следующие обозначения: i – порядковый номер нейрона k -го слоя; j – порядковый номер нейрона $(k-1)$-го слоя; l – порядковый номер нейрона) $(k+1)$- -го слоя (см. рис. 1.18, внизу).

Шаг 1. Инициализация синаптических весов и смещений.

В циклах по $k=1,2,...,K$; $i=1,2,...,H_k$; $j=0,1,2...\ H_{k-1}$ синаптическим весам и смещениям $\omega_{ij}^{(k)}$ датчиком случайных чисел присваиваются малые величины, например, из интервала от –1 до 1.

Шаг 2. Открытие цикла по q = 1, 2, ..., Q. Представление из обучающего множества примеров очередного входного вектора

$$X_q = \left(x_1, x_2, ..., x_N\right)_q$$

и соответствующего ему желаемого выходного вектора

$$D_q = \left(d_1, d_2, ..., d_M\right)_q.$$

где q – номер примера в обучающем множестве.

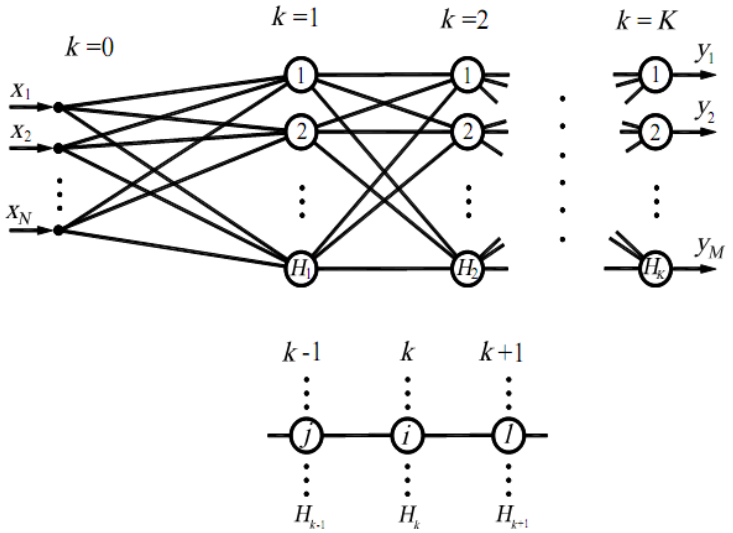


Рис. 1.18. Многослойный персептрон (MLP – MultiLayerPerseptron)

Шаг 3. Прямой проход.

В циклах по $k=1,2,...,K;\ i=1,2,...,H_k;$ вычисляются выходные сигналы i -го нейрона в k -м слое

$$y_i^{(k)} = f_\sigma\left(\sum_{j=0}^{H_{k-1}} w_{ij}^{(k)} y_j^{(k-1)}\right), \qquad (1.38)$$

где

$$y_j^{(0)} = x_j; x_0 = 1; y_0^{(k-1)} = 1$$

выходные сигналы персептрона

$$y_i = y_i^{(K)}.$$

Шаг 4. Обратный проход.

В циклах по $k=1,2,...,K;\ i=1,2,...,H_k;\ j=0,1,2...\ H_{k-1}$ вычисляются синаптические веса на новой эпохе

$$w_{ij}^{(k)}(t+1) = w_{ij}^{(k)}(t) + \Delta w_{ij}^{(k)}, \qquad (1.39)$$

где

$$\Delta w_{ij}^{(k)} = \eta \delta_i^{(k)} y_j^{(k-1)}, \qquad (1.40)$$

причем для выходного слоя $K= k$ согласно (4.33)

$$\delta_i^{(K)} = y_i\left(1 - y_i\right)\left(d_i - y_i\right),$$

а для всех других скрытых слоев согласно (4.36)

$$\delta_i^{(k)} = y_i^{(k)}\left(1 - y_i^{(k)}\right)\sum_{l=1}^{H_{k+1}} \delta_l^{(k+1)} w_{li}^{(k+1)}.$$

Шаг 5. Закрытие цикла по q.

Шаг 6. Повторение шагов 2 – 5 необходимое количество раз.

