Дьяконов В. П., Крутлов В. В.

«СОЛОН»

MATLAB 6.5 SP1/7/7 SP1/7 SP2 Simulink 5/6

Инструменты искусственного интеллекта и биоинформатики

Впервые описаны пакеты расширения:
- по нейронным сетям
- нечеткой логике
- генетическим алгоритмам
- биоинформатике

ОБШИРНАЯ ПОДБОРКА ПРИМЕРОВ

Библиотека Профессионала

УДК 621.396.218
ББК 32.884.1
Д 93

В. П. Дьяконов, В. В. Круглов

Д93 **MATLAB 6.5 SP1/7/7 SP1/7 SP2 + Simulink 5/6. Инструменты искусственного интеллекта и биоинформатики.** Серия «Библиотека профессионала». — М.: СОЛОН-ПРЕСС, 2010. — 456 с.: ил.

ISBN 5-98003-255-X

Пятая книга в серии книг, посвященных последним реализациям мощных матричных систем компьютерной математики MATLAB 6.5 SP1/7/7 SP1/7 SP2 + Simulink 5/6. Впервые дан вводный курс по новейшей версии MATLAB 7 SP 2 + Simulink 6. Детально описаны последние версии пакетов расширения по нейронным сетям и нечеткой логике. Впервые дано описание новейших пакетов расширения по генетическим алгоритмам и биоинформатике. Представлены инструментальные средства проектирования графического интерфейса пользователя, работы в Интернете и компиляции MATLAB-программ. Описано множество примеров применения этих средств. Книга предназначена для научных работников, инженеров, студентов, аспирантов и преподавателей университетов и вузов.

Сайт издательства «СОЛОН-ПРЕСС»: **www.solon-press.ru**
E-mail: **solon-avtor@coba.ru**

ISBN5-98003-255-X

Глава 2. Пакет расширения по нейронным сетям

В этой главе описан пакет расширения Neural Networks Toolbox (Нейронные сети), который может быть использован для решения множества разнообразных задач, таких как обработка сигналов, нелинейное управление, финансовое моделирование и т. п. Так же, как и для ряда других пакетов, описание данного пакета не преследует цели дать по нему исчерпывающую информацию — оно является лишь руководством пользователя и не заменяет обширный справочник по набору функций пакета, некоторые из которых описаны кратко. Для дополнительного знакомства с нейронными цепями можно рекомендовать книги [8, 48—54].

2.1. Введение в пакет Neural Networks Toolbox

2.1.1. Назначение пакета Neural Networks Toolbox

Пакет Neural Networks Toolbox содержит средства для проектирования, моделирования, обучения и использования множества известных парадигм современного аппарата искусственных нейронных сетей (ИНС): от базовых моделей персептрона до самых современных ассоциативных и самоорганизующихся сетей.

Для каждого типа архитектуры и обучающего алгоритма ИНС имеются функции инициализации, обучения, адаптации, создания, моделирования, демонстрации, а также примеры применения. Искусственные многослойные нейронные сети конструируются по принципам построения их биологических аналогов. Они уже сейчас способны решать широкий круг задач распознавания образов, идентификации, управления сложными нелинейными объектами, роботами и т. п. Отметим, что в настоящее время дальнейшее повышение производительности компьютеров связывают с ИНС, в частности, с так называемыми нейрокомпьютерами (НК), основу которых составляет искусственная нейронная сеть.

Пакет Neural Networks Toolbox может работать в среде системы MATLAB 7.0.1 (реализация 14 с Service pack 1 на CD) и более поздних систем. Совместно с Neural Networks Toolbox могут использоваться и другие пакеты расширения, в частности, пакет Simulink, в среде которого весьма удобно создавать, моделировать и исследовать различные нейросетевые системы управления (системы автоматического управления с так называемыми нейросетевыми регуляторами).

Последняя реализация пакета (версия 4.04) сохраняет все основные черты предыдущей реализации Neural Networks Toolbox 3.0, которая использовалась в версиях системы MATLAB 5.*, отличаясь лишь следующим:

1) в пакет добавлен графический интерфейс пользователя (GUI-интерфейс для Neural Networks Toolbox);
2) введены примеры (демоверсии) трех нейросетевых систем управления (в среде Simulink);
3) введено четыре новых функции обучения нейронных сетей, а две исключены из состава пакета (не рекомендованы к дальнейшему применению);
4) расширен перечень иллюстрирующих примеров;

5) устранен ряд недоделок и изменены — в сторону улучшения — отдельные функции, в частности, функция `newlind` (см. ниже, в параграфе 2.4.6), которая позволяет теперь создавать линейную НС со многими входами и выходами и т. д.

Пакет Neural Networks Toolbox имеет обширную справку и документацию в двух файлах формата PDF:

- Neural Networks Toolbox. Release Notes (12 p.);
- Neural Networks Toolbox. For Use with MATLAB. User's Guide (840 p.).

Таким образом, объем документации только в PDF-формате превышает 850 страниц. Практически он продублирован в справке по пакету. Приведенный ниже материал является обзорным описанием пакета, достаточным для начала работы с ним пользователя, не имеющего какой-либо специальной подготовки.

2.1.2. Основные области применения нейронных сетей

Представим некоторые проблемы, решаемые в контексте ИНС и представляющие интерес для пользователей.

