ДОНЕЦКИЙ НАЦИОНАЛЬНЫЙ ТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ

Е.Е. ФЕДОРОВ

ИСКУССТВЕННЫЕ НЕЙРОННЫЕ СЕТИ

Монография

Красноармейск
ДВНЗ «ДонНТУ»

2016

УДК 004.032.26
ББК 32.818
Ф 33

Рекомендовано к печати Ученым Советом Донецкого национального технического университета Министерства образования и науки Украины
Протокол №7 от 31.03.2016

Рецензент **Н.И. Чичикало**, доктор технических наук, профессор

Федоров Е.Е.
Ф 33 **Искусственные нейронные сети**: монография / Е.Е. Федоров. – Красноармейск: ДВНЗ «ДонНТУ», 2016. – 338 с.

ISBN 978-966-377-200-4

В монографии рассматриваются: история искусственных нейронных сетей (ИНС), классификация ИНС (по модульности, связности, обучению, решаемым задачам, наличию скрытых слоев, временных задержек и обратных связей) с примерами, технология настройки ИНС (алгоритмы выбора информативных факторов и понижения размерности данных, способы нормирования информативных факторов, функции активации искусственных нейронов и их особенности, оценки мощности обучающего множества, правила обучения весовых коэффициентов ИНС, условия завершения обучения), приводятся статические ИНС без ассоциативной памяти (MLP, RBFNN, GRNN, PNN, SVM и др.), статические ИНС с ассоциативной памятью (SOM, LVQNN, CPNN и др.), нерекуррентные динамические ИНС (NARNN, NARMANN, TDNN и др.), рекуррентные ИНС без ассоциативной памяти (ENN, RMLP и др.), рекуррентные ИНС с ассоциативной памятью (HNN и др.) и их сравнительная характеристика

У монографії розглядаються: історія штучних нейронних мереж (ШНМ), класифікація ШНМ (модульності, зв'язності, навчання, вирішуваним завданням, наявності прихованих шарів, тимчасових затримок і зворотних зв'язків) з прикладами, технологія налаштування ШНМ (алгоритми вибору інформативних факторів та пониження розмірності даних, способи нормування інформативних факторів, функції активації штучних нейронів та їх особливості, оцінки потужності навчальної множини, правила навчання вагових коефіцієнтів ШНМ, умови завершення навчання), наводяться статичні ШНМ без асоціативної пам'яті (MLP, RBFNN, GRNN, PNN, SVM та ін), статичні ШНМ з асоціативною пам'яттю (SOM, LVQNN, CPNN та ін), нерекурентні динамічні ШНМ (NARNN, NARMANN, TDNN та ін), рекурентні ШНМ без асоціативної пам'яті (ENN, RMLP та ін), рекурентні ШНМ з асоціативною пам'яттю (HNN та ін) та їх порівняльна характеристика

УДК 004.032.26
ББК 32.818

ISBN 978-966-377-200-4

СОДЕРЖАНИЕ

ЧАСТЬ 1
ВВЕДЕНИЕ В ИСКУССТВЕННЫЕ НЕЙРОННЫЕ СЕТИ

ЧАСТЬ 2
НЕРЕКУРРЕНТНЫЕ ИСКУССТВЕННЫЕ НЕЙРОННЫЕ СЕТИ

ЧАСТЬ 3
РЕКУРРЕНТНЫЕ ИСКУССТВЕННЫЕ НЕЙРОННЫЕ СЕТИ

ВВЕДЕНИЕ

Искусственные нейронные сети (ИНС) как научное направление появились в 40-е гг. XX века и стали особенно актуальными в начале 1980-х годов, когда остро встала проблема сверхвысокой производительности вычислительных средств. Для разработки ИНС был создан большой объем как прикладных программ, позволяющих ускорить исследования (например, Matlab, Statistica и др.), так и аппаратных решений. Выросло число научных сотрудников, использующих эти пакеты и аппаратные решения в практических целях.

К достоинствам ИНС относится следующее:

– высокая скорость выполнения вычислительных операций, обусловленная высоким параллелизмом действий;

– возможность использования ассоциативной памяти;

– возможность решения трудно формализуемых задач, в которых совместно используются данные разной физической природы, неполные, неточные, зашумленные и с высокой степень корреляции;

– адаптивность, позволяющая пополнять и усовершенствовать знания;

– надежность, обеспечиваемая параллельной архитектурой;

– возможность построения самообучающихся ИНС;

– сочетаемость с традиционными вычислительными алгоритмами обработки информации;

– взаимосвязи между данными исследуются на уже готовых ИНС;

– не требуются никакие предположения относительно распределения данных, априорная информация о данных может отсутствовать;

– возможен анализ систем с высокой степенью нелинейности;

– разработка структуры и идентификация параметров ИНС осуществляется быстрее, чем в случае традиционных методов моделирования;

– возможен анализ систем с большим количеством факторов;

– не требуется полный перебор всех возможных структур ИНС.

К недостаткам ИНС относится следующее:

– обучение и/или функционирование ИНС занимает значительное время;

– обучение большинства ИНС может привести к локальному минимуму;

– не автоматизирован процесс определения структуры ИНС, а произвольный выбор структуры ИНС и функций активации дает невысокую точность оценки;

– ИНС дает более низкую точность, чем традиционные методы моделирования, которые осуществляют более качественную настройку структуры модели;

– затруднена работа с короткими обучающими выборками;

– в ИНС обычно не используются разные функции активации нейронов в одном слое;

– затруднена работа с плохо формализованными данными;

– для ряда ИНС возможна работа только с бинарными данными.

Сложность и разнообразие исследуемого материала заставили автора изложить только часть проблем, связанных с созданием ИНС. В монографии приведены разработанные автором автоассоциативная нейросеть на основе радиально-базисных функций и автоассоциативная обобщенная регрессионная нейросеть. Автором дана классификация наиболее популярных ИНС, а также установлены взаимосвязи между ИНС и решаемыми ими задачами. В монографии предложена классификация функций активации искусственных нейронов. Автором обобщены условия, накладываемые на мощность обучающего множества для ИНС. В монографии выделены типы алгоритмов обучения, а также установлены взаимосвязи между ИНС и их алгоритмами обучения. Также приведен ряд оригинальных решений научно-прикладного характера, имеющих практическую ценность в образовании студентов и магистрантов.

Предлагаемая монография содержит основные положения методов создания ИНС и их приложения при решении различных технических проблем (например, задача поиска оптимального маршрута). При написании монографии использовались современные зарубежные и отечественные монографии и статьи, которые приведены в списке литературы. В монографии используются материалы курса лекций «Искусственные нейронные сети» и «Интеллектуальный анализ данных», которые читаются в Донецком национальном техническом университете.

Автор

СПИСОК УСЛОВНЫХ ОБОЗНАЧЕНИЙ

ART – adaptive resonance theory
AAMLP – auto-associative multi-layer perceptron
AARBFNN – auto-associative radial-basis function neural network
AAGRNN – auto-associative general regression neural network
BRNN – bidirectional recurrent neural network
BSNN – B-spline neural network
BAM – bidirectional associative memory
BM – Boltzmann machine
BSB – brain-state-in-box
CCNN – cascade-correlation neural network
CM – Cauchy machine
CMAC – cerebella model articulation controller
CRBM – conditional restricted Boltzmann machine
CNN – convolution neural network
CPNN – counter propagation neural network
DBNN – decision-based neural network
DBN – deep belief network
DBM – deep Boltzmann machine
ESN – echo state network
ENN (SRN) – Elman neural network (Simple recurrent network)
FBM – full Boltzmann machine
GM – Gauss machine
GRNN – general regression neural network
HM – Helmholtz machine
HME – Hierarchical mixture of expert
HNN – Hopfield neural network
ICANN – independent component analysis neural network
JNN – Jordan neural network
LVQNN – learning vector quantization neural network
LSTM – long short-term memory
ME – mixture of expert
MLP – multi-layer perceptron
NARNN – nonlinear autoregressive neural network
NARMANN – nonlinear autoregressive-moving average neural network
PCANN – principal component analysis neural network

PCARNN – principal component analysis recurrent neural network
PNN – probabilistic neural network
RBFNN – radial-basis function neural network
RAAM – recurrent auto-associative memory
RCCNN – recurrent cascade-correlation neural network
RMLP – recurrent multi-layer perceptron
RBM – restricted Boltzmann machine
SOM – self-organizing map
SBN – sigmoid belief network
SNN – spline neural network
SVM – support vector machine
TDNN – time-delay neural network
VNN – Volterra neural network
WNN – wavelet neural network

ЧАСТЬ 1
ВВЕДЕНИЕ В ИСКУССТВЕННЫЕ НЕЙРОННЫЕ СЕТИ

1. ИСТОРИЯ И КЛАССИФИКАЦИЯ ИСКУССТВЕННЫХ НЕЙРОННЫХ СЕТЕЙ

1.1. Искусственные нейроны и нейронные сети

Мозг человека представляет собой сложную нелинейную параллельную адаптивную систему обработки информации. Основой человеческого мозга является большое количество (до 10^{21}) связанных между собой и взаимодействующих нервных клеток – нейронов. Искусственные нейроны (ИН) сильно упрощены по сравнению с их биологическими прототипами. ИН представляет собой единицу обработки информации в искусственной нейронной сети. Искусственная нейронная сеть (ИНС) [1] – это система, с помощью которой осуществляется программно-аппаратное моделирование структур, подобных структуре человеческого мозга, и процессов мышления. ИНС реализует как краткосрочную память (в виде состояний нейронов), так и долгосрочную память (в виде весов связи между нейронами).

1.2. История развития искусственных нейронных сетей

Выделяются следующие этапы развития ИНС [1,2]:

1. Первые работы по ИН и ИНС были опубликованы Мак-Каллоком и Питтсом в 40-е гг. XX в. Модель используемого в них ИН представлена на рис.1.1, где w_i – вес (сила) связи (синапса); s – результат суммирования произведения $w_i x_i$; x_i – входные переменные; y – выходной сигнал ИН; $F(s) = \begin{cases} 1, & s \geq 0 \\ 0, & s < 0 \end{cases}$ – функция активации.

Рис.1.1. Искусственный нейрон

В этот период была опубликована работа Хебба, посвященная обучению ИНС, которое заключается в настройке весов связи. Простейшая форма обучения Хебба имеет вид

$$w_{kj}(t+1) = w_{kj}(t) + \eta y_k x_j,$$

где η – константа, определяющая скорость обучения сети, t – такт обучения.

2. В 50-е гг. XXв. начинаются попытки компьютерной реализации ИНС. Розенблатт предложил подход к распознаванию зрительных образов, основанный на использовании персептрона, представляющего собой один нейрон Мак-Каллока-Питтса (рис.1.1). Персептрон умел различать буквы алфавита, но был чувствительным к их написанию. Например, буквы А, *А*, **А** для этого устройства были тремя разными знаками. Другой областью применения персептрона стали электронно-лучевые трубки.

3. В 70-х гг. XXв. интерес к нейросетям ослаб. Это было связано с тем, что в 1969г. Минский и Пайперт доказали ограниченность однослойного персептрона, кроме того для проведения экспериментов не существовало персональных компьютеров.

4. Интерес к ИНС возобновился в 80-е гг. XXв. Было доказано, что на многослойный персептрон не распространяются ограничения, налагаемые на однослойный. В 1982г. Хопфилд предложил ИНС с обратными связями (сети Хопфилда), реализующими ассоциативную память, которая позволяет осуществлять распознавание объектов, информация о которых является неполной или неточной. В 1986 г. появился алгоритм обратного распространения ошибки для обучения многослойных персептронов. В 1982г. Кохоненом были предложены самоорганизующаяся карта признаков, использующая обучение без учителя (на основе конкурентного обучения). В 1987г. Гроссбергом и Карпентером была предложена нейросеть, также использующая обучение без учителя (на основе теории адаптивного резонанса). В 80-е гг. XXв. Хинтоном были предложены стохастическая (со стохастическими нейронами) и детерминированная (с детерминированными нейронами) машины Больцмана, реализующие ассоциативную память и использующие алгоритм имитации отжига. В 1988г. Брумхедом и Лове была предложена процедура построения ИНС на базе радиально-базисных функций, которая стала альтернативой многослойному персептрону. В начале 90-х гг. Ватник и его коллеги предложили машину опорных векторов, которая обеспечивает максимальную разделимость.

1.3. Подходы к реализации искусственных нейронных сетей

Можно выделить три подхода к реализации ИНС [1]:

1. Аппаратный – создание специальных компьютеров, нейрочипов, плат расширения, наборов микросхем.

2. Программный – создание программ и инструментария, рассчитанных на высокопроизводительные компьютеры.

3. Гибридный – комбинация первых двух. Часть вычислений выполняют специальные платы расширения (сопроцессоры), часть – программные средства.

1.4. Классификация искусственных нейронов

ИН классифицируются на [2]:

1. Детерминированные (функция активации не имеет вероятностной интерпретации) и стохастические (функция активации имеет вероятностную интерпретацию).

2. Первого, второго, …, n-го порядка.

3. Статические (без временных задержек) и динамические (с временными задержками).

Также выделяют ИН Фукушимы.

1.5. Классификация искусственных нейронных сетей

Классификация наиболее популярных ИНС предложена в данной монографии и приведена в табл.1.1.

По наличию скрытых слоев ИНС классифицируются на:

1. Однослойные (не содержат слои скрытых нейронов).

2. Многослойные (содержат слои скрытых нейронов).

По наличию временных задержек ИНС классифицируются на:

1. Динамические (с временными задержками).

2. Статические (без временных задержек).

По наличию обратной связи ИНС классифицируются на:

1. Рекуррентные (обратного распространения).

2. Нерекуррентные (прямого распространения).

Рекуррентные ИНС всегда являются динамическими, но обратное неверно.

По модульности ИНС классифицируются на:

1. Модульные (вычисления ИНС распределены по модулям, которые не пересекаются в своей работе друг с другом, а выходные сигналы этих модулей объединены модулем интеграции).

2. Немодульные.

Таблица 1.1 – Классификация наиболее популярных ИНС

№	Название	Категория обучения	Временная задержка	Обратная связь	Модульная
1	ART-1, ART-2	u	+	+	-
2	BRNN	s	+	+	-
3	BSNN	s	-	-	-
4	BAM	s	+	+	-
5	BM	s/u	+	+	-
6	BSB	s	+	+	-
7	CCNN	s	-	-	-
8	CM	s/u	+	+	-
9	CMAC	s	-	-	-
10	Cognitron	s/u	+	-	-
11	CNN	s	+	-	-
12	CPNN	s+u	-	-	-
13	DBNN	s+u	-	-	+
14	ESN	s	+	+	-
15	ENN (SRN), RAAM	s	+	+	-
16	GM	s	+	+	-
17	GRNN, AAGRNN	s	-	-	-
18	Hamming neural network	s	+	+	-
19	HM	s/u	+	+	-
20	HME	s/u	-	-	+
21	HNN	s	+	+	-
22	ICANN	u	-	-	-
23	JNN	s	+	+	-
24	LVQNN	s	-	-	-
25	LSTM	s	+	+	-
26	ME	s/u	-	-	+
27	MLP, AAMLP	s	-	-	-
28	Neocognitron	s/u	+	-	-
29	NARNN	s	+	-	-
30	NARMANN	s	+	+	-
31	PCANN	u	-	-	-
32	PCARNN	u	+	+	-
33	PNN	s	-	-	-
34	RBFNN, AARBFNN	s/s+u	-	-	-
35	RCCNN	s	+	+	-
36	RMLP	s	+	+	-
37	SOM	u	-	-	-
38	Spike neural network	s/u	+	-	-
39	SBN	u	-	-	-
40	SNN	s	-	-	-
41	SVM	s	-	-	-
42	TDNN	s	+	-	-
43	VNN	s	+	-	-
44	WNN	s	-	-	-

По связности ИНС классифицируются на:

1. Полносвязные (каждый нейрон текущего слоя связан со всеми нейронами следующего слоя).

2. Неполносвязные.

По обучению ИНС классифицируются на:

1. Обучающиеся с учителем (supervised).

2. Обучающиеся без учителя (самообучающиеся, самоорганизующееся) (unsupervised).

3. Обучающиеся с подкреплением (разновидность обучающихся без учителя, используется динамическое программирование);

4. Гибридные.

По решаемым задачам ИНС классифицируются на:

1. Распознавание (классификация) образцов. В случае небинарной классификации на вход ИНС подается обучающий вектор значений признаков образца, на выход ИНС подается обучающий бинарный вектор, причем в этом векторе единственная единица подается на нейрон, соответствующий идентификатору класса входного образца. В случае бинарной классификации на вход ИНС подается обучающий вектор значений признаков образца, на выход ИНС подается обучающая единица либо нуль, соответствующие идентификатору класса входного образца.

2. Аппроксимация многомерных функций. На вход ИНС подается обучающий вектор значений переменных аппроксимируемой функции, на выход ИНС подается обучающее значение аппроксимируемой функции.

3. Кластеризация образцов. На вход ИНС подается обучающий вектор значений признаков образца.

4. Автоассоциативное восстановление образцов. В случае самообучения только на вход ИНС подается обучающий вектор значений признаков образца. В случае обучения с учителем на вход и выход ИНС подается один и то же обучающий вектор значений признаков образца.

5. Гетероассоциативное восстановление образцов. На вход ИНС подается один обучающий вектор значений признаков образца, на выход ИНС подается другой обучающий вектор значений признаков образца.

6. Понижение размерности данных. На вход ИНС подается обучающий вектор значений признаков образца.

7. Прогноз. В одном случае на вход ИНС подаются обучающие значения сигнала $x(n-M),...,x(n-1)$, на выход ИНС подается обучающее значение сигнала $y(n)$. В другом случае на вход ИНС подаются обучающие значения сигнала $x(n-M),...,x(n-1)$, на выход ИНС подаются обучающие значения сигнала $y(n-M),...,y(n)$.

8. Фильтрация. В одном случае на вход ИНС подаются обучающие зашумленные значения сигнала $x(n-M),...,x(n-1)$, на выход ИНС подается обучающее незашумленное значение сигнала $y(n)$. В другом случае на вход ИНС подаются обучающие зашумленные значения сигнала $x(n-M),...,x(n-1)$, на выход ИНС подаются обучающие незашумленные значения сигнала $y(n-M),...,y(n)$.

9. Управление. На вход ИНС подается обучающий вектор измеренных значений параметров объекта управления, на выход ИНС подается обучающий вектор значений параметров регулятора.

10. Идентификация параметров модели. На вход ИНС подается обучающий вектор измеренных значений факторов модели объекта, на выход ИНС подается обучающий вектор измеренных значений откликов модели объекта.

Задачи и решающие их наиболее популярные ИНС приведены в табл.1.2. при этом одна ИНС может решать несколько задач.

Таблица 1.2 – Задачи и решающие их наиболее популярные ИНС

№	Задачи	Название ИНС
1	Распознавание (классификация) образцов	BRNN, CCNN, Cognitron, CNN, DBNN, ESN, ENN (SRN), GRNN, HME, JNN, ME, MLP, Neocognitron, PNN, RBFNN, RMLP, Spike neural network, SVM, TDNN
2	Аппроксимация многомерных функций	BSNN, CCNN, GRNN, HME, ME, MLP, RBFNN, SNN, SVM, WNN
3	Кластеризация образцов	ART-1, ART-2, BSB, SOM или LVQNN+SOM
4	Восстановление образцов	AAGRNN, AAMLP, AARBFNN, BAM, BM, BSB, CM, CPNN, GM, Hamming neural network, HNN, RAAM, SBN
5	Понижение размерности данных	ICANN, PCANN, PCARNN
6	Прогноз, фильтрация управление, идентификация параметров модели	BRNN, ENN (SRN), JNN, LSTM, NARNN, NARMANN, RCCNN, RMLP, Spike neural network, VNN

2 ТЕХНОЛОГИЯ НАСТРОЙКИ ИСКУССТВЕННОЙ НЕЙРОННОЙ СЕТИ

Технология настройки ИНС на решаемую задачу представлена на рис.2.1 [3].

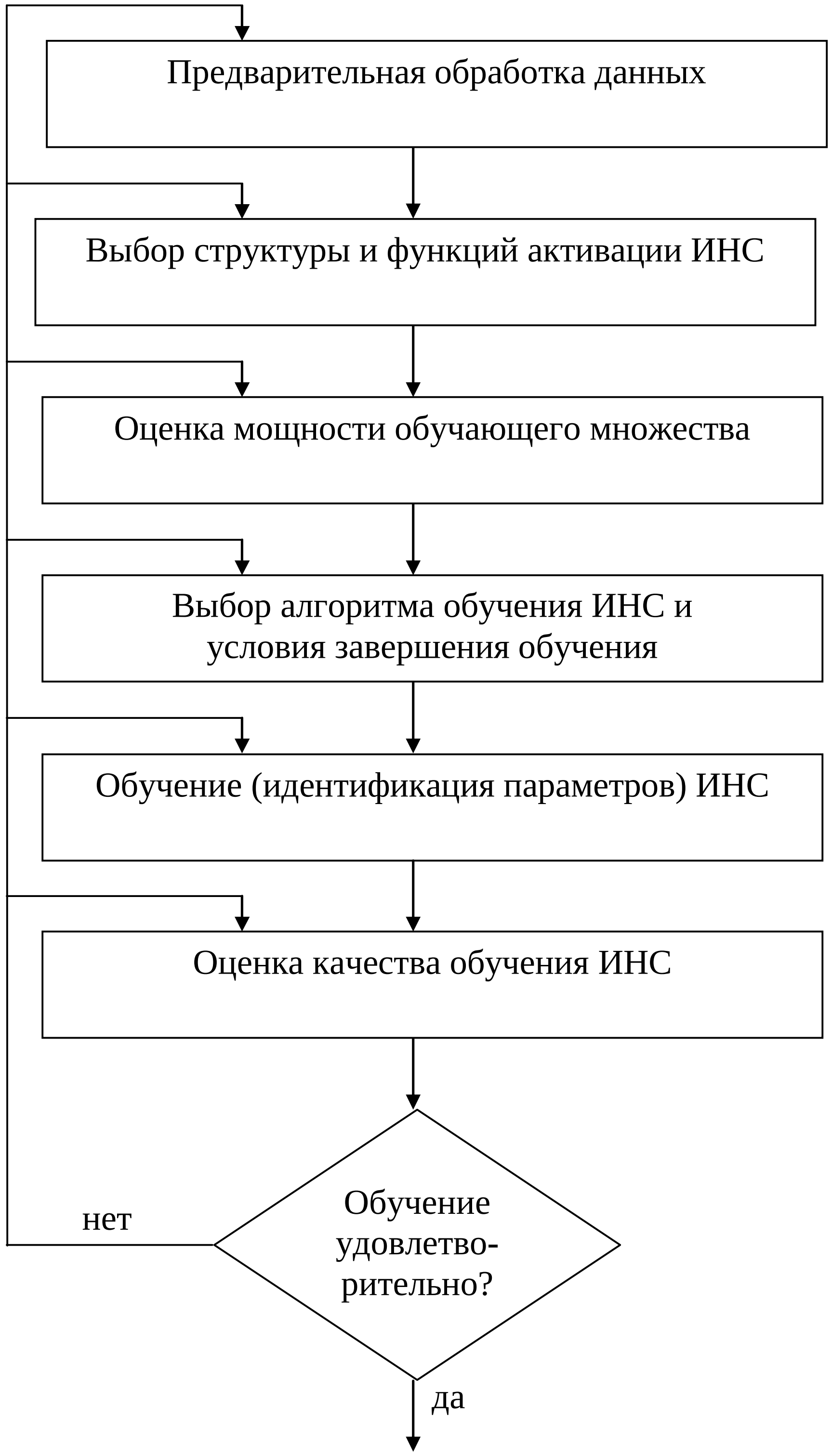


Рис.2.1. Технология настройки ИНС на решаемую задачу

2.1. Предварительная обработка данных

Предварительная обработка данных включает в себя выбор из выделенных факторов наиболее информативных либо понижение размерности данных (замена вектора текущих факторов длины N вектором новых факторов длины $\breve{N}$, $N > \breve{N}$, причем новые факторы некореллированы между собой), после чего выполняется нормирование значений полученных факторов.

2.1.1. Выбор из выделенных факторов наиболее информативных

Согласно [4], выбор из выделенных факторов наиболее информативных чаще всего производится на основе следующих алгоритмов.

Алгоритм последовательного удаления факторов

Оценим ошибку прогноза при использовании всех N факторов. Затем исключим из вектора факторов первый фактор и найдем ошибку e_{11}, которую дают оставшиеся $N-1$ факторов. Поменяем ролями первый и второй факторы и найдем ошибку e_{12} в новом $N-1$-мерном пространстве. Эту операцию поочередного исключения одного фактора проведем N раз. Среди полученных величин $e_{11},...,e_{1N}$ найдем самую малую. Она укажет на фактор, исключение которого из вектора факторов было наименее ощутимым. Исключим этот фактор из вектора факторов и приступим к испытанию оставшихся $N-1$ факторов. Их поочередное исключение из вектора факторов позволит найти самый неинформативный и снизить размерность пространства до $N-2$. Эти процедуры повторяются $N-\breve{N}$ раз, т.е. до тех пор, пока в векторе факторов не останется заданное число факторов $\breve{N}$.

Алгоритм последовательного добавления факторов

Вначале все N факторы проверяются на информативность. Для этого делается прогноз по контрольной последовательности по каждому из N факторов в отдельности и в информативный вектор включается фактор, давший наименьшее число ошибок. Затем к нему по очереди добавляются все $N-1$ факторов по одному. Получающиеся двумерные подпространства оцениваются по количеству ошибок прогноза. Выбирается самая информативная пара факторов. К ней таким же путем подбирается наилучший третий фактор из оставшихся $N-2$ и так продолжается до получения вектора $\breve{N}$ факторов.

Алгоритм случайного поиска с адаптацией

Разобьем отрезок (0,1) на N одинаковых участков и сопоставим каждый участок своему фактору: 1-й участок соответствует первому фактору, 2-й — второму и т. д. Каждый участок имеет ширину $1/N$. Запускается датчик случайных чисел с равномерным распределением в диапазоне (0,1). Если число попадает в *j*-й участок, то *j*-й фактор включается в испытуемый вектор факторов. После $\breve{N}$ шагов работы датчика выбранными оказываются $\breve{N}$ факторов. Качество этой случайно выбранной подсистемы факторов оценивается по одному из критериев, например по числу получаемых ошибок прогноза e_1. Описанная процедура случайного выбора $\breve{N}$-мерных подсистем факторов и оценки их качества повторяется *r* раз. Рассмотрение полученного списка оценок $e_1,...,e_r$ позволяет выбрать наилучшую и наихудшую из подсистем. На этом основании делается процедура «поощрения» и «наказания»: участки, соответствующие факторам, попавшим в наилучшую подсистему, поощряются путем расширения их границ на величину h, а участки, соответствующие признакам из самой неинформативной подсистемы, наказываются тем, что их ширина уменьшается на величину h ($h<1/N$). Суммарная длина всех участков по-прежнему равна единице. После этого случайным образом выбираются и испытываются *r* новых подсистем. Но теперь вероятность попадания признаков в эти подсистемы не одинакова: поощренные факторы, представленные более широкими отрезками, имеют больше шансов войти в очередную подсистему, чем наказанные. По результатам испытания этой партии подсистем процедура адаптации (наказания и поощрения) повторяется. Если некоторый фактор случайно попадает и в самую лучшую и самую худшую подсистемы, то длина его участка остается неизменной. Если же он регулярно оказывается в составе самой информативной подсистемы, то длина его участка растет с каждым шагом адаптации. Точно так же фактор, систематически попадающий в самую неинформативную подсистему, с каждым шагом сокращает длину своего участка и тем самым уменьшает вероятность включения в испытуемые подмножества факторов. После некоторого количества (*R*) циклов поиска и адаптации процесс стабилизируется: участки удачливых факторов занимают практически весь отрезок (0,1) и в испытуемую подсистему выбираются одни и те же факторы. Этот факт служит сигналом к окончанию процесса выбора $\breve{N}$-мерной подсистемы наиболее информативных факторов. Скорость сходимости и качество получаемого решения зависят от величины *h*. При больших значениях *h* процесс останавливается раньше, но качество полученного решения обычно хуже, чем при малых *h*. Малые значения *h* соответствуют более мягкой стратегии поощрений и наказаний. Если исходить из того, что фактор, наказываемый на всех *R* шагах адаптации, все еще сохраняет ненулевое значение ($l_{\min}$) длины своего участка, то величина *h* должна выбираться из соотношения $(1/N - Rh) \geq l_{\min}$.

Алгоритм направленного таксономического поиска признаков

В таксоны объединяются факторы, одинаково проявляющие себя на объектах обучающей выборки. Так как в один таксон группируются факторы по принципу максимального сходства (зависимости), то между таксонами обеспечивается максимальное несходство (независимость). Выбрав затем по одному типичному фактору из каждого таксона, получим $\breve{N}$-мерную подсистему, которая включает в свой состав факторы, отличающиеся максимальной независимостью друг от друга. В реальных задачах может встретиться случай, когда в исходной системе имеются группы факторов, связанных между собой сильной зависимостью, но не несущих информации, полезной для прогноза. Такие факторы объединятся в таксоны и их представители попадут в $\breve{N}$-мерную подсистему. Чтобы избежать такой ловушки, в данном алгоритме предусмотрен следующий прием. Таксономия N факторов делается на N' таксонов ($N > N' > \breve{N}$). Выбирается N' факторов и делается перебор $C_{N'}^{\breve{N}}$ сочетаний из N' по N факторам. Эти сочетания сравниваются между собой по качеству прогноза и выбирается такое сочетание, которое приводит к наименьшей величине ошибок. Оно и принимается за наиболее информативную $\breve{N}$-мерную подсистему факторов.

2.1.2. Понижение размерности данных

Согласно [3], понижение размерности данных чаще всего производится на основе метода анализа главных компонент, который заключается в следующем:

1. Формирование матрицы нормированных и центрированных данных

$$\hat{X} = \begin{bmatrix} \hat{x}_{11} & ... & \hat{x}_{1N} \\ ... & ... & ... \\ \hat{x}_{M1} & ... & \hat{x}_{MN} \end{bmatrix},$$

$$\hat{x}_{ij} = \frac{x_{ij} - \mu_j}{\sigma_j},$$

где μ_j – математическое ожидание j-го фактора,

σ_j – среднеквадратичное отклонение j-го фактора,

x_{ij} – i-е значение j-го фактора.

2. Вычисление ковариационной матрицы C

$$C = \frac{1}{N-1} \hat{X}^T \hat{X}$$

3. Нахождение собственных значений λ_j и собственных векторов $\mathbf{e}_j$ (соответствуют главным компонентам) матрицы C

4. Сортировка собственных значений λ_j и собственных векторов $\mathbf{e}_j$ по убыванию собственных значений λ_j

5. Отбор собственных векторов $\mathbf{e}_j$ согласно одному из трех следующих правил:

– правило на основе объясненной и выборочной дисперсии

$$\lambda_j < \alpha \sum_{i=1}^{N} \lambda_i - \sum_{i=1}^{j-1} \lambda_i \text{; обычно } \alpha = 0.9 \text{ или } \alpha = 0.95,$$

$\sum_{i=1}^{N} \lambda_i$ – выборочная дисперсия, $\sum_{i=1}^{j-1} \lambda_i$ – объясненная дисперсия.

– правило Кайзера (имеет тенденцию завышать количество главных компонент)

$$\lambda_j > \frac{1}{N} \sum_{i=1}^{N} \lambda_i;$$

– правило сломанной трости (имеет тенденцию занижать количество главных компонент)

$$\lambda_j > \frac{1}{N} \left(\sum_{i=1}^{N} \lambda_i \right) \left(\sum_{i=j}^{N} \frac{1}{i} \right).$$

6. Формирование из $\breve{N}$ отобранных собственных векторов $\mathbf{e}_j$ матрицы весов W

$$W = \begin{bmatrix} e_{11} & \dots & e_{1N} \\ \dots & \dots & \dots \\ e_{\breve{N}1} & \dots & e_{\breve{N}N} \end{bmatrix}.$$

7. Формирование новой матрицы данных меньшей размерности

$$\breve{X} = \widehat{X} W^T .$$

2.1.3. Нормирование значений полученных факторов

Согласно [3], чаще всего используются следующие способы нормирования значений факторов:

1. $\widehat{x}_i = \dfrac{x_i - x_i^{\min}}{x_i^{\max} - x_i^{\min}}, \ \widetilde{x}_i \in [0,1],$

где $x_i^{\min}, x_i^{\max}$ – минимальное и максимальное значение фактора x_i.

2. $\widehat{x}_i = \dfrac{x_i - \mu_i}{\sigma_i}, \ \widetilde{x}_i \in [0,1],$

где μ_i – математическое ожидание фактора x_i,

σ_i – среднеквадратичное отклонение фактора x_i.

3. $\widehat{x}_i = \dfrac{x_i - \mu_i}{\max\limits_m | x_m - \mu_m |}, \ \widetilde{x}_i \in [0,1].$

2.2. Выбор структуры и функций активации искусственной нейронной сети

Выбор структуры подразумевает, что если ИНС имеет скрытые слои, то необходимо задать количество скрытых слоев и количества нейронов в скрытых слоях. Их определение обычно производится эмпирическим путем и только для CCNN автоматически.

Классификация функций активации ИН предложена в данной монографии и приведена ниже:

1. Пороговые:

– пороговая бинарная

$$f(s) = \begin{cases} 0 & s \le 0 \\ 1, & s > 0 \end{cases}, \ f(s) \in \{0,1\};$$

– пороговая биполярная (знаковая, сигнатурная)

$$f(s) = \operatorname{sgn}(s) = \begin{cases} -1 & s \le 0 \\ 1, & s > 0 \end{cases}, \; f(s) \in \{-1,1\}.$$

2. Линейная полиномиальная

$$f(s) = s, \; f(s) \in R.$$

3. Нелинейные полиномиальные:

– квадратичная

$$f(s) = (s+1)^2 \text{ или } f(s) = s^2, \; f(s) \in R;$$

– кубическая

$$f(s) = (s+1)^3 \text{ или } f(s) = s^3, \; f(s) \in R.$$

4. Линейные кусочно-полиномиальные:

– полулинейная (псевдолинейная)

$$f(s) = \begin{cases} 0 & s \le 0 \\ s, & s > 0 \end{cases}, \; f(s) \in R_+;$$

– линейная с насыщением

$$f(s) = \begin{cases} -1, & s \le -1 \\ s, & -1 < s < 1, \\ 1, & s \ge 1 \end{cases} \; f(s) \in [-1,1];$$

– полулинейная (псевдолинейная) с насыщением

$$f(s) = \begin{cases} 0, & s \le 0 \\ s, & 0 < s < 1, \\ 1, & s \ge 1 \end{cases} \; f(s) \in [0,1];$$

– модульная

$$f(s) = |s|, \; f(s) \in R_+;$$

– треугольная

$$f(s) = \begin{cases} 1-|s| & |s| \le 1 \\ 0, & |s| > 1 \end{cases}, \; f(s) \in [0,1].$$

5. Сигмоидальные:

– логистическая

$$f(s) = \operatorname{sigm}(s) = \frac{1}{1+e^{-s}}, \; f(s) \in (0,1), \; s \in [-1,1];$$

– гиперболический тангенс

$$f(s) = \tanh(s), \; f(s) \in (-1,1), \; s \in [-1,1];$$

– арктангенс

$$f(s) = \frac{2}{\pi} \arctan(s), \; f(s) \in (-1,1), \; s \in [-1,1];$$

– синусоидальная

$f(s) = \sin(s\pi / 2)$, $f(s) \in [-1,1]$, $s \in [-1,1]$;

– алгебраическая сигмоида

$$f(s) = \frac{s}{\sqrt{1+s^2}}, \; f(s) \in (-1,1), \; s \in [-1,1].$$

6. Радиально-базисные:

– Гаусса

$$f(s) = \exp\left(-\frac{s^2}{2\sigma^2}\right), \; f(s) \in (0,1], \; \sigma > 0;$$

– Лапласа

$$f(s) = \exp\left(-\left|\frac{s}{\sigma}\right|\right), \; f(s) \in [1,\infty), \; \sigma > 0;$$

– Коши

$$f(s) = \left(1 + \left|\frac{s}{\sigma}\right|\right)^{-1}, \; f(s) \in (0,1], \; \sigma > 0;$$

– мультиквадратичная

$$f(s) = \sqrt{s^2 + \sigma^2}, \; f(s) \in [\sigma,\infty), \; \sigma > 0;$$

– обратная мультиквадратичная

$$f(s) = \left(\sqrt{s^2 + \sigma^2}\right)^{-1}, \; f(s) \in (0,\sigma^{-1}], \; \sigma > 0;$$

– плоский сплайн

$$f(s) = \left(\frac{s}{\sigma}\right)^2 \ln\left(\frac{s}{\sigma}\right), \; f(s) \in (-\infty,0) \cup (0,\infty), \; \sigma > 0;$$

– косинусоидальная

$$f(s) = \cos\left(\frac{s}{\sigma}\right), \; f(s) \in [-1,1], \; \sigma > 0;$$

– квадратичная полиномиальная

$$f(s) = \left(1 + \frac{s}{\sigma}\right)^2, \; f(s) \in R, \; \sigma > 0;$$

– кубическая полиномиальная

$$f(s) = \left(1 + \frac{s}{\sigma}\right)^3, \; f(s) \in R, \; \sigma > 0;$$

– треугольная

$$f(s)=\begin{cases}1-\left|\frac{s}{\sigma}\right| & \left|\frac{s}{\sigma}\right|\le 1\\ 0, & \left|\frac{s}{\sigma}\right|>1\end{cases},\ f(s)\in[0,1].$$

Особенности функций активации:

1. Пороговые функции не дифференцируемы. Они использовались в первых персептронах.