Векторы обучающих примеров X_q и D_q на шаге 2 алгоритма обычно представляются последовательно от первого до последнего, т.е. q = 1, 2, ...,Q, где Q – общее количество примеров. Например, в случае распознавания букв

русского алфавита 33 = Q. После того, как для каждого обучающего примера будут скорректированы весовые коэффициенты персептрона, т.е. шаги 2–4 будут повторены 33 раза, на шаге 6 алгоритма вычисляется среднеквадратичная ошибка, усредненная по всем обучающим примерам:

$$\varepsilon = \frac{1}{QM}\sum_{q=1}^{Q}\sum_{i=1}^{M}\left(\left(d_i - y_i\right)_q^2\right). \qquad (1.41)$$

Помимо среднеквадратичной ошибки может быть также оценена максимальная разность между желаемым и прогнозным (то, что вычислил персептрон) выходами персептрона:

$$\varepsilon = \max\left(\left(\left|d_i - y_i\right|\right)_q\right); \quad i = 1,2,...,M; q = 1,2,...,Q. \qquad (1.42)$$

Итерационный процесс, задаваемый шагом 6, заканчивается после того, как ошибка ε, вычисляемая по формулам (1.41) или (1.42), достигнет заданной величины, либо когда будет достигнуто предельное количество эпох обучения. В результате персептрон обучится выполнять нужное отображение любого входного вектора X_q на выходной вектор Y_q , отличающийся от желаемого вектора D_q на некоторую малую величину.

Вывод

Первым алгоритмом обучения нейронной сети были правила Хебба, предназначенные для обучения однослойного персептрона с нейронами, имеющими ступенчатые активационные функции. Затем было введено понятие нейронной ошибки как разницы между требуемым выходом нейрона d_i и его реальным значением y_i. В результате алгоритм обучения персептрона с помощью правил Хебба был обобщен в виде алгоритма дельта-правила. В итерационных формулах алгоритма дельта-правила появился коэффициент скорости обучения η, позволяющий влиять на величину итерационного шага.

Затем была предложена сигмоидная активационная функция и было введено понятие квадратичной ошибки обучения персептрона. В результате появилось обобщенное дельта-правило, реализующее метод градиентного спуска, и позволяющий работать не только с бинарными, но и с непрерывными сигналами. Алгоритм обратного распространения ошибки является следующим обобщением обобщенного дельта-правила и позволяет обучать не только однослойные, но и многослойные персептроны.

4. Виды активационных функций

В современных нейронных сетях и нейропакетах наиболее часто применяются следующие виды активационных функций.

Пороговые активационные функции (функции-ступеньки). Пороговые активационные функции-ступеньки могут иметь как несимметричный (рис. 1.6, *а*), так и симметричный (рис. 1.19, *б*) относительно начала координат вид.

а) б)

Рис. 1.19. Пороговые активационные функции-ступеньки

Их аналитическое представление соответственно для рис. 1.6,*а* и рис. 1.6,*б*:

$$y = \begin{cases} 1, \text{если } S \geq 0; \\ 0, \text{если } S < 0; \end{cases} \qquad y = \begin{cases} 1, \text{если } S \geq 0; \\ -1, \text{если } S < 0, \end{cases}$$

где

$$S = \sum_{i=0}^{I} x_i w_i,$$

причем $x_0 = 1$– величина сигнала дополнительного входа, а w_0– его вес, имитирующий нейронное смещение b (которое равно порогу чувствительности нейрона, взятому с противоположным знаком*:* $b = -\theta$).

Ступенчатые активационные функции обычно используются в задачах распознавания образов. Персептроны со ступенчатыми активационными функциями могут обучаться с помощью правил Хебба и дельта-правила.

Обобщенное дельта правило и алгоритм обратного распространения ошибки для обучения таких персептронов не годятся, так как в эти алгоритмы включают нахождение производных от активационных функций, что невозможно для функций, имеющих разрыв.

Линейные активационные функции. На рис. 1.20,а представлен график линейной активационной функции $y = S$.

Область изменения этой функции неограниченна.

Такие активационные функции обычно применяются в нейронах входного слоя. Они также неплохо работают при решении простых линейно разделимых задач, причем с обучением таких персептронов могут справляться алгоритмы дельта-правила, обобщенного дельта-правила и алгоритм обратного распространения ошибки.

Иногда применяют линейные активационные функции с ограниченной областью изменения (рис. 1.20, б):

$$y = \begin{cases} -1, \text{если } S < -1 \\ S, \text{ если } -1 \leq S \leq 1 \\ 1, \text{ если } S > 1 \end{cases}$$

а) б)

Рис. 1.20. Линейные активационные функции с неограниченной (*а*) и ограниченной (*б*) областями изменения

Сигмоидные активационные функции. На рис. 1.21,а изображен график сигмоидной функции, заданной уравнением

$$y = \frac{1}{1+e^{-\alpha S}},$$

а на рис. 1.21,*б* – график функции, заданной уравнением

$$y = \frac{1-e^{-\alpha S}}{1+e^{-\alpha S}}.$$

В этих уравнениях коэффициент α влияет на угол наклона линий к оси *S*.