Классификация образов. Задача состоит в указании принадлежности входного образа (например, речевого сигнала или рукописного символа), представленного вектором признаков, одному или нескольким предварительно определенным классам. К известным приложениям относятся распознавание букв, распознавание речи, классификация сигнала электрокардиограммы, классификация клеток крови.

Кластеризация/категоризация. При решении задачи кластеризации, которая известна также как классификация образов «без учителя» , отсутствует обучающая выборка с метками классов. Алгоритм кластеризации основан на подобии образов и помещает близкие образы в один кластер. Известны случаи применения кластеризации для извлечения знаний, сжатия данных и исследования свойств данных.

Аппроксимация функций. Предположим, что имеется обучающая выборка $((x^1,y^1), (x^2,y^2)$ ..., $(x^N,y^N))$ (пары данных вход-выход), которая генерируется неизвестной функцией $F(x)$, искаженной шумом. Задача аппроксимации состоит в нахождении оценки неизвестной функции $F(x)$. Аппроксимация функций необходима при решении многочисленных инженерных и научных задач моделирования.

Предсказание/прогноз. Пусть заданы n дискретных отсчетов $\{y(t_1), y(t_2)$ ..., $y(t_k)\}$ в последовательные моменты времени t_1, t_2, ..., t_k. Задача состоит в предсказании значения $y(t_{k+1})$ в некоторый будущий момент времени t_{k+1}. Предсказание/прогноз имеют значительное влияние на принятие решений в бизнесе, науке и технике. Предсказание цен на фондовой бирже и прогноз погоды являются типичными приложениями техники предсказания/прогноза.

Оптимизация. Многочисленные проблемы в математике, статистике, технике, науке, медицине и экономике могут рассматриваться как проблемы оптимизации. Задачей алгоритма оптимизации является нахождение такого решения, которое удовлетворяет системе ограничений и максимизирует или минимизирует целевую функцию.

Память, адресуемая по содержимому. Ассоциативная память, или память, адресуемая по содержимому, доступна по указанию заданного содержимого. Содержимое памяти может быть вызвано даже по частичному входу или искаженному содержимому. Ассоциативная память чрезвычайно желательна при создании мультимедийных информационных баз данных.

Управление. Рассмотрим динамическую систему, заданную совокупностью $\{u(t), y(t)\}$, где $u(t)$ является входным управляющим воздействием, а $y(t)$ — выхо-

дом системы в момент времени *t*. В системах управления с эталонной моделью целью управления является расчет такого входного воздействия $u(t)$, при котором система следует по желаемой траектории, диктуемой эталонной моделью. Примером является оптимальное управление двигателем.

Эти и подобные задачи успешно решаются средствами пакета Neural Networks, который обеспечивает довольно мощные инструментальные возможности по созданию нейронных сетей и их применению.

2.2. Краткие сведения об искусственных нейронных сетях

2.2.1. Появление искусственных нейронных сетей

Термин *нейронные сети* сформировался в 40-х годах XX века в среде исследователей, изучавших принципы организации и функционирования биологических нейронных сетей. Основные результаты, полученные в этой области, связаны с именами американских исследователей У. Маккалоха, Д. Хебба, Ф. Розенблатта, М. Минского, Дж. Хопфилда и других.

Под нейронными сетями подразумеваются вычислительные структуры, которые моделируют простые биологические процессы, обычно ассоциируемые с процессами человеческого мозга. Адаптируемые и обучаемые, они представляют собой распараллеленные системы, способные к обучению путем анализа положительных и отрицательных воздействий. Элементарным преобразователем в данных сетях является *искусственный нейрон* или просто *нейрон*, названный так по аналогии с биологическим прототипом.

К настоящему времени предложено и изучено большое количество моделей нейроподобных элементов и нейронных сетей, некоторые из которых рассмотрены ниже. Рассмотрение ограничено сетями, которые могут быть реализованы в пакете расширения Neural Networks Toolbox.

2.2.2. Структура искусственного нейрона

Структура искусственного нейрона показана на рис. 2.1.

В состав нейрона входят умножители (синапсы), сумматор и нелинейный преобразователь. Синапсы осуществляют связь между нейронами и умножают вход-

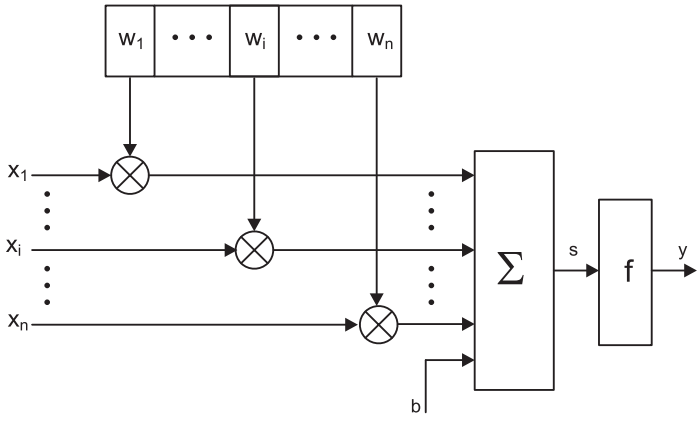

Рис. 2.1. Структура искусственного нейрона

ной сигнал на число, характеризующее силу связи, — вес синапса. Сумматор выполняет сложение сигналов, поступающих по синаптическим связям от других нейронов, и внешних входных сигналов. Нелинейный преобразователь реализует нелинейную функцию одного аргумента — выхода сумматора. Эта функция называется *функцией активации* или *передаточной функцией* нейрона. Нейрон в целом реализует скалярную функцию векторного аргумента. Математическая модель нейрона описывается соотношениями