2. Линейные и кусочно-линейные полиномиальные функции используются в MLP только в случае линейной разделимости классов.

3. Нелинейные полиномиальные функции обычно используются в полиномиальных машинах для аппроксимации.

4. Сигмоидальные функции являются непрерывными, монотонными, возрастающими, ограниченными и гладкими (всюду дифференцируемы на $[-1,1]$). Они обычно используются в MLP (особенно логистическая и гиперболический тангенс) для аппроксимации и классификации.

5. Радиально-базисные функции (RBF) являются симметричными. Они обычно используются в сетях RBFNN (особенно Гаусса) для аппроксимации и классификации. В качестве RBF могут также использоваться непрерывные вейвлет-функции (вейвлетная нейросеть WNN), базисные функции квадратичного или кубического B-сплайна (B-сплайновая нейросеть BSNN), базисные функции сплайна Катмулла-Рома (сплайновая нейросеть SNN).

2.3. Оценки мощности обучающего множества

На основе обобщения работы [2] в монографии предлагается для ИНС со скрытыми слоями наложить на мощность обучающего множества P следующие условия:

1. Необходимое условие (для функций активаций всех классов)

$$P>\frac{N_w}{1-a},\ N_w=(N^{(0)}+N^{(L)})N_H,$$

где N_w – количество весов (связей между нейронами),

$N^{(0)}$ – количество выходных нейронов,

N_H – количество нейронов в скрытых слоях,

$N^{(L)}$ – количество выходных нейронов,

a – требуемая вероятность распознавания.

2. Достаточное условие (зависит от функции активации):

– для пороговой (использовалась бинарная) $P > \frac{N_w}{1-a} \ln \frac{N_w}{1-a}$;

– для полиномиальной (линейной и нелинейной) $P > \frac{N_w}{1-a}$;

– для линейной кусочно-полиномиальной $P > \left(\frac{N_w}{1-a} \right)^2$;

– для сигмоидальной (использовалась логистическая) и радиально-базисной (использовалась Гаусса) $P > \left(\frac{N_w}{1-a} \right)^4$,

На практике часто ограничиваются только необходимым условием.

2.4. Выбор алгоритма обучения искусственной нейронной сети

Выбор алгоритма обучения ИНС осуществляется эмпирическим путем. В монографии выделены следующие типы алгоритмов обучения:

– обучение на основе коррекции ошибок (обучение с учителем);

– обучение Хебба (обучение с учителем и без учителя);

– стохастическое обучение (обучение с учителем и без учителя);

– конкурентное (соревновательное) обучение (обучение с учителем и без учителя);

– обучение SVM (обучение с учителем);

– алгоритмы k-mean, EM (обучение без учителя);

– обучение с подкреплением/антиподкреплением (обучение с учителем);

– одношаговое обучение (инициализация опорными образцами) (обучение с учителем).

Также возможны гибридные типы алгоритмов обучения.

Каждый алгоритм использует правила обучения весовых коэффициентов.

Наиболее распространенные следующие правила обучения:

1. Дельта (Уидроу-Хоффа, Adaline или LMS) правило (обучение с учителем)

$$w_{ij}(n+1) = w_{ij}(n) + \eta(d_j - y_j)x_i .$$

2. Обобщенное дельта правило (обучение с учителем)

$$w_{ij}^{(k)}(n+1) = w_{ij}^{(k)}(n) + \eta g_j^{(k)} y_i^{(k-1)},$$

$$g_j^{(k)} = \begin{cases} (d_j - y_j^{(L)}) f'^{(L)}(s_j^{(L)}), & k = L \\ f'^{(k)}(s_j^{(k)}) \sum_l w_{jl}^{(k+1)}(n) g_l^{(k+1)}, & k < L \end{cases},$$

$$s_j^{(k)} = b_j^{(k)}(n) + \sum_i w_{ij}^{(k)}(n) y_i^{(k-1)},$$

где η – параметр, определяющий скорость обучения, $0 < \eta < 1$,

$b_j^{(k)}(n)$ – смещения (пороги) *k*-го слоя в момент времени n,

$w_{ij}^{(k)}(n)$ – вес связи от *i*-го нейрона *k*-1-го слоя к *j*-му нейрону *k*-го слоя в момент времени n,

$y_j^{(k)}$ – выход *j*-го нейрона *k*-го слоя,

$y_j^{(L)}$ – выход *j*-го нейрона выходного слоя,

x_i – обучающий выход *i*-го нейрона входного слоя,

d_j – обучающий выход *j*-го нейрона выходного слоя,

$f^{(k)}$ – функция активация нейронов *k*-го слоя,

L – число слоев.

3. Правило Хебба (обучение без учителя)

$$w_{ij}(n+1) = w_{ij}(n) + \eta y_j x_i.$$

4. Корреляционное правило (обучение с учителем)

$w_{ij}(n+1) = w_{ij}(n) + \eta d_j x_i$ или $w_{ij}(n+1) = w_{ij}(n) + \eta x_j x_i$.

5. Правило BSB (обучение с учителем)

$$w_{ij}(n+1) = w_{ij}(n) + \eta(x_j - y_j)x_i.$$

6. Правило Ойя (обучение без учителя)

$$w_i(n+1) = w_i(n) + \eta y(x_i - w_i(n)y).$$

7. Правило Сенгера (обучение без учителя)

$$w_{ij}(n+1) = w_{ij}(n) + \eta y_j \left(x_i - \sum_k^j w_{ik}(n) y_k \right).$$

8. Правило Infomax (обучение без учителя)

$$\mathbf{W}(n+1) = \mathbf{W}(n) + \eta(\mathbf{I} - 2\tanh(\mathbf{y}(n))\mathbf{y}^T(n))\mathbf{W}(n).$$

9. Правило STDP (обучение без учителя)

$$w_{ij}(n+1) = w_{ij}(n) + \eta\phi(t_j^y, t_i^x),$$

$$\phi(t_j^y, t_i^x) = \begin{cases} A^+ \exp\left(\dfrac{t_j^y - t_i^x}{\tau^+}\right), & t_j^y - t_i^x < 0 \\ -A^- \exp\left(-\dfrac{t_j^y - t_i^x}{\tau^-}\right), & t_j^y - t_i^x \ge 0 \end{cases},$$

где A^-, A^+, τ^-, τ^+ – положительные параметры,

t_i^x – время срабатывания *i*-го предсинаптического нейрона,

t_j^y – время срабатывания *j*-го постсинаптического нейрона.

10. Стохастическое правило пробуждения-засыпания (обучение с учителем, если имеются стохастические нейроны выходного слоя, иначе, обучение без учителя)

$$w_{ij}(n+1) = w_{ij}(n) + \eta(y_j - P(y_j \mid \mathbf{x}))x_i;$$

где x_i – состояние *i*-го стохастического нейрона,

y_j – состояние *j*-го стохастического нейрона.

11. Стохастическое правило Больцмана (обучение с учителем, если имеются стохастические нейроны выходного слоя, иначе, обучение без учителя)

$$w_{ij}(n+1) = w_{ij}(n) + \eta(\rho_{ij}^+ - \rho_{ij}^-),$$

$$\rho_{ij}^+ =< \varphi(x1_i, x1_j) >, \ \rho_{ij}^- =< \varphi(x2_i, x2_j) >,$$

$$\varphi(x_i, x_j) = \begin{cases} 1, & x_i = x_j \\ 0, & x_i \ne x_j \end{cases},$$

где $x1_i, x2_i$ – состояние *i*-го стохастического нейрона в положительной и отрицательной фазах,

$< \cdot >$ – усреднение.

12. Детерминированное правило Больцмана (обучение с учителем)

$$w_{ij}(n+1) = w_{ij}(n) + \eta(\rho_{ij}^+ - \rho_{ij}^-),$$

$$\rho_{ij}^+ =< x1_i >< x1_j >, \ \rho_{ij}^- =< x2_i >,< x2_j >,$$

где $< x1_i >, < x2_i >$ – среднее значение состояния *i*-го стохастического нейрона в положительной и отрицательной фазах.

13. Стохастическое правило SBN (обучение без учителя)

$$w_{ij}(n+1) = w_{ij}(n) + \eta\rho_{ij},$$

$$\rho_{ij} = < \varphi(x1_j x1_i) P_j >,$$

$$\varphi(x_i, x_j) = \begin{cases} 1, & x_i = x_j \\ 0, & x_i \neq x_j \end{cases},$$

где $x1_i$ – состояние i-го стохастического нейрона в положительной фазе,

P_j – вероятность того, что j-й стохастический нейрон находится в состоянии $x1_j$,

$< \cdot >$ – усреднение.

14. Правило Кохонена (обучение без учителя)

$$w_{ij}(n+1) = w_{ij}(n) + \eta h(j, j^*)(x_i - w_{ij}(n)),$$

где h – функция соседства,

j^* – номер нейрона-победителя.

15. Правило LVQ (обучение с учителем)

$$w_{ij}(n+1) = w_{ij}(n) + \eta h(j^d, j, j^*)(x_i - w_{ij}(n)),$$

$$h(j^d, j, j^*) = \begin{cases} 1, & j = j^* \wedge j^* = j^d \\ -1, & j = j^* \wedge j^* \neq j^d \\ 0, & j \neq j^* \end{cases},$$

где j^d – обучающий номер нейрона-победителя.

Относится как к обучению Хебба, так и к обучению с подкреплением/антиподкреплением.

16. Правило входной звезды (instar) (обучение без учителя)

$$w_{ij}(n+1) = w_{ij}(n) + \eta h(j, j^*)(x_i - w_{ij}(n)),$$

$$h(j, j^*) = \begin{cases} 1, & j = j^* \\ 0, & j \neq j^* \end{cases},$$

где j^* – номер нейрона-победителя.

17. Правило выходной звезды (outstar) (обучение с учителем)

$$w_{ij}(n+1) = w_{ij}(n) + \eta h(i, i^*)(d_j - w_{ij}(n)),$$

$$h(i, i^*) = \begin{cases} 1, & i = i^* \\ 0, & i \neq i^* \end{cases},$$

где i^* – номер нейрона-победителя.

Также используются правила когнитрона и неокогнитрона.

Тип алгоритма обучения и обучаемая на основе него наиболее популярные ИНС приведены в табл.2.1. при этом одна ИНС может обучаться на основе нескольких алгоритмов обучения.

Таблица 2.1 – Тип алгоритма обучения наиболее популярных ИНС

№	Тип алгоритма обучения	Название ИНС
1	Обучение на основе коррекции ошибок (дельта правило, обобщенное дельта правило, правило BSB)	BRNN, BSNN, BSB, CCNN, CMAC, CNN, ESN, ENN (SRN), GRNN, HME, JNN, LSTM, ME, MLP, NARNN, NARMANN, RBFNN, RCCNN, Spike neural network, SNN, TDNN, VNN, WNN
2	Обучение Хебба (детерминированное правило Больцмана, корреляционное правило, правило Infomax, правило Сенгера, правило Ойя, правило STDP)	детерминированные BM и CM, BSB, ICANN, PCANN, PCARNN, Spike neural network
3	Стохастическое обучение (стохастическое правило Больцмана, стохастическое правило SBN, стохастическое правило пробуждения-засыпания)	стохастические BM, CM, HM, SBN
4	Конкурентное обучение (правило instar, правило outstar, правило LVQ, правило Кохонена, правило когнитрона и неокогнитрона)	ART-1, ART-2, Cognitron, CPNN, LVQNN, Neocognitron, SOM
5	Обучение SVM	SVM
6	Обучение на основе алгоритма EM	HME, ME
7	Одношаговое обучение	Hamming neural network, PNN
8	Одношаговое обучение и обучение Хебба (корреляционное правило)	BAM, GM, HNN
9	Обучение на основе алгоритма k-mean и обучение на основе коррекции ошибок (обобщенное дельта правило)	RBFNN
10	Обучение на основе алгоритмов k-mean либо EM, и обучение с подкреплением/антиподкреплением	DBNN

2.5. Выбор условия завершения обучения искусственной нейронной сети

Для условия завершения обучения чаще всего используются следующие ошибки, характеризующие качество обучения [3]:

1. Среднеквадратичная ошибка

$$E = \frac{1}{P}\sum_{m=1}^{P} \| \mathbf{y}_m - \mathbf{d}_m \|^2 ,$$

где $\mathbf{y}_m$ – m-й полученный выходной вектор ИНС,

$\mathbf{d}_m$ – m-й тестовый выходной вектор,

P – мощность обучающего множества.

2. Взвешенная среднеквадратичная ошибка

$$E = \frac{1}{P}\sum_{m=1}^{P} v_m \| \mathbf{y}_m - \mathbf{d}_m \|^2 ,$$

где v_m – вес m-го тестового вектора.

Обучение завершается, если ошибка E меньше заданного ε.

2.6. Оценка качества обучения искусственной нейронной сети

Оценка качества обучения ИНС чаще всего производится на основе следующих критериев:

1. Вероятность правильной работы ИНС, полученная на тестовом множестве

2. Среднеквадратичная или взвешенная среднеквадратичная ошибка, полученная на тестовом множестве.

ЧАСТЬ 2
НЕРЕКУРРЕНТНЫЕ ИСКУССТВЕННЫЕ НЕЙРОННЫЕ СЕТИ

ГЛАВА 3
СТАТИЧЕСКИЕ ИСКУССТВЕННЫЕ НЕЙРОННЫЕ СЕТИ БЕЗ АССОЦИАТИВНОЙ ПАМЯТИ

3.1. Многослойный персептрон

На рис. 3.1 приведен многослойный персептрон (MLP) [5], который является нерекуррентной статической многослойной ИНС, содержащей один или более скрытых слоев и выходной слой. Классы разделяются гиперплоскостями.

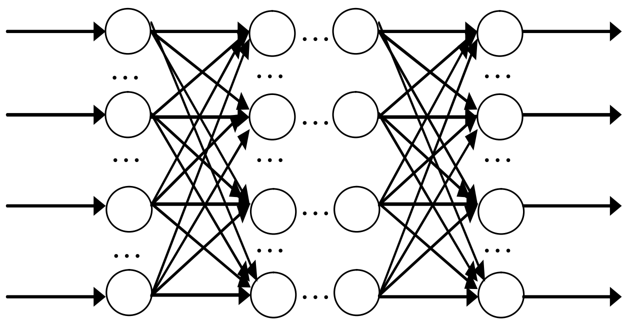


Рис. 3.1. Многослойный персептрон (MLP)

Для MLP используется обучение на основе коррекции ошибок (обучение с учителем), при этом чаще всего применяется алгоритм обратного распространения (BP). Это итеративный градиентный алгоритм обучения, который обеспечивает минимизацию среднеквадратичной ошибки.

Обучение ИНС (алгоритм обратного распространения)

1. Номер итерации обучения $n=1$, инициализация посредством равномерного распределения на интервале (0,1) или [-0.5, 0.5] смещений (порогов) $b_j^{(k)}(n)$ и весов $w_{ij}^{(k)}(n)$, $i \in \overline{1, N^{(k-1)}}$, $j \in \overline{1, N^{(k)}}$, $k \in \overline{1, L}$, где $N^{(k)}$ – количество нейронов в k-м слое, L – количество слоев.

2. Задается обучающее множество $\{(\mathbf{x}_\mu, \mathbf{d}_\mu) \mid \mathbf{x}_\mu \in R^{N^{(0)}}, \mathbf{d}_\mu \in R^{N^{(L)}}\}$, $\mu \in \overline{1, P}$, где $\mathbf{x}_\mu$ – μ-й обучающий входной вектор, $\mathbf{d}_\mu$ – μ-й обучающий выходной вектор, $N^{(0)}$ – количество нейронов входного слоя, $N^{(L)}$ – количество нейронов

выходного слоя, P – мощность обучающего множества. Номер текущей пары из обучающего множества $\mu = 1$.

3. Вычисление выходного сигнала для каждого слоя (прямой ход)

$$y_i^{(0)}(n) = x_{\mu i},$$

$$y_j^{(k)}(n) = f^{(k)}(s_j^{(k)}(n)), s_j^{(k)}(n) = \sum_{i=0}^{N^{(k-1)}} w_{ij}^{(k)}(n) y_i^{(k-1)}(n),$$

$$j \in \overline{1, N^{(k)}}, k \in \overline{1, L},$$

где $N^{(k)}$ – число нейронов в k-м слое, k – номер слоя, L – число слоев, $w_{ij}^{(k)}(n)$ – вес связи от i-го нейрона к j-му нейрону на k-м слое в момент времени n, $y_j^{(k)}(n)$ – выход j-го нейрона на k-м слое, $f^{(k)}$ – функция активация нейронов k-го слоя.

Считается, что $w_{0j}^{(k)}(n) = b_j^{(k)}(n), y_0^{(k-1)}(n) = 1$.

4. Вычисление энергии ошибки ИНС

$$E(n) = \frac{1}{2} \sum_{j=1}^{N^{(L)}} e_j^2(n), \; e_j(n) = y_j^{(L)}(n) - d_{\mu j}$$

5. Настройка синаптических весов на основе обобщенного дельта правила (обратный ход)

$$w_{ij}^{(k)}(n+1) = w_{ij}^{(k)}(n) - \eta \frac{\partial E(n)}{\partial w_{ij}^{(k)}(n)},$$

где η – параметр, определяющий скорость обучения (при большом η обучение происходит быстрее, но увеличивается опасность получить неверное решение), $0 < \eta < 1$.

$$\frac{\partial E(n)}{\partial w_{ij}^{(k)}(n)} = y_i^{(k-1)}(n) g_j^{(k)}(n), \; i \in \overline{0, N^{(k-1)}}, \; j \in \overline{1, N^{(k)}}, \; k \in \overline{1, L-1},$$

$$g_j^{(k)}(n) = \begin{cases} f'^{(L)}(s_j^{(L)}(n))(y_j^{(L)}(n) - d_{\mu j}), & k = L \\ f'^{(k)}(s_j^{(k)}(n)) \sum_{l=1}^{N^{(k+1)}} w_{jl}^{(k+1)}(n) g_l^{(k+1)}(n), & k < L \end{cases}$$

6. Проверка условия завершения

Если $n \bmod P > 0$, то $\mu = \mu + 1$, $n = n + 1$, переход к 3.

Если $n \bmod P = 0$ и $\frac{1}{P}\sum_{s=1}^{P} E(n-P+s) > \varepsilon$, то $n = n+1$, переход к 2.

Если $n \bmod P = 0$ и $\frac{1}{P}\sum_{s=1}^{P} E(n-P+s) < \varepsilon$, то завершиться.

Функционирование ИНС

$y_i^{(0)} = x_i$,

$$y_j^{(k)} = f^{(k)}(s_j^{(k)}),\ s_j^{(k)} = b_j^{(k)} + \sum_{i=1}^{N^{(k-1)}} w_{ij}^{(k)} y_i^{(k-1)},\ j \in \overline{1, N^{(k)}}, k \in \overline{1, L}.$$

Оценки количества скрытых слоев и количества нейронов в скрытых слоях для многослойного персептрона

На основе теоремы Колмогорова для MLP с сигмоидальной функцией достаточно одного скрытого слоя [2], но на практике для решения сложных задач их обычно больше. Например, если их два, то в первом скрытом слое извлекаются локальные признаки, т.е. нейроны этого слоя обучаются локальным признакам, характеризующим области входного пространства. Во втором скрытом слое извлекаются глобальные признаки, т.е. нейрон второго слоя обобщает выходы нейронов первого слоя, относящиеся к конкретной области входного пространства, и во всех областях, кроме этой области, его выход равен нулю.

Согласно [6], для MLP с сигмоидальной функцией количество нейронов в скрытых слоях N_H, т.е. $N_H = \sum_{k=1}^{L-1} N^{(k)}$, на основе теоремы Колмогорова определяется в виде $N_H = 2N^{(0)} + 1$, но на практике обычно $N^{(0)} \le N_H \le 3N^{(0)}$. Согласно [7], для оптимальной структуры MLP с сигмоидальной функцией количество нейронов в каждом скрытом слое должно равняться количеству входных нейронов, т.е. $N^{(k)} = N^{(0)}, k \in \overline{1, L-1}$.

Методы нелинейного программирования, используемые для обучения многослойного персептрона

Согласно [2], для обучения MLP чаще всего используются следующие методы нелинейного программирования:

1. Методы первого порядка:

1.1. Метод наискорейшего спуска (на нем основан BP).

1.2. Метод сопряженного градиента Флетчера-Ривса.

1.3. Квазиньютоновские методы (например, метод Бройдена-Флетчера-Гольдфарба-Шанно).

2. Методы второго порядка (например, методы Гаусса-Ньютона, Ньютона-Рафсона, Левенберга-Маркуардта).

Достоинства

1. Используется для классификации образцов.

2. Является универсальным аппроксиматором. Обеспечивает глобальную аппроксимацию нелинейного отображения входного сигнала в выходной.

3. Обеспечивает хорошее качество обобщения (т.е. хорошо работает с тестовыми данными, которые в процессе обучения не предъявлялись).

4. Количество классов может быть больше двух (в выходном слое может быть больше одного нейрона).

5. Количество нейронов в скрытых слоях меньше, чем у RBFNN, PNN, SVM, что приводит к более быстрому функционированию MLP.

Недостатки

1. Обучение происходит медленнее, чем в случае RBFNN, PNN, сети Хемминга, SOM, CPNN.

2. Отсутствует автоматическое определение числа скрытых слоев и числа нейронов в этих слоях. Это может привести к следующим проблемам:

– из-за избыточного количества нейронов в скрытых слоях, высокая точность, получаемая на обучающей выборке, может привести к неустойчивости полученных результатов (ИНС «переобучилась»);

– из-за недостаточного количества скрытых слоев и нейронов в скрытых слоях, глобальный минимум может быть не обнаружен.

3. В отличие от методов обучения SVM, градиентные методы обучения MLP выбирают положение разделяющей гиперплоскости произвольным образом.

4. Обучение не сводится к задаче квадратичного программирования в выпуклой области, имеющей единственное решение.

5. В отличие от ART, не решает проблему пластичности-стабильности (способность к восприятию новых образцов при сохранении старых образцов).

6. Представляет изображение вектором.

3.2. Нейросеть на основе радиально-базисных функций

На рис. 3.2 приведена нейросеть на основе радиально-базисных функций (RBFNN) [8-11], которая является нерекуррентной статической двухслойной ИНС. В отличие от MLP классы разделяются не гиперплоскостями, а гиперсферами. В отличие от MLP выходной слой является линейным.

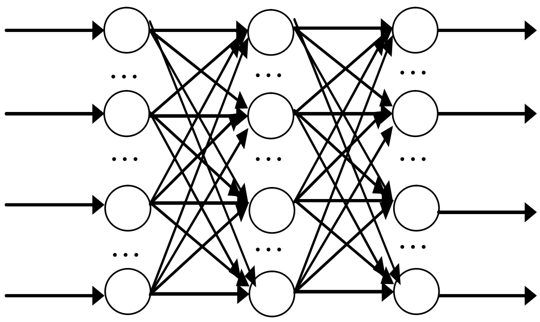


Рис. 3.2. Нейросеть на основе радиально-базисных функций (RBFNN)

Обычно используются три алгоритма обучения:

– обучение на основе коррекции ошибок (обучение с учителем) для выбора центров RBF и настройки весов;

– алгоритмы кластеризации (k-средних либо EM) для выбора центров RBF и обучение на основе коррекции ошибок (обучение с учителем) для настройки весов;

– случайный выбор центров RBF и метод псевдообращения для настройки весов.

Обычно для обучения RBFNN используется первый алгоритм, который дает наиболее точный результат. Это итеративный градиентный алгоритм обучения, который обеспечивает минимизацию среднеквадратичной ошибки.

Обучение ИНС

1. Номер итерации обучения $n=0$, инициализация посредством равномерного распределения на интервале (0,1) или [-0.5, 0.5] смещений (порогов) $b_j(n)$ и весов $w_{ij}(n)$, центров RBF (векторов математических ожиданий) $\mathbf{m}_i(n)$ размерности $N^{(0)}$ ($\mathbf{m}_i(n)$ интерпретируется как вектор весов i-го нейрона первого слоя), диагональных ковариационных матриц $\mathbf{C}_i(n)$ размерности $N^{(0)}\times N^{(0)}$, $\mathbf{C}_i(n)=\mathrm{diag}(\sigma_{i1}^2,...,\sigma_{iN^{(0)}}^2)$, $i\in\overline{1,N^{(1)}}$, $j\in\overline{1,N^{(2)}}$, где $N^{(0)}$ – количество нейронов входного слоя, $N^{(1)}$ – количество нейронов в первом слое, $N^{(2)}$ – количество нейронов во втором слое.

2. Задается обучающее множество $\{(\mathbf{x}_\mu,\mathbf{d}_\mu)\,|\,\mathbf{x}_\mu\in R^{N^{(0)}},\mathbf{d}_\mu\in R^{N^{(2)}}\}$, $\mu\in\overline{1,P}$, где $\mathbf{x}_\mu$ – μ-й обучающий входной вектор, $\mathbf{d}_\mu$ – μ-й обучающий выходной вектор, P – мощность обучающего множества.

3. Вычисление выходного сигнала для выходного слоя

$$y_{\mu j}(n)=\sum_{i=0}^{N^{(1)}}w_{ij}(n)G_i(\mathbf{x}_\mu),\ \mu\in\overline{1,P},\ j\in\overline{1,N^{(2)}},$$

$G_i(\mathbf{x}_\mu)=\exp\left(-\frac{1}{2}(\mathbf{x}_\mu-\mathbf{m}_i(n))^T\mathbf{C}_i^{-1}(n)(\mathbf{x}_\mu-\mathbf{m}_i(n))\right)$ – многомерная функция Гаусса.

Считается, что $w_{0j}(n)=b_j(n), G_0(\mathbf{x}_\mu)=1$.

4. Вычисление энергии ошибки ИНС

$$E(n)=\frac{1}{2P}\sum_{\mu=1}^{P}\sum_{j=1}^{N^{(2)}}e_{\mu j}^2(n),\ e_{\mu j}(n)=y_{\mu j}(n)-d_{\mu j}.$$

5. Весовые коэффициенты выходного слоя, вектора математических ожиданий и диагональные ковариационные матрицы вычисляются на основе обобщенного дельта правила

$$w_{ij}(n+1)=w_{ij}(n)-\eta_1\frac{\partial E(n)}{\partial w_{ij}(n)},\ i\in\overline{0,N^{(1)}},\ j\in\overline{1,N^{(2)}},$$

$$\mathbf{m}_i(n+1)=\mathbf{m}_i(n)-\eta_2\frac{\partial E(n)}{\partial \mathbf{m}_i(n)},\ i\in\overline{1,N^{(1)}},$$

$$\mathbf{C}_i^{-1}(n+1) = \mathbf{C}_i^{-1}(n) - \eta_3 \frac{\partial E(n)}{\partial \mathbf{C}_i^{-1}(n)},\ i \in \overline{1, N^{(1)}},$$

где η_1, η_2, η_3 – параметры, определяющие скорость обучения, $0 < \eta_1 < 1, 0 < \eta_2 < 1, 0 < \eta_3 < 1$.

$$\frac{\partial E(n)}{\partial w_{ij}(n)} = \frac{1}{P}\sum_{\mu=1}^{P}(y_{\mu j}(n) - d_{\mu j})G_i(\mathbf{x}_\mu),$$

$$\frac{\partial E(n)}{\partial \mathbf{m}_i(n)} = 2w_{ij}(n)\frac{1}{P}\sum_{\mu=1}^{P}(y_{\mu j}(n) - d_{\mu j})G_i(\mathbf{x}_\mu)\mathbf{C}_i^{-1}(\mathbf{x}_\mu - \mathbf{m}_i(n)),$$

$$\frac{\partial E(n)}{\partial \mathbf{C}_i^{-1}(n)} = -w_{ij}(n)\frac{1}{P}\sum_{\mu=1}^{P}(y_{\mu j}(n) - d_{\mu j})G_i(\mathbf{x}_\mu)(\mathbf{x}_\mu - \mathbf{m}_i(n))^T(\mathbf{x}_\mu - \mathbf{m}_i(n)).$$

6. Проверка условия завершения

Если $E(n) < \varepsilon$, то завершиться, иначе $n = n+1$, переход к 2.

Функционирование ИНС

$$y_j = f_j(\mathbf{x}) = b_j + \sum_{i=1}^{N^{(1)}} w_{ij}G_i(\mathbf{x}),\ j \in \overline{1, N^{(2)}}.$$

Достоинства

1. Используется для классификации образцов.

2. Является универсальным аппроксиматором. Обеспечивает локальную аппроксимацию нелинейного отображения входного сигнала в выходной (решает задачу аппроксимации кривой по точкам в многомерном пространстве, т.е. обучение эквивалентно нахождению такой поверхности в многомерном пространстве, которая наиболее точно соответствует данным обучения, т.е. интерполирует их).

3. Обеспечивает хорошее качество обобщения в случае выбора в качестве процедуры обучения обучение на основе коррекции ошибок.

4. Автоматически определяется количество скрытых слоев (равно одному).

5. Количество нейронов в скрытых слоях меньше, чем у PNN, поэтому RBFNN функционирует быстрее, чем PNN.

Недостатки

1. Отсутствует автоматическое определение числа нейронов в скрытом слое.

2. Количество нейронов в скрытом слое больше, чем у MLP, поэтому RBFNN функционирует медленнее, чем MLP.

3. В отличие от методов обучения SVM, градиентные методы обучения RBFNN выбирают положение разделяющей гиперплоскости произвольным образом.

4. Обучение не сводится к задаче квадратичного программирования в выпуклой области, имеющей единственное решение.

5. Обучение происходит медленнее, чем в случае PNN, сети Хемминга, SOM, CPNN.

6. В отличие от ART, не решает проблему пластичности-стабильности.

7. Представляет изображение вектором.

3.3. Обобщенная регрессионная нейросеть

На рис. 3.3 приведена обобщенная регрессионная нейросеть (GRNN) [12], которая является нерекуррентной статической двухслойной ИНС. В отличие от MLP классы разделяются не гиперплоскостями, а гиперсферами. В отличие от MLP выходной слой является линейным. GRNN является модификацией RBFNN.

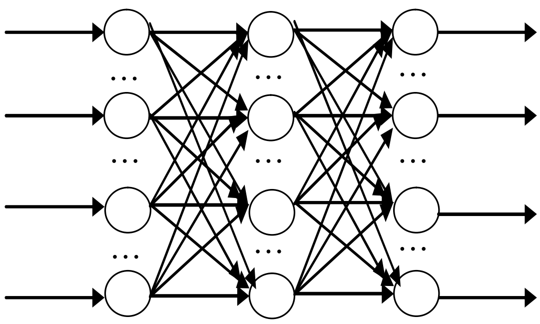


Рис. 3.3. Обобщенная регрессионная нейросеть (GRNN)

Обычно для GRNN используется обучение на основе коррекции ошибок (обучение с учителем) для выбора центров RBF и настройки весов. Это итеративный градиентный алгоритм обучения, который обеспечивает минимизацию среднеквадратичной ошибки.

Обучение ИНС

1. Номер итерации обучения $n = 0$, инициализация посредством равномерного распределения на интервале (0,1) или [-0.5, 0.5] смещений (порогов) $b_j(n)$ и весов $w_{ij}(n)$, центров RBF (векторов математических ожиданий) $\mathbf{m}_i(n)$ размерности $N^{(0)}$ ($\mathbf{m}_i(n)$ интерпретируется как вектор весов i-го нейрона первого слоя), диагональных ковариационных матриц $\mathbf{C}_i(n)$ размерности $N^{(0)} \times N^{(0)}$, $\mathbf{C}_i(n) = \operatorname{diag}(\sigma_{i1}^2, ..., \sigma_{iN^{(0)}}^2)$, $i \in \overline{1, N^{(1)}}$, $j \in \overline{1, N^{(2)}}$, где $N^{(0)}$ – количество нейронов входного слоя, $N^{(1)}$ – количество нейронов в первом слое, $N^{(2)}$ – количество нейронов во втором слое, $N^{(1)} = N^{(2)}$.

2. Задается обучающее множество $\{(\mathbf{x}_\mu, \mathbf{d}_\mu) \mid \mathbf{x}_\mu \in R^{N^{(0)}}, \mathbf{d}_\mu \in R^{N^{(2)}}\}$, $\mu \in \overline{1, P}$, где $\mathbf{x}_\mu$ – μ-й обучающий входной вектор, $\mathbf{d}_\mu$ – μ-й обучающий выходной вектор, P – мощность обучающего множества.

3. Вычисление выходного сигнала для первого слоя

$$y_{\mu j}(n) = \frac{\sum_{i=0}^{N^{(1)}} w_{ij}(n) G_i(\mathbf{x}_\mu)}{\sum_{i=0}^{N^{(1)}} G_i(\mathbf{x}_\mu)}, \ \mu \in \overline{1, P}, \ j \in \overline{1, N^{(2)}},$$

$G_i(\mathbf{x}_\mu) = \exp\left(-\frac{1}{2}(\mathbf{x}_\mu - \mathbf{m}_i(n))^T \mathbf{C}_i^{-1}(n)(\mathbf{x}_\mu - \mathbf{m}_i(n))\right)$ – многомерная функция Гаусса.

Считается, что $w_{0j}(n) = b_j(n), G_0(\mathbf{x}_\mu) = 1$.

4. Вычисление энергии ошибки ИНС

$$E(n) = \frac{1}{2P} \sum_{\mu=1}^{P} \sum_{j=1}^{N^{(2)}} e_{\mu j}^2(n), \ e_{\mu j}(n) = y_{\mu j}(n) - d_{\mu j}.$$

5. Весовые коэффициенты выходного слоя, вектора математических ожиданий и диагональные ковариационные матрицы вычисляются на основе обобщенного дельта правила

$$w_{ij}(n+1) = w_{ij}(n) - \eta_1 \frac{\partial E(n)}{\partial w_{ij}(n)}, \ i \in \overline{0, N^{(1)}}, \ j \in \overline{1, N^{(2)}},$$

$$\mathbf{m}_i(n+1) = \mathbf{m}_i(n) - \eta_2 \frac{\partial E(n)}{\partial \mathbf{m}_i(n)}, \ i \in \overline{1, N^{(1)}},$$

$$\mathbf{C}_i^{-1}(n+1) = \mathbf{C}_i^{-1}(n) - \eta_3 \frac{\partial E(n)}{\partial \mathbf{C}_i^{-1}(n)}, \ i \in \overline{1, N^{(1)}},$$

где η_1, η_2, η_3 – параметры, определяющие скорость обучения, $0 < \eta_1 < 1, 0 < \eta_2 < 1, 0 < \eta_3 < 1$.

$$\frac{\partial E(n)}{\partial w_{ij}(n)} = \frac{1}{P} \sum_{\mu=1}^{P} \frac{(y_{\mu j}(n) - d_{\mu j}) G_i(\mathbf{x}_\mu)}{\sum_{i=0}^{N^{(1)}} G_i(\mathbf{x}_\mu)},$$

$$\frac{\partial E(n)}{\partial \mathbf{m}_i(n)} = 2 \frac{1}{P} \sum_{\mu=1}^{P} \frac{G_i(\mathbf{x}_\mu) \mathbf{C}_i^{-1} (\mathbf{x}_\mu - \mathbf{m}_i(n))(w_{ij}(n) - y_{\mu j}(n))}{\sum_{i=0}^{N^{(1)}} G_i(\mathbf{x}_\mu)},$$

$$\frac{\partial E(n)}{\partial \mathbf{C}_i^{-1}(n)} = -\frac{1}{P} \sum_{\mu=1}^{P} \frac{G_i(\mathbf{x}_\mu)(\mathbf{x}_\mu - \mathbf{m}_i(n))^T (\mathbf{x}_\mu - \mathbf{m}_i(n))(w_{ij}(n) - y_{\mu j}(n))}{\sum_{i=0}^{N^{(1)}} G_i(\mathbf{x}_\mu)}$$

6. Проверка условия завершения

Если $E(n) < \varepsilon$, то завершиться, иначе $n = n + 1$, переход к 2.