Аналогичный представленному на последнем графике вид имеют функции арктангенса

$$y = \frac{2}{\pi} arctg \alpha S$$

и гиперболического тангенса

$$y = \tan \alpha S,$$

а также функция

$$y = \frac{\alpha S}{1+|\alpha S|},$$

которые тоже называют сигмоидными.

а) б)

Рис. 1.21. Сигмоидные активационные функции с несимметричной (*а*) и симметричной (*б*) областями изменения

Персептроны с сигмоидными активационными функциями хорошо обучаются с помощью алгоритма обратного распространения ошибки, а также с помощью дельта правила и обобщенного дельта-правила (если персептрон однослойный).

Логарифмические активационные функции.

На рис. 1.22 представлен график активационной функции, заданной уравнением

$$y = \ln\left(S + \sqrt{S^2 + 1}\right). \qquad (1.43)$$

В отличие от сигмоидной эта функция имеет неограниченную область изменения. Иногда это удобно, т.к. не требуется масштабирования выходных сигналов персептрона (подробнее об этом см. теоретический материал к практической работе №8). Кроме того, логарифмические активационные функции позволяют избегать нежелательного эффекта, называемого *параличем сети* – потерей чувствительности сети к вариациям весовых коэффициентов и, как следствие, замирание процесса обучения при попадании взвешенных сумм входных сигналов нейрона в область насыщения сигмоиды (см. также практическую работу №8).

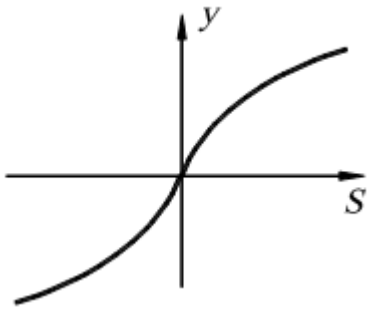


Рис. 1.22. Логарифмическая активационная функция

Радиально-базисные активационные функции. В последнее время получают распространение нейросети, нейроны которых имеют активационные функции в форме функции Гаусса (см. рис. 1.23):

$$y = e^{-\frac{S^2}{2\sigma^2}},$$

где − S= ||X – C|| евклидово расстояние между входным вектором X и центром активационной функции C; σ – параметр гауссовой кривой, называемый шириной окна. Такие активационные функции называют радиальнобазисными (RBF), а соответствующие нейронные сети – RBF-сетями. Методы обучения RBF-сетей, их преимущества и недостатки в нашем учебном курсе не рассматриваются.

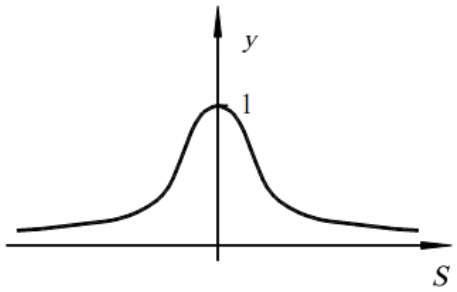


Рис. 1.23. Радиально-базисная активационная функция

Отметим, что все приведенные выше активационные функции, за исключением пороговой и кусочно-линейной, являются непрерывно дифференцируемыми. Линейная функция и логарифмическая функция выполняют преобразование бесконечного входного множества значений переменных S в бесконечное множество переменных y. Пороговые активационные функции преобразуют множество значений S в бинарные множества $y = 0$ и $y = 1$ или $y = -1$ и $y = 1$. Остальные активационные функции преобразуют бесконечное входное множество значений S в ограниченные выходные множества:), y∈ (0, 1), y∈ (-1, 1) и y∈ (0, 1]. От вида используемых активационных функций зависят функциональные возможности нейронных сетей, а также выбор способов их обучения.