$$s = \sum_{i=1}^{n} w_i x_i + b,$$

$$y = f(s),$$

где w_i — вес синапса, ($i = 1...n$); b — значение смещения; s — результат суммирования; x_i — компонент входного вектора (входной сигнал), ($i = 1...n$); y — выходной сигнал нейрона; n — число входов нейрона; f — нелинейное преобразование (функция активации или передаточная функция).

В общем случае входной сигнал, весовые коэффициенты и значения смещения могут принимать вещественные значения. Выход (y) определяется видом функции активации и может быть как действительным, так и целым. Во многих практических задачах входы, веса и смещения могут принимать лишь некоторые фиксированные значения.

Синаптические связи с положительными весами называют *возбуждающими*, с отрицательными весами — *тормозящими*.

Таким образом, нейрон полностью описывается своими весами w_i и передаточной функцией $f(s)$. Получив набор чисел (вектор) x_i в качестве входов, нейрон выдает некоторое число y на выходе.

Описанный вычислительный элемент можно считать упрощенной математической моделью биологических нейронов — клеток, из которых состоит нервная система человека и животных.

Чтобы подчеркнуть различие нейронов биологических и искусственных (математических), вторые иногда называют *нейроноподобными элементами* или *формальными нейронами*.

На входной сигнал s нелинейный преобразователь отвечает выходным сигналом $f(s)$, который представляет собой выход нейрона y. Примеры активационных функций представлены в табл. 2.1 и на рис. 2.2.

Таблица 2.1. Перечень функций активации нейронов

Название	Формула	Область значений
Пороговая	$f(s)=\begin{cases}0, & s<\theta,\\ 1, & s\geq\theta\end{cases}$	(0, 1)
Знаковая (сигнатурная)	$f(s)=\begin{cases}1, & s>0,\\ -1, & s\leq 0\end{cases}$	(–1, 10
Сигмоидальная (логистическая)	$F(S)=\frac{1}{1+e^{-s}}$	(0, 1)
Полулинейная	$f(s)=\begin{cases}s, & s>0,\\ 0, & s\leq 0\end{cases}$	(o, ∞)

Название	Формула	Область значений
Линейная	$f(s) = s$	$(-\infty, \infty)$
Радиальная базисная (гауссова)	$f(s) = \exp(-s^2)$	(0, 1)
Полулинейная с насыщением	$f(s) = \begin{cases} 0, & s \le 0, \\ s, & 0 < s < 1, \\ 1, & s \ge 1 \end{cases}$	(0, 1)
Линейная с насыщением	$f(s) = \begin{cases} -1, & s \le -1 \\ s, & -1 < s < 1, \\ 1, & s \ge 1 \end{cases}$	(–1, 1)
Гиперболический тангенс (сигмоидальная)	$f(s) = \dfrac{e^s - e^{-s}}{e^s + e^{-s}}$	(–1, 1)
Треугольная	$f(s) = \begin{cases} 1 - \lvert s \rvert, & \lvert s \rvert \le 1, \\ 0, & \lvert s \rvert > 1 \end{cases}$	(0, 1)

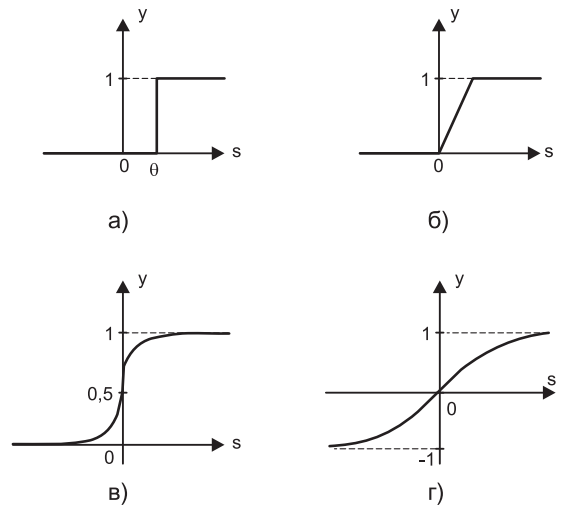

Рис. 2.2. Примеры активационных функций: а — пороговая; б — полулинейная с насыщением; в — сигмоид (логистическая функция); г — сигмоид (гиперболический тангенс)

Одной из наиболее распространенных является нелинейная функция с насыщением, так называемая *логистическая функция* или *сигмоид* (то есть функция S-образного вида):

$$f(s) = \frac{1}{1 + e^{-s}}.$$

Из выражения для сигмоида очевидно, что выходное значение нейрона лежит в диапазоне [0, 1]. Одно из ценных свойств сигмоидальной функции — простое выражение для ее производной:

$$f'(s) = f(s) \cdot [1 - f(s)].$$

Оно используется в некоторых алгоритмах обучения (см. ниже). Кроме того, данная функция обладает свойством «усиливать» слабые сигналы лучше, чем большие, и предотвращает насыщение от больших сигналов, так как они соответствуют областям аргументов, где сигмоид имеет пологий наклон.