Функционирование ИНС

$$y_j = f_j(\mathbf{x}) = \frac{b_j + \sum_{i=1}^{N^{(1)}} w_{ij} G_i(\mathbf{x})}{\sum_{i=1}^{N^{(1)}} G_i(\mathbf{x})}, \ j \in \overline{1, N^{(2)}}.$$

Достоинства

1. Используется для классификации образцов.

2. Является универсальным аппроксиматором. Обеспечивает локальную аппроксимацию нелинейного отображения входного сигнала в выходной (решает задачу аппроксимации кривой по точкам в многомерном пространстве, т.е. обучение эквивалентно нахождению такой поверхности в многомерном пространстве,

которая наиболее точно соответствует данным обучения, т.е. интерполирует их).

3. Обеспечивает хорошее качество обобщения в случае выбора в качестве процедуры обучения алгоритма выборе центров радиально-базисных функций с учителем.

4. Автоматически определяется количество скрытых слоев (равно одному).

5. Количество нейронов в скрытых слоях меньше, чем у PNN, поэтому GRNN функционирует быстрее, чем PNN.

Недостатки

1. Отсутствует автоматическое определение числа нейронов в скрытом слое.

2. Количество нейронов в скрытом слое больше, чем у MLP, поэтому GRNN функционирует медленнее, чем MLP.

3. В отличие от методов обучения SVM, градиентные методы обучения GRNN выбирают положение разделяющей гиперплоскости произвольным образом.

4. Обучение не сводится к задаче квадратичного программирования в выпуклой области, имеющей единственное решение.

5. Обучение происходит медленнее, чем в случае PNN, сети Хемминга, SOM, CPNN.

6. В отличие от ART, не решает проблему пластичности-стабильности.

7. Представляет изображение вектором.

3.4. Вероятностная нейросеть

Вероятностная нейросеть (PNN) [13,14] является нерекуррентной статической двухслойной (рис. 3.4) или трехслойной (рис.3.5) ИНС. В отличие от MLP классы разделяются не гиперплоскостями, а гиперсферами. Выходной слой является линейным.

Обычно для обучения PNN используется одношаговое обучение (обучение с учителем).

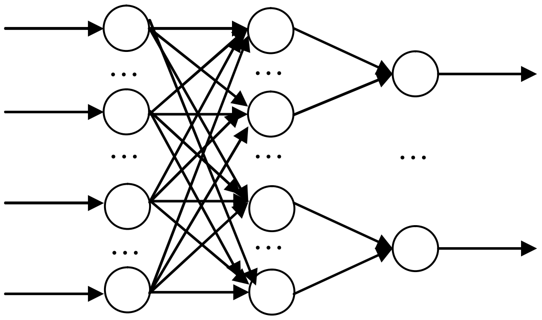


Рис. 3.4. Вероятностная нейросеть (PNN) с двумя слоями

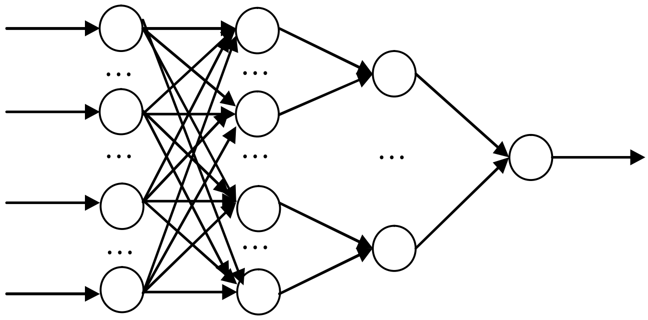


Рис. 3.5. Вероятностная нейросеть (PNN) с тремя слоями

Обучение ИНС

Для двухслойной и трехслойной PNN задаются:

– вектора математических ожиданий $\mathbf{m}_i$ размерности $N^{(0)}$ ($\mathbf{m}_i$ интерпретируется как вектор весов i-го нейрона первого слоя) и диагональные ковариационные матрицы $\mathbf{C}_i$ размерности $N^{(0)} \times N^{(0)}$, $\mathbf{C}_i = \mathrm{diag}(\sigma_{i1}^2,...,\sigma_{iN^{(0)}}^2)$, $i \in \overline{1, N^{(1)}}$,

– веса второго слоя w_{ij}, $w_{ij} \in \{1,0\}$, $i \in \overline{1, N^{(1)}}$, $j \in \overline{1, N^{(2)}}$, причем, если i-й нейрон первого слоя связан с j-м нейроном второго слоя, $w_{ij} = 1$, иначе $w_{ij} = 0$, где $N^{(0)}$ – количество нейронов входного слоя, $N^{(1)}$ – количество нейронов в первом слое, $N^{(2)}$ – количество нейронов во втором слое.

Для трехслойной PNN также задаются:

– стоимость (цена) неправильной классификации (величина потери) для j-го класса l_j, $j \in \overline{1, N^{(2)}}$;

– априорная вероятность появления объекта из j-го класса h_j, причем $\sum_{j=1}^{N^{(2)}} h_j = 1$.

Функционирование ИНС

Для двухслойной PNN выходной в виде

$$y_j = f_j(\mathbf{x}) = \frac{1}{n_j} \sum_{i=1}^{N^{(1)}} w_{ij} \frac{1}{\sqrt{(2\pi)^{N^{(0)}} \det \mathbf{C}_i}} \exp\left(-\frac{1}{2} (\mathbf{x} - \mathbf{m}_i)^T \mathbf{C}_i^{-1} (\mathbf{x} - \mathbf{m}_i) \right),$$

$$\det \mathbf{C}_i = \prod_{k=1}^{N^{(0)}} \sigma_{ik}^2 , \; n_j = \sum_{i=1}^{N^{(1)}} w_{ij} , \; j \in \overline{1, N^{(2)}} .$$

Для трехслойной PNN выходной сигнал вычисляется на основе минимума среднего риска в виде

$$y = f(\mathbf{x}) = \arg\max_j h_j l_j f_j(\mathbf{x}), \; j \in \overline{1, N^{(2)}} ,$$

причем $f_j(\mathbf{x})$ интерпретируется как функция правдоподобия.

Достоинства

1. Используется для классификации образцов.

2. Является универсальным аппроксиматором. Обеспечивает локальную аппроксимацию нелинейного отображения входного сигнала в выходной.

3. Обеспечивает хорошее качество обобщения.

4. Автоматически определяется количество скрытых слоев (равно одному или двум).

5. Автоматически определяется число нейронов первого скрытого слоя (равно числу обучающих данных) и количество нейронов второго скрытого слоя (равно числу классов).

6. Обучение отсутствует (все параметры задаются).

7. Количество классов может быть больше двух (во втором слое может быть больше одного нейрона).

8. Трехслойная PNN позволяет интерпретировать уровни выходного сигнала, считая их вероятностями. При этом ИНС сообщает, насколько можно доверять ее решению.

Недостатки

1. В отличие от других ИНС, классическая PNN ограничивается только задачами классификации.

2. Количество нейронов в первом скрытом слое больше, чем у RBFNN (необходимо хранить все обучающие данные). Это приводит к более медленному функционированию PNN по сравнению с другими ИНС.

3. В отличие от ART, не решает проблему пластичности-стабильности.

3.5. Машина опорных векторов

На рис. 3.6 приведена машина опорных векторов (SVM) [15], которая является нерекуррентной статической двухслойной ИНС. Два класса разделяются гиперплоскостью, но в отличие от MLP эта гиперплоскость обеспечивает максимальное разделение.

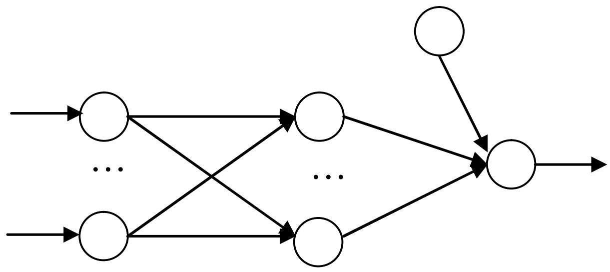


Рис. 3.6. Машина опорных векторов (SVM)

Основная идея создания SVM состоит в выборе подмножества обучающих данных в качестве опорных векторов. Это подмножество представляет устойчивые свойства всей обучающей выборки. Алгоритм обучения SVM (обучение с учителем) обеспечивает максимизацию функции Лагранжа. Этот алгоритм основан на минимизации эмпирического риска (т.е. ошибок обучения).

Обучение ИНС

1. Задается обучающее множество $\{(\mathbf{x}_\mu, d_\mu) \mid \mathbf{x}_\mu \in R^{N^{(0)}}, d_\mu \in \{-1,1\}\}$, $\mu \in \overline{1,P}$, где $\mathbf{x}_\mu$ – μ-й обучающий входной вектор, d_μ – μ-е обучающее выходное значение, $N^{(0)}$ – количество нейронов входного слоя, P – мощность обучающего множества, и параметр $C > 0$.

2. Вычисление выходного сигнала для первого слоя в виде ядра $K(\mathbf{x}_i, \mathbf{x}_j)$, $i, j \in \overline{1,P}$

– если хотим получить полиномиальную машину обучения, то

$K(\mathbf{x}_i, \mathbf{x}_j) = (\mathbf{x}_i^T \mathbf{x}_j)^p$ или $K(\mathbf{x}_i, \mathbf{x}_j) = (\mathbf{x}_i^T \mathbf{x}_j + 1)^p$;

– если хотим получить двухслойный персептрон, то

$K(\mathbf{x}_i, \mathbf{x}_j) = \tanh(k_0 + k_1 \mathbf{x}_i^T \mathbf{x}_j)$,

где параметры k_0, k_1 удовлетворяют условию $k_0 > 0$ или $k_1 > 0$;

– если хотим получить радиально-базисную функцию (RBF), то

$$K(\mathbf{x}_i, \mathbf{x}_j) = \exp\left(-\frac{1}{2\pm^2} \left\| \mathbf{x}_i - \mathbf{x}_j \right\|^2 \right),$$

где параметр ± – ширина функции $K(\mathbf{x}_i, \mathbf{x}_j)$,

$$\mathbf{x}_i^T \mathbf{x}_j = \sum_{k=1}^{N^{(0)}} x_{ik} x_{jk} \text{ , } \left\| \mathbf{x}_i - \mathbf{x}_j \right\| = \sqrt{\sum_{k=1}^{N^{(0)}} (x_{ik} - x_{jk})^2} \text{ .}$$

3. Определяются переменные (множители Лагранжа) ª$_i$, максимизирующие функцию Лагранжа, т.е.

$$L(\boldsymbol{\lambda}) = \sum_{i=1}^{P} \lambda_i - \frac{1}{2} \sum_{i=1}^{P} \sum_{j=1}^{P} \lambda_i \lambda_j d_i d_j K(\mathbf{x}_i, \mathbf{x}_j) \to \max_{\boldsymbol{\lambda}},$$

при ограничениях

$$0 \le \lambda_i \le C,$$

$$\sum_{i \in I} \lambda_i y_i = 0, \; I = \{ i : 0 \le \lambda_i \le C \}$$

Если $0 < \lambda_i < C$, то пара $(\mathbf{x}_i, d_i)$ является опорным вектором, т.е. находится на границе разделяющей полосы.

Количество опорных векторов обозначим как $N^{(1)}$, поскольку она соответствует количеству нейронов первого слоя. Перенумеруем обучающее множество и множители Лагранжа, чтобы вначале шли опорные вектора и связанные с ними множители Лагранжа.

4. Определяем оптимальное значение вектора весовых коэффициентов $\mathbf{w}$

$$\mathbf{w} = \sum_{i=1}^{N^{(1)}} \lambda_i d_i \mathbf{x}_i \text{ .}$$

5. Определяем оптимальное смещение (порог) b, используя любой опорный вектор $\mathbf{x}_i$

$$b = \frac{1}{d_i} - \mathbf{w}^T \mathbf{x}_i \text{ .}$$

Функционирование ИНС

$$y = f(\mathbf{x}) = \mathrm{sgn}\left(b + \mathbf{w}^T \mathbf{x} \right) = \mathrm{sgn}\left(b + \sum_{i=1}^{N^{(1)}} \lambda_i d_i K(\mathbf{x}, \mathbf{x}_i) \right).$$

Замечание. Чтобы разделяющая гиперплоскость как можно дальше отстояла от точек выборки, ширина полосы $\frac{2}{\|\mathbf{w}\|}$ должна быть максимальной.

Достоинства

1. Используется для классификации образцов.
2. Является универсальным аппроксиматором. Обеспечивает глобальную аппроксимацию нелинейного отображения входного сигнала в выходной.
3. Обеспечивает хорошее качество обобщения.
4. Автоматически определяется количество скрытых слоев (равно одному).
5. Автоматически определяется число нейронов скрытого слоя (равно числу опорных векторов).
6. В отличие от градиентных методов обучения MLP, RBFNN, ME, HME, которые выбирают положение разделяющей гиперплоскости произвольным образом, обучение SVM базируется на принципе оптимальной разделяющей гиперплоскости, что приводит к максимизации ширины разделяющей полосы между классами, следовательно, к более уверенной классификации.
7. В отличие от градиентных методов обучения MLP, RBFNN, ME, HME, сводящихся к многоэкстремальным задачам, обучение SVM сводится к задаче квадратичного программирования в выпуклой области, имеющей единственное решение.

Недостатки

1. Количество нейронов в скрытом слое больше, чем у MLP, RBFNN, особенно в случае линейной неразделимости и зашумленности данных. Это приводит к более медленному функционированию SVM по сравнению с MLP, RBFNN.
2. Может быть только два класса, поэтому приходится использовать бинарное дерево, узлами которого являются SVM.
3. Не существует общего подхода к автоматическому выбору ядра (и построению спрямляющего подпространства в целом) в случае линейной неразделимости классов.
4. В общем случае, когда линейная разделимость не гарантируется, приходится подбирать управляющий параметр C.

5. В отличие от ART, не решает проблему пластичности-стабильности.

6. Представляет изображение вектором.

3.6. Нейросеть на основе субкластерного принятия решений

На рис. 3.7 приведена нейросеть на основе субкластерного принятия решений (SDBNN) [16], которая является нерекуррентной статической модульной ИНС. Каждый модуль представляет собой дискриминантную функцию. Каждая подсеть состоит из модулей и соответствует определенному классу.

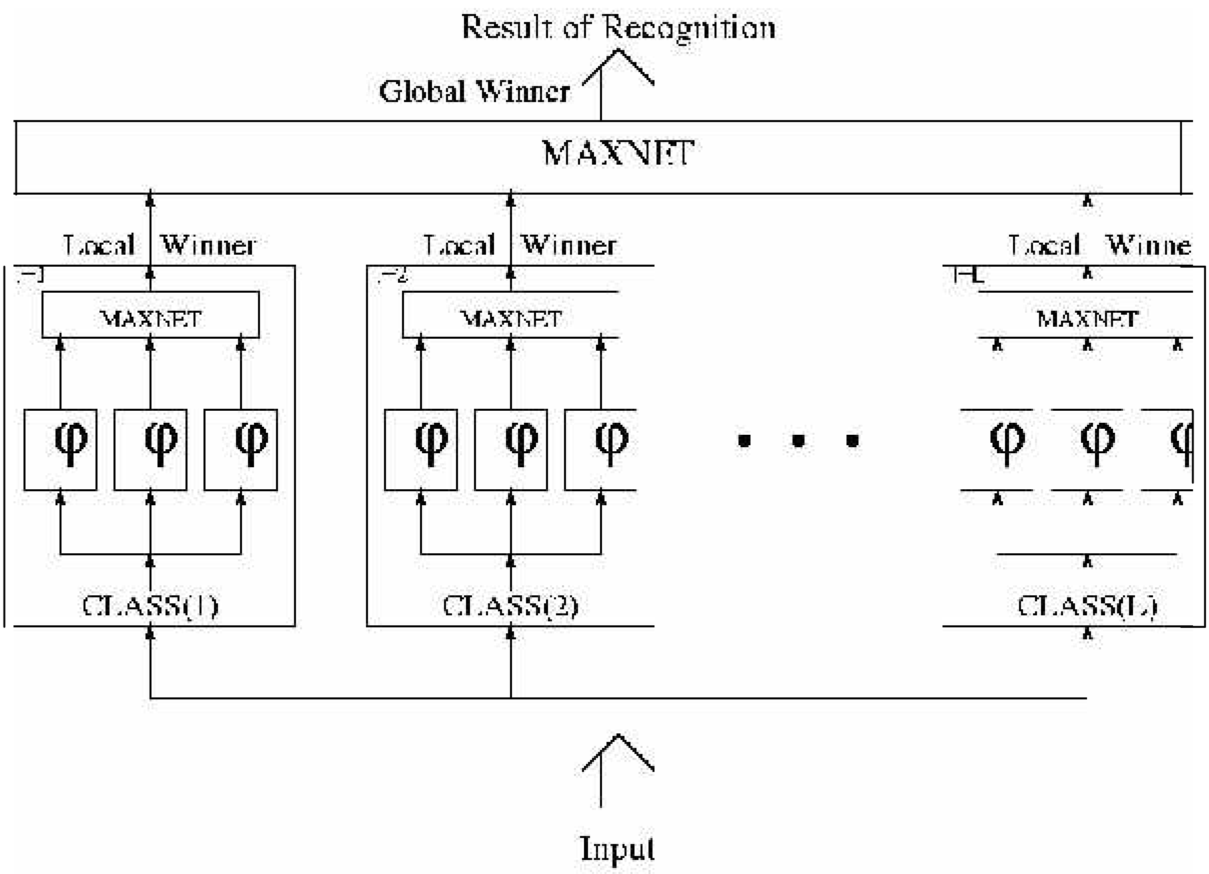


Рис. 3.7. Нейросеть на основе субкластерного принятия решений (SDBNN)

Для SDBNN обучение состоит из двух фаз – локального обучения без учителя (при этом чаще всего используется алгоритм k-средних, который обеспечивает формирование центров кластеров) и глобального обучения с учителем (используется обучение с подкреплением/антиподкреплением, которое обеспечивает максимизацию вероятности классификации).

Обучение ИНС

Локальное обучение (алгоритм k-средних) k-й подсети

1. Инициализация посредством равномерного распределения на интервале (0,1) или [-0.5, 0.5] весов $w_{ijk}(n)$, $i \in \overline{1, N^{(0)}}$, $j \in \overline{1, J}$, где $N^{(0)}$ – количество нейронов входного слоя, J – количество модулей подсети, L – количество слоев.

2. Задается случайным образом разбиение множества $\mathbf{X}_k = \{\mathbf{x}_k \mid \mathbf{x}_k \in R^{N^{(0)}}\}$, содержащего образцы k-го класса на множества $\mathbf{D}_{jk} = \{\mathbf{x}_t \mid \mathbf{x}_t \in R^{N^{(0)}}\}$, $j \in \overline{1, J}$.

3. Вычисление значения функции цели

$$F(\mathbf{w}_{jk}) = \sum_{\mathbf{x}_t \in \mathbf{D}_{jk}} \sum_{j=1}^{J} \left\| \mathbf{x}_t - \mathbf{w}_{jk} \right\|^2 .$$

4. Вычисление центров кластеров

$$\mathbf{w}_{jk} = \frac{\sum\limits_{\mathbf{x}_t \in \mathbf{D}_{jk}} \mathbf{x}_t}{\sum\limits_{t} \mathbf{x}_t}, \quad j \in \overline{1, J} .$$

5. Сформировать заново множества $\mathbf{D}_{jk} = \{\mathbf{x}_t \mid \mathbf{x}_t \in R^{N^{(0)}}\}$, $j \in \overline{1, J}$, согласно следующему правилу

если $j^* = \arg\min\limits_{j} \left\| \mathbf{x}_t - \mathbf{w}_{jk} \right\|^2$, то $\mathbf{D}_{jk^*} = \mathbf{D}_{jk^*} \bigcup \{\mathbf{x}_t\}$ и $\forall j \in \overline{1, J}, j \neq j^*$ $\mathbf{D}_{jk^*} = \mathbf{D}_{jk^*} \setminus \{\mathbf{x}_t\}$.

6. Вычисление значения функции цели

$$F(\hat{\mathbf{w}}_{jk}) = \sum_{\mathbf{x}_t \in \mathbf{D}_{jk}} \sum_{j=1}^{J} \left\| \mathbf{x}_t - \tilde{\mathbf{w}}_{jk} \right\|^2 .$$

7. Проверка условия завершения

Если $| F(\mathbf{w}_{jk}) - F(\hat{\mathbf{w}}_{jk}) | > \varepsilon$, то переход на 2, иначе завершиться.

Глобальное обучение (обучение с подкреплением/антиподкреплением) всей ИНС

1. Задается обучающее множество $\mathbf{X} = \{\mathbf{x}_t \mid \mathbf{x}_t \in R^{N^{(0)}}\}$, $t \in \overline{1, T}$, содержащее образцы всех классов.

2. Выполнить классификацию по всем модулям

$$y_{tk} = \max_j f(\mathbf{x}_t, \mathbf{w}_{jk}),\ j \in \overline{1,J},\ k \in \overline{1,K},\ t \in \overline{1,T},$$

$$z_t = \arg\max_k y_{tk},\ k \in \overline{1,K}.$$

Обычно $\varphi(\mathbf{x}_t, \mathbf{w}_{jk}) = \mathbf{x}_t^T \mathbf{w}_{jk}$ или $\varphi(\mathbf{x}_t, \mathbf{w}_{jk}) = \frac{1}{2}\left\|\mathbf{x}_t - \mathbf{w}_{jk}\right\|^2$

3. Сформировать для каждой подсети на основе выполненной классификации множество ложно отклоненных образцов $\mathbf{D}2_k \subset \mathbf{X}$, $k \in \overline{1,K}$, и множество ложно принятых образцов $\mathbf{D}3_k \subset \mathbf{X}$, $k \in \overline{1,K}$.

4. Вычислить новый вектор весов $\tilde{\mathbf{w}}_{ir}$ для каждого модуля

Если $\varphi(\mathbf{x}_t, \mathbf{w}_{jk}) = \mathbf{x}_t^T \mathbf{w}_{jk}$, то $\tilde{\mathbf{w}}_{jk} = \mathbf{w}_{jk} + \eta \sum\limits_{\mathbf{x}_t \in \mathbf{D}2_k} \mathbf{x}_t - \eta \sum\limits_{\mathbf{x}_t \in \mathbf{D}3_k} \mathbf{x}_t$.

Если $\varphi(\mathbf{x}_t, \mathbf{w}_{jk}) = \frac{1}{2}\left\|\mathbf{x}_t - \mathbf{w}_{jk}\right\|^2$, то

$$\tilde{\mathbf{w}}_{jk} = \mathbf{w}_{jk} + \eta \sum_{\mathbf{x}_t \in \mathbf{D}2_k} (\mathbf{x}_t - \mathbf{w}_{jk}) - \eta \sum_{\mathbf{x}_t \in \mathbf{D}3_k} (\mathbf{x}_t - \mathbf{w}_{jk})$$

5. Проверка условия завершения

$\sum\limits_{k=1}^{K} |\,\mathbf{D}2_k + \mathbf{D}3_k\,| > \varepsilon$, то $\mathbf{w}_{jk} = \tilde{\mathbf{w}}_{jk}$, $k \in \overline{1,K}$, то переход к 2, иначе завершиться.

Функционирование ИНС

$$y_k = \max_j \varphi(\mathbf{x}, \mathbf{w}_{jk}),\ j \in \overline{1,J},\ k \in \overline{1,K},$$

$$z = \arg\max_k y_k,\ k \in \overline{1,K}.$$

Достоинства

1. Используется для классификации образцов.

2. Является универсальным аппроксиматором. SDBNN использует нелинейность, аналогично MLP, но не для построения отображения входного сигнала в выходной, а с целью разбиения входного пространства между модулями.

3. Количество классов может быть больше двух.

4. Использует подсеть шлюзов для объединения знаний, накопленных разными модулями, в общее решение, которое имеет приоритет над каждым решением отдельного модуля.

5. Обеспечивает хорошее качество обобщения.

Недостатки

1. Отсутствует автоматическое определение числа модулей.

2. В отличие от методов обучения SVM, методы обучения SDBNN выбирают положение разделяющей гиперплоскости произвольным образом.

3. Обучение не сводится к задаче квадратичного программирования в выпуклой области, имеющей единственное решение.

4. Обучение происходит медленнее, чем в случае PNN, сети Хемминга, SOM, CPNN.

5. В отличие от ART, не решает проблему пластичности-стабильности.

3.7. Нейросеть на основе вероятностного принятия решений

На рис. 3.8 приведена нейросеть на основе вероятностного принятия решений (PDBNN) [17,18], которая является нерекуррентной статической модульной ИНС. Каждый модуль представляет собой дискриминантную функцию, которая является распределением Гаусса. Каждая подсеть состоит из модулей, представляет собой смесь распределений Гаусса и соответствует определенному классу.

Для PDBNN обучение состоит из двух фаз – локального обучения без учителя (при этом чаще всего используется алгоритм EM, который обеспечивает формирование центров кластеров) и глобального обучения с учителем (используется обучение с подкреплением/антиподкреплением, которое обеспечивает максимизацию вероятности классификации).

Обучение ИНС

Локальное обучение (EM-алгоритм) k-й подсети (смеси)

1. Инициализация посредством равномерного распределения на интервале (0,1) или [-0.5, 0.5] векторов параметров PDBNN $\boldsymbol{\theta}_{j|k} = (P_{j|k}, \mathbf{m}_{j|k}, \mathbf{C}_{j|k})$, $j \in \overline{1, J}$, где $P_{j|k}$ – априорная вероятность j-го модуля k-й подсети, $\mathbf{m}_{j|k}$ – вектор математических ожиданий размерности $N^{(0)}$ j-го модуля k-й подсети, $\mathbf{C}_{j|k}$ – диагональная ковариационная матрица размерности $N^{(0)} \times N^{(0)}$ j-го модуля k-й

подсети, J – количество модулей подсети, $N^{(0)}$ – количество нейронов входного слоя.

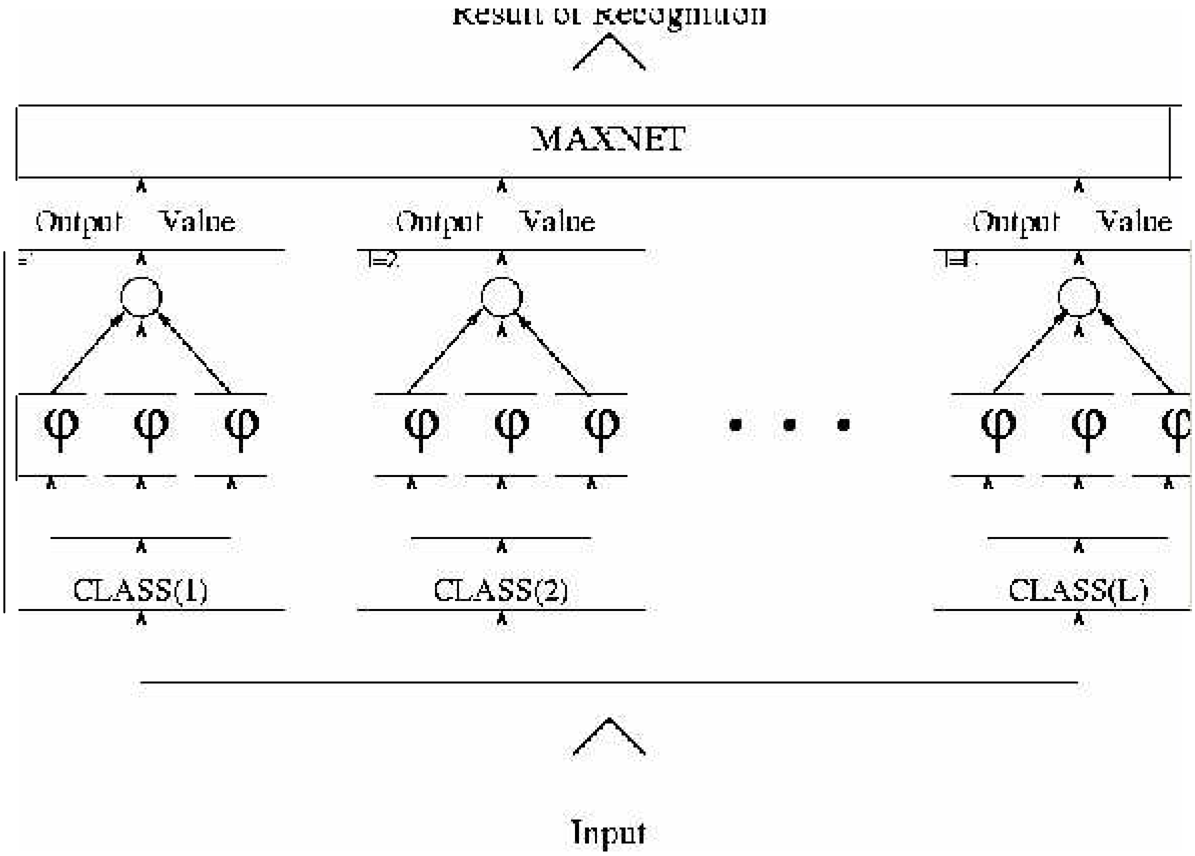


Рис. 3.8. Нейросеть на основе вероятностного принятия решений (PDBNN)

2. Задается обучающее множество $\mathbf{X}_i = \{\mathbf{x}_t \mid \mathbf{x}_t \in R^{N^{(0)}}\}$, $t \in \overline{1,T}$, содержащее образцы k-го класса.

3. Вычислить логарифмированную функцию правдоподобия

$$p_{j|k}(\mathbf{x}_t \mid \boldsymbol{\theta}_{j|k}) = \frac{1}{\sqrt{(2\pi)^N \det \mathbf{C}_{j|k}}} \exp\left(-\frac{1}{2}(\mathbf{x}_t - \mathbf{m}_{j|k})^T \mathbf{C}_{j|k}^{-1} (\mathbf{x}_t - \mathbf{m}_{j|k})\right),$$

$j \in \overline{1,J}$,

$$\varphi(\mathbf{x}_t, \boldsymbol{\Theta}_k) = \ln \sum_{j=1}^{J} P_{j|k}\, p_{j|k}(\mathbf{x}_t \mid \boldsymbol{\theta}_{j|k}),$$

$$\ln L(\mathbf{X} \mid \boldsymbol{\Theta}_k) = \sum_{t=1}^{T} \varphi(\mathbf{x}_t \mid \boldsymbol{\theta}_k).$$

4. Вычислить апостериорные вероятности того, что вектор $\mathbf{x}_t$ получен из j-го модуля k-й подсети (шаг E)

$$h_{j|k}(\mathbf{x}_t) = \frac{P_{j|k}\, p_{j|k}(\mathbf{x}_t)}{\sum_{s=1}^{J} P_{s|k}\, p_{s|k}(\mathbf{x}_t)}, \; j \in \overline{1,J}, \; t \in \overline{1,T}.$$

5. Вычислить новый вектор параметров $\tilde{\mathbf{\Theta}}_k$, максимизирующий логарифмированную функцию правдоподобия $\ln L(\mathbf{X} \,|\, \mathbf{\Theta}_k)$ (шаг M)

$$\tilde{P}_{j|k} = \frac{1}{T}\sum_{t=1}^{T} h_{j|k}(\mathbf{x}_t), \; j \in \overline{1,J},$$

$$\tilde{\mathbf{m}}_{j|k} = \frac{\sum_{t=1}^{T} h_{j|k}(\mathbf{x}_t)\mathbf{x}_t}{\sum_{t=1}^{T} h_{j|k}(\mathbf{x}_t)}, \; j \in \overline{1,J},$$

$$\tilde{\mathbf{C}}_{j|k} = \frac{\sum_{t=1}^{T} h_{j|k}(\mathbf{x}_t)(\mathbf{x}_t - \mathbf{m}_{j|k})^T(\mathbf{x}_t - \mathbf{m}_{j|k})}{\sum_{t=1}^{T} h_{j|k}(\mathbf{x}_t)}, \; j \in \overline{1,J}.$$

6. Вычислить логарифмированную функцию правдоподобия

$$\tilde{p}_{j|k}(\mathbf{x}_t \,|\, \mathbf{\theta}_{j|k}) = \frac{1}{\sqrt{(2\pi)^N \det \tilde{\mathbf{C}}_{j|k}}} \exp\left(-\frac{1}{2}(\mathbf{x}_t - \tilde{\mathbf{m}}_{j|k})^T \tilde{\mathbf{C}}_{j|k}^{-1}(\mathbf{x}_t - \tilde{\mathbf{m}}_{j|k})\right),$$

$j \in \overline{1,J}$,

$$\varphi(\mathbf{x}_t, \hat{\mathbf{\Theta}}_k) = \ln \sum_{j=1}^{J} P_{j|k}\, p_{j|k}(\mathbf{x}_t),$$

$$\ln L(\mathbf{X} \,|\, \hat{\mathbf{\Theta}}_k) = \sum_{t=1}^{T} \varphi(\mathbf{x}_t \,|\, \hat{\mathbf{\Theta}}_k).$$

7. Проверка условия завершения

Если $|\ln L(\mathbf{X} \,|\, \tilde{\mathbf{\Theta}}_k) - \ln L(\mathbf{X} \,|\, \mathbf{\Theta}_k)| > \varepsilon$, то $\mathbf{\Theta}_k = \tilde{\mathbf{\Theta}}_k$, переход к 3.

Если $|\ln L(\mathbf{X} \,|\, \tilde{\mathbf{\Theta}}_k) - \ln L(\mathbf{X} \,|\, \mathbf{\Theta}_k)| < \varepsilon$, то завершиться.

Глобальное обучение (обучение с подкреплением/антиподкреплением) всей ИНС

1. Инициализация (например, случайным образом) порогов $\hat{\gamma}_k$, $k \in \overline{1,K}$.

2. Задается обучающее множество $\mathbf{X}=\{\mathbf{x}_t \mid \mathbf{x}_t \in R^{N^{(0)}}\}$, $t\in\overline{1,T}$, содержащее образцы всех классов.

3. Выполнить классификацию по всем подсетям

$$\varphi(\mathbf{x}_t,\boldsymbol{\Theta}_k)=\ln\sum_{j=1}^{J}P_{j|k}\,p_{j|k}(\mathbf{x}_t \mid \boldsymbol{\theta}_{j|k}),\ k\in\overline{1,K},\ t\in\overline{1,T},$$

$$k_t^*=\begin{cases}\arg\max\limits_{k\in I_t}\varphi(\mathbf{x}_t \mid \boldsymbol{\Theta}_k), & |I_t|>0\\ 0, & |I_t|=0\end{cases},\ t\in\overline{1,T},$$

где $I_t=\{i \mid i\in\overline{1,K}\wedge\varphi(\mathbf{x}_t \mid \boldsymbol{\Theta}_i)-\gamma_i>0\}$.

4. Сформировать для каждой подсети на основе выполненной классификации множество ложно отклоненных образцов $\mathbf{D}2_k\subset\mathbf{X}$, $k\in\overline{1,K}$, и множество ложно принятых образцов $\mathbf{D}3_k\subset\mathbf{X}$, $k\in\overline{1,K}$.