Вывод

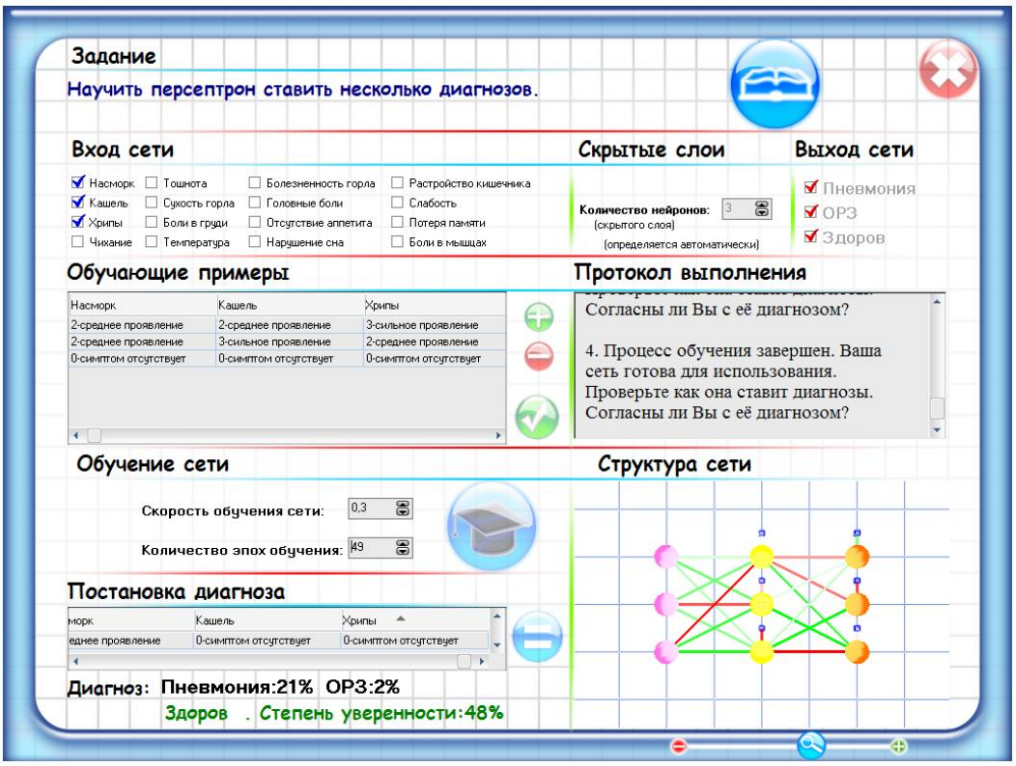
Активационные функции осуществляют преобразование взвешенной суммы входных сигналов нейрона в его выходной сигнал. От вида активационных функций зависят функциональные возможности нейронных сетей, а также выбор способов их обучения.

Запустите программу и приступайте к исследованиям.

Эта работа, по существу, является продолжением первой практической работы. Задание состоит в моделировании логических функций, но только в место одного нейрона используется двухслойный персептрон. Выполняя задание, обучающиеся задают структуру персептрона – количество нейронов на скрытом слое, вид активационных функций, скорость обучения и количество эпох обучения.

Практические работы № 6,7

Время выполнения работы 2 часа (90 мин.).



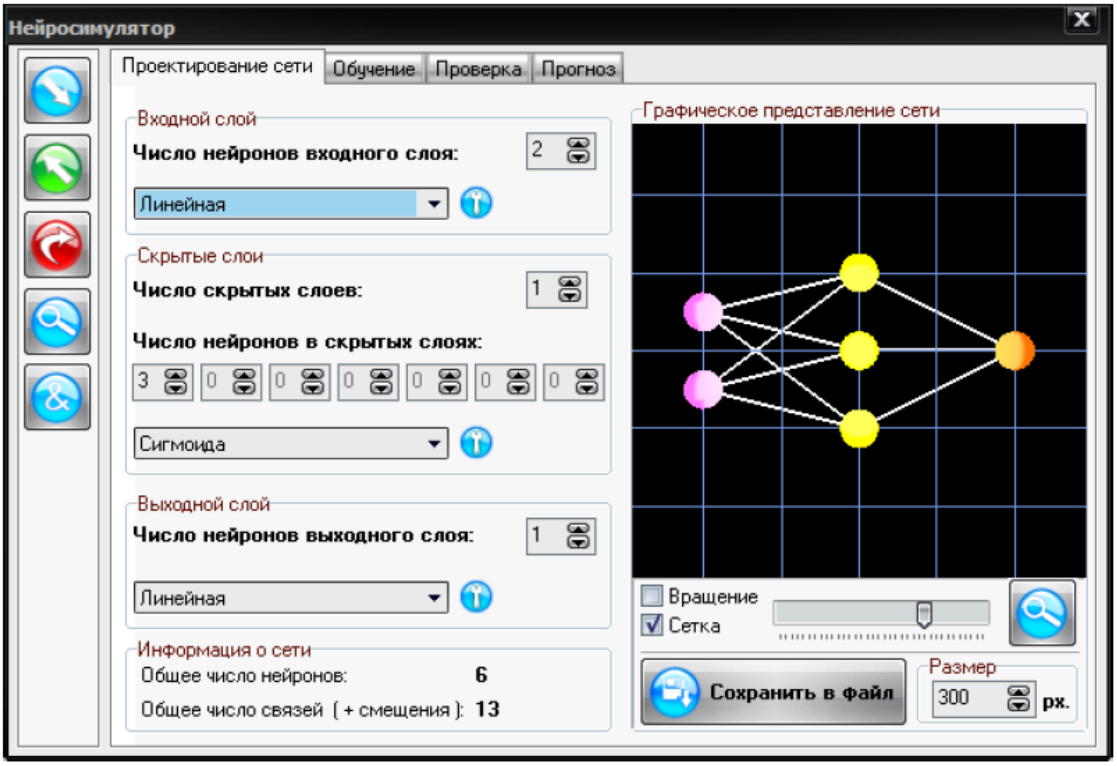
Обучающиеся, исходя из собственных медицинских знаний, обучают нейронную сеть ставить такие диагнозы заболеваний, как грипп, пневмония, ОРЗ. Для этого они выбирают симптомы, которые считают важными для постановки диагноза, и формируют множество примеров постановки диагнозов пациентам, которых они сами придумывают.