2.2.3. Классификация нейронных сетей и их свойства

Искусственная нейронная сеть — это набор нейронов, соединенных между собой. Как правило, передаточные (активационные) функции всех нейронов в сети фиксированы, а веса являются параметрами сети и могут изменяться. Некоторые входы нейронов помечены как внешние входы сети, а некоторые выходы — как внешние выходы сети. Подавая любые числа на входы сети, мы получаем какой-то набор чисел на выходах сети. Таким образом, работа нейросети состоит в преобразовании входного вектора $\boldsymbol{X}$ в выходной вектор $\boldsymbol{Y}$, причем это преобразование задается весами сети.

Практически любую задачу можно свести к задаче, решаемой нейросетью. В табл. 2.2 показано, каким образом следует сформулировать в терминах нейросети задачу распознавания рукописных букв.

Таблица 2.2. Задача распознавания рукописных букв в терминах нейросети

Задача распознавания рукописных букв	
Дано: растровое черно-белое изображение буквы размером 30×30 пикселей	**Надо:** определить, какая это буква (в алфавите 33 буквы)
Формулировка для нейросети	
Дано: входной вектор из 900 двоичных символов (900 = 30×30)	**Надо:** построить нейросеть с 900 входами и 33 выходами, которые помечены буквами. Если на входе сети изображение буквы «А», то максимальное значение выходного сигнала достигается на выходе «А». Аналогично сеть работает для всех 33 букв

Поясним, зачем требуется выбирать выход с максимальным уровнем сигнала. Дело в том, что уровень выходного сигнала, как правило, может принимать любые значения из какого-то диапазона. Однако в данной задаче нас интересует не аналоговый ответ, а всего лишь номер категории (номер буквы в алфавите). Поэтому используется следующий подход — каждой категории сопоставляется свой выход, а ответом сети считается та категория, на чьем выходе уровень сигнала максимален. В определенном смысле уровень сигнала на выходе «А »— это достоверность того, что на вход была подана рукописная буква «А» .Задачи, в которых нужно отнести входные данные к одной из известных категорий, называются задачами классификации. Изложенный подход — стандартный способ классификации с помощью нейронных сетей.

Теперь, когда стало ясно, что именно мы хотим построить, можно переходить к вопросу «как строить такую сеть» . Этот вопрос решается в два этапа:

1) выбор типа (архитектуры) сети;
2) подбор весов (обучение) сети.

На первом этапе следует выбрать следующее:

1) какие нейроны мы хотим использовать (число входов, передаточные функции);
2) каким образом следует соединить их между собой;
3) что взять в качестве входов и выходов сети.

Эта задача на первый взгляд кажется необозримой, но, к счастью, нам необязательно придумывать нейросеть «с нуля» — существует несколько десятков различных нейросетевых архитектур, причем эффективность многих из них доказана математически. Наиболее популярные и изученные архитектуры — это многослойный персептрон, нейросеть с общей регрессией, сети Кохонена и другие, которые будут рассмотрены ниже.

На втором этапе нам следует «обучить» выбранную сеть, то есть подобрать такие значения ее весов, чтобы сеть работала нужным образом. Необученная сеть подобна ребенку — ее можно научить чему угодно. В используемых на практике нейросетях количество весов может составлять несколько десятков тысяч, поэтому обучение — действительно сложный процесс. Для многих архитектур разработаны специальные алгоритмы обучения, которые позволяют настроить веса сети определенным образом.

В зависимости от функций, выполняемых нейронами в сети, можно выделить три их типа:

1) *входные нейроны* — на которые подается входной вектор, кодирующий входное воздействие или образ внешней среды; в них обычно не осуществляется вычислительных процедур, информация передается со входа на выход нейрона путем изменения его активации;
2) *выходные нейроны* — выходные значения которых представляют выход сети; преобразования в таких нейронах осуществляются по приведенным выше выражениям;
3) *промежуточные нейроны* — составляют основу искусственных нейронных сетей, преобразования в них выполняются по этим же выражениям.

В большинстве нейронных моделей тип нейрона связан с его расположением в сети. Если нейрон имеет только выходные связи, то это входной нейрон, если наоборот — выходной нейрон. Однако может встретиться случай, когда выход внутреннего нейрона рассматривается как часть выхода сети. В процессе функционирования сети осуществляется преобразование входного вектора в выходной, то есть некоторая переработка информации. Конкретный вид выполняемого сетью преобразования информации обусловливается не только характеристиками нейроподобных элементов (функциями активации и т. п.), но и особенностями ее архитектуры. В частности, той или иной топологией межнейронных связей, выбором определенных подмножеств нейроподобных элементов для ввода и вывода информации, способами обучения сети, наличием или отсутствием конкуренции между нейронами, направлением и способами передачи информации между нейронами.