5. Вычислить новый вектор параметров $\tilde{\boldsymbol{\Theta}}_k$ для каждой подсети

$$\tilde{P}_{j|k}=\frac{1}{T}\sum_{t=1}^{T}h_{j|k}(\mathbf{x}_t),\ j\in\overline{1,J},\ k\in\overline{1,K}.$$

$$\tilde{\mathbf{m}}_{j|k}=\mathbf{m}_{j|k}+\eta_1\left(\sum_{\mathbf{x}_t\in\mathbf{D}2_k}h_{j|k}(\mathbf{x}_t)\mathbf{C}_{j|k}^{-1}(\mathbf{x}_t-\mathbf{m}_{j|k})\right)-$$

$$-\eta_1\left(\sum_{\mathbf{x}_t\notin\mathbf{D}3_k}h_{j|k}(\mathbf{x}_t)\mathbf{C}_{j|k}^{-1}(\mathbf{x}_t-\mathbf{m}_{j|k})\right),\ j\in\overline{1,J},\ k\in\overline{1,K},$$

$$\mathbf{H}_{j|k}=\mathbf{C}_{j|k}^{-1}(\mathbf{x}_t-\mathbf{m}_{j|k})(\mathbf{x}_t-\mathbf{m}_{j|k})^T\mathbf{C}_{j|k}^{-1},$$

$$\tilde{\mathbf{C}}_{j|k}=\mathbf{C}_{j|k}+\frac{1}{2}\eta_2\left(\sum_{\mathbf{x}_t\in\mathbf{D}2_k}h_{j|k}(\mathbf{x}_t)(\mathbf{H}_{j|k}-\mathbf{C}_{j|k}^{-1})\right)-$$

$$-\frac{1}{2}\eta_2\left(\sum_{\mathbf{x}_t\notin\mathbf{D}3_k}h_{j|k}(\mathbf{x}_t)(\mathbf{H}_{j|k}-\mathbf{C}_{j|k}^{-1}\right),\ j\in\overline{1,J},\ k\in\overline{1,K}.$$

6. Вычислить новый порог γ_k для каждой подсети

$$\hat{\gamma}_k=\begin{cases}\gamma_k-\eta_3 f'(\gamma_k-\varphi(\mathbf{x}_t \mid \boldsymbol{\Theta}_k)), & \mathbf{x}_t\in\mathbf{D}2_k\\ \gamma_k+\eta_3 f'(\gamma_k-\varphi(\mathbf{x}_t \mid \boldsymbol{\Theta}_k)), & \mathbf{x}_t\in\mathbf{D}3_k\end{cases},\ k\in\overline{1,K},\ t\in\overline{1,T},$$

$f(s)=\dfrac{1}{1+e^{-as}}$ – штрафная функция.

7. Проверка условия завершения

$\sum_{k=1}^{K} | \mathbf{D}2_k + \mathbf{D}3_k | > \varepsilon$, то $\mathbf{\Theta}_k = \tilde{\mathbf{\Theta}}_k$, $k \in \overline{1,K}$, то переход к 3, иначе завершиться.

Функционирование ИНС

$$\varphi(\mathbf{x}, \mathbf{\Theta}_k) = \ln \sum_{j=1}^{J} P_{j|k} \, p_{j|k}(\mathbf{x} \,|\, \mathbf{\theta}_{j|k}), \; k \in \overline{1,K},$$

$$k^* = \begin{cases} \arg\max\limits_{k \in I} \varphi(\mathbf{x} \,|\, \mathbf{\Theta}_k), & |I| > 0 \\ 0, & |I| = 0 \end{cases},$$

где $I = \{ i \,|\, i \in \overline{1,K} \wedge \varphi(\mathbf{x} \,|\, \mathbf{\Theta}_i) - \gamma_i > 0 \}$.

Достоинства

1. Используется для классификации образцов.
2. Является универсальным аппроксиматором. ME использует нелинейность, аналогично MLP, но не для построения отображения входного сигнала в выходной, а с целью разбиения входного пространства между модулями.
3. Количество классов может быть больше двух.
4. Использует подсеть шлюзов для объединения знаний, накопленных разными модулями, в общее решение, которое имеет приоритет над каждым решением отдельного модуля.
5. Обеспечивает хорошее качество обобщения.

Недостатки

1. Отсутствует автоматическое определение числа модулей.
2. В отличие от методов обучения SVM, методы обучения PDBNN выбирают положение разделяющей гиперплоскости произвольным образом.
3. Обучение не сводится к задаче квадратичного программирования в выпуклой области, имеющей единственное решение.
4. В отличие от ART, не решает проблему пластичности-стабильности.

3.8. Смесь экспертов

На рис. 3.9 приведена смесь экспертов (ME) [19], которая является нерекуррентной статической модульной ИНС. ME состоит из модулей (сетей экспертов или просто экспертов), а интегрирующий элемент называется сетью шлюза и является посредником между модулями (экспертами). Выходные сигналы модулей (экспертов) и общий выходной сигнал могут быть вектором в случае нескольких классов или нескольких выходных переменных. Каждый модуль (эксперт) представляет собой линейный фильтр.

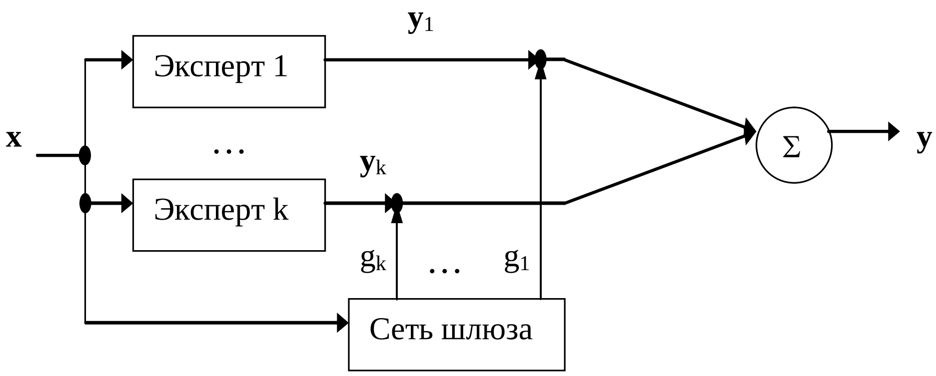


Рис. 3.9. Смесь экспертов (ME)

Обычно используются два алгоритма обучения:

– алгоритм EM (обучение без учителя), который обеспечивает максимизацию логарифмической функции правдоподобия;

– обучение на основе коррекции ошибок (обучение с учителем), при этом чаще всего используется итеративный градиентный алгоритм обучения, который обеспечивает минимизацию среднеквадратичной ошибки и максимизацию логарифмической функции правдоподобия.

Поскольку для ME первый метод требует громоздких вычислений, ограничимся рассмотрением второго метода.

Обучение ИНС

1. Номер итерации обучения $n=1$, инициализация посредством равномерного распределения на интервале (0,1) или [-0.5, 0.5] весов сети экспертов $w_{jls}(n)$, весов сети шлюзов $a_{js}(n)$, $j \in \overline{1,J}$, $l \in \overline{1,N^{(0)}}, s \in \overline{1,N^{(L)}}$, где $N^{(0)}$ – количество нейронов входного слоя, $N^{(L)}$ – количество нейронов выходного слоя, J – количество модулей (экспертов).

2. Задается обучающее множество $\{(\mathbf{x}_\mu, \mathbf{d}_\mu) \mid \mathbf{x}_\mu \in R^{N^{(0)}}, \mathbf{d}_\mu \in R^{N^{(L)}}\}$, $\mu \in \overline{1,P}$, где $\mathbf{x}_\mu$ – μ-й обучающий входной вектор, $\mathbf{d}_\mu$ – μ-й обучающий выходной вектор, P – мощность обучающего множества. Номер текущей пары из обучающего множества $\mu = 1$.

3. Вычисление априорных вероятностей (значений функций g_j) для каждого модуля

$$u_j(n) = \sum_{s=1}^{N^{(0)}} a_{js}(n) x_{\mu s}, \ j \in \overline{1,J},$$

$$g_j(n) = \frac{\exp(u_j(n))}{\sum_{s=1}^{J} \exp(u_s(n))}, \ j \in \overline{1,J}.$$

4. Вычисление апостериорных вероятностей (значений функций h_j) для каждого модуля

$$y_{jl}(n) = \sum_{s=1}^{N^{(0)}} w_{jls}(n) x_{\mu s}, \ j \in \overline{1,J}, \ l \in \overline{1,N^{(L)}},$$

$$G_j(n) = \frac{1}{\sqrt{(2\pi)^{N^{(L)}}}} \exp\left(-\frac{1}{2} (\mathbf{d}_\mu - \mathbf{y}_j(n))^T (\mathbf{d}_\mu - \mathbf{y}_j(n)) \right),$$

$$h_j(n) = \frac{g_j(n) G_j(n)}{\sum_{s=1}^{J} g_s(n) G_s(n)}, \ j \in \overline{1,J}.$$

5. Вычисление выходного сигнала всей ИНС

$$y_l(n) = \sum_{j=1}^{J} g_j(n) y_{jl}(n), \ l \in \overline{1,N^{(L)}}.$$

6. Вычисление энергии ошибки всей ИНС

$$E(n) = \frac{1}{2} \sum_{l=1}^{N^{(L)}} (d_{\mu l} - y_l(n))^2.$$

7. Настройка синаптических весов на основе обобщенного дельта правила

$$w_{jls}(n+1) = w_{jls}(n+1) + \eta_1 \frac{\partial L(n+1)}{\partial w_{jls}(n+1)},$$

$$a_{js}(n+1) = a_{js}(n+1) + \eta_2 \frac{\partial L(n+1)}{\partial a_{js}(n+1)},$$

$$\frac{\partial L(n+1)}{\partial w_{jls}(n+1)} = h_j(n)(d_{\mu l} - y_{jl}(n))x_{\mu s},$$

$$\frac{\partial L(n+1)}{\partial a_{js}(n+1)} = (h_j(n) - g_j(n))x_{\mu s},$$

$$j \in \overline{1, J},\ l \in \overline{1, N^{(L)}},\ s \in \overline{1, N^{(0)}},$$

где $\ln L(\mathbf{X} \mid \mathbf{\Theta}) = \sum_{\mu=1}^{P} \ln \sum_{j=1}^{J} g_j(\mu) G_j(\mu)$ – логарифмическая функция правдоподобия, вектор параметров $\mathbf{\Theta}$ содержит веса сети экспертов w_{jls} и веса сети шлюзов a_{jks}.

8. Проверка условия завершения

Если $n \bmod P > 0$, то $\mu = \mu + 1$, $n = n + 1$, переход к 3.

Если $n \bmod P = 0$ и $\frac{1}{P}\sum_{s=1}^{P} E(n - P + s) > \varepsilon$, то $n = n + 1$, переход к 2.

Если $n \bmod P = 0$ и $\frac{1}{P}\sum_{s=1}^{P} E(n - P + s) < \varepsilon$, то завершиться.

Функционирование ИНС

$$u_j = \sum_{s=1}^{N^{(0)}} a_{js} x_s,\ j \in \overline{1, J},$$

$$g_j = \frac{\exp(u_j)}{\sum_{s=1}^{J} \exp(u_s)},\ j \in \overline{1, J},$$

$$y_{jl} = \sum_{s=1}^{N^{(0)}} w_{jls} x_s,\ j \in \overline{1, J},\ l \in \overline{1, N^{(L)}},$$

$$y_l = \sum_{j=1}^{J} g_j y_{jl},\ l \in \overline{1, N^{(L)}}$$

Достоинства

1. Используется для классификации образцов.

2. Является универсальным аппроксиматором. ME использует нелинейность, аналогично MLP, но не для построения отображения входного сигнала в выходной, а с целью разбиения входного пространства между экспертами.

3. Количество классов может быть больше двух.

4. Использует сеть шлюзов для объединения знаний, накопленных разными модулями, в общее решение, которое имеет приоритет над каждым решением отдельного модуля.

5. Обеспечивает хорошее качество обобщения.

Недостатки

1. В ряде случаев MLP может дать более точный результат.

2. Модули – только линейные фильтры.

3. Отсутствует автоматическое определение числа модулей.

4. В отличие от методов обучения SVM, градиентные методы обучения ME выбирают положение разделяющей гиперплоскости произвольным образом.

5. Обучение не сводится к задаче квадратичного программирования в выпуклой области, имеющей единственное решение.

6. В отличие от ART, не решает проблему пластичности-стабильности.

3.9. Иерархическая смесь экспертов

На рис. 3.10 приведен простейший вариант иерархической смеси экспертов (HME) [20], которая является нерекуррентной статической модульной ИНС. Архитектура HME подобна дереву, в котором сети шлюзов являются ветвями, а отдельные модули (эксперты) – листьями. Выходные сигналы модулей (экспертов) и общий выходной сигнал могут быть вектором в случае нескольких классов или нескольких выходных переменных. Уровней иерархии может быть два и более. Модулей (экспертов), связанных с определенным шлюзом, в HME два. Каждый модуль (эксперт) представляет собой линейный фильтр.

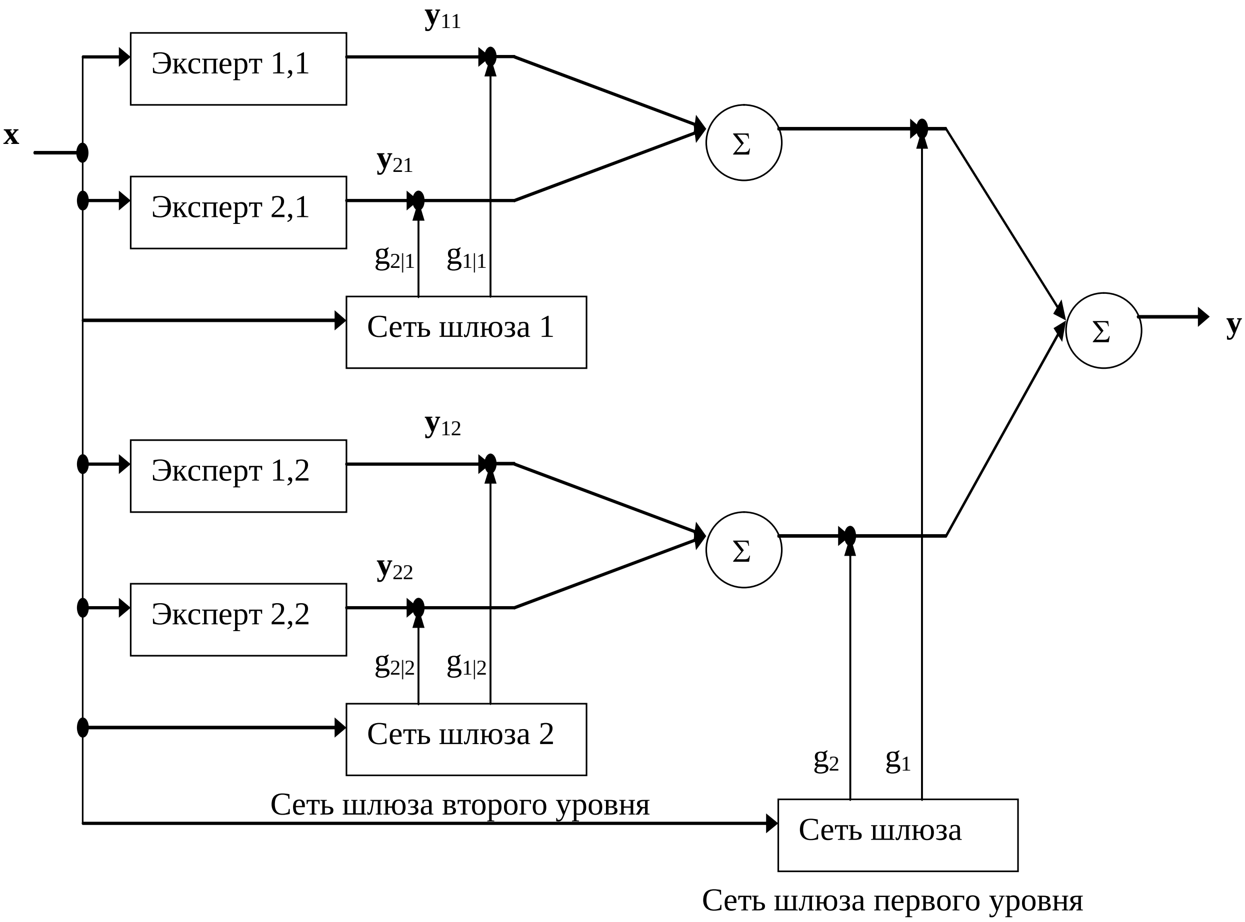


Рис. 3.10. Иерархическая смесь экспертов (HME) (два уровня)

Обычно используются два алгоритма обучения:

– алгоритм EM (обучение без учителя), который обеспечивает максимизацию логарифмической функции правдоподобия;

– обучение на основе коррекции ошибок (обучение с учителем), при этом чаще всего используется итеративный градиентный алгоритм обучения, который обеспечивает минимизацию среднеквадратичной ошибки и максимизацию логарифмической функции правдоподобия.

Поскольку для HME первый алгоритм требует громоздких вычислений, ограничимся рассмотрением второго алгоритма.

Обучение ИНС

1. Номер итерации обучения $n=1$, инициализация посредством равномерного распределения на интервале (0,1) или [-0.5, 0.5] весов сети экспертов $w_{jkls}(n)$, весов сети шлюзов первого $a_{ks}(n)$ и второго $a_{jks}(n)$ уровней, $j \in \overline{1,2}$, $k \in \overline{1,K}$, $l \in \overline{1,N^{(0)}}, s \in \overline{1,N^{(L)}}$, где $N^{(0)}$ –

количество нейронов входного слоя, $N^{(L)}$ – количество нейронов выходного слоя, K – количество сетей шлюзов на втором уровне.

2. Задается обучающее множество $\{(\mathbf{x}_\mu, \mathbf{d}_\mu) \mid \mathbf{x}_\mu \in R^{N^{(0)}}, \mathbf{d}_\mu \in R^{N^{(L)}}\}$, $\mu \in \overline{1, P}$, где $\mathbf{x}_\mu$ – μ-й обучающий входной вектор, $\mathbf{d}_\mu$ – μ-й обучающий выходной вектор, P – мощность обучающего множества. Номер текущей пары из обучающего множества $\mu = 1$.

3. Вычисление априорных вероятностей (значений функций g_k и $g_{j|k}$) для каждого модуля

$$u_k(n) = \sum_{s=1}^{N^{(0)}} a_{ks}(n) x_{\mu s}, \quad k \in \overline{1, K},$$

$$u_{jk}(n) = \sum_{s=1}^{N^{(0)}} a_{jks}(n) x_{\mu s}, \quad j \in \overline{1, 2}, \quad k \in \overline{1, K},$$

$$g_k(n) = \frac{\exp(u_k(n))}{\sum_{s=1}^{K} \exp(u_s(n))}, \quad k \in \overline{1, K},$$

$$g_{j|k}(n) = \frac{\exp(u_{jk}(n))}{\sum_{s=1}^{2} \exp(u_{sk}(n))}, \quad j \in \overline{1, 2}, \quad k \in \overline{1, K}.$$

4. Вычисление апостериорных вероятностей (значений функций h_k, $h_{j|k}$ и h_{jk}) для каждого модуля

$$y_{jkl}(n) = \sum_{s=1}^{N^{(0)}} w_{jkls}(n) x_{\mu s}, \quad j \in \overline{1, 2}, \quad k \in \overline{1, K}, \quad l \in \overline{1, N^{(L)}},$$

$$G_{j|k}(n) = \frac{1}{\sqrt{(2\pi)^{N^{(L)}}}} \exp\left(-\frac{1}{2} (\mathbf{d}_\mu - \mathbf{y}_{jk}(n))^T (\mathbf{d}_\mu - \mathbf{y}_{jk}(n)) \right),$$

$$h_k(n) = \frac{g_k(n) \sum_{j=1}^{2} g_{j|k}(n) G_{j|k}(n)}{\sum_{k=1}^{K} g_k(n) \sum_{j=1}^{2} g_{j|k}(n) G_{j|k}(n)}, \quad j \in \overline{1, 2}, \quad k \in \overline{1, K},$$

$$h_{j|k}(n) = \frac{g_{j|k}(n) G_{j|k}(n)}{\sum\limits_{j=1}^{2} g_{j|k}(n) G_{j|k}(n)}, \; j \in \overline{1,2}, \; k \in \overline{1,K},$$

$$h_{jk}(n) = h_k(n) h_{j|k}(n) = \frac{g_k(n) g_{j|k}(n) G_{j|k}(n)}{\sum\limits_{k=1}^{K} g_k(n) \sum\limits_{j=1}^{2} g_{j|k}(n) G_{j|k}(n)}, \; j \in \overline{1,2}, k \in \overline{1,K}.$$

5. Вычисление выходного сигнала всей ИНС

$$y_l(n) = \sum_{k=1}^{K} g_k(n) \sum_{j=1}^{2} g_{j|k}(n) y_{jkl}(n), \; l \in \overline{1, N^{(L)}}$$

6. Вычисление энергии ошибки всей ИНС

$$E(n) = \frac{1}{2} \sum_{l=1}^{N^{(L)}} (d_{\mu l} - y_l(n))^2 .$$

7. Настройка синаптических весов на основе обобщенного дельта правила

$$w_{jkls}(n+1) = w_{jkls}(n+1) + \text{¦}_1 \frac{\partial L(n+1)}{\partial w_{jkls}(n+1)},$$

$$a_{ks}(n+1) = a_{ks}(n+1) + \text{¦}_2 \frac{\partial L(n+1)}{\partial a_{kl}(n+1)},$$

$$a_{jks}(n+1) = a_{jks}(n+1) + \text{¦}_3 \frac{\partial L(n+1)}{\partial a_{jkl}(n+1)},$$

где η_1, η_2, η_3 – параметры, определяющие скорость обучения (при больших η_1, η_2, η_3 обучение происходит быстрее, но увеличивается опасность получить неверное решение), $0 < \eta_1 < 1, 0 < \eta_2 < 1, 0 < \eta_3 < 1$.

$$\frac{\partial L(n+1)}{\partial w_{jkls}(n+1)} = h_{j|k}(n) h_k(n) (d_{\mu l} - y_{jkl}(n)) x_{\mu s},$$

$$\frac{\partial L(n+1)}{\partial a_{ks}(n+1)} = (h_k(n) - g_k(n)) x_{«s},$$

$$\frac{\partial L(n+1)}{\partial a_{jks}(n+1)} = h_k(n)(h_{j|k}(n) - g_{j|k}(n)) x_{«s},$$

$j \in \overline{1,2}, \; k \in \overline{1,K}, \; l \in \overline{1, N^{(L)}}, \; s \in \overline{1, N^{(0)}},$

где $\ln L(\mathbf{X}|\mathbf{\Theta}) = \sum_{\mu=1}^{P} \ln \sum_{k=1}^{K} g_k(\mu) \sum_{j=1}^{2} g_{j|k}(\mu) G_{j|k}(\mu)$ – логарифмированная функция правдоподобия, вектор параметров $\mathbf{\Theta}$ содержит веса сети экспертов w_{jkls} и веса сети шлюзов первого a_{ks} и второго a_{jks} уровней.

8. Проверка условия завершения

Если $n \bmod P > 0$, то $\mu = \mu + 1$, $n = n + 1$, переход к 3.

Если $n \bmod P = 0$ и $\frac{1}{P} \sum_{s=1}^{P} E(n - P + s) > \varepsilon$, то $n = n + 1$, переход к 2.

Если $n \bmod P = 0$ и $\frac{1}{P} \sum_{s=1}^{P} E(n - P + s) < \varepsilon$, то завершиться.

Функционирование ИНС

$$u_k = \sum_{s=1}^{N^{(0)}} a_{ks} x_s, \ k \in \overline{1, K},$$

$$u_{jk} = \sum_{s=1}^{N^{(0)}} a_{jks} x_s, \ j \in \overline{1, 2}, \ k \in \overline{1, K},$$

$$g_k = \frac{\exp(u_k)}{\sum_{s=1}^{K} \exp(u_s)}, \ k \in \overline{1, K},$$

$$g_{j|k} = \frac{\exp(u_{jk})}{\sum_{s=1}^{2} \exp(u_{sk})}, \ j \in \overline{1, 2}, \ k \in \overline{1, K},$$

$$y_{jkl} = \sum_{s=1}^{N^{(0)}} w_{jkls} x_s, \ j \in \overline{1, 2}, \ k \in \overline{1, K}, \ l \in \overline{1, N^{(L)}},$$

$$y_l = \sum_{k=1}^{K} g_k \sum_{j=1}^{2} g_{j|k} y_{jkl}, \ l \in \overline{1, N^{(L)}}.$$

Достоинства

1. Используется для классификации образцов.

2. Является универсальным аппроксиматором. В отличие от дерева решений, HME осуществляет не «жесткое», а «мягкое» разбиение входного пространства, и поэтому является деревом мягких решений.

3. Обеспечивает хорошее качество обобщения.

4. Использует нелинейность, аналогично MLP, но не для построения отображения входного сигнала в выходной, а с целью разбиения входного пространства между модулями.

5. Использует сеть шлюзов для объединения знаний, накопленных разными модулями, в общее решение, которое имеет приоритет над каждым решением отдельного модуля.

6. В отличие от ME, входное пространство разбивается на множество вложенных подпространств, а информация объединяется и перераспределяется между модулями под управлением нескольких сетей шлюзов, организованных в иерархическую структуру.

7. Структура HME может быть определена с помощью алгоритма CART (дерево классификации и регрессии).

Недостатки

1. В ряде случаев MLP может дать более точный результат.

2. Эксперты – только линейные фильтры.

3. Отсутствует автоматическое определение числа уровней иерархии.

4. В отличие от методов обучения SVM, градиентные методы обучения HME выбирают положение разделяющей гиперплоскости произвольным образом.

5. Обучение не сводится к задаче квадратичного программирования в выпуклой области, имеющей единственное решение.

6. В отличие от ART, не решает проблему пластичности-стабильности.

3.10. Сплайновая нейронная сеть

На рис. 3.11 приведена сплайновая нейронная сеть (SNN) на основе сплайна Катмулла-Рома [21], которая является нерекуррентной статической двухслойной ИНС. Базисным функциям $C_j(x)$ соответствуют нейроны скрытого слоя. В отличие от MLP выходной слой является линейным.

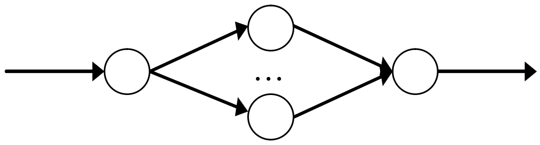


Рис. 3.11. Сплайновая нейронная сеть (SNN)

Сплайн Катмулла-Рома является кусочно-полиномиальной функцией и представлен в виде [22]

$$S(x)=\sum_{j=0}^{r-4}P_jC_j(x),\ x\in[\lambda_K,\lambda_{r-K-1}],$$

$$C_j(x)=\frac{-x^3+2x^2-x}{2},\qquad C_{j+1}(x)=\frac{3x^3-5x^2+2}{2},$$

$$C_{j+2}(x)=\frac{-3x^3+4x^2+x}{2},\ C_{j+3}(x)=\frac{x^3-x^2}{2},\ j\in\overline{0,r-4},$$

$$x^{\min}<\lambda_0\le\ldots\le\lambda_{r-1}<x^{\max},$$

где $C_j(x)$ – базисные функции степени 3 (порядка 4),

P_j – контрольные (опорные, базовые) точки,

λ_j – узлы,

r – количество узлов.

Узловой вектор $\lambda=(\lambda_0,\ldots,\lambda_{r-1})$ является равномерным (все узлы находятся на равном расстоянии) и $x\in[0,1]$ (т.е. сплайн является нормализованным).

Для SNN используется обучение на основе коррекции ошибок (обучение с учителем), при этом чаще всего применяется алгоритм обратного распространения (BP). Это итеративный градиентный алгоритм обучения, который обеспечивает минимизацию среднеквадратичной ошибки.

Обучение ИНС (алгоритм обратного распространения)

1. Номер итерации обучения $n=1$, инициализация посредством равномерного распределения на интервале (0,1) или [-0.5, 0.5] весов $w_j(n)$, $j\in\overline{0,N^{(1)}}$, где $N^{(1)}$ – количество нейронов в первом слое (количество базисных функций). Задаются все узлы λ_j, $j\in\overline{0,r-1}$.

2. Задается обучающее множество $\{(x_\mu, d_\mu) \mid x_\mu \in R, d_\mu \in R\}$, $\mu \in \overline{1, P}$, где x_μ – μ-е обучающее входное значение, d_μ – μ-е обучающее выходное значение. Номер текущей пары из обучающего множества $\mu = 1$.

3. Вычисление выходного сигнала для первого слоя (прямой ход)

$$f_\mu(n) = \sum_{j=0}^{N^{(1)}} w_j(n) C_j(x), \ \mu \in \overline{1, P}$$

4. Вычисление энергии ошибки ИНС

$$E(n) = \frac{1}{2P} \sum_{\mu=1}^{P} e_\mu^2(n), \ e_\mu(n) = f_\mu(n) - d_\mu$$

5. Весовые коэффициенты выходного слоя вычисляются на основе обобщенного дельта правила (обратный ход)

$$w_j(n+1) = w_j(n) - \eta \frac{\partial E(n)}{\partial w_j(n)},$$

где η – параметры, определяющие скорость обучения, $0 < \eta < 1$.

$$\frac{\partial E(n)}{\partial w_j(n)} = \frac{1}{P} \sum_{\mu=1}^{P} (f_\mu(n) - d_\mu) C_j(x)$$

6. Проверка условия завершения

Если $E(n) < \varepsilon$, то завершиться, иначе $n = n + 1$, переход к 2.

Функционирование ИНС

$$f(x) = \sum_{j=0}^{N^{(1)}} w_j C_j(x)$$

Достоинства

1. Является универсальным аппроксиматором. Обеспечивает локальную аппроксимацию нелинейного отображения входного сигнала в выходной.

2. Обеспечивает хорошее качество обобщения.

3. Автоматически определяется количество скрытых слоев (равно одному).

4. Обеспечивает аппроксимацию лучше, чем MLP, RBFNN.

5. Количество нейронов в скрытых слоях меньше, чем у PNN, поэтому SNN функционирует быстрее, чем PNN.

6. Сплайн Катмулла-Рома в отличие от обычного кубического сплайна не требует больших расчетов.

Недостатки

1. Отсутствует автоматическое определение числа нейронов в скрытом слое.

2. Количество нейронов в скрытом слое больше, чем у MLP, поэтому SNN функционирует медленнее, чем MLP.

3. В отличие от методов обучения SVM, градиентные методы обучения SNN выбирают положение разделяющей гиперплоскости произвольным образом.

4. Обучение не сводится к задаче квадратичного программирования в выпуклой области, имеющей единственное решение.

5. Обучение происходит медленнее, чем в случае PNN, сети Хемминга, SOM, CPNN.

6. В отличие от ART, не решает проблему пластичности-стабильности.

Замечание. Дальнейшим развитием SNN является адаптивная сплайновая нейросеть (ASNN) [23].

3.11. B-сплайновая нейронная сеть

На рис. 3.12 приведена B-сплайновая нейронная сеть (BSNN) [24], которая является нерекуррентной статической двухслойной ИНС. Базисным функциям $N_K^j(x)$ соответствуют нейроны скрытого слоя. В отличие от MLP выходной слой является линейным.

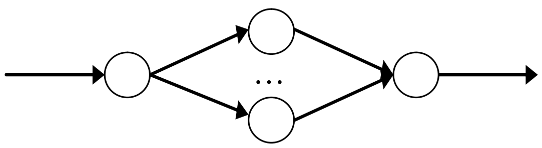


Рис. 3.12. B-сплайновая нейронная сеть (BSNN)

B-сплайн является кусочно-полиномиальной функцией и представлен в виде [22]

$$S(x) = \sum_{j=0}^{r-K-2} P_j N_K^j(x), \; x \in [\lambda_K, \lambda_{r-K-1}],$$

$$N_0^j(x)=\begin{cases}1, & x\in I_j\\ 0, & x\notin I_j\end{cases},\ I_j=[\lambda_j,\lambda_{j+1}],\ j\in\overline{0,r-2},$$

$$N_k^j(x)=\left(\frac{x-\lambda_j}{\lambda_{j+k}-\lambda_j}\right)N_{k-1}^j(x)+\left(\frac{\lambda_{j+k+1}-x}{\lambda_{j+k+1}-\lambda_{j+1}}\right)N_{k-1}^{j+1}(x),$$

$j\in\overline{0,r-k-2}$, $k\in\overline{1,K}$,

$$x^{\min}<\lambda_0\le\ldots\le\lambda_{r-1}<x^{\max},$$

где $N_K^j(x)$ – базисные функции степени K (порядка $K+1$),

P_j – контрольные (опорные, базовые) точки,

λ_j – узлы,

r – количество узлов.

На степень K налагается условие $0\le K\le r-2$.

Для базисных функций выполняются условия

$\forall x\ \ N_k^j(x)\ge 0$ при этом если $x\in[\lambda_j,\lambda_{j+k+1}]$, то $N_k^j(x)>0$

$$\forall x\in[\lambda_k,\lambda_{r-k-1}]\ \ \sum_{j=0}^{r-k-2}N_k^j(x)=1,\ k\in\overline{1,K}$$

Узловой вектор $\lambda=(\lambda_0,\ldots,\lambda_{r-1})$ может быть равномерным (все узлы находятся на равном расстоянии) или неравномерным (узлы на разном расстоянии или повторяются).

B-сплайн степени $K=0$ является константным, степени $K=1$ – линейным, степени $K=2$ – квадратичным, степени $K=3$ – кубическим.

Для равномерных узловых векторов и $x\in[0,1]$ справедливы формулы:

– для нормализованного квадратичного B-сплайна

$$N_2^j(x)=\frac{x^2-2x+1}{2},\qquad N_2^{j+1}(x)=\frac{-2x^2+2x+1}{2},\qquad N_2^{j+2}(x)=\frac{x^2}{2},$$

$j\in\overline{0,r-3}$;

– для нормализованного кубического B-сплайна

$$N_3^j(x)=\frac{-x^3+3x^2-3x+1}{6},\qquad N_3^{j+1}(x)=\frac{3x^3+6x^2+4}{6},$$

$$N_3^{j+2}(x)=\frac{-x^3+3x^2+3x+1}{6},\ N_3^{j+3}(x)=\frac{x^3}{6},\ j\in\overline{0,r-4}.$$

Для BSNN используется обучение на основе коррекции ошибок (обучение с учителем), при этом чаще всего применяется алгоритм обратного распространения (BP). Это итеративный градиентный алгоритм обучения, который обеспечивает минимизацию среднеквадратичной ошибки.

Обучение ИНС (алгоритм обратного распространения)

1. Номер итерации обучения $n=1$, инициализация посредством равномерного распределения на интервале (0,1) или [-0.5, 0.5] весов $w_j(n)$, $j \in \overline{0, N^{(1)}}$, где $N^{(1)}$ – количество нейронов в первом слое (количество базисных функций). Задаются все узлы λ_j, $j \in \overline{0, r-1}$.

2. Задается обучающее множество $\{(x_\mu, d_\mu) \mid x_\mu \in R, d_\mu \in R\}$, $\mu \in \overline{1, P}$, где x_μ – μ-е обучающее входное значение, d_μ – μ-е обучающее выходное значение. Номер текущей пары из обучающего множества $\mu = 1$.

3. Вычисление выходного сигнала для первого слоя (прямой ход)

$$f_\mu(n) = \sum_{j=0}^{N^{(1)}} w_j(n) N_K^j(x), \ \mu \in \overline{1, P}$$

4. Вычисление энергии ошибки ИНС

$$E(n) = \frac{1}{2P} \sum_{\mu=1}^{P} e_\mu^2(n), \ e_\mu(n) = f_\mu(n) - d_\mu$$

5. Весовые коэффициенты выходного слоя вычисляются на основе обобщенного дельта правила (обратный ход)

$$w_j(n+1) = w_j(n) - \eta \frac{\partial E(n)}{\partial w_j(n)},$$

где η – параметры, определяющие скорость обучения, $0 < \eta < 1$.

$$\frac{\partial E(n)}{\partial w_j(n)} = \frac{1}{P} \sum_{\mu=1}^{P} (f_\mu(n) - d_\mu) N_K^j(x)$$

6. Проверка условия завершения

Если $E(n) < \varepsilon$, то завершиться, иначе $n = n+1$, переход к 2.

Функционирование ИНС

$$f(x)=\sum_{j=0}^{N^{(1)}} w_j N_K^j(x)$$

Достоинства

1. Является универсальным аппроксиматором. Обеспечивает локальную аппроксимацию нелинейного отображения входного сигнала в выходной.

2. Обеспечивает хорошее качество обобщения.

3. Автоматически определяется количество скрытых слоев (равно одному).

4. Обеспечивает аппроксимацию лучше, чем MLP, RBFNN.

5. Количество нейронов в скрытых слоях меньше, чем у PNN, поэтому BSNN функционирует быстрее, чем PNN.

6. B-сплайны являются обобщением кривых Безье и имеют преимущества перед ними:

–влияние каждой вершины на B-сплайн проявляется только при тех значениях параметра, где соответствующая базисная функция не равна нулю;

– возможность изменять порядок базисных функций и, следовательно, всего B-сплайна без изменения количества вершин.