После традиционного повторения теоретического материала и напоминания обучающимся, чтобы они внимательно читали и выполняли пункты «Протокола выполнения», преподаватель может подойти к справившемуся с работой студенты и проверить с помощью созданной обучающимся интеллектуальной диагностической системы состояние своего здоровья.

Практическая работа №8

Время выполнения работы 1 час (45 мин.)

Эта Практическая работа представляет собой нейропакет, т.е. программу, предназначенную для проектирования, обучения, тестирования и использования нейронных сетей.



В отличие от предыдущих лабораторных работ, освоение этой программы производится под руководством преподавателя согласно рекомендациям, данным в **Учебно-методическом комплексе «Искусственный интеллект и нейронные сети»**. Эту программу обучающиеся обычно скачивают на свои компьютеры и используют для выполнения самостоятельных (курсовых) работ. Круг решаемых с помощью нее задач чрезвычайно широк. Вот далеко не полный перечень интеллектуальных информационных систем, созданных с помощью этой программы:

1. Интеллектуальный детектор лжи.
2. Интеллектуальный антиспамер.
3. Интеллектуальная система диагностики сложных технических устройств.
4. Интеллектуальная система диагностики здоровья человека.
5. Интеллектуальная система распознавания автомобильных номерных знаков.
6. Интеллектуальная система распознавания криминальных ситуаций по данным видеонаблюдений.
7. Интеллектуальная система оценки жилой недвижимости.

8. Интеллектуальная система оценки стоимости подержанных автомобилей.

9. Интеллектуальная система прогнозирования курсов валют, котировок акций и ценных бумаг (с учетом влияния различных факторов).

10. Интеллектуальная система оценки банковских рисков.

11. Интеллектуальная система оценки кредитоспособности физических лиц.

12. Интеллектуальная система выявления клиентов-мошенников страховых компаний.

13. Интеллектуальная система оценки вероятности банкротств организаций.

14. Интеллектуальная система прогнозирования расхода зданиями тепловой и электрической энергии.

15. Интеллектуальная система прогнозирования индексов потребительских цен.

16. Интеллектуальная система прогнозирования результатов голосований.

17. Интеллектуальная система прогнозирования результатов выборов президента страны.

18. Интеллектуальная система прогнозирования результатов выборов в законодательное собрание области, края.

19. Интеллектуальная система оценки шансов поступления абитуриента в вуз.

20. Интеллектуальная система-советчик выбора профессии.

21. Интеллектуальная система-советчик выбора партнера супружеской пары.

22. Интеллектуальная система прогнозирования пола будущего ребенка.

23. Интеллектуальная система поддержки принятия решений руководителя.

24. Интеллектуальная система формирования коэффициентов исхода спортивных матчей (прогнозирование букмекерских коэффициентов).

25. Интеллектуальная система распознавания лиц.

26. Интеллектуальная система прогнозирования результатов автомобильных гонок, скачек и пр.

27. Интеллектуальная система прогнозирования вероятности дорожно-транспортных происшествий.

Понимание теории, умение пользоваться этим нейропакетом и создавать подобного рода интеллектуальные информационные системы как раз и является конечной целью изучения учебно-методического комплекса «Искусственный интеллект и нейронные сети».