Классифицируя нейронные сети по топологии, можно выделить три основных типа таких сетей:

1) полносвязные сети (рис. 2.3, а);
2) многослойные или слоистые сети (рис. 2.3, б);

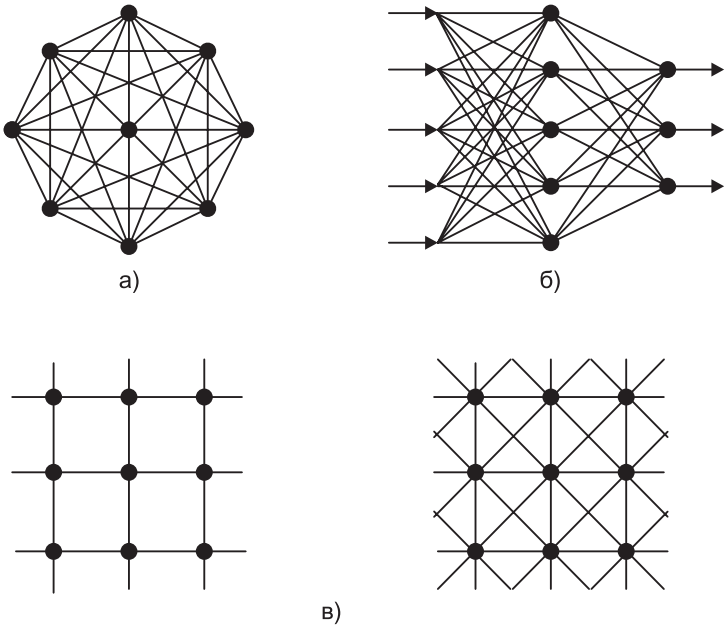

Рис. 2.3. Архитектуры нейронных сетей: а — полносвязная сеть, б — многослойная сеть с последовательными связями, в — слабосвязные сети

3) слабосвязные сети (нейронные сети с локальными связями) (рис. 2.3, *в*).

Полносвязные сети представляют собой ИНС, каждый нейрон которой передает свой выходной сигнал остальным нейронам, в том числе и самому себе. *Все* входные сигналы подаются *всем* нейронам. Выходными сигналами сети могут быть все или некоторые выходные сигналы нейронов после нескольких тактов функционирования сети.

В многослойных сетях нейроны объединяются в *слои*. Слой содержит совокупность нейронов с едиными входными сигналами. Число нейронов в каждом слое может быть любым и никак заранее не связано с количеством нейронов в других слоях. В общем случае сеть состоит из Q слоев, пронумерованных слева направо. Внешние входные сигналы подаются на входы нейронов первого слоя (входной слой часто нумеруют как нулевой), а выходами сети являются выходные сигналы последнего слоя. Вход нейронной сети можно рассматривать как выход «нулевого слоя» вырожденных нейронов, которые служат лишь в качестве распределительных точек, суммирования и преобразования сигналов здесь не производится. Кроме входного и выходного слоев в многослойной нейронной сети есть один или несколько промежуточных (скрытых) слоев. Связи от выходов нейронов некоторого слоя q ко входам нейронов следующего слоя (q+1) называются последовательными.

В свою очередь, среди слоистых сетей выделяют следующие типы.

Сети без обратных связей. В таких сетях нейроны входного слоя получают входные сигналы, преобразуют их и передают нейронам 1-го скрытого слоя, далее срабатывает 1-й скрытый слой и т. д. до Q-го слоя, который выдает выходные сигналы для интерпретатора и пользователя. Если не оговорено противное, то каждый выходной сигнал q-го слоя подается на вход всех нейронов (q+1)-го слоя; однако возможен вариант соединения q-го слоя с произвольным (q+p)-м слоем.

Следует отметить, что классическим вариантом слоистых сетей являются сети прямого распространения (рис. 2.4).

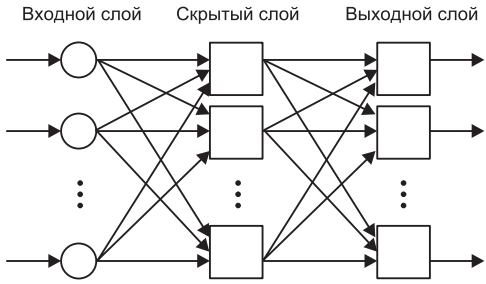

Рис. 2.4. Многослойная (двухслойная) сеть прямого распространения

Сети с обратными связями. Это сети, у которых информация с последующих слоев передается на предыдущие.

В качестве примера сетей с обратными связями на рис. 2.5 представлены так называемые *частично-рекуррентные сети* Элмана и Жордана.

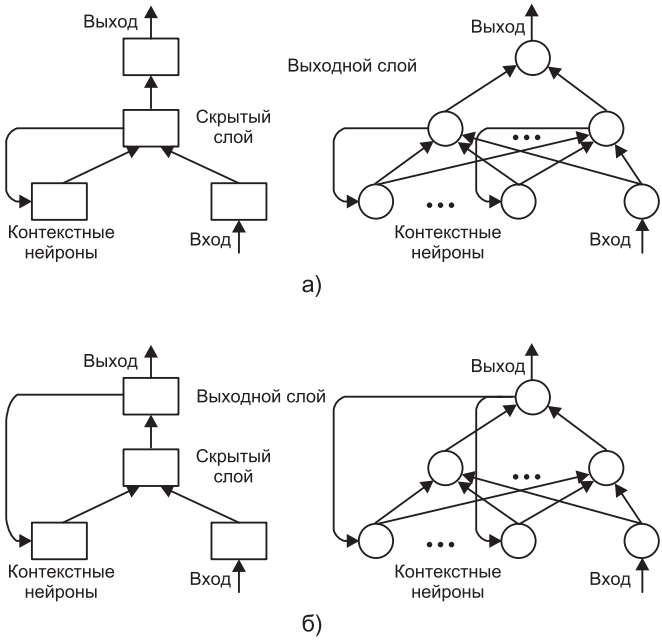

Рис. 2.5. Частично-рекуррентные сети: а — Элмана, б — Жордана

Сети можно классифицировать также по числу слоев.