Недостатки

1. Отсутствует автоматическое определение числа нейронов в скрытом слое.

2. Количество нейронов в скрытом слое больше, чем у MLP, поэтому BSNN функционирует медленнее, чем MLP.

3. В отличие от методов обучения SVM, градиентные методы обучения BSNN выбирают положение разделяющей гиперплоскости произвольным образом.

4. Обучение не сводится к задаче квадратичного программирования в выпуклой области, имеющей единственное решение.

5. Обучение происходит медленнее, чем в случае PNN, сети Хемминга, SOM, CPNN.

6. В отличие от ART, не решает проблему пластичности-стабильности.

3.12. Вейвлетная нейронная сеть

На рис.3.13 приведена вейвлетная нейронная сеть (WNN) [25], которая является нерекуррентной статической двухслойной ИНС. Вейвлет-функциям соответствуют нейроны скрытого слоя. В отличие от MLP выходной слой является линейным.

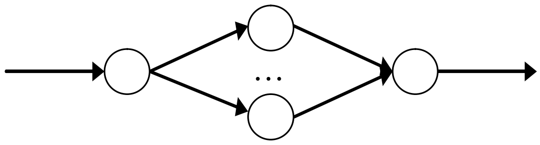


Рис. 3.13. Вейвлетная нейронная сеть (WNN)

Для дискретных значений параметров масштаба (a) и сдвига (b) вейвлет-функция представляется в виде

$$\psi_{ml}(t) = \frac{1}{\sqrt{a_m}}\psi\left(\frac{t-b_l}{a_m}\right),$$

Для речевых сигналов обычно используют вейвлет Морле

$$\psi(\xi) = (2\pi)^{-1/2}\cos(\omega_0\xi)e^{-\xi^2/2},$$

где ω_0 – константа.

Для WNN используется обучение на основе коррекции ошибок (обучение с учителем), при этом чаще всего применяется алгоритм обратного распространения (BP). Это итеративный градиентный алгоритм обучения, который обеспечивает минимизацию среднеквадратичной ошибки.

Обучение ИНС (алгоритм обратного распространения)

1. Номер итерации обучения $n=1$, инициализация посредством равномерного распределения на интервале (0,1) или [-0.5, 0.5] смещения (порога) θ_0 и весов $w_i(n)$, инициализация положительными числами параметров масштаба a_i и параметров сдвига b_i, $i \in \overline{1, N^{(1)}}$, где $N^{(1)}$ – количество нейронов в первом слое.

2. Задается обучающее множество $\{(x_\mu, d_\mu) \mid x_\mu \in R, d_\mu \in R\}$, $n \in \overline{1, P}$, где x_μ – μ-е обучающее входное значение, d_μ – μ-е обучающее выходное значение. Номер текущей пары из обучающего множества $\mu = 1$.

3. Вычисление выходного сигнала для первого слоя (прямой ход)

$$f_\mu(n)=\sum_{i=0}^{N^{(1)}} w_i(n)\frac{1}{\sqrt{a_i}}\psi\left(\frac{x_\mu - b_i}{a_i}\right),\ \mu\in\overline{1,P}.$$

Считается, что $w_0(n)=\theta_0, \frac{1}{\sqrt{a_0}}\psi\left(\frac{x_\mu - b_0}{a_0}\right)=1.$

4. Вычисление энергии ошибки ИНС

$$E(n)=\frac{1}{2P}\sum_{\mu=1}^{P} e_\mu^2(n),$$

$$e_\mu(n)=f_\mu(n)-d_\mu$$

5. Весовые коэффициенты выходного слоя, параметры масштаба и сдвига вычисляются на основе обобщенного дельта правила (обратный ход)

$$w_i(n+1)=w_i(n)-\eta_1\frac{\partial E(n)}{\partial w_i(n)},$$

$$a_i(n+1)=a_i(n)-\eta_2\frac{\partial E(n)}{\partial a_i(n)},$$

$$b_i(n+1)=b_i(n)-\eta_3\frac{\partial E(n)}{\partial b_i(n)},$$

где η_1,η_2,η_3 – параметры, определяющие скорость обучения, $0<\eta_1<1, 0<\eta_2<1, 0<\eta_3<1.$

$$\frac{\partial E(n)}{\partial w_i(n)}=\frac{1}{P}\sum_{\mu=1}^{P}(f_\mu(n)-d_\mu)\frac{1}{\sqrt{a_i}}\psi\left(\frac{x_\mu - b_i}{a_i}\right),$$

$$\frac{\partial E(n)}{\partial a_i(n)}=-\frac{1}{P}\sum_{\mu=1}^{P}(f_\mu(n)-d_\mu)\frac{w_i}{a_i\sqrt{a_i}}\cdot$$

$$\cdot\left(\frac{1}{2}\psi\left(\frac{x_\mu - b_i}{a_i}\right)+\frac{x_\mu - b_i}{a_i}\psi'\left(\frac{x_\mu - b_i}{a_i}\right)\right)$$

$$\frac{\partial E(n)}{\partial b_i(n)}=-\frac{1}{P}\sum_{\mu=1}^{P}(f_\mu(n)-d_\mu)\frac{w_i}{a_i\sqrt{a_i}}\psi'\left(\frac{x_\mu - b_i}{a_i}\right),$$

Например, для Морле

$$\psi'(\xi)=-(2\pi)^{-1/2}[\omega_0\sin(\omega_0\xi)+\xi\cos(\omega_0\xi)]e^{-\xi^2/2}$$

6. Проверка условия завершения

Если $E(n)<\varepsilon$, то завершиться, иначе $n=n+1$, переход к 2.

Функционирование ИНС

$$f(x) = \theta_0 + \sum_{i=1}^{N^{(1)}} w_i \frac{1}{\sqrt{a_i}} \psi\left(\frac{x - b_i}{a_i} \right).$$

Достоинства

1. Является универсальным аппроксиматором. Обеспечивает локальную аппроксимацию нелинейного отображения входного сигнала в выходной.

2. Обеспечивает хорошее качество обобщения.

3. Автоматически определяется количество скрытых слоев (равно одному).

4. Обеспечивает аппроксимацию лучше, чем MLP, RBFN.

5. Количество нейронов в скрытых слоях меньше, чем у PNN, поэтому BSNN функционирует быстрее, чем PNN.

Недостатки

1. Отсутствует автоматическое определение числа нейронов в скрытом слое.

2. Количество нейронов в скрытом слое больше, чем у MLP, поэтому WNN функционирует медленнее, чем MLP.

3. В отличие от методов обучения SVM, градиентные методы обучения WNN выбирают положение разделяющей гиперплоскости произвольным образом.

4. Обучение не сводится к задаче квадратичного программирования в выпуклой области, имеющей единственное решение.

5. Обучение происходит медленнее, чем в случае PNN, сети Хемминга, SOM, CPNN.

6. В отличие от ART, не решает проблему пластичности-стабильности.

3.13. Каскадно-корреляционная нейросеть

На рис. 3.14 приведена каскадно-корреляционная нейросеть (CCNN) [26], который является нерекуррентной статической многослойной ИНС, содержащей один или более скрытых слоев и выходной слой. Классы разделяются гиперплоскостями.

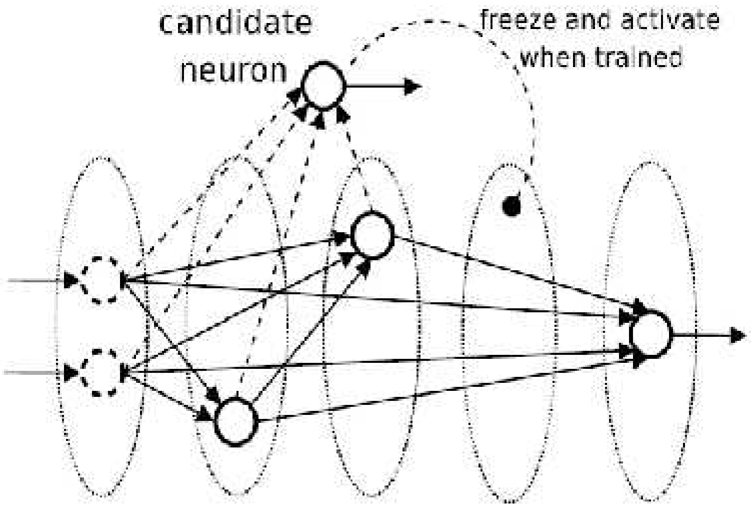


Рис. 3.14. Каскадно-корреляционная нейросеть (CCNN)

Для CCNN используется обучение на основе коррекции ошибок (обучение с учителем), при этом применяется алгоритм обратного распространения (BP) или аналогичные ему, но более быстрые алгоритмы. Это итеративные градиентные алгоритмы обучения, которые обеспечивают минимизацию среднеквадратичной ошибки.

На начальном этапе формируется ИНС, состоящая только из входного и выходного слоев. Происходит обучение ИНС. Если ИНС дает небольшую ошибку, то скрытые слои не добавляются и формирование ИНС завершается. Если ИНС дает большую ошибку, то создается новый слой. Процесс создания нового слоя заключается в следующем. По обученной ИНС вычисляются ошибки каждого нейрона выходного слоя и средняя ошибка нейрона выходного слоя. Затем входной слой соединяется с новым скрытым слоем. Настраиваются только веса между нейронами входного слоя и единственным нейроном нового слоя, используя ковариацию между выходом нового нейрона и ошибкой каждого нейрона выходного слоя. Затем выходной слой соединяется с новым скрытым слоем. Настраиваются только веса между единственным нейроном нового слоя и нейронами выходного слоя. Процесс создания новых слоев с единственным нейроном продолжается, пока ИНС дает большую ошибку. Для более точной работы ИНС может параллельно использоваться не один нейрон, а несколько нейронов-кандидатов (от 5 до 10) с разными функциями активации и начальными весами, из которых будет выбран лучший.

Обучение ИНС (алгоритм обратного распространения)

Обучение весов между входным и выходным слоем

1. Номер итерации обучения $n = 1$, инициализация посредством равномерного распределения на интервале (0,1) или [-0.5, 0.5] весов $w_{ij}^{(0,1)}(n)$, $i \in \overline{1, N^{(0)}}$, $j \in \overline{1, N^{(1)}}$, где $N^{(k)}$ – количество нейронов в k-м слое.

2. Задается обучающее множество $\{(\mathbf{x}_\mu, \mathbf{d}_\mu) \mid \mathbf{x}_\mu \in R^{N^{(0)}}, \mathbf{d}_\mu \in R^{N^{(1)}}\}$, $\mu \in \overline{1, P}$, где $\mathbf{x}_\mu$ – μ-й обучающий входной вектор, $\mathbf{d}_\mu$ – μ-й обучающий выходной вектор, $N^{(0)}$ – количество нейронов входного слоя, $N^{(1)}$ – количество нейронов выходного слоя, P – мощность обучающего множества. Номер текущей пары из обучающего множества $\mu = 1$.

3. Вычисление выходного сигнала для выходного слоя (прямой ход)

$$y_i^{(0)}(n) = x_{\mu i},$$

$$y_j^{(1)}(n) = f^{(1)}(s_j^{(1)}(n)), s_j^{(1)}(n) = \sum_{i=1}^{N^{(0)}} w_{ij}^{(0,1)}(n) y_i^{(0)}(n), \; j \in \overline{1, N^{(1)}},$$

где $N^{(k)}$ – число нейронов в k-м слое, k – номер слоя, $w_{ij}^{(0,1)}(n)$ – вес связи от i-го нейрона входного слоя к j-му нейрону выходного слоя в момент времени n, $y_j^{(k)}(n)$ – выход j-го нейрона на k-м слое, $f^{(k)}$ – функция активация нейронов k-го слоя.

4. Вычисление энергии ошибки ИНС

$$E(n) = \frac{1}{2} \sum_{j=1}^{N^{(1)}} e_j^2(n), \; e_j(n) = y_j^{(1)}(n) - d_{\mu j}$$

5. Настройка синаптических весов на основе обобщенного дельта правила (обратный ход)

$$w_{ij}^{(0,1)}(n+1) = w_{ij}^{(0,1)}(n) - \eta \frac{\partial E(n)}{\partial w_{ij}^{(0,1)}(n)},$$

где η – параметр, определяющий скорость обучения (при большом η обучение происходит быстрее, но увеличивается опасность получить неверное решение), $0 < \eta < 1$.

$$\frac{\partial E(n)}{\partial w_{ij}^{(0,1)}(n)} = y_i^{(0)}(n) f'^{(1)}(s_j^{(1)}(n))(y_j^{(1)}(n) - d_{\mu j}), \qquad i \in \overline{1, N^{(0)}},$$

$j \in \overline{1, N^{(1)}}$

6. Проверка условия завершения

Если $n \bmod P > 0$, то $\mu = \mu + 1$, $n = n + 1$, переход к 3.

Если $n \bmod P = 0$ и $\frac{1}{P}\sum_{s=1}^{P} E(n - P + s) > \varepsilon$, то $n = n + 1$, переход к 2.

Если $n \bmod P = 0$ и $\frac{1}{P}\sum_{s=1}^{P} E(n - P + s) < \varepsilon$, то $w_{ij}^{(0,1)} = w_{ij}^{(0,1)}(n)$.

Обучение весов между нейронами слоев, предшествующих новому скрытому слою, и единственным нейроном нового скрытого слоя

1. Номер итерации обучения $n = 1$, начальная ковариация $\sigma_j(n-1) = 0$, $j \in \overline{1, N^{(L)}}$, инициализация посредством равномерного распределения на интервале (0,1) или [-0.5, 0.5] весов $w_{i1}^{(k,new)}(n)$, $i \in \overline{1, N^{(k)}}, k \in \overline{0, L-1}$, где $N^{(k)}$ – количество нейронов в k-м слое, L – количество слоев.

2. Задается обучающее множество $\{(\mathbf{x}_\mu, \mathbf{d}_\mu) \mid \mathbf{x}_\mu \in R^{N^{(0)}}, \mathbf{d}_\mu \in R^{N^{(L)}}\}$, $\mu \in \overline{1, P}$, где $\mathbf{x}_\mu$ – μ-й обучающий входной вектор, $\mathbf{d}_\mu$ – μ-й обучающий выходной вектор, $N^{(0)}$ – количество нейронов входного слоя, $N^{(L)}$ – количество нейронов выходного слоя, P – мощность обучающего множества. Номер текущей пары из обучающего множества $\mu = 1$.

3. Вычисление выходного сигнала для каждого слоя

$$y_{\mu i}^{(0)} = x_{\mu i},$$

$$y_{\mu j}^{(k)} = f^{(k)}(s_{\mu j}^{(k)}), \; s_{\mu j}^{(k)} = \sum_{l=0}^{k-1} \sum_{i=1}^{N^{(l)}} w_{ij}^{(l,k)} y_{\mu i}^{(l)}, \; j \in \overline{1, N^{(k)}}, k \in \overline{1, L},$$

где $N^{(k)}$ – число нейронов в k-м слое, k – номер слоя, L – число слоев, $w_{ij}^{(l,k)}$ – вес связи от i-го нейрона на l-м слое к j-му нейрону на k-м слое, $y_{\mu j}^{(k)}$ – выход *j*-го нейрона на *k*-м слое для μ-го обучающего входного вектора, $f^{(k)}$ – функция активация нейронов *k*-го слоя.

4. Вычисление ошибки ИНС для каждого выходного нейрона

$e_{\mu j} = y_{\mu j}^{(1)} - d_{\mu j}$

5. Проверка условия завершения

Если $\mu < P$, то $\mu = \mu + 1$, переход к 3.

6. Вычисление средней ошибки ИНС для каждого выходного нейрона

$$\bar{e}_j = \frac{1}{P}\sum_{s=1}^{P} e_{\mu j}$$

7. Вычисление выходного сигнала для единственного нейрона нового слоя

$$y_{\mu 1}^{(new)}(n) = f^{(new)}(s_{\mu 1}^{(new)}(n)),\ s_{\mu 1}^{(new)}(n) = \sum_{l=0}^{L-1}\sum_{i=1}^{N^{(l)}} w_{i1}^{(l,new)}(n) y_{\mu i}^{(l)},\ \mu \in \overline{1,P},$$

где $N^{(k)}$ – число нейронов в *k*-м слое, *k* – номер слоя, *L* – число слоев, $w_{i1}^{(l,new)}(n)$ – вес связи от *i*-го нейрона на *l*-м слое к единственному нейрону на новом слое в момент времени n, $y_j^{(k)}$ – выход *j*-го нейрона на *k*-м слое, $f^{(k)}$ – функция активация нейронов *k*-го слоя.

8. Вычисление среднего выходного сигнала для единственного нейрона нового слоя

$$\bar{y}_1^{(new)}(n) = \frac{1}{P}\sum_{s=1}^{P} y_{\mu 1}^{(new)}(n)$$

9. Вычисление ковариации между выходными сигналами единственного нейрона на новом слое и нейронами входного слоя

$$\sigma_j(n) = \frac{1}{P}\sum_{\mu=1}^{P}(y_{\mu 1}^{(new)}(n) - \bar{y}_1^{(new)}(n))(e_{\mu j} - \bar{e}_j),\ j \in \overline{1, N^{(L)}}.$$

10. Настройка синаптических весов на основе расширенного обобщенного дельта правила (обратный ход)

$$w_{i1}^{(k,new)}(n+1) = w_{i1}^{(k,new)}(n) - \eta \frac{\partial S(n)}{\partial w_{i1}^{(k,new)}(n)},$$

где η – параметр, определяющий скорость обучения (при большом η обучение происходит быстрее, но увеличивается опасность получить неверное решение), $0 < \eta < 1$.

$$\frac{\partial S(n)}{\partial w_{i1}^{(k,new)}(n)} = \sum_{\mu=1}^{P}\sum_{j=1}^{L} y_{\mu i}^{(k)} f'^{(new)}(s_{\mu 1}^{(new)}(n))\sigma_j(n)(e_{\mu j} - \bar{e}_j),$$

$i \in \overline{1, N^{(k)}}, k \in \overline{0, L-1}$.

11. Проверка условия завершения

Если $\left|\sum_{j=1}^{L}\sigma_j(n-1) - \sum_{j=1}^{L}\sigma_j(n)\right| > \varepsilon$, то $n = n+1$, переход к 7.

Если $\left|\sum_{j=1}^{L}\sigma_j(n-1) - \sum_{j=1}^{L}\sigma_j(n)\right| < \varepsilon$, то $w_{i1}^{(k,new)} = w_{i1}^{(k,new)}(n)$, $y_{\mu 1}^{(new)} = y_{\mu 1}^{(new)}(n)$.

Обучение весов между единственным нейроном нового скрытого слоя и нейронами выходного слоя

1. Номер итерации обучения $n = 1$, инициализация посредством равномерного распределения на интервале (0,1) или [-0.5, 0.5] весов $w_{1j}^{(new,L)}(n)$, $j \in \overline{1, N^{(L)}}$.

2. Номер текущей пары из обучающего множества $\mu = 1$.

3. Вычисление выходного сигнала для выходного слоя (прямой ход)

$$y_j^{(L)}(n) = f^{(L)}(s_j^{(L)}(n)),$$

$$s_j^{(L)}(n) = w_{1j}^{(new,L)}(n) y_{\mu 1}^{(new)} + \sum_{l=0}^{L-1}\sum_{i=1}^{N^{(l)}} w_{ij}^{(l,k)} y_{\mu i}^{(l)}, \; j \in \overline{1, N^{(L)}},$$

где $N^{(k)}$ – число нейронов в *k*-м слое, *k* – номер слоя, $w_{ij}^{(l,k)}$ – вес связи от *i*-го нейрона на *l*-м слое к *j*-му нейрону на *k*-м слое, $y_j^{(k)}(n)$ – выход *j*-го нейрона на *k*-м слое, $f^{(k)}$ – функция активация нейронов *k*-го слоя.

4. Вычисление энергии ошибки ИНС

$$E(n) = \frac{1}{2}\sum_{j=1}^{N^{(L)}} e_j^2(n), \; e_j(n) = y_j^{(L)}(n) - d_{\mu j}$$

5. Настройка синаптических весов на основе обобщенного дельта правила (обратный ход)

$$w_{1j}^{(new,L)}(n+1) = w_{1j}^{(new,L)}(n) - \eta \frac{\partial E(n)}{\partial w_{1j}^{(new,L)}(n)},$$

где η – параметр, определяющий скорость обучения (при большом η обучение происходит быстрее, но увеличивается опасность получить неверное решение), $0 < \eta < 1$.

$$\frac{\partial E(n)}{\partial w_{1j}^{(new,L)}(n)} = y_1^{(new)}(n) f'^{(L)}(s_j^{(L)}(n))(y_j^{(L)}(n) - d_{\mu j}), \ j \in \overline{1, N^{(L)}}.$$

6. Проверка условия завершения

Если $n \bmod P > 0$, то $\mu = \mu + 1$, $n = n + 1$, переход к 3.

Если $n \bmod P = 0$ и $\frac{1}{P}\sum_{s=1}^{P} E(n - P + s) > \varepsilon$, то $n = n + 1$, переход к 2.

Если $n \bmod P = 0$ и $\frac{1}{P}\sum_{s=1}^{P} E(n - P + s) < \varepsilon$, то $w_{1j}^{(new,L)} = w_{1j}^{(new,L)}(n)$.

Функционирование ИНС

$$y_i^{(0)} = x_i,$$

$$y_j^{(k)} = f^{(k)}(s_j^{(k)}), \ s_j^{(k)} = \sum_{l=0}^{k-1}\sum_{i=1}^{N^{(l)}} w_{ij}^{(l,k)} y_i^{(l)}, \ j \in \overline{1, N^{(k)}}, k \in \overline{1, L}.$$

Достоинства

1. Используется для классификации образцов.

2. Является универсальным аппроксиматором. Обеспечивает глобальную аппроксимацию нелинейного отображения входного сигнала в выходной.

3. Обеспечивает хорошее качество обобщения.

4. Количество классов может быть больше двух (в выходном слое может быть больше одного нейрона).

5. Автоматически определяет число скрытых слоев и число нейронов в этих слоях (равно 1).

Недостатки

1. Обучение происходит медленнее, чем в случае RBFNN, PNN, сети Хемминга, SOM, CPNN.

2. В отличие от методов обучения SVM, градиентные методы обучения MLP выбирают положение разделяющей гиперплоскости произвольным образом.

3. Обучение не сводится к задаче квадратичного программирования в выпуклой области, имеющей единственное решение.

4. В отличие от ART, не решает проблему пластичности-стабильности.

ГЛАВА 4
СТАТИЧЕСКИЕ ИСКУССТВЕННЫЕ НЕЙРОННЫЕ СЕТИ С АССОЦИАТИВНОЙ ПАМЯТЬЮ

4.1. Автоассоциативный многослойный персептрон

На рис. 4.1 приведен автоассоциативный двухслойный персептрон (AAMLP) [27, 28], который является нерекуррентной статической многослойной ИНС. Классы разделяются гиперплоскостями. В отличие от MLP выходной слой является линейным. В отличие от MLP число входных и выходных нейронов совпадает.

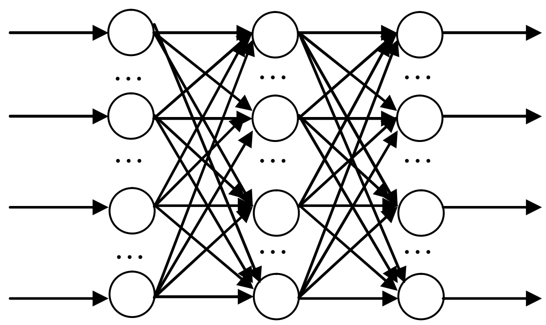


Рис. 4.1. Автоассоциативный многослойный персептрон (AAMLP)

AAMLP реализует автоассоциативную память (ее элемент представлен парой образцов $(\mathbf{m}_x, \mathbf{m}_y)$, $\mathbf{m}_y = \mathbf{m}_x$) и восстанавливает запомненный образец $\mathbf{m}_y$ по ключевому образцу $\mathbf{m}_x$, соответствующему входному вектору $\mathbf{x}$.

Важнейшим свойством AAMLP является то, что одна и та же ИНС с одними и теми же весами связей может хранить и воспроизводить несколько различных запомненных образцов.

Для AAMLP используется обучение на основе коррекции ошибок (обучение с учителем), при этом чаще всего применяется алгоритм обратного распространения (BP). Это итеративный градиентный алгоритм обучения, который обеспечивает минимизацию среднеквадратичной ошибки.

Обучение ИНС (алгоритм обратного распространения)

1. Номер итерации обучения $n = 1$, инициализация посредством равномерного распределения на интервале (0,1) или [-0.5, 0.5]

смещений (порогов) $b_j^{(k)}(n)$ и весов $w_{ij}^{(k)}(n)$, $i \in \overline{1, N^{(k-1)}}$, $j \in \overline{1, N^{(k)}}, k \in \overline{1,2}$, где $N^{(k)}$ – количество нейронов в k-м слое.

2. Задается обучающее множество $\{\mathbf{x}_\mu \mid \mathbf{x}_\mu \in R^{N^{(0)}}\}$, $\mu \in \overline{1, P}$, где $\mathbf{x}_\mu$ – μ-й обучающий входной вектор, $N^{(0)}$ – количество нейронов входного слоя, $N^{(2)}$ – количество нейронов выходного слоя, $N^{(0)} = N^{(2)}$, P – мощность обучающего множества. Номер текущей пары из обучающего множества $\mu = 1$.

3. Вычисление выходного сигнала для каждого слоя (прямой ход)

$$y_i^{(0)}(n) = x_{\mu i},$$

$$y_j^{(1)}(n) = f^{(1)}(s_j^{(1)}(n)), s_j^{(1)}(n) = \sum_{i=0}^{N^{(0)}} w_{ij}^{(1)}(n) y_i^{(0)}(n), \; j \in \overline{1, N^{(1)}},$$

$$y_j^{(2)}(n) = \sum_{i=0}^{N^{(1)}} w_{ij}^{(2)}(n) y_i^{(1)}(n), \; j \in \overline{1, N^{(2)}},$$

где $N^{(k)}$ – число нейронов в k-м слое, k – номер слоя, $w_{ij}^{(k)}(n)$ – вес связи от i-го нейрона к j-му нейрону на k-м слое в момент времени n, $y_j^{(k)}(n)$ – выход j-го нейрона на k-м слое, $f^{(1)}$ – функция активация нейронов первого слоя.

Считается, что $w_{0j}^{(k)}(n) = b_j^{(k)}(n), y_0^{(k-1)}(n) = 1$.

4. Вычисление энергии ошибки ИНС

$$E(n) = \frac{1}{2} \sum_{j=1}^{N^{(2)}} e_j^2(n), \; e_j(n) = y_j^{(2)}(n) - x_{\mu j}$$

5. Настройка синаптических весов на основе обобщенного дельта правила (обратный ход)

$$w_{ij}^{(k)}(n+1) = w_{ij}^{(k)}(n) - \eta \frac{\partial E(n)}{\partial w_{ij}^{(k)}(n)},$$

где η – параметр, определяющий скорость обучения (при большом η обучение происходит быстрее, но увеличивается опасность получить неверное решение), $0 < \eta < 1$.

$$\frac{\partial E(n)}{\partial w_{ij}^{(k)}(n)} = y_i^{(k-1)}(n) g_j^{(k)}(n),\ i \in \overline{0, N^{(k-1)}},\ j \in \overline{1, N^{(k)}},\ k \in \overline{1,2},$$

$$g_j^{(k)}(n) = \begin{cases} (y_j^{(2)}(n) - x_{\mu j}), & k = 2 \\ f'^{(1)}(s_j^{(1)}(n)) \sum\limits_{l=1}^{N^{(2)}} w_{jl}^{(2)}(n) g_l^{(2)}(n), & k = 1 \end{cases}$$

6. Проверка условия завершения

Если $n \bmod P > 0$, то $\mu = \mu + 1$, $n = n + 1$, переход к 3.

Если $n \bmod P = 0$ и $\frac{1}{P}\sum\limits_{s=1}^{P} E(n - P + s) > \varepsilon$, то $n = n + 1$, переход к 2.

Если $n \bmod P = 0$ и $\frac{1}{P}\sum\limits_{s=1}^{P} E(n - P + s) < \varepsilon$, то завершиться.

Функционирование ИНС

$y_i^{(0)} = x_i$,

$$y_j^{(1)} = f^{(1)}(s_j^{(1)}), s_j^{(1)} = \sum_{i=0}^{N^{(0)}} w_{ij}^{(1)} y_i^{(0)},\ j \in \overline{1, N^{(1)}},$$

$$y_j^{(2)} = \sum_{i=0}^{N^{(1)}} w_{ij}^{(2)} y_i^{(1)},\ j \in \overline{1, N^{(2)}}.$$

Результатом является образец $\mathbf{y}^{(2)}$.

Достоинства

1. Используется для восстановления образцов.
2. Обеспечивает хорошее качество обобщения.
3. Автоматически определяется количество скрытых слоев (равно одному).
4. Количество классов может быть больше двух.
5. В отличие от DHNN, GM и BAM работает с вещественными данными.

Недостатки

1. Отсутствует автоматическое определение числа нейронов в скрытом слое.

2. В отличие от ART, не решает проблему пластичности-стабильности.

3. В отличие от SOM требует больше объемов вычислений и занимаемой памяти.

4. В отличие от BM не гарантирует нахождение глобального минимума.

4.2. Автоассоциативная нейросеть на основе радиально-базисных функций

На рис. 4.2 приведена авторская автоассоциативная нейросеть на основе радиально-базисных функций (AARBFNN), которая является нерекуррентной статической двухслойной ИНС. В отличие от MLP классы разделяются не гиперплоскостями, а гиперсферами. В отличие от MLP выходной слой является линейным. В отличие от MLP число входных и выходных нейронов совпадает.

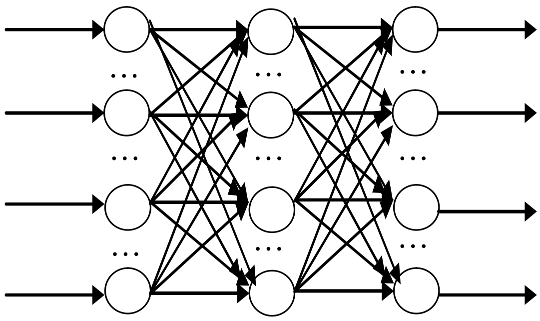


Рис. 4.2. Автоассоциативная нейросеть на основе радиально-базисных функций (AARBFNN)

AARBFNN реализует автоассоциативную память (ее элемент представлен парой образцов $(\mathbf{m}_x, \mathbf{m}_y)$, $\mathbf{m}_y = \mathbf{m}_x$) и восстанавливает запомненный образец $\mathbf{m}_y$ по ключевому образцу $\mathbf{m}_x$, соответствующему входному вектору $\mathbf{x}$.

Важнейшим свойством AARBFNN является то, что одна и та же ИНС с одними и теми же весами связей может хранить и воспроизводить несколько различных запомненных образцов.

Обычно используются три алгоритма обучения:

– обучение на основе коррекции ошибок (обучение с учителем) для выбора центров RBF и настройки весов;

– конкурентное обучение (обучение без учителя) для выбора центров RBF и обучение на основе коррекции ошибок (обучение с учителем) для настройки весов;

– случайный выбор центров RBF и метод псевдообращения для настройки весов.

Обычно для обучения AARBFNN используется первый алгоритм, который дает наиболее точный результат. Это итеративный градиентный алгоритм обучения, который обеспечивает минимизацию среднеквадратичной ошибки.

Обучение ИНС

1. Номер итерации обучения $n=0$, инициализация посредством равномерного распределения на интервале (0,1) или [-0.5, 0.5] смещений (порогов) $b_j(n)$ и весов $w_{ij}(n)$, центров RBF (векторов математических ожиданий) $\mathbf{m}_i(n)$ размерности $N^{(0)}$ ($\mathbf{m}_i(n)$ интерпретируется как вектор весов i-го нейрона первого слоя), диагональных ковариационных матриц $\mathbf{C}_i(n)$ размерности $N^{(0)} \times N^{(0)}$, $\mathbf{C}_i(n) = \operatorname{diag}(\sigma_{i1}^2, \dots, \sigma_{iN^{(0)}}^2)$, $i \in \overline{1, N^{(1)}}$, $j \in \overline{1, N^{(2)}}$, где $N^{(0)}$ – количество нейронов входного слоя, $N^{(1)}$ – количество нейронов в первом слое, $N^{(2)}$ – количество нейронов во втором слое, $N^{(0)} = N^{(2)}$.

2. Задается обучающее множество $\{\mathbf{x}_\mu \mid \mathbf{x}_\mu \in R^{N^{(0)}}\}$, $\mu \in \overline{1, P}$, где $\mathbf{x}_\mu$ – μ-й обучающий входной вектор, P – мощность обучающего множества.

3. Вычисление выходного сигнала для выходного слоя

$$y_{\mu j}(n) = \sum_{i=0}^{N^{(1)}} w_{ij}(n) G_i(\mathbf{x}_\mu), \ \mu \in \overline{1, P}, \ j \in \overline{1, N^{(2)}},$$

$$G_i(\mathbf{x}_\mu) = \exp\left(-\frac{1}{2} (\mathbf{x}_\mu - \mathbf{m}_i(n))^T \mathbf{C}_i^{-1}(n) (\mathbf{x}_\mu - \mathbf{m}_i(n)) \right)$$ – многомерная функция Гаусса.

Считается, что $w_{0j}(n) = b_j(n), G_0(\mathbf{x}_\mu) = 1$.

4. Вычисление энергии ошибки ИНС

$$E(n) = \frac{1}{2P} \sum_{\mu=1}^{P} \sum_{j=1}^{N^{(2)}} e_{\mu j}^2(n), \ e_{\mu j}(n) = y_{\mu j}(n) - x_{\mu j}.$$

5. Весовые коэффициенты выходного слоя, вектора математических ожиданий и диагональные ковариационные матрицы вычисляются на основе обобщенного дельта правила

$$w_{ij}(n+1) = w_{ij}(n) - \eta_1 \frac{\partial E(n)}{\partial w_{ij}(n)},\ i \in \overline{0, N^{(1)}},\ j \in \overline{1, N^{(2)}},$$

$$\mathbf{m}_i(n+1) = \mathbf{m}_i(n) - \eta_2 \frac{\partial E(n)}{\partial \mathbf{m}_i(n)},\ i \in \overline{1, N^{(1)}},$$

$$\mathbf{C}_i^{-1}(n+1) = \mathbf{C}_i^{-1}(n) - \eta_3 \frac{\partial E(n)}{\partial \mathbf{C}_i^{-1}(n)},\ i \in \overline{1, N^{(1)}},$$

где η_1, η_2, η_3 – параметры, определяющие скорость обучения, $0 < \eta_1 < 1, 0 < \eta_2 < 1, 0 < \eta_3 < 1$.

$$\frac{\partial E(n)}{\partial w_{ij}(n)} = \frac{1}{P}\sum_{\mu=1}^{P}(y_{\mu j}(n) - x_{\mu j})G_i(\mathbf{x}_\mu),$$

$$\frac{\partial E(n)}{\partial \mathbf{m}_i(n)} = 2w_{ij}(n)\frac{1}{P}\sum_{\mu=1}^{P}(y_{\mu j}(n) - x_{\mu j})G_i(\mathbf{x}_\mu)\mathbf{C}_i^{-1}(\mathbf{x}_\mu - \mathbf{m}_i(n)),$$

$$\frac{\partial E(n)}{\partial \mathbf{C}_i^{-1}(n)} = -w_{ij}(n)\frac{1}{P}\sum_{\mu=1}^{P}(y_{\mu j}(n) - x_{\mu j})G_i(\mathbf{x}_\mu)(\mathbf{x}_\mu - \mathbf{m}_i(n))^T(\mathbf{x}_\mu - \mathbf{m}_i(n)).$$

6. Проверка условия завершения

Если $E(n) < \varepsilon$, то завершиться, иначе $n = n+1$, переход к 2.

Функционирование ИНС

$$y_j = f_j(\mathbf{x}) = b_j + \sum_{i=1}^{N^{(1)}} w_{ij} G_i(\mathbf{x}),\ j \in \overline{1, N^{(2)}}.$$

Результатом является образец $\mathbf{y}$.

Достоинства

1. Используется для восстановления образцов.
2. Обеспечивает хорошее качество обобщения.
3. Автоматически определяется количество скрытых слоев (равно одному).
4. Количество классов может быть больше двух.
5. В отличие от DHNN, GM и BAM работает с вещественными данными.

Недостатки

1. Отсутствует автоматическое определение числа нейронов в скрытом слое.

2. В отличие от ART, не решает проблему пластичности-стабильности.

3. В отличие от SOM требует больше объемов вычислений и занимаемой памяти.

4. В отличие от BM не гарантирует нахождение глобального минимума.

4.3. Автоассоциативная обобщенная регрессионная нейросеть

На рис. 4.3 приведена авторская автоассоциативная обобщенная регрессионная нейросеть (AAGRNN), которая является нерекуррентной статической двухслойной ИНС. В отличие от MLP классы разделяются не гиперплоскостями, а гиперсферами. В отличие от MLP выходной слой является линейным. GRNN является модификацией RBFNN. В отличие от MLP число входных и выходных нейронов совпадает.

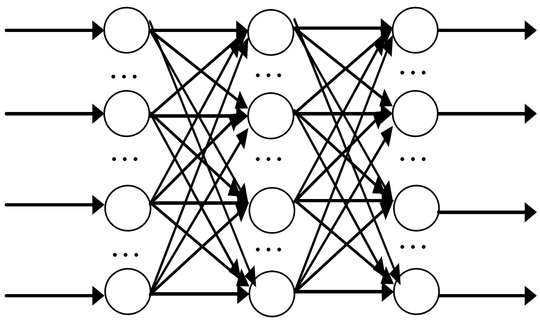


Рис. 4.3. Автоассоциативная обобщенная регрессионная нейросеть (AAGRNN)

AAGRNN реализует автоассоциативную память (ее элемент представлен парой образцов $(\mathbf{m}_x, \mathbf{m}_y)$, $\mathbf{m}_y = \mathbf{m}_x$) и восстанавливает запомненный образец $\mathbf{m}_y$ по ключевому образцу $\mathbf{m}_x$, соответствующему входному вектору $\mathbf{x}$.

Важнейшим свойством AAGRNN является то, что одна и та же ИНС с одними и теми же весами связей может хранить и воспроизводить несколько различных запомненных образцов.

Обычно для обучения AAGRNN используется обучение на основе коррекции ошибок (обучение с учителем) для выбора центров RBF и настройки весов. Это итеративный градиентный алгоритм обучения, который обеспечивает минимизацию среднеквадратичной ошибки.

Обучение ИНС

1. Номер итерации обучения $n = 0$, инициализация посредством равномерного распределения на интервале (0,1) или [-0.5, 0.5] смещений (порогов) $b_j(n)$ и весов $w_{ij}(n)$, центров RBF (векторов математических ожиданий) $\mathbf{m}_i(n)$ размерности $N^{(0)}$ ($\mathbf{m}_i(n)$ интерпретируется как вектор весов i-го нейрона первого слоя), диагональных ковариационных матриц $\mathbf{C}_i(n)$ размерности $N^{(0)} \times N^{(0)}$, $\mathbf{C}_i(n) = \mathrm{diag}(\sigma_{i1}^2, ..., \sigma_{iN^{(0)}}^2)$, $i \in \overline{1, N^{(1)}}$, $j \in \overline{1, N^{(1)}}$, где $N^{(0)}$ – количество нейронов входного слоя, $N^{(1)}$ – количество нейронов в первом и втором (выходном) слое, $N^{(0)} = N^{(2)}$.

2. Задается обучающее множество $\{\mathbf{x}_\mu \mid \mathbf{x}_\mu \in R^{N^{(0)}}\}$, $\mu \in \overline{1, P}$, где $\mathbf{x}_\mu$ – μ-й обучающий входной вектор, P – мощность обучающего множества.

3. Вычисление выходного сигнала для первого слоя

$$y_{\mu j}(n) = \frac{\sum_{i=0}^{N^{(1)}} w_{ij}(n) G_i(\mathbf{x}_\mu)}{\sum_{i=0}^{N^{(1)}} G_i(\mathbf{x}_\mu)}, \ \mu \in \overline{1, P}, \ j \in \overline{1, N^{(1)}},$$

$G_i(\mathbf{x}_\mu) = \exp\left(-\frac{1}{2} (\mathbf{x}_\mu - \mathbf{m}_i(n))^T \mathbf{C}_i^{-1}(n) (\mathbf{x}_\mu - \mathbf{m}_i(n)) \right)$ – многомерная функция Гаусса.

Считается, что $w_{0j}(n) = b_j(n), G_0(\mathbf{x}_\mu) = 1$.

4. Вычисление энергии ошибки ИНС

$$E(n) = \frac{1}{2P} \sum_{\mu=1}^{P} \sum_{j=1}^{N^{(1)}} e_{\mu j}^2(n), \ e_{\mu j}(n) = y_{\mu j}(n) - x_{\mu j}.$$

5. Весовые коэффициенты выходного слоя, вектора математических ожиданий и диагональные ковариационные матрицы вычисляются на основе обобщенного дельта правила

$$w_{ij}(n+1) = w_{ij}(n) - \eta_1 \frac{\partial E(n)}{\partial w_{ij}(n)}, \ i \in \overline{0, N^{(1)}}, \ j \in \overline{1, N^{(1)}},$$

$$\mathbf{m}_i(n+1) = \mathbf{m}_i(n) - \eta_2 \frac{\partial E(n)}{\partial \mathbf{m}_i(n)}, \ i \in \overline{1, N^{(1)}},$$

$$\mathbf{C}_i^{-1}(n+1) = \mathbf{C}_i^{-1}(n) - \eta_3 \frac{\partial E(n)}{\partial \mathbf{C}_i^{-1}(n)}, \ i \in \overline{1, N^{(1)}},$$

где η_1, η_2, η_3 – параметры, определяющие скорость обучения, $0 < \eta_1 < 1, 0 < \eta_2 < 1, 0 < \eta_3 < 1$.

$$\frac{\partial E(n)}{\partial w_{ij}(n)} = \frac{1}{P} \sum_{\mu=1}^{P} \frac{(y_{\mu j}(n) - x_{\mu j}) G_i(\mathbf{x}_\mu)}{\sum_{i=0}^{N^{(1)}} G_i(\mathbf{x}_\mu)},$$

$$\frac{\partial E(n)}{\partial \mathbf{m}_i(n)} = 2 \frac{1}{P} \sum_{\mu=1}^{P} \frac{G_i(\mathbf{x}_\mu) \mathbf{C}_i^{-1} (\mathbf{x}_\mu - \mathbf{m}_i(n))(w_{ij}(n) - x_{\mu j}(n))}{\sum_{i=0}^{N^{(1)}} G_i(\mathbf{x}_\mu)},$$

$$\frac{\partial E(n)}{\partial \mathbf{C}_i^{-1}(n)} = -\frac{1}{P} \sum_{\mu=1}^{P} \frac{G_i(\mathbf{x}_\mu)(\mathbf{x}_\mu - \mathbf{m}_i(n))^T (\mathbf{x}_\mu - \mathbf{m}_i(n))(w_{ij}(n) - x_{\mu j}(n))}{\sum_{i=0}^{N^{(1)}} G_i(\mathbf{x}_\mu)}$$

6. Проверка условия завершения

Если $E(n) < \varepsilon$, то завершиться, иначе $n = n + 1$, переход к 2.

Функционирование ИНС

$$y_j = f_j(\mathbf{x}) = \frac{b_j + \sum_{i=1}^{N^{(1)}} w_{ij} G_i(\mathbf{x})}{\sum_{i=1}^{N^{(1)}} G_i(\mathbf{x})}, \ j \in \overline{1, N^{(1)}}.$$

Результатом является образец $\mathbf{y}$.

Достоинства

1. Используется для восстановления образцов.
2. Обеспечивает хорошее качество обобщения.
3. Автоматически определяется количество скрытых слоев (равно одному).

4. Количество классов может быть больше двух.

5. В отличие от DHNN, GM и BAM работает с вещественными данными.

Недостатки

1. Отсутствует автоматическое определение числа нейронов в скрытом слое.

2. В отличие от ART, не решает проблему пластичности-стабильности.

3. В отличие от SOM требует больше объемов вычислений и занимаемой памяти.

4. В отличие от BM не гарантирует нахождение глобального минимума.

4.4. Самоорганизующаяся карта признаков

На рис. 4.4-4.5 приведена самоорганизующаяся карта признаков (SOM) [29-31], которая является нерекуррентной статической однослойной ИНС. Выход ИНС является линейным.

Нейроны выходного слоя соответствуют кластерам и образуют решетку (карту), обычно одно– или двухмерную. Элементы входных сигналов подаются на входы всех нейронов ИНС без задания желаемых выходных сигналов.

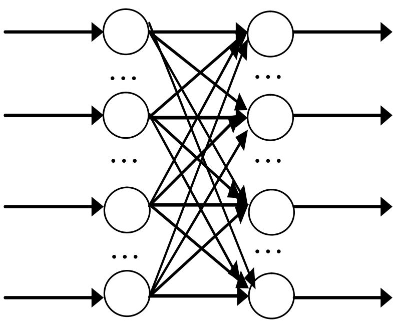


Рис. 4.4. Самоорганизующаяся карта признаков (SOM) одномерная

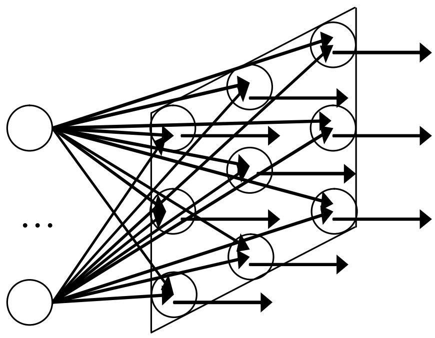


Рис. 4.5. Самоорганизующаяся карта признаков (SOM) двухмерная

Эта ИНС обучается без учителя, т.е. заранее не известна желаемая реакция ИНС на заданный входной вектор, и, следовательно, невозможно использовать ошибку ИНС для изменения весов связей. В этом случае ИНС должна обучаться, обнаруживая закономерности, присущие пространству входных сигналов. Поэтому она называется самоорганизующейся.

Алгоритм обучения SOM дает возможность строить ИНС для разделения векторов входных сигналов на подгруппы. После того, как было предъявлено достаточное число входных векторов, синаптические веса ИНС определяют кластеры. Кроме того, веса организуются так, что топологически близкие узлы чувствительны к похожим внешним воздействиям (входным сигналам).

SOM моделирует в некоторой степени процессы, происходящие в мозге. В мозге нейроны располагаются в определенном порядке так, что некоторые внешние физические воздействия вызывают ответную реакцию нейронов из определенной области мозга. Например, в той части мозга, которая отвечает за восприятие звуковых сигналов, нейроны группируются в соответствии с частотами входного сигнала, на которых они резонируют. Хотя строение мозга в значительной степени предопределяется генетически, отдельные структуры мозга формируются в процессе самоорганизации.

Для SOM используется соревновательное (конкурентное) обучение – выигрывает тот нейрон, чей вектор весов наиболее близок к текущему входному вектору по расстоянию Хемминга. Веса изменяются только у нейрона-победителя и его окружения, т.е. нейронная активность остальных падает.

Кроме конкуренции также реализуется кооперация. Нейрон-победитель находится в центре топологической окрестности сотрудничающих нейронов. На рис. 4.6 показаны зоны топологического соседства нейронов на двумерной карте признаков в различные моменты времени, где $NE_j(t)$ – множество нейронов, которые считаются соседями нейрона j в момент времени t. Зоны соседства уменьшаются с течением времени.

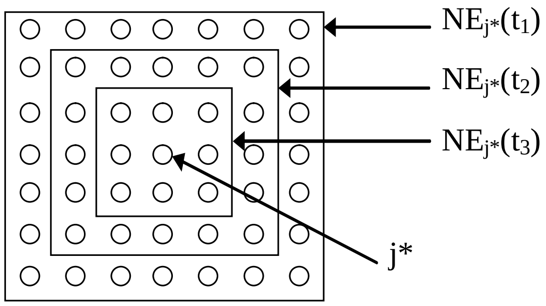


Рис. 4.6. Зоны топологического соседства на двумерной карте признаков в различные моменты времени

SOM реализует автоассоциативную память (ее элемент представлен парой образцов $(\mathbf{m}_x, \mathbf{m}_y)$, $\mathbf{m}_y = \mathbf{m}_x$) и восстанавливает запомненный образец $\mathbf{m}_y$ по ключевому образцу $\mathbf{m}_x$, соответствующему входному вектору $\mathbf{x}$.

Важнейшим свойством SOM является то, что одна и та же ИНС с одними и теми же весами связей может хранить и воспроизводить несколько различных запомненных образцов.

Обучение ИНС

1. Номер итерации обучения $n = 0$, инициализация посредством равномерного распределения на интервале (0,1) или [-0.5, 0.5] весов $w_{ij}(n)$, $i \in \overline{1, N^{(0)}}$, $j \in \overline{1, N^{(1)}}$, где $N^{(0)}$ – количество нейронов входного слоя, $N^{(1)}$ – количество нейронов в решетке (выходном слое).

Задается обучающее множество $\{\mathbf{x}_\mu \mid \mathbf{x}_\mu \in R^{N^{(0)}}\}$, $\mu \in \overline{1, P}$, где $\mathbf{x}_\mu$ – μ-й обучающий входной вектор, P – мощность обучающего множества. Номер вектора из обучающего множества $\mu = 1$.

Начальное наименьшее расстояние $d(0)$=0.

2. Вычисление расстояния до всех нейронов ИНС

Расстояние $d_{«j}$ от μ-го входного сигнала до каждого j-го нейрона определяется по формуле:

$$d_{\mu j} = \sum_{i=1}^{N^{(0)}} (x_{\mu i} - w_{ij}(n))^2 \,, \; j \in \overline{1, N^{(1)}}$$

где $w_{ij}(n)$ – вес связи от i-го элемента входного сигнала к j-му нейрону в момент времени n.

3. Вычисление наименьшего расстояния и выбор нейрона с наименьшим расстоянием

Вычисляется наименьшее расстояние

$$d_\mu = \min_j d_{\mu j} \,, \; j \in \overline{1, N^{(1)}}$$

и выбирается нейрон-победитель j^*, для которого расстояние $d_{«j}$ наименьшее.

$$j^* = \arg\min_j d_{\mu j} \,, \; j \in \overline{1, N^{(1)}} \,.$$

4. Настройка весов нейрона-победителя j^* и его соседей на основе правила Кохонена

$$w_{ij}(n+1) = w_{ij}(n) + \eta(n)h(j, j^*, n)(x_{\mu j} - w_{ij}(n)),$$

где $\eta(n)$ – коэффициент обучения, уменьшающийся с течением времени, $0 < \eta(n) < 1$, например

$$\eta(n) = 1/n,\ \eta(n) = \eta_0 e^{-\frac{n}{\tau_2}} \text{ или } \eta(n) = \eta_0 \left(\frac{\eta_{min}}{\eta_0}\right)^{n/n_{max}},$$

где η_{min}, η_0, τ_2 – константы, обычно η_0=0.1, τ_2=1000, n_{max} – максимальное количество итераций.

$h(j, j^*, n)$ – функция топологического соседства:

– прямоугольная (для алгоритма WTA)

$$h(j, j^*, n) = h(j, j^*) = \begin{cases} 1, & j = j^* \\ 0, & j \neq j^* \end{cases};$$

– гауссова (для алгоритма WTM)

$$h(j, j^*, n) = \exp\left(-\frac{\left\| r_j - r_{j^*} \right\|}{2\sigma(n)^2}\right),$$

– для алгоритма нейронного газа

$$h(j, j^*, n) = \exp\left(-\frac{m(j)}{\sigma(n)}\right),$$

где $r_j = (m, n)$ – позиция нейрона в двумерной карте, $r_j = j$ – позиция нейрона в одномерной карте,

$\| r_j - r_{j^*} \| = | j - j^* |$ – расстояние для одномерной карты,

$\| r_j - r_{j^*} \| = (m_j - m_{j^*})^2 + (n_j - n_{j^*})^2$ – расстояние для двумерной карты,

$m(j)$ – номер в последовательности нейронов, полученной путем их сортировки в порядке возрастания по расстоянию $\| r_j - r_{j^*} \|$,

$\sigma(n)$ – эффективная ширина топологической окрестности («диаметр» функции топологического соседства), уменьшающаяся с течением времени,

$$\sigma(n) = 1/n,\ \sigma(n) = \sigma_0 e^{\frac{-n}{\tau_1}} \text{ или } \sigma(n) = \sigma_{max} \left(\frac{\sigma_{min}}{\sigma_{max}}\right)^{n/n_{max}},$$

где $\pm_0$, τ_1 – константы, обычно $\pm_0$ – радиус карты, ${}^{2}{}_1 = \frac{1000}{\log \pm_0}$, $\sigma_{\min}$, $\sigma_{\max}$ – минимальное и максимальное значение σ, $n_{\max}$ – максимальное количество итераций.

5. Если $\mu < P$, то $\mu = \mu + 1$, перейти к 2, иначе к 6

6. Проверка условия завершения

$$d(n+1) = \frac{1}{P}\sum_{\mu=1}^{P} d_\mu ,$$

Если $d(n+1) - d(n) \le \varepsilon$, то завершиться иначе $\mu = 1, n = n+1$, переход к 2.

Функционирование ИНС

$$d_j = \sum_{i=1}^{N^{(0)}} (x_i - w_{ij}(n))^2 , \quad j \in \overline{1, N^{(1)}}$$

$$j^* = \arg\min_j d_j , \quad j \in \overline{1, N^{(1)}}$$

Результатом является образец $(w_{1j^*},...,w_{N^{(0)}j^*})$.

Использование одномерной самоорганизующейся карты признаков для задачи поиска оптимального маршрута

Для задачи поиска оптимального маршрута нейроны входного слоя одномерной самоорганизующейся карты признаков соответствуют координатам вершин. Нейроны выходного слоя перед началом работы ИНС соответствуют точкам многомерной сферы (если вершины заданы на плоскости, то эти нейроны соответствуют точкам кольца), а после работы ИНС соответствуют точкам в полученном субоптимальном маршруте.

Процедура поиска оптимального маршрута

1. Номер итерации обучения $n = 1$, инициализация весов $w_{ij}(n)$, $i \in \overline{1, N^{(0)}}$, $j \in \overline{1, N^{(1)}}$, таким образом, что точка с координатами $(w_{i1}(n),...,w_{iN^{(0)}}(n))$ находится на поверхности $N^{(0)}$-мерной сферы радиуса $R \in (0,1]$, где $N^{(0)}$ – количество нейронов входного слоя (или

длина вектора координат вершины), $N^{(1)}$ – количество нейронов выходного слоя, обычно $N \le N^{(1)} \le 3N$, N – количество вершин.

Задается множество координат вершин $\{\mathbf{x}_\mu \mid \mathbf{x}_\mu \in R^{N^{(0)}}\}$, $\mu \in \overline{1, N}$, где $\mathbf{x}_\mu$ – μ-й вектор координат вершин. Номер вершины $\mu = 1$.

Начальное наименьшее расстояние $d(n)$=0.

Задаётся максимальное количество итераций $n_{\max}$ и порог ε.

2. Вычисление расстояния до всех нейронов ИНС

Расстояние $d_{\mu j}$ от μ-й вершины до каждого j-го нейрона определяется по формуле:

$$d_{\mu j} = \sum_{i=1}^{N^{(0)}} (x_{\mu i} - w_{ij}(n))^2 , \; j \in \overline{1, N^{(1)}}$$

3. Вычисление наименьшего расстояния и выбор нейрона с наименьшим расстоянием

Вычисляется наименьшее расстояние

$$d_\mu = \min_j d_{\mu j} , \; j \in \overline{1, N^{(1)}}$$

и выбирается нейрон-победитель j^*, для которого расстояние $d_{\mu j}$ наименьшее.

$$j^* = \arg\min_j d_{\mu j} , \; j \in \overline{1, N^{(1)}} .$$

4. Настройка весов нейрона-победителя j^* и его соседей на основе правила Кохонена

$$w_{ij}(n+1) = w_{ij}(n) + \eta(n) h_{j, j^*}(n)(x_{\mu i} - w_{ij}(n)),$$

$h(j, j^*, n)$ – функция топологического соседства:

$$h_{j,j^*}(n) = \begin{cases} \exp\left(-\dfrac{\rho^2(j, j^*)}{2\sigma^2(n)} \right), & \rho(j, j^*) < \sigma(n) \\ 0, & \rho(j, j^*) \ge \sigma(n) \end{cases}$$

или

$$h_{j,j^*}(n) = \begin{cases} 1 - \dfrac{\rho(j, j^*)}{\sigma(n)+1}, & \rho(j, j^*) < \sigma(n) \\ 0, & \rho(j, j^*) \ge \sigma(n) \end{cases} ,$$

$$\rho(j, j^*) = \min\{| j - j^* |, N^{(1)} - | j - j^* |\},$$

$\eta(n)$ – коэффициент обучения, уменьшающийся с течением времени, $0 < \eta(n) < 1$, например,

$$\eta(n) = \eta_{\max} \exp\left(-\frac{n}{\gamma}\right),\ 1 < \gamma \le n_{\max}$$

или

$$\eta(n) = \eta_{\max} \alpha^n,\ 0 < \alpha < 1$$

или

$$\eta(n) = \eta_{\max} n^{-\alpha},\ 0 < \alpha \le 1,$$

$\sigma(n)$ – эффективная ширина топологической окрестности («диаметр» функции топологического соседства), уменьшающаяся с течением времени, $0 < \sigma(n) < N^{(1)}/2$, например,

$$\sigma(n) = \max\left\{\sigma_{\min}, \sigma_{\max} \exp\left(-\frac{n \ln \sigma_{\max}}{\gamma}\right)\right\},\ 1 < \gamma \le n_{\max}$$

или

$$\sigma(n) = \max\{\sigma_{\min}, \sigma_{\max}\beta^n\},\ 0 < \beta < 1$$

или

$$\sigma(n) = \max\{\sigma_{\min}, \sigma_{\max} n^{-\beta}\},\ 0 < \beta \le 1$$

5. Если $\mu < N$, то $\mu := \mu + 1$, переход к 2

6. Проверка условия завершения обучения

$$d(n+1) = \frac{1}{N} \sum_{\mu=1}^{N} d_\mu,$$

Если $d(n+1) - d(n) > \varepsilon$ или $n < n_{\max}$, то $\mu = 1, n = n+1$, переход на шаг 2.

7. Номер вершины $\mu = 1$.

8. Вычисление расстояния до всех нейронов ИНС

$$d_{\mu j} = \sum_{i=1}^{N^{(0)}} (x_{\mu i} - w_{ij}(n))^2,\ j \in \overline{1, N^{(1)}}$$

9. Выбор нейрона с наименьшим расстоянием

$$j_\mu^* = \arg\min_j d_{\mu j},\ j \in \overline{1, N^{(1)}}$$

10. Если $\mu < N$, то $\mu = \mu + 1$, переход к 8

Результатом является вектор пар «вершина-ранг» $((1, j_1^*), ..., (N, j_N^*))$

Достоинства

1. Используется для кластеризации.

2. Обеспечивает хорошее качество обобщения.

3. Автоматически определяется количество скрытых слоев и нейронов в скрытых слоях (равно нулю).

4. Количество кластеров может быть больше двух (в выходном слое может быть больше одного нейрона).

5. SOM, использующая соревновательный алгоритм, обучается быстрее, чем MLP, RBFNN, ME, HME, с градиентными методами обучения, поэтому может использоваться в тех приложениях, где долгая обучающая процедура невозможна.

6. Функционирует быстрее ИНС со скрытыми слоями.

7. Компрессия данных (образующие кластер группы данных представляются единственным вектором весов нейрона-победителя).

Недостатки:

1. Может быть использована для кластерного анализа только в том случае, если заранее известно число кластеров.

2. Метод обучения является эвристическим, поэтому завершение процедуры обучения не основывается на оптимизации математической модели процесса.

3. В отличие от ART, не решает проблему пластичности-стабильности.

4.5. Нейросеть квантования вектора обучения

На рис. 4.7 приведена нейросеть квантования вектора обучения (LVQNN) [32], которая является нерекуррентной статической однослойной ИНС. Выход ИНС является линейным.

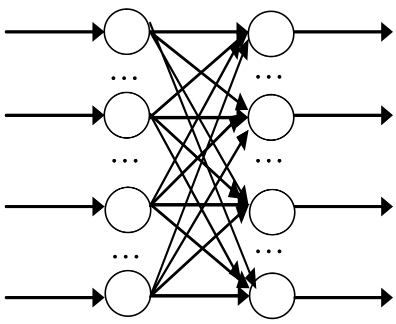


Рис. 4.7. Нейросеть квантования вектора обучения (LVQNN)

Как и для SOM, нейроны выходного слоя соответствуют кластерам.

В отличие от SOM, реализует гетероассоциативную память (ее элемент представлен парой образцов $(\mathbf{m}_x, \mathbf{m}_y)$, $\mathbf{m}_y \neq \mathbf{m}_x$) и восстанавливает запомненный образец $\mathbf{m}_y$ по ключевому образцу $\mathbf{m}_x$, соответствующему входному вектору $\mathbf{x}$.

Важнейшим свойством LVQNN является то, что одна и та же ИНС с одними и теми же весами связей может хранить и воспроизводить несколько различных запомненных образцов.

В отличие от SOM, обучается с учителем, т.е. заранее известна желаемая реакция ИНС на заданный входной вектор. С другой, подобно SOM, используется конкурентное обучение – выигрывает тот нейрон, чей вектор весов наиболее близок к текущему входному вектору по расстоянию Хемминга. Веса изменяются только у нейрона-победителя.

Алгоритм LVQ (квантование вектора обучения) дает возможность строить ИНС для разделения векторов входных сигналов на подгруппы. После того, как было предъявлено достаточное число входных векторов, синаптические веса ИНС определяют кластеры.

Частот используется комбинация SOM с LVQNN – входом LVQNN являются выходы SOM.

Обучение ИНС (алгоритм квантования вектора обучения)

1. Номер итерации обучения $n = 0$, инициализация посредством равномерного распределения на интервале (0,1) или [-0.5, 0.5] весов $w_{ij}(n)$, $i \in \overline{1, N^{(0)}}$, $j \in \overline{1, N^{(1)}}$, где $N^{(0)}$ – количество нейронов входного слоя, $N^{(1)}$ – количество нейронов в выходном слое.

Задается обучающее множество $\{(\mathbf{x}_\mu, j_\mu^d) \mid \mathbf{x}_\mu \in R^{N^{(0)}}, j_\mu^d \in \{1,..., N^{(1)}\}\}$, $\mu \in \overline{1, P}$, где $\mathbf{x}_\mu$ – μ-й обучающий входной вектор, j_μ^d – μ-й обучающий номер кластера, P – мощность обучающего множества. Номер текущей пары из обучающего множества $\mu = 1$.

Начальное наименьшее расстояние $d(0)$=0.

2. Вычисление расстояния до всех нейронов ИНС

Расстояние $D_{\mu j}$ от μ-го входного сигнала до каждого j-го нейрона определяется по формуле:

$$D_{\mu j} = \sum_{i=1}^{N^{(0)}} (x_{\mu i} - w_{ij}(n))^2 , \; j \in \overline{1, N^{(1)}}$$

где $w_{ij}(n)$ – вес связи от i-го элемента входного сигнала к j-му нейрону в момент времени n.

3. Вычисление наименьшего расстояния и выбор нейрона с наименьшим расстоянием

Вычисляется наименьшее расстояние

$$D_{\mu} = \min_j D_{\mu j} , \; j \in \overline{1, N^{(1)}}$$

и выбирается нейрон-победитель $j*$, для которого расстояние $d_{«j}$ наименьшее.

$$j* = \arg\min_j D_{\mu j} , \; j \in \overline{1, N^{(1)}} .$$

4. Настройка весов нейрона-победителя $j*$ и его соседей на основе правила LVQ

$$w_{ij}(n+1) = w_{ij}(n) + \eta(n) h(j_{\mu}^{d}, j, j^{*})(x_i - w_{ij}(n)),$$

$h(j_{\mu}^{d}, j, j^{*})$ – функция топологического соседства

$$h(j_{\mu}^{d}, j, j^{*}) = \begin{cases} 1, & j = j^{*} \wedge j^{*} = j_{\mu}^{d} \\ -1, & j = j^{*} \wedge j^{*} \neq j_{\mu}^{d} \\ 0, & j \neq j^{*} \end{cases} ,$$

где $\eta(n)$ – коэффициент обучения, уменьшающийся с течением времени, $0 < \eta(n) < 1$, например

$$\eta(n) = 1/n , \; \eta(n) = \eta_0 e^{-\frac{n}{\tau_2}} \text{ или } \sigma(n) = \eta_0 \left(\frac{\eta_{\min}}{\eta_0} \right)^{n/n_{\max}} ,$$

где $\eta_{\min}$, η_0, τ_2 – константы, обычно η_0=0.1, τ_2=1000, $n_{\max}$ – максимальное количество итераций.

5. Если $\mu < P$, то $\mu = \mu + 1$, перейти к 2, иначе к 6

6. Проверка условия завершения

$$D(n+1) = \frac{1}{P} \sum_{\mu=1}^{P} D_{\mu} ,$$

Если $D(n+1) - D(n) \le \varepsilon$, то завершиться иначе $\mu = 1, n = n+1$, переход к 2.

Функционирование ИНС

$$D_j = \sum_{i=1}^{N^{(0)}} (x_i - w_{ij})^2 ,$$

$$j^* = \arg\min_j D_j , \; j \in \overline{1, N^{(1)}} .$$

Результатом является образец $(w_{1j^*},..., w_{N^{(0)}j^*})$.

Достоинства

1. Используется для кластеризации.

2. Обеспечивает хорошее качество обобщения.

3. Автоматически определяется количество скрытых слоев и нейронов в скрытых слоях (равно нулю).

4. Количество кластеров может быть больше двух (в выходном слое может быть больше одного нейрона).

6. LVQNN, использующая алгоритм векторного квантования, обучается быстрее, чем MLP, RBFNN, ME, HME, с градиентными методами обучения, поэтому может использоваться в тех приложениях, где долгая обучающая процедура невозможна.

7. Функционирует быстрее ИНС со скрытыми слоями.

8. Компрессия данных (образующие кластер группы данных представляются единственным вектором весов нейрона-победителя).

Недостатки:

1. Может быть использована для кластерного анализа только в том случае, если заранее известно число кластеров.

2. Метод обучения является эвристическим, поэтому завершение процедуры обучения не основывается на оптимизации математической модели процесса.

3. В отличие от ART, не решает проблему пластичности-стабильности.

4.6. Однонаправленная нейросеть встречного распространения

На рис.4.8 приведена однонаправленная нейросеть встречного распространения (FOCPNN) [33,34], которая является нерекуррентной статической двухслойной ИНС. Скрытый слой соответствует слою Кохонена, Выходной слой соответствует слою Гроссберга. Выход ИНС является линейным.

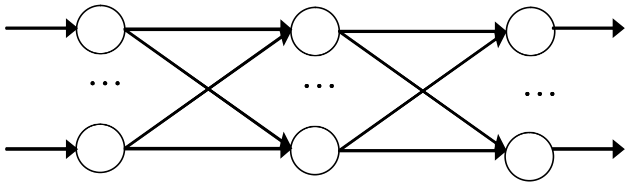


Рис. 4.8. Однонаправленная нейросеть встречного распространения (FOCPNN)

В отличие от SOM, реализует гетероассоциативную память (ее элемент представлен парой образцов $(\mathbf{m}_x, \mathbf{m}_y)$, $\mathbf{m}_y \neq \mathbf{m}_x$) и восстанавливает запомненный образец $\mathbf{m}_y$ по ключевому образцу $\mathbf{m}_x$, соответствующему входному вектору $\mathbf{x}$.

Важнейшим свойством FOCPNN является то, что одна и та же ИНС с одними и теми же весами связей может хранить и воспроизводить несколько различных запомненных образцов.

Для FOCPNN используется конкуррентное обучение (комбинация обучение с учителем и без учителя).

Обучение ИНС

Первая фаза (обучение скрытого слоя) (шаги 1-6)

1. Номер итерации обучения $n = 0$, инициализация посредством равномерного распределения на интервале (0,1) или [-0.5, 0.5] весов $w_{ij}^{(1)}(n)$, $i \in \overline{1, N^{(0)}}$, $j \in \overline{1, N^{(1)}}$, где $N^{(0)}$ – количество нейронов входного слоя, $N^{(1)}$ – количество нейронов в скрытом слое.

Задается обучающее множество $\{\mathbf{x}_\mu \mid \mathbf{x}_\mu \in R^{N^{(0)}}\}$, $\mu \in \overline{1, P}$, где $\mathbf{x}_\mu$ – μ-й обучающий входной вектор, P – мощность обучающего множества. Номер вектора из обучающего множества $\mu = 1$.

Начальное наименьшее расстояние $\bar{z}(0)$=0.

2. Вычисление расстояния до всех скрытых нейронов ИНС

Расстояние $z_{\mu i}$ от μ-го входного сигнала до каждого *i*-го нейрона определяется по формуле:

$$z_{\mu i} = \sum_{k=1}^{N^{(0)}} (x_{\mu k} - w_{ki}^{(1)}(n))^2, \ j \in \overline{1, N^{(1)}}$$

где $w_{ki}^{(1)}(n)$ – предварительно обученный вес связи от k-го элемента входного сигнала к i-му нейрону в момент времени n.

3. Вычисление наименьшего расстояния и выбор нейрона с наименьшим расстоянием

Вычисляется наименьшее расстояние

$$z_\mu = \min_i z_{\mu i}, \ i \in \overline{1, N^{(1)}}$$

и выбирается нейрон-победитель $i*$, для которого расстояние $z_{\mu i}$ наименьшее

$$i* = \arg\min_i z_{\mu i}, \ i \in \overline{1, N^{(1)}}.$$

4. Настройка весов нейронов выходного слоя, связанных с нейроном-победителем $i*$ и его соседей на основе правила выходной звезды

$$w_{ki}^{(1)}(n+1) = w_{ki}^{(1)}(n) + \eta(n)h(i,i^*)(x_{\mu k} - w_{ki}^{(1)}(n)),$$

где $\eta(n)$ – коэффициент обучения, уменьшающийся с течением времени, $0 < \eta(n) < 1$, например

$$\eta(n) = 1/n, \ \eta(n) = \eta_0 e^{-\frac{n}{\tau_2}} \text{ или } \eta(n) = \eta_0 \left(\frac{\eta_{\min}}{\eta_0}\right)^{n/n_{\max}},$$

где $\eta_{\min}$, η_0, τ_2 – константы, обычно η_0=0.1, τ_2=1000, $n_{\max}$ – максимальное количество итераций,

$h(i,i^*)$ – прямоугольная функция топологического соседства

$$h(i,i^*) = \begin{cases} 1, & i = i* \\ 0, & i \neq i* \end{cases}.$$

5. Если $\mu < P$, то $\mu = \mu + 1$, перейти к 2, иначе к 6

6. Проверка условия завершения

$$\bar{z}(n+1) = \frac{1}{P}\sum_{\mu=1}^{P} z_\mu,$$

Если $\bar{z}(n+1) - \bar{z}(n) \le \varepsilon$, то завершиться иначе $\mu = 1, n = n+1$, переход к 2.

Вторая фаза (обучение выходного слоя) (шаги 7-12)

7. Номер итерации обучения $n = 0$, инициализация посредством равномерного распределения на интервале (0,1) или [-0.5, 0.5] весов

$w_{ij}^{(2)}(n)$, $i \in \overline{1, N^{(1)}}$, $j \in \overline{1, N^{(2)}}$, где $N^{(1)}$ – количество нейронов в скрытом слое, $N^{(2)}$ – количество нейронов в выходном слое.

Задается обучающее множество $\{(\mathbf{x}_\mu, \mathbf{d}_\mu) \mid \mathbf{x}_\mu \in R^{N^{(0)}}, \mathbf{d}_\mu \in R^{N^{(2)}}\}$, $\mu \in \overline{1, P}$, где $\mathbf{x}_\mu$ – μ-й обучающий входной вектор, $\mathbf{d}_\mu$ – μ-й обучающий выходной вектор, P – мощность обучающего множества, $N^{(0)}$ – количество нейронов входного слоя. Номер текущей пары из обучающего множества $\mu = 1$.

Начальное наименьшее расстояние $\bar{z}(0)$=0.