Теоретически число слоев и число нейронов в каждом слое может быть произвольным, однако фактически оно ограничено ресурсами компьютера или специализированной микросхемы, на которых обычно реализуются ИНС. Чем сложнее ИНС, тем масштабнее задачи, подвластные ей.

Выбор структуры ИНС осуществляется в соответствии с особенностями и сложностью задачи. Для решения некоторых отдельных типов задач уже существу-

ют оптимальные (на сегодняшний день) конфигурации, описанные ниже. Если же задача не может быть сведена ни к одному из известных типов, разработчику приходится решать сложную проблему синтеза новой конфигурации. При этом он руководствуется несколькими основополагающими принципами:

1) возможности сети возрастают с увеличением числа ячеек сети, плотности связей между ними и числом выделенных слоев;
2) введение обратных связей наряду с увеличением возможностей сети поднимает вопрос о так называемой динамической устойчивости сети;
3) сложность алгоритмов функционирования сети (в том числе, например, введение нескольких типов синапсов — возбуждающих, тормозящих и др.) также способствует усилению возможностей ИНС.

Вопрос о необходимых и достаточных свойствах сети для решения того или иного рода задач представляет собой целое направление нейрокомпьютерной науки. Так как проблема синтеза ИНС сильно зависит от решаемой задачи, дать общие подробные рекомендации затруднительно. В большинстве случаев оптимальный вариант получается на основе интуитивного подбора, хотя в литературе приведены доказательства того, что для любого алгоритма существует нейронная сеть, которая может его реализовать. Остановимся на этом подробнее.

Многие задачи: распознавания образов (зрительных, речевых), выполнения функциональных преобразований при обработке сигналов, управления, прогнозирования, идентификации сложных систем, сводятся к следующей математической постановке. Необходимо построить отображение $X \rightarrow Y$ такое, чтобы в ответ на каждый возможный входной сигнал X формировался правильный выходной сигнал Y. Отображение задается конечным набором пар (<вход>, <известный выход>). Число таких пар (обучающих примеров) существенно меньше общего числа возможных сочетаний значений входных и выходных сигналов. Совокупность всех обучающих примеров носит название обучающей выборки.

В задачах распознавания образов X — некоторое представление образа (изображение, вектор чисел), Y — номер класса, к которому принадлежит входной образ.

В задачах управления X — набор контролируемых параметров управляемого объекта, Y — код, определяющий управляющее воздействие, соответствующее текущим значениям контролируемых параметров.

В задачах прогнозирования в качестве входных сигналов используются временные ряды, представляющие значения контролируемых переменных на некотором интервале времени. Выходной сигнал — множество переменных, которое является подмножеством переменных входного сигнала.

При идентификации X и Y представляют входные и выходные сигналы системы соответственно.

Вообще говоря, большая часть прикладных задач может быть сведена к реализации некоторого сложного многомерного функционального преобразования $X \rightarrow Y$.

В результате построения такого преобразования (отображения) необходимо добиться того, чтобы обеспечивалось формирование правильных выходных сигналов в соответствии:

1) со всеми примерами обучающей выборки;
2) со всеми возможными входными сигналами, которые не вошли в обучающую выборку.

Второе требование в значительной степени усложняет задачу формирования обучающей выборки. В общем виде эта задача в настоящее время еще не решена, однако во всех известных случаях было найдено частное решение. Дальнейшие рассуждения предполагают, что обучающая выборка уже сформирована.

Отметим, что теоретической основой для построения нейронных сетей является следующее утверждение: для любого множества пар входных-выходных векторов произвольной размерности $\{(\boldsymbol{X}^k, \boldsymbol{Y}^k), k = 1...N\}$ существует двухслойная однородная нейронная сеть с последовательными связями, с сигмоидальными передаточными функциями и с конечным числом нейронов, которая для каждого входного вектора $\boldsymbol{X}^k$ формирует соответствующий ему выходной вектор $\boldsymbol{Y}^k$.

Таким образом, для представления многомерных функций многих переменных может быть использована однородная нейронная сеть, имеющая всего один скрытый слой, с сигмоидальными передаточными функциями нейронов.

Для оценки числа нейронов в скрытых слоях однородных нейронных сетей можно воспользоваться эвристической формулой для оценки необходимого числа синаптических весов L_w (в многослойной сети с сигмоидальными передаточными функциями):

$$\frac{mN}{1+\log_2 N} \le L_w \le m\left(\frac{N}{m}+1\right)(n+m+1)+m,$$

где n — размерность входного сигнала, m — размерность выходного сигнала,

N — число элементов обучающей выборки.