8. Вычисление расстояния до всех скрытых нейронов ИНС

Расстояние $z_{\mu i}$ от μ-го входного сигнала до каждого i-го нейрона определяется по формуле:

$$z_{\mu i} = \sum_{k=1}^{N^{(0)}} (x_{\mu k} - w_{ki}^{(1)})^2, \ i \in \overline{1, N^{(1)}}$$

где $w_{ki}^{(1)}(n)$ – предварительно обученный вес связи от k-го элемента входного сигнала к i-му нейрону в момент времени n.

9. Вычисление наименьшего расстояния и выбор нейрона с наименьшим расстоянием

Вычисляется наименьшее расстояние

$$z_\mu = \min_i z_{\mu i}, \ i \in \overline{1, N^{(1)}}$$

и выбирается нейрон-победитель i^*, для которого расстояние $z_{\mu i}$ наименьшее.

$$i^* = \arg\min_i z_{\mu i}, \ i \in \overline{1, N^{(1)}}.$$

10. Настройка весов нейронов выходного слоя, связанных с нейроном-победителем i^* и его соседей на основе правила выходной звезды

$$w_{ij}^{(2)}(n+1) = w_{ij}^{(2)}(n) + \eta(n) h(i, i^*)(d_{\mu j} - w_{ij}^{(2)}(n)),$$

где $\eta(n)$ – коэффициент обучения, уменьшающийся с течением времени, $0 < \eta(n) < 1$, например

$$\eta(n) = 1/n, \ \eta(n) = \eta_0 e^{-\frac{n}{\tau_2}} \text{ или } \eta(n) = \eta_0 \left(\frac{\eta_{\min}}{\eta_0} \right)^{n/n_{\max}},$$

где η_{min}, η_0, τ_2 – константы, обычно η_0=0.1, τ_2=1000, n_{max} – максимальное количество итераций.

$h(i,i^*)$ – прямоугольная функция топологического соседства

$$h(i,i^*) = \begin{cases} 1, & i = i^* \\ 0, & i \neq i^* \end{cases}.$$

11. Если $\mu < P$, то $\mu = \mu + 1$, перейти к 8, иначе к 12

12. Проверка условия завершения

$$\bar{z}(n+1) = \frac{1}{P}\sum_{\mu=1}^{P} z_\mu ,$$

Если $\bar{z}(n+1) - \bar{z}(n) \leq \varepsilon$, то завершиться иначе $\mu = 1, n = n+1$, переход к 8.

Функционирование ИНС

1. $z_i = \sum_{k=1}^{N^{(0)}} (x_k - w_{ki}^{(1)})^2$, $i \in \overline{1, N^{(1)}}$

2. $i^* = \arg\min_i z_i$, $i \in \overline{1, N^{(1)}}$

Результатом является образец $(w_{i^*1}^{(2)}, ..., w_{i^*N^{(2)}}^{(2)})$.

Достоинства

1. Используется для восстановления образцов.
2. Обеспечивает хорошее качество обобщения.
3. Не требует определения количества скрытых слоев.
4. Не требует определения количества нейронов скрытого слоя.
5. Количество классов может быть больше двух.
6. В отличие от DHNN, GM и BAM работает с вещественными данными.

Недостатки

1. В отличие от ART, не решает проблему пластичности-стабильности.
2. В отличие от SOM, требует больше объемов вычислений и занимаемой памяти.
3. В отличие от BM не гарантирует нахождение глобального минимума.

4.7. Полная (двунаправленная) нейросеть встречного распространения

На рис.4.9 приведена полная (двунаправленная) нейросеть встречного распространения (FCPNN) [6], которая является нерекуррентной статической двухслойной ИНС. Скрытый слой соответствует слою Кохонена, Выходной слой соответствует слою Гроссберга. Выход ИНС является линейным.

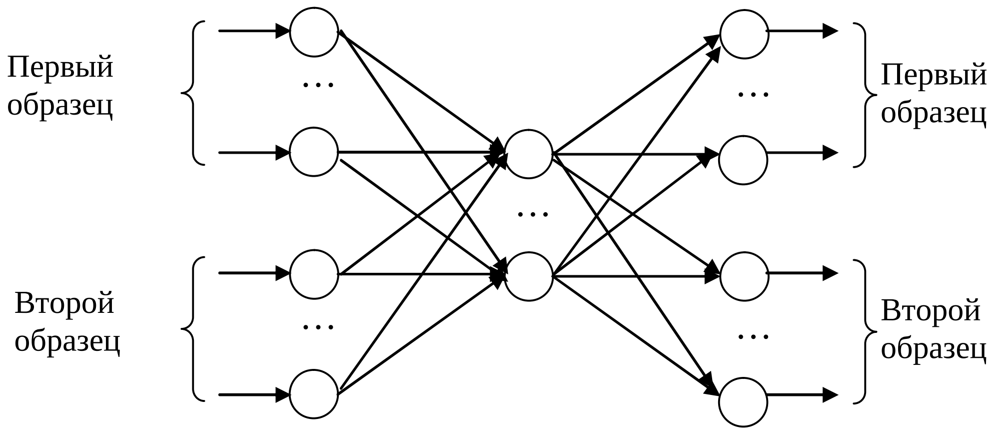


Рис. 4.9. Полная (двунаправленная) нейросеть встречного распространения (FCPNN)

В отличие от FOCPNN реализует гетероассоциативную память (ее элемент представлен парой образцов $(\mathbf{m}_x, \mathbf{m}_y)$, $\mathbf{m}_y \neq \mathbf{m}_x$) и автоассоциативную память (ее элемент представлен парой образцов $(\mathbf{m}_x, \mathbf{m}_y)$, $\mathbf{m}_y = \mathbf{m}_x$) и восстанавливает запомненный образец $\mathbf{m}_y$ по ключевому образцу $\mathbf{m}_x$, соответствующему входному вектору $\mathbf{x}$.

Важнейшим свойством FCPNN является то, что одна и та же ИНС с одними и теми же весами связей может хранить и воспроизводить несколько различных запомненных образцов.

Для FCPNN используется конкуррентное обучение (комбинация обучение с учителем и без учителя).

Обучение ИНС

Первая фаза (обучение скрытого слоя) (шаги 1-6)

1. Номер итерации обучения $n = 0$, инициализация посредством равномерного распределения на интервале (0,1) или [-0.5, 0.5] весов $w_{ki}^{(1)}(n)$, $v_{si}^{(1)}(n)$, $k \in \overline{1, NX^{(0)}}$, $s \in \overline{1, NY^{(0)}}$, $i \in \overline{1, N^{(1)}}$, где $NX^{(0)}$ – количество нейронов обучающего входного слоя, $NY^{(0)}$ – количество

нейронов обучающего выходного слоя, $N^{(1)}$ – количество нейронов в скрытом слое.

Задается обучающее множество $\{(\mathbf{x}_\mu, \mathbf{d}_\mu) \mid \mathbf{x}_\mu \in R^{NX^{(0)}}, \mathbf{d}_\mu \in R^{NY^{(0)}}\}$, $\mu \in \overline{1,P}$, где $\mathbf{x}_\mu$ – μ-й обучающий входной вектор, $\mathbf{d}_\mu$ – μ-й обучающий выходной вектор, P – мощность обучающего множества. Номер текущей пары из обучающего множества $\mu = 1$.

Начальное наименьшее расстояние $\bar{z}(0)=0$.

2. Вычисление расстояния до всех скрытых нейронов ИНС

Сумма расстояний $z_{\mu i}$ от μ-го обучающего входного сигнала до каждого i-го нейрона скрытого слоя и от каждого μ-го обучающего выходного сигнала до каждого i-го нейрона скрытого слоя определяется по формуле:

$$z_{\mu i} = \sum_{k=1}^{NX^{(0)}} (x_{\mu k} - w_{ki}^{(1)}(n))^2 + \sum_{s=1}^{NY^{(0)}} (d_{\mu s} - v_{si}^{(1)}(n))^2 \,,\ i \in \overline{1, N^{(1)}}\,,$$

где $w_{ki}^{(1)}(n)$ – вес связи от k-го элемента обучающего входного сигнала к i-му нейрону в момент времени n, $v_{ki}^{(1)}(n)$ – вес связи от s-го элемента обучающего выходного сигнала к i-му нейрону в момент времени n.

3. Вычисление наименьшего расстояния и выбор нейрона с наименьшим расстоянием

Вычисляется наименьшее расстояние

$$z_\mu = \min_i z_{\mu i},\ i \in \overline{1, N^{(1)}}$$

и выбирается нейрон-победитель i^*, для которого расстояние $z_{\mu i}$ наименьшее.

$$i^* = \arg\min_i z_{\mu i},\ i \in \overline{1, N^{(1)}}\,.$$

4. Настройка весов нейрона-победителя i^* и его соседей на основе правила входной звезды и правила выходной звезды

$$w_{ki}^{(1)}(n+1) = w_{ki}^{(1)}(n) + \eta(n) h(i, i^*)(x_{\mu i} - w_{ki}^{(1)}(n))\,,$$

$$v_{si}^{(1)}(n+1) = v_{si}^{(1)}(n) + \eta(n) h(i, i^*)(d_{\mu s} - v_{si}^{(1)}(n))\,,$$

где $\eta(n)$ – коэффициент обучения, уменьшающийся с течением времени, $0 < \eta(n) < 1$, например

$$\eta(n) = 1/n,\ \eta(n) = \eta_0 e^{-\frac{n}{\tau_2}} \text{ или } \eta(n) = \eta_0 \left(\frac{\eta_{min}}{\eta_0}\right)^{n/n_{max}},$$

где η_{min}, η_0, τ_2 – константы, обычно η_0=0.1, τ_2=1000, n_{max} – максимальное количество итераций,

$h(i,i^*)$ – прямоугольная функция топологического соседства,

$$h(i,i^*) = \begin{cases} 1, & i = i^* \\ 0, & i \neq i^* \end{cases}.$$

5. Если $\mu < P$, то $\mu = \mu + 1$, перейти к 2, иначе к 6

6. Проверка условия завершения

$$\bar{z}(n+1) = \frac{1}{P}\sum_{\mu=1}^{P} z_\mu,$$

Если $\bar{z}(n+1) - \bar{z}(n) \le \varepsilon$, то завершиться иначе $\mu = 1, n = n+1$, переход к 2.

Вторая фаза (обучение выходного слоя) (шаги 7-12)

7. Номер итерации обучения $n = 0$, инициализация посредством равномерного распределения на интервале (0,1) или [-0.5, 0.5] весов $w_{ij}^{(2)}(n)$, $v_{iq}^{(2)}(n)$, $i \in \overline{1, N^{(1)}}$, $j \in \overline{1, NY^{(2)}}$, $q \in \overline{1, NX^{(2)}}$, где $N^{(1)}$ – количество нейронов в скрытом слое, $NX^{(2)}$ – количество нейронов обучаемого входного слоя, $NY^{(2)}$ – количество нейронов обучаемого выходного слоя, $NX^{(0)} = NX^{(2)}$, $NY^{(0)} = NY^{(2)}$.

Задается обучающее множество $\{(\mathbf{x}_\mu, \mathbf{d}_\mu) \mid \mathbf{x}_\mu \in R^{NX^{(0)}}, \mathbf{d}_\mu \in R^{NY^{(0)}}\}$, $\mu \in \overline{1, P}$, где $\mathbf{x}_\mu$ – μ-й обучающий входной вектор, $\mathbf{d}_\mu$ – μ-й обучающий выходной вектор, P – мощность обучающего множества. Номер текущей пары из обучающего множества $\mu = 1$.

Начальное наименьшее расстояние $\bar{z}(0)$=0.

8. Вычисление расстояния до всех скрытых нейронов ИНС

Сумма расстояний $z_{\mu i}$ от μ-го обучающего входного сигнала до каждого *i*-го нейрона скрытого слоя и от каждого μ-го обучающего выходного сигнала до каждого *i*-го нейрона скрытого слоя определяется по формуле:

$$z_{\mu i} = \sum_{k=1}^{NX^{(0)}} (x_{\mu k} - w_{ki}^{(1)}(n))^2 + \sum_{s=1}^{NY^{(0)}} (d_{\mu s} - v_{si}^{(1)}(n))^2 \;,\; i \in \overline{1, N^{(1)}} \;,$$

где $w_{ki}^{(1)}(n)$ – вес связи от k-го элемента обучающего входного сигнала к i-му нейрону в момент времени n, $v_{ki}^{(1)}(n)$ – вес связи от s-го элемента обучающего выходного сигнала к i-му нейрону в момент времени n.

9. Вычисление наименьшего расстояния и выбор нейрона с наименьшим расстоянием

Вычисляется наименьшее расстояние

$$z_{\mu} = \min_i z_{\mu i} \;,\; i \in \overline{1, N^{(1)}}$$

и выбирается нейрон-победитель i^*, для которого расстояние $z_{\mu i}$ наименьшее.

$$i^* = \arg\min_i z_{\mu i} \;,\; i \in \overline{1, N^{(1)}} \;.$$

10. Настройка весов нейрона-победителя i^* и его соседей на основе правила входной звезды и правила выходной звезды

$$w_{ij}^{(2)}(n+1) = w_{ij}^{(2)}(n) + \eta(n) h(i, i^*)(d_{\mu j} - w_{ij}^{(2)}(n)) \,,$$

$$v_{is}^{(2)}(n+1) = v_{is}^{(2)}(n) + \eta(n) h(i, i^*)(x_{\mu i} - v_{is}^{(2)}(n)) \,,$$

где $\eta(n)$ – коэффициент обучения, уменьшающийся с течением времени, $0 < \eta(n) < 1$, например

$$\eta(n) = 1/n \,,\; \eta(n) = \eta_0 e^{-\frac{n}{\tau_2}} \text{ или } \eta(n) = \eta_0 \left(\frac{\eta_{\min}}{\eta_0} \right)^{n/n_{\max}} \,,$$

где $\eta_{\min}$, η_0, τ_2 – константы, обычно η_0=0.1, τ_2=1000, $n_{\max}$ – максимальное количество итераций,

$h(i, i^*)$ – прямоугольная функция топологического соседства,

$$h(i, i^*) = \begin{cases} 1, & i = i^* \\ 0, & i \neq i^* \end{cases} .$$

11. Если $\mu < P$, то $\mu = \mu + 1$, перейти к 8, иначе к 12.

12. Проверка условия завершения

$$\bar{z}(n+1) = \frac{1}{P} \sum_{\mu=1}^{P} z_{\mu} \;,$$

Если $\bar{z}(n+1)-\bar{z}(n)\le\varepsilon$, то завершиться иначе $\mu=1, n=n+1$, переход к 8.

Функционирование ИНС

1. Если по первому образцу восстанавливаем второй образец, то

1. $z_i = \sum_{k=1}^{NX^{(0)}} (x_k - w_{ki}^{(1)})^2$, $i \in \overline{1, N^{(1)}}$

2. $i^* = \arg\min_i z_i$, $i \in \overline{1, N^{(1)}}$

Результатом является образец $(w_{i^*1}^{(2)},...,w_{i^*NY^{(2)}}^{(2)})$.

2. Если по второму образцу восстанавливаем первый образец, то

1. $z_i = \sum_{s=1}^{NY^{(0)}} (d_s - v_{si}^{(1)})^2$, $i \in \overline{1, N^{(1)}}$

2. $i^* = \arg\min_i z_i$, $i \in \overline{1, N^{(1)}}$

Результатом является образец $(v_{i^*1}^{(2)},...,v_{i^*NX^{(2)}}^{(2)})$.

Достоинства

1. Используется для восстановления или классификации образцов.
2. Обеспечивает хорошее качество обобщения.
3. Не требует определения количества скрытых слоев.
4. Не требует определения количества нейронов скрытого слоя.
5. Количество классов может быть больше двух.
6. В отличие от DHNN, GM и BAM работает с вещественными данными.

Недостатки

1. В отличие от ART, не решает проблему пластичности-стабильности.
2. В отличие от SOM требует больше объемов вычислений и занимаемой памяти.
3. В отличие от BM не гарантирует нахождение глобального минимума.

4.8. Нейросеть анализа главных компонент

На рис.4.10 приведена нейросеть анализа главных компонент (PCANN) [35], которая является нерекуррентной статической однослойной ИНС. Выход ИНС является линейным. PCANN выделяет для классифицирующих ИНС информативные признаки, сокращая размерность данных. Таким образом, пространство данных (входное пространство) преобразуется в пространство информативных признаков. Для PCANN используется обучение Хебба (обучение без учителя), при этом применяются алгоритмы повторной оценки (например, GHA).

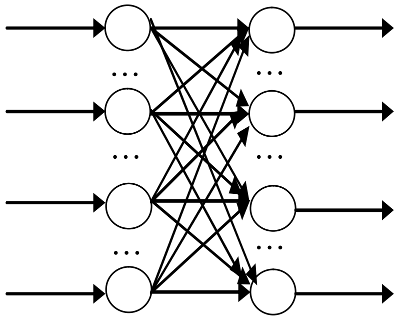


Рис.4.10. Нейросеть анализа главных компонент (PCANN)

PCANN реализует гетероассоциативную память (ее элемент представлен парой образцов $(\mathbf{m}_x, \mathbf{m}_y)$, $\mathbf{m}_y \neq \mathbf{m}_x$) и восстанавливает запомненный образец $\mathbf{m}_y$ по ключевому образцу $\mathbf{m}_x$, соответствующему входному вектору $\mathbf{x}$.

Важнейшим свойством PCANN является то, что одна и та же ИНС с одними и теми же весами связей может хранить и воспроизводить несколько различных запомненных образцов.

Обучение ИНС (алгоритм GHA)

1. Номер итерации обучения $n = 1$, инициализация посредством равномерного распределения на интервале (0,1) или [-0.5, 0.5] весов $w_{ij}(n)$, $i \in \overline{1, N^{(0)}}$, $j \in \overline{1, N^{(1)}}$, где $N^{(0)}$ – количество нейронов входного слоя, $N^{(1)}$ – количество нейронов выходного слоя, $N^{(1)} < N^{(0)}$.

2. Задается обучающее множество $\{\mathbf{x}_\mu \mid \mathbf{x}_\mu \in R^{N^{(0)}}\}$, $\mu \in \overline{1, P}$, где $\mathbf{x}_\mu$ – μ-й обучающий входной вектор, P – мощность обучающего множества. Номер вектора из обучающего множества $\mu = 1$.

3. Вычисление выходного сигнала ИНС

$$y_j(n) = \sum_{i=1}^{N^{(0)}} w_{ij} x_{\mu i}, \quad j \in \overline{1, N^{(1)}}.$$

4. Настройка синаптических весов на основе правила Сенгера

$$w_{ij}(n+1) = w_{ij}(n) + \eta y_j(n)(x_{\mu i}(n) - \sum_{k=1}^{j} w_{ik}(n) y_k(n)), \qquad i \in \overline{1, N^{(0)}},$$

$j \in \overline{1, N^{(1)}}$.

5. Проверка условия завершения

Если $\sum_{i=1}^{N^{(0)}} \sum_{j=1}^{N^{(1)}} | w_{ij}(n+1) - w_{ij}(n) | > \varepsilon$, то $n = n+1$, переход к 3.

Если $\mu < P$, то $n = 1, \mu = \mu + 1$, перейти к 3, иначе завершиться.

Функционирование ИНС

$$y_j = \sum_{i=1}^{N^{(0)}} w_{ij} x_i, \quad j \in \overline{1, N^{(1)}}.$$

Результатом является образец (вектор главных компонент) $\mathbf{y}$.

Достоинства

Сокращает размерность данных.

Недостатки

Отсутствует автоматическое определение количества нейронов выходного слоя (информативных признаков).

4.9.Нейросеть анализа независимых компонент

На рис.4.11 приведена нейросеть анализа независимых компонент (ICANN) [36,37], которая является нерекуррентной статической однослойной ИНС. Выход ИНС является линейным. ICANN выделяют для классифицирующих ИНС информативные признаки, сокращая размерность данных. Таким образом, пространство данных (входное пространство) преобразуется в пространство признаков. Для обучения ICANN используется обучение Хебба (обучение без учителя), при этом применяются алгоритмы Infomax и FastICA.

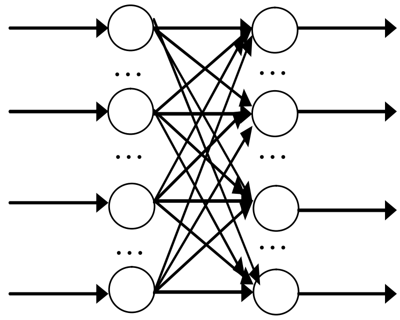


Рис. 4.11. Нейросеть анализа независимых компонент (ICANN)

ICANN реализует гетероассоциативную память (ее элемент представлен парой образцов $(\mathbf{m}_x, \mathbf{m}_y)$, $\mathbf{m}_y \neq \mathbf{m}_x$) и восстанавливает запомненный образец $\mathbf{m}_y$ по ключевому образцу $\mathbf{m}_x$, соответствующему входному вектору $\mathbf{x}$.

Важнейшим свойством ICANN является то, что одна и та же ИНС с одними и теми же весами связей может хранить и воспроизводить несколько различных запомненных образцов.

Обучение ИНС (алгоритм Infomax)

1. Номер итерации обучения $n = 1$, инициализация посредством равномерного распределения на интервале (0,1) или [-0.5, 0.5] весов $w_{ij}(n)$, $i \in \overline{1, N^{(0)}}, j \in \overline{1, N^{(1)}}$, где $N^{(0)}$ – количество нейронов входного слоя, $N^{(1)}$ – количество нейронов выходного слоя.

2. Задается обучающее множество $\{\mathbf{x}_\mu \mid \mathbf{x}_\mu \in R^{N^{(0)}}\}$, $\mu \in \overline{1, P}$, где $\mathbf{x}_\mu$ – μ-й обучающий входной вектор, P – мощность обучающего множества. Номер вектора из обучающего множества $\mu = 1$.

3. Вычисление выходного сигнала ИНС

$$\mathbf{y}(n) = \mathbf{W}(n)\mathbf{x}_\mu$$

4. Настройка синаптических весов на основе правила Infomax

$$\mathbf{W}(n+1) = \mathbf{W}(n) + \eta(\mathbf{I} - 2f(\mathbf{y}(n))\mathbf{y}^T(n))\mathbf{W}(n),$$

где $f(\mathbf{y}) = (f(y_1), ..., f(y_{N^{(1)}}))$, $f(y_j) = \tanh(y_j) = \dfrac{e^{ay_j} - e^{-ay_j}}{e^{ay_j} + e^{-ay_j}}$

или

$$\mathbf{W}(n+1) = \mathbf{W}(n) + \eta(n)(\mathbf{I} + (1 - 2f(\mathbf{y}(n)))\mathbf{y}^T(n))\mathbf{W}(n),$$

где $f(\mathbf{y}) = (f(y_1),..., f(y_{N^{(1)}}))$, $f(y_j) = \dfrac{1}{1+e^{-y_j}}$

или

$$\mathbf{W}(n+1) = \mathbf{W}(n) + \eta(\mathbf{I} - f(\mathbf{y}(n))\mathbf{y}^T(n))\mathbf{W}(n),$$

где $f(\mathbf{y}) = (f(y_1),..., f(y_{N^{(1)}}))$, $f(y_j) = \frac{1}{2}(y_j)^5 + \frac{2}{3}(y_j)^7 +$

$+\frac{15}{2}(y_j)^9 + \frac{2}{15}(y_j)^{11} - \frac{112}{3}(y_j)^{13} + 128(y_j)^{15} - \frac{512}{3}(y_j)^{17}$

5. Проверка условия завершения

Если $\sum\limits_{i=1}^{N^{(0)}} \sum\limits_{j=1}^{N^{(1)}} | w_{ij}(n+1) - w_{ij}(n) | > \varepsilon$, то $n = n+1$, переход к 3.

Если $\mu < P$, то $n = 1, \mu = \mu + 1$, перейти к 3, иначе завершиться.

Обучение ИНС (алгоритм FastICA)

1. Номер итерации обучения $n = 1$, инициализация посредством равномерного распределения на интервале (0,1) или [-0.5, 0.5] весов $w_{ij}(n)$, $i \in \overline{1, N^{(0)}}, j \in \overline{1, N^{(1)}}$, где $N^{(0)}$ – количество нейронов входного слоя, $N^{(1)}$ – количество нейронов выходного слоя.

2. Задается обучающее множество $\{\mathbf{x}_\mu \mid \mathbf{x}_\mu \in R^{N^{(0)}}\}$, $\mu \in \overline{1, P}$, где $\mathbf{x}_\mu$ – μ-й обучающий входной вектор, P – мощность обучающего множества. Номер вектора из обучающего множества $\mu = 1$.

3. $\mathbf{W}^+(n+1) = M[\mathbf{x}_\mu f(\mathbf{W}^T(n)\mathbf{x}_\mu)] - M[f'(\mathbf{W}^T(n)\mathbf{x}_\mu)]\mathbf{W}(n)$,

где $f(\mathbf{y}) = (f(y_1),..., f(y_{N^{(1)}}))$,

$$f(y_j) = \tanh(y_j) = \frac{e^{ay_j} - e^{-ay_j}}{e^{ay_j} + e^{-ay_j}}, \; a \in [1,2],$$

или

$$f(y_j) = y_j \exp(-(y_j)^2 / 2),$$

или

$$f(y_j) = (y_j)^3$$

4. $\mathbf{W}(n+1) = \dfrac{\mathbf{W}^+(n+1)}{\| \mathbf{W}^+(n+1) \|}$, где $\| \mathbf{W}^+(n) \|$ – норма матрицы,

$$\| \mathbf{W}^{+}(n) \| = \max_{i \in \overline{1,N^{(0)}}} \sum_{j=1}^{N^{(1)}} w_{ij}^{+}(k) \text{ или } \| \mathbf{W}^{+}(n) \| = \max_{j \in \overline{1,N^{(1)}}} \sum_{i=1}^{N^{(0)}} w_{ij}^{+}(k)$$

5. Если $n = 0$, то переход к 8

6. $\mathbf{W}^{+}(n+1) = \mathbf{W}(n+1) - \sum_{s=1}^{N^{(0)}} \mathbf{W}^{T}(n+1)\mathbf{W}(s)\mathbf{W}(s)$

7. $\mathbf{W}(n+1) = \dfrac{\mathbf{W}^{+}(n+1)}{\| \mathbf{W}^{+}(n+1) \|}$

8. Проверка условия завершения

Если $\sum_{i=1}^{N^{(0)}} \sum_{j=1}^{N^{(1)}} | w_{ij}(n+1) - w_{ij}(n) | > \varepsilon$, то $n = n+1$, переход к 3.

Если $\mu < P$, то $n = 1, \mu = \mu + 1$, перейти к 3, иначе завершиться.

Функционирование ИНС

$$y_j = \sum_{i=1}^{N^{(0)}} w_{ij} x_i , \ j \in \overline{1, N^{(1)}} .$$

Результатом является образец (вектор независимых компонент) $\mathbf{y}$.

Замечание. Часто для уменьшения размерности вначале применяется PCA, а затем ICA.

Достоинства

Сокращает размерность данных.

Недостатки

Отсутствует автоматическое определение количества независимых компонент.

4.10. Церебральная модель артикуляционного контроллера

Рассмотрим исходную и современную церебральную модель артикуляционного контроллера.

4.10.1. Исходная церебральная модель артикуляционного контроллера

На рис.4.12 приведена исходная церебральная модель артикуляционного контроллера (CMAC) [38], которая является нерекуррентной статической двухслойной ИНС. Выход ИНС является линейным.

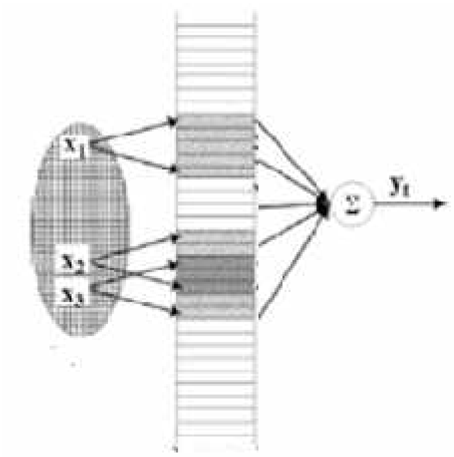


Рис.4.12. Исходная церебральная модель артикуляционного контроллера (CMAC)

На рис.4.13 приведен пример многослойного кодирования, которое представляет собой отображение предварительно квантованного входного вектора (оба элемента входного вектора представлены 16 состояниями, пронумерованными от 0 до 15) в ячейки ассоциативной (концептуальной или виртуальной) памяти. Каждый слой кодирования разбит на блоки (на рис.4.13 представлены прямоугольниками и подписаны большими и малыми буквами), причем в каждом последующем слое эти блоки обычно смещаются на 1, а длина блока и количество слоев кодирования обычно совпадают (на рис.4.13 они равны 4, при этом межблочное перекрытие равно 3).

Важнейшим свойством исходной CMAC является то, что одна и та же ИНС с одними и теми же весами связей может хранить и воспроизводить несколько различных запомненных образцов.

Для обучения CMAC используется обучение на основе коррекции ошибок (обучение с учителем).

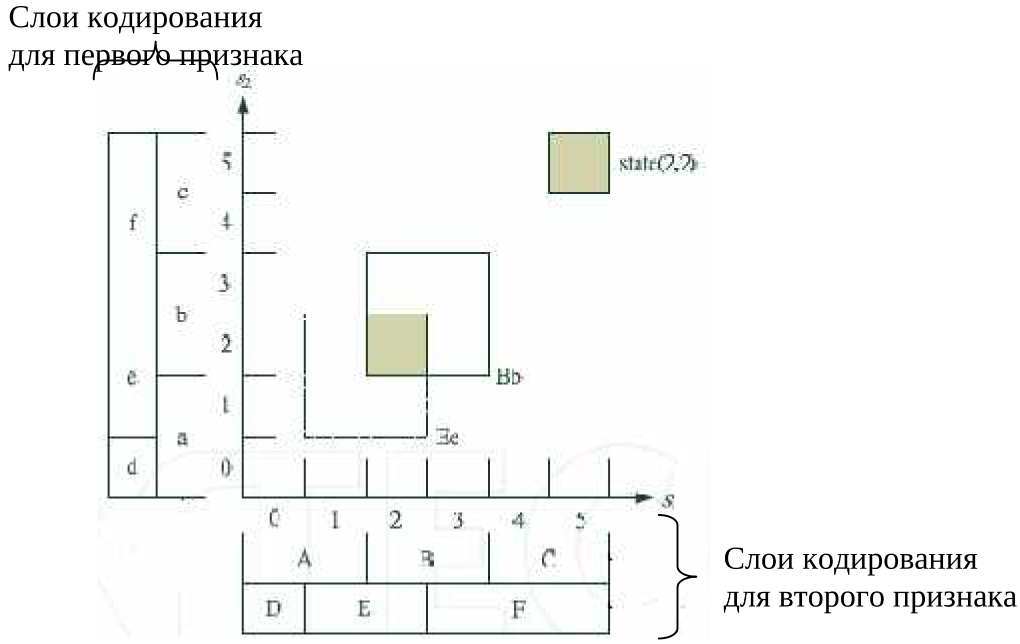


Рис.4.13. Пример многослойного кодирования предварительно квантованного двумерного входного вектора

Обучение ИНС

1. Номер итерации обучения $n=1$, инициализация посредством равномерного распределения на интервале (0,1) или [-0.5, 0.5] весов $w_{ij}(n)$, $i \in \overline{1,N^{(1)}}$, $j \in \overline{1,N^{(2)}}$, где $N^{(1)}$ – количество нейронов скрытого слоя, $N^{(2)}$ – количество нейронов выходного слоя, задание количества слоев кодирования N^e, задание количества состояний N^s на которые квантуется входной вектор, причем чаще всего выполняется $N^s \bmod N^e = 0$.

2. Задается обучающее множество $\{(\mathbf{x}_\mu, \mathbf{d}_\mu) \mid \mathbf{x}_\mu \in R^{N^{(0)}}, \mathbf{d}_\mu \in R^{N^{(2)}}\}$, $\mu \in \overline{1,P}$, где $\mathbf{x}_\mu$ – μ-й обучающий входной вектор, $\mathbf{d}_\mu$ – μ-й обучающий выходной вектор, $N^{(0)}$ – количество нейронов входного слоя, обычно $N^{(0)} = N^{(2)}$, P – мощность обучающего множества. Номер текущей пары из обучающего множества $\mu = 1$.

3. Квантование входного вектора

Преобразование вещественного вектора $\mathbf{x}_\mu$ в целочисленный вектор $\mathbf{z}(n)$ выполняется в виде

$$z_i(n) = \min\left(floor\left(N^s \frac{x_{\mu i} - \min_i}{\max_i - \min_i} \right), N^s - 1 \right) + 1,\ i \in \overline{1, N^{(0)}},$$

где $floor()$ – целая часть числа.

Таким образом $z_i(n) \in \{1,.., N^s\}$

4. Активизация нейронов скрытого слоя (соответствует ассоциативной, концептуальной или виртуальной памяти)

Количество нейронов в первом слое предложим определить как

$$N^{(1)} = \left(\frac{N^s}{N^e} \right)^{N^{(0)}} + \left(\frac{N^s}{N^e} + 1 \right)^{N^{(0)}} \left(N^e - 1 \right).$$

Каждый вектор $\mathbf{z}(n)$ активирует N^e ячеек бинарного вектора $\mathbf{a}(n)$, содержащего текущее состояние всех нейронов скрытого слоя, в виде $\mathbf{a}(n) = M(\mathbf{z}(n))$.

5. Вычисление выходного сигнала выходного слоя

$$y_j(n) = \sum_{i=1}^{N^{(1)}} w_{ij}(n) a_i(n),\ j \in \overline{1, N^{(2)}}.$$

6. Вычисление энергии ошибки ИНС

$$E(n) = \frac{1}{2} \sum_{j=1}^{N^{(3)}} e_j^2(n),\ e_j(n) = y_j(n) - d_{\mu j}.$$

7. Настройка синаптических весов на основе дельта правила

$$w_{ij}(n+1) = w_{ij}(n) + \eta a_i(n)(d_{\mu j} - y_j(n)) / N^e,\ i \in \overline{1, N^{(1)}},\ j \in \overline{1, N^{(2)}},$$

где η – параметр, определяющий скорость обучения (при большом η обучение происходит быстрее, но увеличивается опасность получить неверное решение), $0 < \eta < 1$.

8. Проверка условия завершения

Если $n \bmod P > 0$, то $\mu = \mu + 1$, $n = n + 1$, переход к 3.

Если $n \bmod P = 0$ и $\frac{1}{P} \sum_{s=1}^{P} E(n - P + s) > \varepsilon$, то $n = n + 1$, переход к 2.

Если $n \bmod P = 0$ и $\frac{1}{P} \sum_{s=1}^{P} E(n - P + s) < \varepsilon$, то завершиться.

Функционирование ИНС

$$z_i(n) = \min\left(floor\left(N^s \frac{x_{\mu i} - \min_i}{\max_i - \min_i} \right), N^s - 1 \right) + 1,\ i \in \overline{1, N^{(0)}},$$

$$\mathbf{a}(n) = M(\mathbf{z}(n)),$$

$$y_j = \sum_{i=1}^{N^{(1)}} w_{ij} a_i,\ j \in \overline{1, N^{(2)}}.$$

Результатом является образец $\mathbf{y}$.

4.10.2. Современная церебральная модель артикуляционного контроллера

На рис.4.14 приведена современная церебральная модель артикуляционного контроллера (CMAC) [32], которая является нерекуррентной статической трехслойной ИНС. Выход ИНС является линейным.

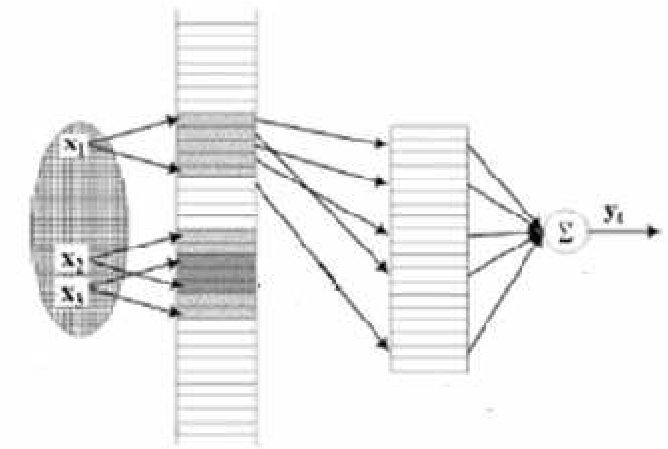


Рис.4.14. Современная церебральная модель артикуляционного контроллера (CMAC)

На рис.4.15 приведен пример многослойного кодирования, которое представляет собой отображение предварительно квантованного входного вектора (оба элемента входного вектора представлены 16 состояниями, пронумерованными от 0 до 15) в ячейки ассоциативной (концептуальной или виртуальной) памяти. Каждый слой кодирования разбит на блоки (на рис.4.15 представлены прямоугольниками и подписаны большими и малыми буквами), причем в каждом последующем слое эти блоки обычно смещаются на 1, а длина блока и количество слоев кодирования обычно совпадают (на рис.4.15 они равны 4, при этом межблочное перекрытие равно 3).