Оценив необходимое число весов, можно рассчитать число нейронов в скрытых слоях. Например, число нейронов в двухслойной сети составит

$$L = \frac{L_w}{n+m}.$$

Известны и другие подобные формулы, например, вида

$$2\,(L+n+m) \le N \le 10\,(L+n+m),$$

$$\frac{N}{10} - n - m \le L \le \frac{N}{2} - n - m.$$

Точно так же можно рассчитать число нейронов в сетях с большим числом слоев, которые иногда целесообразно использовать: такие многослойные нейронные сети могут иметь меньшие размерности матриц весов нейронов одного слоя, чем двухслойные сети, реализующие то же самое отображение. К сожалению, строгая методика построения данных сетей пока отсутствует.

2.2.4. Теорема о полноте

Отметим, что отечественному читателю приведенные результаты обычно известны в виде так называемой *теоремы о полноте*. Ниже дано ее определение без имеющегося доказательства.

Теорема о полноте. Любая непрерывная функция на замкнутом ограниченном множестве может быть равномерно приближена функциями, вычисляемыми нейронными сетями, если функция активации нейрона дважды непрерывно дифференцируема и нелинейна (существуют и другие варианты теорем о полноте).

Таким образом, нейронные сети являются универсальными аппроксимирующими системами.

Очевидно, что процесс функционирования ИНС, то есть сущность действий, которые она способна выполнять, зависит от величин синаптических связей, поэтому, задавшись определенной структурой ИНС, отвечающей какой-либо задаче, разработчик сети должен найти оптимальные значения всех весовых коэффициентов.

Этот этап называется обучением ИНС, и от того, насколько качественно он будет выполнен, зависит способность сети решать поставленные перед ней проблемы во время функционирования.

2.2.5. Обучение нейронных сетей

Обучить нейросеть — значит сообщить ей, чего мы от нее добиваемся. Этот процесс очень похож на обучение ребенка алфавиту. Показав ребенку изображение буквы «А », мы спрашиваем его: «Какая это буква?» Если ответ неверен, мы сообщаем ребенку тот ответ, который хотели бы от него получить: «Это буква А» . Ребенок запоминает этот пример вместе с верным ответом, то есть в его памяти происходят некоторые изменения в нужном направлении. Мы будем повторять процесс предъявления букв снова и снова до тех пор, когда все буквы будут твердо запомнены. Такой процесс называют «обучением с учителем» (рис. 2.6).

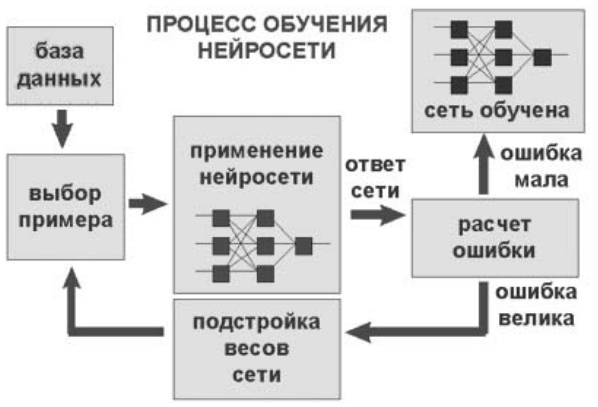

Рис. 2.6. Иллюстрация процесса обучения НС

При обучении сети мы действуем совершенно аналогично. У нас имеется некоторая база данных, содержащая примеры (набор рукописных изображений букв). Предъявляя изображение буквы «А» на вход сети, мы получаем от нее некоторый ответ, не обязательно верный. Нам известен и верный (желаемый) ответ — в данном случае нам хотелось бы, чтобы на выходе с меткой «А» уровень сигнала был максимален. Обычно в качестве желаемого выхода в задаче классификации берут набор (1, 0, 0, ...), где 1 стоит на выходе с меткой «А» ,а 0 — на всех остальных выходах. Вычисляя разность между желаемым ответом и реальным ответом сети, мы получаем (для букв русского алфавита) 33 числа — вектор ошибки. Алгоритм обучения — это набор формул, который позволяет по вектору ошибки вычислить требуемые поправки для весов сети. Одну и ту же букву (а также различные изображения одной и той же буквы) мы можем предъявлять сети много раз. В этом смысле обучение скорее напоминает повторение упражнений в спорте — тренировку.

Оказывается, что после многократного предъявления примеров веса сети стабилизируются, причем сеть дает правильные ответы на все (или почти все) примеры из базы данных. В таком случае говорят, что «сеть выучила все примеры» ,

«сеть обучена» ,или «сеть натренирована ».В программных реализациях можно видеть, что в процессе обучения функция ошибки (например, сумма квадратов ошибок по всем выходам) постепенно уменьшается. Когда функция ошибки достигает нуля или приемлемого малого уровня, тренировку останавливают, а полученную сеть считают натренированной и готовой к применению на новых данных.

Важно отметить, что вся информация, которую сеть имеет о задаче, содержится в наборе примеров. Поэтому качество обучения сети напрямую зависит от количества примеров в обучающей выборке, а также от того, насколько полно эти примеры описывают данную задачу. Так, например, бессмысленно использовать сеть для предсказания финансового кризиса, если в обучающей выборке кризисов не представлено. Считается, что для полноценной тренировки требуется хотя бы несколько десятков (а лучше сотен) примеров.

Математически процесс обучения можно описать следующим образом.

В процессе функционирования нейронная сеть формирует выходной сигнал $\boldsymbol{Y}$ в соответствии с входным сигналом $\boldsymbol{X}$, реализуя некоторую функцию $\mathbf{Y} = \mathbf{G}(\mathbf{X})$. Если архитектура сети задана, то вид функции **G** определяется значениями синаптических весов и смещений сети.

Пусть решением некоторой задачи является функция $\boldsymbol{Y} = \boldsymbol{F}(\boldsymbol{X})$, заданная парами входных-выходных данных $(\boldsymbol{X}^1, \boldsymbol{Y}^1)$, $(\boldsymbol{X}^2, \boldsymbol{Y}^2)$, ..., $(\boldsymbol{X}^N, \boldsymbol{Y}^N)$, для которых $\boldsymbol{Y}^k = \boldsymbol{F}(\boldsymbol{X}^k)$ $(k = 1, 2, ..., N)$.

Обучение состоит в поиске (синтезе) функции $\boldsymbol{G}$, близкой к $\boldsymbol{F}$ в смысле некоторой функции ошибки E (см. рис. 2.6).

Если выбрано множество обучающих примеров — пар $(\boldsymbol{X}^k, \boldsymbol{Y}^k)$ (где $k = 1, 2, ..., N$) и способ вычисления функции ошибки E, то обучение нейронной сети превращается в задачу многомерной оптимизации, имеющую очень большую размерность, при этом, поскольку функция E может иметь произвольный вид, обучение в общем случае — многоэкстремальная невыпуклая задача оптимизации.

2.2.6. Алгоритмы обучения нейронных сетей

Для решения этой задачи могут быть использованы следующие (итерационные) алгоритмы:

1) алгоритмы локальной оптимизации с вычислением частных производных первого порядка:
   - градиентный алгоритм (метод скорейшего спуска),
   - методы с одномерной и двумерной оптимизацией целевой функции в направлении антиградиента,
   - метод сопряженных градиентов,
   - методы, учитывающие направление антиградиента на нескольких шагах алгоритма;
2) алгоритмы локальной оптимизации с вычислением частных производных первого и второго порядка:
   - метод Ньютона,
   - методы оптимизации с разреженными матрицами Гессе,
   - квазиньютоновские методы,
   - метод Гаусса—Ньютона,
   - метод Левенберга—Марквардта и др.;
3) стохастические алгоритмы оптимизации

- поиск в случайном направлении,
- имитация отжига,
- метод Монте-Карло (численный метод статистических испытаний);

4) алгоритмы глобальной оптимизации (задачи глобальной оптимизации решаются с помощью перебора значений переменных, от которых зависит целевая функция).

Алгоритм обратного распространения. Рассмотрим идею одного из самых распространенных алгоритмов обучения — алгоритма обратного распространения ошибки (back propagation). Это итеративный градиентный алгоритм обучения, который используется с целью минимизации среднеквадратичного отклонения текущего выхода от желаемого выхода в многослойных нейронных сетях.

Алгоритм обратного распространения используется для обучения многослойных нейронных сетей с последовательными связями вида рис. 2.4. Как отмечено выше, нейроны в таких сетях делятся на группы с общим входным сигналом — слои, при этом на каждый нейрон первого слоя подаются все элементы внешнего входного сигнала, а все выходы нейронов q-го слоя подаются на каждый нейрон слоя $(q+1)$. Нейроны выполняют взвешенное (с синаптическими весами) суммирование элементов входных сигналов; к данной сумме прибавляется смещение нейрона. Над полученным результатом затем выполняется нелинейное преобразование с помощью активационной функции. Значение функции активации есть выход нейрона.

В многослойных сетях оптимальные выходные значения нейронов всех слоев, кроме последнего, как правило, неизвестны, и трех- или более слойный персептрон уже невозможно обучить, руководствуясь только величинами ошибок на выходах НС. Наиболее приемлемым вариантом обучения в таких условиях оказался градиентный метод поиска минимума функции ошибки с рассмотрением сигналов ошибки от выходов НС к ее входам, то есть в направлении, обратном прямому распространению сигналов в обычном режиме работы. Этот алгоритм обучения НС получил название процедуры обратного распространения.

В данном алгоритме функция ошибки представляет собой сумму квадратов рассогласования (ошибки) желаемого выхода сети и реального. При вычислении элементов вектора градиента использован своеобразный вид производных функций активации сигмоидального типа. Алгоритм действует циклически (итеративно), и его циклы принято называть *эпохами*. На каждой эпохе на вход сети поочередно подаются все обучающие наблюдения, выходные значения сети сравниваются с целевыми значениями и вычисляется ошибка. Значение ошибки, а также градиента поверхности ошибок используется для корректировки весов, после чего все действия повторяются. Начальная конфигурация сети выбирается случайным образом, и процесс обучения прекращается либо когда пройдено определенное количество эпох, либо когда ошибка достигнет некоторого определенного уровня малости, либо когда ошибка перестанет уменьшаться (пользователь может сам выбрать нужное условие остановки).

Приведем словесное описание алгоритма.

Шаг 1. Весам сети присваиваются небольшие начальные значения.

Шаг 2. Выбирается очередная обучающая пара ($\boldsymbol{X}$, $\boldsymbol{Y}$) из обучающего множества; вектор $\boldsymbol{X}$ подается на вход сети.

Шаг 3. Вычисляется выход сети.

Содержание