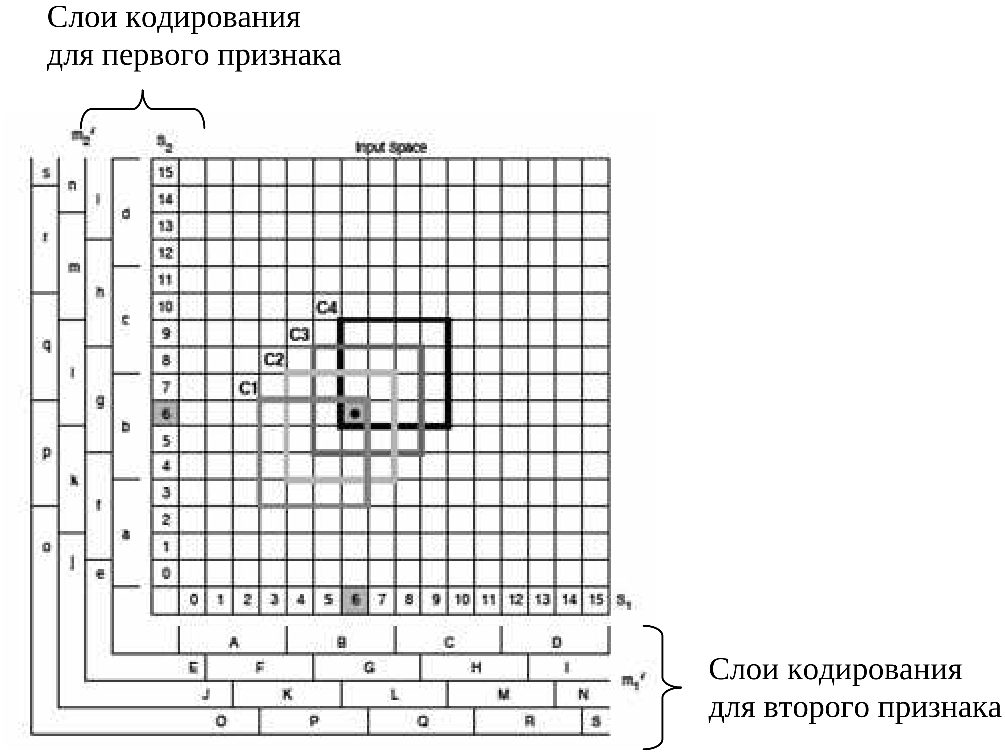


Рис.4.15. Пример многослойного кодирования предварительно квантованного двумерного входного вектора

На рис.4.16 приведен пример однослойного кодирования, которое представляет собой отображение предварительно квантованного входного вектора (оба элемента входного вектора представлены 9 состояниями, пронумерованными от 0 до 8) в ячейки ассоциативной (концептуальной или виртуальной) памяти.

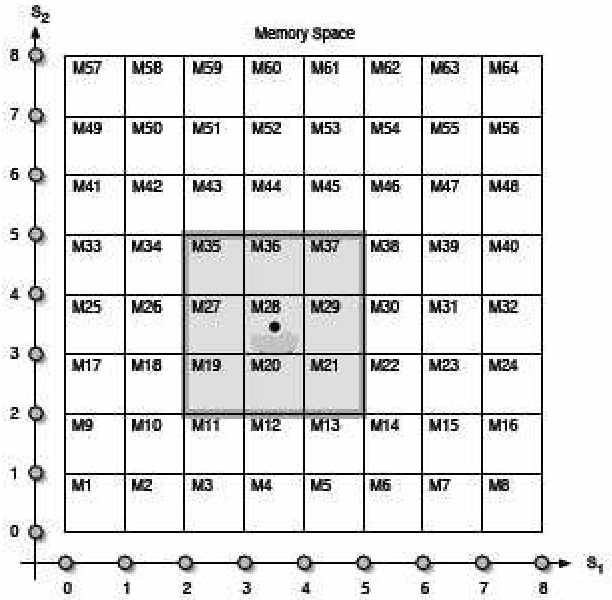
Ячейки ассоциативной памяти отображаются в ячейки физической памяти посредством хеширования.

Современная CMAC реализует гетероассоциативную память (ее элемент представлен парой образцов $(\mathbf{m}_x, \mathbf{m}_y)$, $\mathbf{m}_y \neq \mathbf{m}_x$) и восстанавливает запомненный образец $\mathbf{m}_y$ по ключевому образцу $\mathbf{m}_x$, соответствующему входному вектору $\mathbf{x}$.

Важнейшим свойством современной CMAC является то, что одна и та же ИНС с одними и теми же весами связей может хранить и воспроизводить несколько различных запомненных образцов.

Для обучения CMAC используется обучение на основе коррекции ошибок (обучение с учителем).

Слои кодирования
для первого признака



Слои кодирования
для второго признака

Рис.4.16. Пример однослойного кодирования предварительно квантованного двумерного входного вектора

Обучение ИНС для случая многослойного кодирования

1. Номер итерации обучения $n=1$, инициализация посредством равномерного распределения на интервале (0,1) или [-0.5, 0.5] весов $w_{ij}(n)$, $i \in \overline{1, N^{(2)}}$, $j \in \overline{1, N^{(3)}}$, где $N^{(2)}$ – количество нейронов второго скрытого слоя, $N^{(3)}$ – количество нейронов выходного слоя (или длина вектора откликов), задание количества слоев квантования N^e, задание количества состояний N^s на которые квантуется входной вектор, причем чаще всего выполняется $N^s \bmod N^e = 0$.

2. Задается обучающее множество $\{(\mathbf{x}_\mu, \mathbf{d}_\mu) \mid \mathbf{x}_\mu \in R^{N^{(0)}}, \mathbf{d}_\mu \in R^{N^{(3)}}\}$, $\mu \in \overline{1, P}$, где $\mathbf{x}_\mu$ – μ-й обучающий входной вектор, $\mathbf{d}_\mu$ – μ-й обучающий выходной вектор, $N^{(0)}$ – количество нейронов входного слоя, обычно $N^{(0)} = N^{(3)}$, P – мощность обучающего множества. Номер текущей пары из обучающего множества $\mu = 1$.

3. Квантование входного вектора

Преобразование вещественного вектора $\mathbf{x}_\mu$ в целочисленный вектор $\mathbf{z}(n)$ выполняется в виде

$$z_i(n)=\min\left(floor\left(N^s\frac{x_{\mu i}-\min_i}{\max_i-\min_i}\right),N^s-1\right)+1,\ i\in\overline{1,N^{(0)}},$$

где $floor()$ – целая часть числа.

Таким образом $z_i(n)\in\{1,..,N^s\}$.

4. Активизация нейронов первого слоя (соответствует ассоциативной, концептуальной или виртуальной памяти)

Количество нейронов в первом слое предложим определить как

$$N^{(1)}=\left(\frac{N^s}{N^e}\right)^{N^{(0)}}+\left(\frac{N^s}{N^e}+1\right)^{N^{(0)}}\left(N^e-1\right).$$

Каждый вектор $\mathbf{z}(n)$ активирует N^e ячеек бинарного вектора $\mathbf{a}(n)$, содержащего текущее состояние всех нейронов первого слоя, в виде $\mathbf{a}(n)=M(\mathbf{z}(n))$.

5. Активизация нейронов второго слоя (соответствует физической памяти или хеш-памяти)

$$\breve{a}_{H(i)}(n)=a_i(n),\ i\in\overline{1,N^{(1)}},$$

где $H()$ – хеш-функция, $H(i)\in\overline{1,N^{(2)}}$.

6. Вычисление выходного сигнала выходного слоя

$$y_j(n)=\sum_{i=1}^{N^{(2)}}w_{ij}(n)\breve{a}_i(n),\ j\in\overline{1,N^{(3)}}.$$

7. Вычисление энергии ошибки ИНС

$$E(n)=\frac{1}{2}\sum_{j=1}^{N^{(3)}}e_j^2(n),\ e_j(n)=y_j(n)-d_{\mu j}.$$

8. Настройка синаптических весов на основе дельта правила

$$w_{ij}(n+1)=w_{ij}(n)+\eta\breve{a}_i(n)(d_{\mu j}-y_j(n))/N^e,\ i\in\overline{1,N^{(2)}},\ j\in\overline{1,N^{(3)}},$$

где η – параметр, определяющий скорость обучения (при большом η обучение происходит быстрее, но увеличивается опасность получить неверное решение), $0<\eta<1$.

9. Проверка условия завершения

Если $n \bmod P>0$, то $\mu=\mu+1$, $n=n+1$, переход к 3.

Если $n \bmod P=0$ и $\frac{1}{P}\sum_{s=1}^{P}E(n-P+s)>\varepsilon$, то $n=n+1$, переход к 2.

Если $n \bmod P = 0$ и $\frac{1}{P}\sum_{s=1}^{P} E(n-P+s) < \varepsilon$, то завершиться.

Функционирование ИНС для случая многослойного кодирования

$$z_i(n) = \min\left(floor\left(N^s \frac{x_{\mu i} - \min_i}{\max_i - \min_i} \right), N^s - 1 \right) + 1,\ i \in \overline{1, N^{(0)}},$$

$$\mathbf{a}(n) = M(\mathbf{z}(n)),$$

$$\breve{a}_{H(i)}(n) = a_i(n),\ i \in \overline{1, N^{(1)}},$$

$$y_j = \sum_{i=1}^{N^{(2)}} w_{ij} \breve{a}_i,\ j \in \overline{1, N^{(3)}}.$$

Результатом является образец $\mathbf{y}$.

Обучение ИНС для случая однослойного кодирования

1. Номер итерации обучения $n = 1$, инициализация посредством равномерного распределения на интервале (0,1) или [-0.5, 0.5] весов $w_{ij}(n)$, $i \in \overline{1, N^{(2)}}$, $j \in \overline{1, N^{(3)}}$, где $N^{(2)}$ – количество нейронов второго скрытого слоя, $N^{(3)}$ – количество нейронов выходного слоя, задание радиуса окрестности r для ячейки ассоциативной памяти, задание количества состояний N^s на которые квантуется входной вектор.

2. Задается обучающее множество $\{(\mathbf{x}_\mu, \mathbf{d}_\mu) \mid \mathbf{x}_\mu \in R^{N^{(0)}}, \mathbf{d}_\mu \in R^{N^{(3)}}\}$, $\mu \in \overline{1, P}$, где $\mathbf{x}_\mu$ – μ-й обучающий входной вектор, $\mathbf{d}_\mu$ – μ-й обучающий выходной вектор, $N^{(0)}$ – количество нейронов входного слоя, обычно $N^{(0)} = N^{(3)}$, P – мощность обучающего множества. Номер текущей пары из обучающего множества $\mu = 1$.

3. Квантование входного вектора

Преобразование вещественного вектора $\mathbf{x}_\mu$ в целочисленный вектор $\mathbf{z}(n)$ выполняется в виде

$$z_i(n) = \min\left(floor\left(N^s \frac{x_{\mu i} - \min_i}{\max_i - \min_i} \right), N^s - 1 \right) + 1,\ i \in \overline{1, N^{(0)}},$$

где $floor()$ – целая часть числа.

Таким образом $z_i(n) \in \{1,.., N^s\}$.

4. Активизация нейронов первого слоя (соответствует ассоциативной, концептуальной или виртуальной памяти)

Каждый вектор $\mathbf{z}(n)$ для гиперкубической окрестности активирует N^e, $(r+1)^{N^{(0)}} \le N^e \le (2r+1)^{N^{(0)}}$, ячеек бинарного вектора $\mathbf{a}(n)$, содержащего текущее состояние всех нейронов первого слоя, в виде $\mathbf{a}(n) = M(\mathbf{z}(n))$.

5. Взвешивание выходов нейронов первого слоя

Взвешивание выходов нейронов первого слоя осуществляется путем умножения на базисную функцию

$$\hat{a}_i(n) = a_i(n)\Phi(z_i(n)),\ i \in \overline{1, N^{(1)}}.$$

6. Активизация нейронов второго слоя (соответствует физической памяти или хеш-памяти)

$$\breve{a}_{H(i)}(n) = \hat{a}_i(n),\ i \in \overline{1, N^{(1)}},$$

где $H()$ – хеш-функция, $H(i) \in \overline{1, N^{(2)}}$.

7. Вычисление выходного сигнала выходного слоя

$$y_j(n) = \sum_{i=1}^{N^{(2)}} w_{ij}(n)\breve{a}_i(n),\ j \in \overline{1, N^{(3)}}.$$

8. Вычисление энергии ошибки ИНС

$$E(n) = \frac{1}{2}\sum_{j=1}^{N^{(3)}} e_j^2(n),\ e_j(n) = y_j(n) - d_{\mu j}.$$

9. Настройка синаптических весов на основе дельта правила

$$w_{ij}(n+1) = w_{ij}(n) + \eta\breve{a}_i(n)(d_{\mu j} - y_j(n)) / N^e,\ i \in \overline{1, N^{(2)}},\ j \in \overline{1, N^{(3)}},$$

где η – параметр, определяющий скорость обучения (при большом η обучение происходит быстрее, но увеличивается опасность получить неверное решение), $0 < \eta < 1$.

10. Проверка условия завершения

Если $n \bmod P > 0$, то $\mu = \mu + 1$, $n = n + 1$, переход к 3.

Если $n \bmod P = 0$ и $\frac{1}{P}\sum_{s=1}^{P} E(n - P + s) > \varepsilon$, то $n = n + 1$, переход к 2.

Если $n \bmod P = 0$ и $\frac{1}{P}\sum_{s=1}^{P} E(n - P + s) < \varepsilon$, то завершиться.

Функционирование ИНС для случая однослойного кодирования

$$z_i(n) = \min\left(floor\left(N^s \frac{x_{\mu i} - \min_i}{\max_i - \min_i} \right), N^s - 1 \right) + 1,\ i \in \overline{1, N^{(0)}},$$

$$\hat{\mathbf{a}}(n) = M(\mathbf{z}(n))\Phi(\mathbf{z}(n)),$$

$$\breve{a}_{H(i)}(n) = \hat{a}_i(n),\ i \in \overline{1, N^{(1)}},$$

$$y_j = \sum_{i=1}^{N^{(2)}} w_{ij}\breve{a}_i,\ j \in \overline{1, N^{(3)}}.$$

Результатом является образец $\mathbf{y}$.

Виды хеш-функций

1. $H(k) = 1 + k \bmod N^{(2)}$

2. $H(k) = 1 + round\left(N^{(2)} round\left(\frac{Fk}{w} \bmod 1 \right) \right),$

$w = 2^{30},\ F = 663608941,$

где $round()$ – округление до ближайшего целого.

Виды базисных функций

1. Прямоугольная (B-сплайн первого порядка)

$\Phi_i(x) = 1.$

2. Тригонометрическая (косинусоидальная)

$$\Phi_i(x) = \begin{cases} \cos\left(\frac{\pi}{2r}(x - \mu_i) \right), & x \in [\mu_i - r, \mu_i + r] \\ 0, & \textit{иначе} \end{cases},$$

где μ_i – центр области квантования.

3. Гауссова

$$\Phi_i(x) = \exp\left(-\frac{(x - \mu_i)^2}{\sigma_i^2} \right),$$

где μ_i – центр области квантования, σ_i – среднеквадратичное отклонение.

4. Модифицированная Гауссова

$$\Phi_i(x) = \begin{cases} \exp\left(-\dfrac{(\lambda_2 - \lambda_1)^2 / 4}{(x-\lambda_1)(\lambda_2 - x)}\right), & x \in (\lambda_1, \lambda_2) \\ 0, & \textit{иначе} \end{cases},$$

где λ_i – узел сплайна (центр области квантования).

5. Сплайны

$$B_{n,i}(x) = \frac{x - \lambda_{i-n}}{\lambda_{i-1} - \lambda_{i-n}} B_{n-1,i-1}(x) + \frac{\lambda_i - x}{\lambda_i - \lambda_{i-n+1}} B_{n-1,i}(x),$$

$$B_{0,i}(x) = \begin{cases} 1, & x \in [\lambda_{i-1}, \lambda_i) \\ 0, & x \notin [\lambda_{i-1}, \lambda_i) \end{cases}.$$

6. Сплайны порядка δ

$$B_{n,i}^{\delta}(x) = \frac{n-1}{n-\delta-1}\left(\left(\frac{x-\lambda_{i-n}}{\lambda_{i-1}-\lambda_{i-n}}\right)^{\delta} B_{n-1,i-1}(x) + \left(\frac{\lambda_i - x}{\lambda_i - \lambda_{i-n+1}}\right)^{\delta} B_{n-1,i}(x)\right),$$

$$B_{0,i}^{\delta}(x) = \begin{cases} 1, & x \in [\lambda_{i-1}, \lambda_i) \\ 0, & x \notin [\lambda_{i-1}, \lambda_i) \end{cases},$$

$$B_{n,i}^{\delta}(x) = \frac{B_{n-1,i-1}^{\delta-1}(x)}{\lambda_{i-1} - \lambda_{i-n}} - \frac{B_{n-1,i}^{\delta-1}(x)}{\lambda_i - \lambda_{i-n+1}}.$$

Пример активизации нейронов первого слоя в случае многослойного кодирования

Пусть имеются предварительно квантованные вектора $z(1) = (2{,}1)^T, z(2) = (2{,}2)^T, z(3) = (3{,}2)^T, z(4) = (4{,}6)^T$, $P = 4$, $N^{(0)} = 2$, $N^e = 3$, $N^s = 6$. Схема многослойного кодирования изображена на рис.4.17.

$a(n) = (a_{aA}, a_{aB}, a_{bA}, a_{bB}, a_{cC}, a_{cD}, a_{cE}, a_{dC}, a_{dD}, a_{dE}, a_{eC}, a_{eD}, a_{eE}, a_{fF}, a_{fG}, a_{fH}, a_{gF}, a_{gG}, a_{gH}, a_{hF}, a_{hG}, a_{hH})^T$, $N^{(1)} = 22$.

Таким образом

при $x(1) = (2{,}1)^T$ $a(1) = (0{,}0{,}0{,}1{,}0{,}0{,}0{,}0{,}0{,}0{,}0{,}0{,}1{,}0{,}0{,}0{,}0{,}0{,}1{,}0{,}0{,}0)^T$,

при $x(2) = (2{,}2)^T$ $a(2) = (0{,}0{,}0{,}1{,}0{,}0{,}0{,}0{,}0{,}0{,}0{,}0{,}1{,}0{,}0{,}0{,}0{,}1{,}0{,}0{,}0{,}0)^T$,

при $x(3) = (3{,}2)^T$ $a(3) = (0{,}0{,}0{,}1{,}0{,}0{,}0{,}0{,}0{,}1{,}0{,}0{,}0{,}0{,}0{,}0{,}0{,}1{,}0{,}0{,}0{,}0)^T$,

при $x(4) = (4{,}6)^T$ $a(4) = (1{,}0{,}0{,}0{,}0{,}0{,}0{,}1{,}0{,}0{,}0{,}0{,}0{,}0{,}0{,}0{,}1{,}0{,}0{,}0{,}0{,}0)^T$.

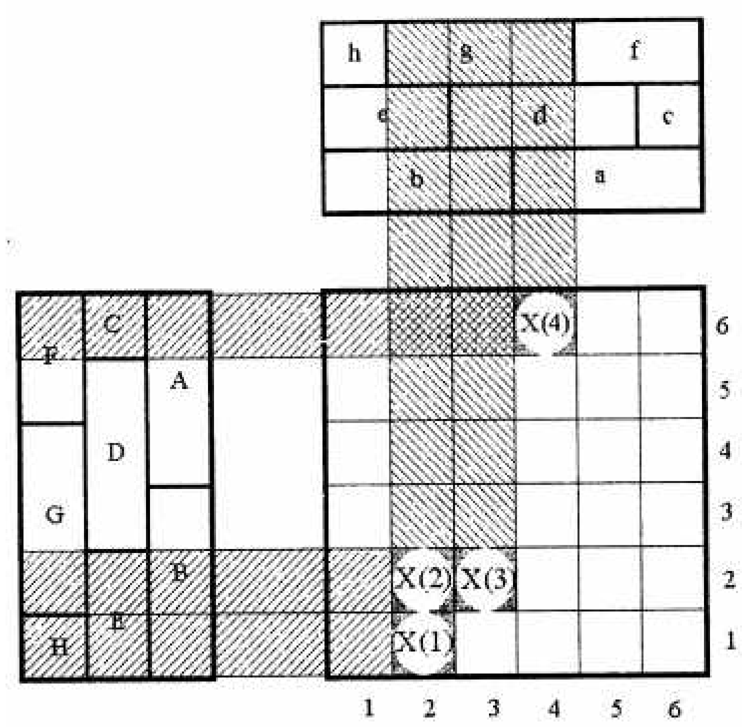


Рис.4.17. Многослойное кодирование квантованного двумерного вектора

Пример активизации нейронов первого слоя в случае однослойного кодирования

Пусть имеются предварительно квантованные вектора $z(1)=(2{,}1)^T$, $z(2)=(2{,}2)^T$, $z(3)=(3{,}2)^T$, $z(4)=(4{,}6)^T$, $P=4$, $N^{(0)}=2$, $r=1$. Схема однослойного кодирования изображена на рис.4.18.

$a(n)=(a_{11},a_{21},a_{31},a_{41},a_{51},a_{61},a_{12},a_{22},a_{32},a_{42},a_{52},a_{62},$
$a_{13},a_{23},a_{33},a_{43},a_{53},a_{63},a_{14},a_{24},a_{34},a_{44},a_{54},a_{64},$
$a_{15},a_{25},a_{35},a_{45},a_{55},a_{65},a_{16},a_{26},a_{36},a_{46},a_{56},a_{66})^T$, $N^{(1)}=36$.

Таким образом

при $x(1)=(2{,}1)^T$

$a(1)=(1,1,1,0,0,0,1,1,1,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0)^T$,

при $x(2)=(2{,}2)^T$

$a(2)=(1,1,1,0,0,0,1,1,1,0,0,0,1,1,1,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0)^T$,

при $x(3)=(3{,}2)^T$

$a(3)=(0,1,1,1,0,0,0,1,1,1,0,0,0,1,1,1,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0)^T$,

при $x(4)=(4{,}6)^T$

$a(4)=(0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,1,1,1,0,0,0,1,1,1,0)^T$.

			$x(4)$			6
						5
						4
						3
	$x(2)$	$x(3)$				2
	$x(1)$					1
1	2	3	4	5	6	

Рис.4.18. Однослойное кодирование квантованного двумерного вектора

Достоинства

1. Является универсальным аппроксиматором. Обеспечивает локальную аппроксимацию нелинейного отображения входного сигнала в выходной.
2. Обеспечивает хорошее качество обобщения.

Недостатки

1. Не существует оптимальной схемы кодирования информации.
2. Имеет ограниченную область применения (обычно используется только для аппроксимации функций).
3. Отсутствует автоматическое определение числа нейронов в скрытых слоях.
4. Эффективность CMAC зависит от вида восстанавливаемой функции.
5. Использование хеширования приводит к коллизиям (несколько ячеек ассоциативной памяти соответствуют одной ячейке физической памяти), а его неиспользование – к нестабильному обучению и слабому использованию ассоциативной памяти.

ГЛАВА 5
НЕРЕКУРРЕНТНЫЕ ДИНАМИЧЕСКИЕ ИСКУССТВЕННЫЕ НЕЙРОННЫЕ СЕТИ

5.1. Нейросеть нелинейной авторегрессии

Нейросеть нелинейной авторегрессии (NARNN) является нерекуррентной динамической двуслойной ИНС [39]. На рис.5.1 скрытый слой изображен прямоугольником, за основу бралась модель нелинейной авторегрессии NAR(5). NARNN является особым классом многослойного персептрона

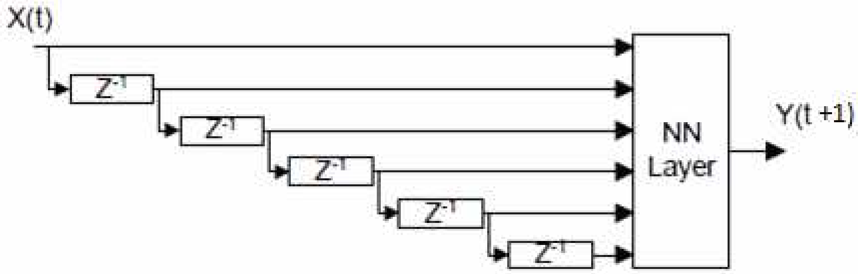


Рис. 5.1. Нейросеть нелинейной авторегрессии (NARNN)

Для NARNN используется обучение на основе коррекции ошибок (обучение с учителем), при этом чаще всего применяется алгоритм обратного распространения (BP). Это итеративный градиентный алгоритм обучения, который обеспечивает минимизацию среднеквадратичной ошибки.

Обучение ИНС (алгоритм обратного распространения)

1. Номер итерации обучения $n=1$, инициализация смещений (порогов) $b^{(1)}(n), b^{(2)}(n)$ и весов $w_{lj}^{(1)}(n)$, $w_i^{(2)}(n)$, $i, j \in \overline{1, N^{(1)}}, l \in \overline{0, M^{(0)}}$, где $N^{(1)}$ – количество нейронов в первом слое, $M^{(0)}$ – задержка во входном слое.

2. Задается обучающее множество $\{(x_\mu, d_\mu) \mid x_\mu \in R, d_\mu \in R\}$, $\mu \in \overline{1, P}$, где x_μ – μ-е обучающее входное значение, d_μ – μ-е обучающее выходное значение, P – мощность обучающего множества. Номер текущей пары из обучающего множества $\mu = 1$.

3. Начальное вычисление выходного сигнала для входного слоя

$y^{(0)}(n - v) = 0$, $v \in \overline{1, M^{(0)}}$.

4. Вычисление выходного сигнала для каждого слоя (прямой ход)

$y^{(0)}(n) = x_\mu$,

$$y_j^{(1)}(n) = f^{(1)}(s_j^{(1)}(n)), \qquad s_j^{(1)}(n) = b_j^{(1)} + \sum_{l=0}^{M^{(0)}} w_{lj}^{(1)}(n) y^{(0)}(n-l),$$

$j \in \overline{1, N^{(1)}}$,

$$y^{(2)}(n) = f^{(2)}(s^{(2)}(n)),\ s^{(2)}(n) = \sum_{i=0}^{N^{(1)}} w_i^{(2)}(n) y_i^{(1)}(n),$$

где $N^{(1)}$ – число нейронов в первом слое, $w_{lj}^{(1)}(n)$ – вес связи от входного нейрона в момент времени $n-l$ к j-му нейрону в первом слое в момент времени n, $w_i^{(2)}(n)$ – вес связи от i-го нейрона к выходному нейрону в момент времени n, $y_j^{(1)}(n)$ – выход j-го нейрона в первом слое, $y^{(2)}(n)$ – выход нейрона во втором слое, $f^{(k)}$ – функция активация нейронов k-го слоя.

Считается, что $w_0^{(2)}(n) = b^{(2)}(n), y_0^{(1)}(n) = 1$.

5. Вычисление энергии ошибки ИНС

$$E(n) = \frac{1}{2} e^2(n),\ e(n) = y^{(2)}(n) - d_\mu.$$

6. Настройка синаптических весов на основе обобщенного дельта правила (обратный ход)

$$w_i^{(2)}(n+1) = w_i^{(2)}(n) - \eta \frac{\partial E(n)}{\partial w_i^{(2)}(n)},$$

$$w_{lj}^{(1)}(n+1) = w_{lj}^{(1)}(n) - \eta \frac{\partial E(n)}{\partial w_{lj}^{(1)}(n)},$$

$$\frac{\partial E(n)}{\partial w_i^{(2)}(n)} = y_i^{(1)}(n) g^{(2)}(n),\ i \in \overline{0, N^{(1)}},$$

$$\frac{\partial E(n)}{\partial w_{lj}^{(1)}(n)} = y^{(0)}(n-l) g_j^{(1)}(n),\ j \in \overline{1, N^{(1)}}, l \in \overline{0, M^{(0)}},$$

$$\frac{\partial E(n)}{\partial b_j^{(1)}(n)} = g_j^{(1)}(n),\ j \in \overline{1, N^{(1)}},$$

$$g^{(2)}(n) = f'^{(2)}(s^{(2)}(n))(y^{(2)}(n) - d_\mu),$$

$$g_j^{(1)}(n) = f'^{(1)}(s_j^{(1)}(n))w_j^{(2)}(n)g^{(2)}(n)$$

7. Проверка условия завершения

Если $n \bmod P > 0$, то $\mu = \mu + 1$, $n = n + 1$, переход к 4.

Если $n \bmod P = 0$ и $\frac{1}{P}\sum_{s=1}^{P}(y(n - P + s) - d_s) > \varepsilon$, то $n = n + 1$, переход к 2.

Если $n \bmod P = 0$ и $\frac{1}{P}\sum_{s=1}^{P}(y(n - P + s) - d_s) < \varepsilon$, то завершиться.

Функционирование ИНС

1. Инициализация

$n = 1$.

$y^{(0)}(n - v) = 0$, $v \in \overline{1, M^{(0)}}$.

2. Прогноз

$y(n) = x_1$.

$$y_j^{(1)}(n) = f^{(1)}(b^{(1)} + \sum_{l=0}^{M^{(0)}} w_{lj}^{(1)} y^{(0)}(n - l)).$$

$$y^{(2)}(n) = f^{(2)}(b^{(2)} + \sum_{i=1}^{N^{(1)}} w_i^{(2)} y_i^{(1)}(n)).$$

Достоинства

1. Является универсальным аппроксиматором. Обеспечивает глобальную аппроксимацию нелинейного отображения входного сигнала в выходной.

2. Обеспечивает хорошее качество обобщения.

3. Автоматически определяется количество скрытых слоев (равно одному).

4. В отличие от статических ИНС позволяет осуществлять адаптивную фильтрацию, прогноз, адаптивное управление, параметрическую идентификацию модели, классификацию нестационарных сигналов.

5. В отличие от NARMANN обучается быстрее.

Недостатки

1. Обучение происходит медленнее, чем в случае MLP, RBFNN, PNN, сети Хемминга, SOM, CPNN.

2. Отсутствует автоматическое определение числа нейронов в скрытом слое.

3. В отличие от методов обучения SVM, градиентные методы обучения NARNN выбирают положение разделяющей гиперплоскости произвольным образом.

4. Обучение не сводится к задаче квадратичного программирования в выпуклой области, имеющей единственное решение.

5. В отличие от ART, не решает проблему пластичности-стабильности.

6. Может быть только два класса, что не всегда подходит для задач классификации.

7. В отличие от NARMANN хуже решает задачи адаптивной фильтрации, прогноза, адаптивного управления, параметрической идентификации модели, классификации нестационарных сигналов.

8. В отличие от LSTM не работает с нестационарными последовательностями.

5.2. Нейросеть Вольтерри

На рис.5.2 приведен пример нейросети Вольтерри (VNN) [40], которая является нерекуррентной динамической многослойной ИНС. Классы разделяются гиперплоскостями. В общем случае количество уровней может быть больше трех. Эта ИНС относятся к ИНС высокого порядка.

Для VNN используется обучение на основе коррекции ошибок (обучение с учителем), при этом чаще всего применяется алгоритм обратного распространения (BP). Это итеративный градиентный алгоритм обучения, который обеспечивает минимизацию среднеквадратичной ошибки.

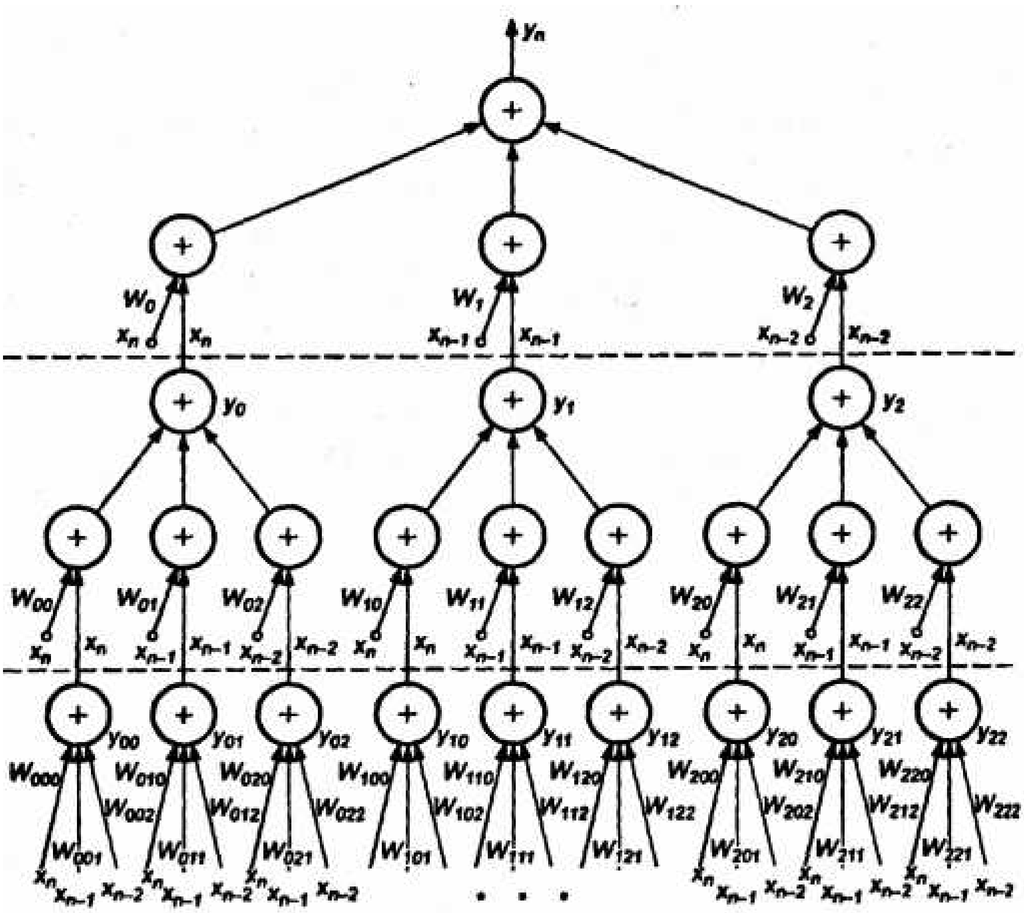


Рис. 5.2. Трехуровневая нейросеть Вольтерри (VNN)

Обучение ИНС (алгоритм обратного распространения во времени)

1. Номер итерации обучения $n=1$, инициализация посредством равномерного распределения на интервале (0,1) или [-0.5, 0.5] весов $w_{i_1},\ldots,w_{i_1\ldots i_K}$, $i_1,\ldots,i_K \in \overline{0,M}$, где M – задержка, K – количество уровней.

2. Задается обучающее множество $\{(x_\mu, d_\mu) \mid x_\mu \in R, d_\mu \in R\}$, $\mu \in \overline{1,P}$, где x_μ – μ-е обучающее входное значение, d_μ – μ-е обучающее выходное значение P – мощность обучающего множества. Номер текущей пары из обучающего множества $\mu = 1$.

3. Начальное вычисление выходного сигнала для нулевого слоя

$z(n-v)=0$, $v \in \overline{1,M}$.

4. Вычисление выходного сигнала (прямой ход)

$z(n) = x_\mu$.

$y(n) = f(s(n))$,

$$s(n) = \sum_{i_1=0}^{M} w_{i_1}(n) z(n-i_1) + \ldots + \sum_{i_1=0}^{M} \ldots \sum_{i_K=0}^{M} w_{i_1 \ldots i_K}(n) z(n-i_1) \ldots z(n-i_K) .$$

5. Вычисление энергии ошибки ИНС

$$E(n) = \frac{1}{2} e_j^2(n), \; e_j(n) = y(n) - d_{\mu j} .$$

6. Настройка синаптических весов на основе обобщенного дельта правила (обратный ход)

$$w_{i_1}(n) = w_{i_1}(n) - \mu \frac{\partial E(n)}{\partial w_{i_1}(n)}$$

$$\frac{\partial E(n)}{\partial w_{i_1}(n)} = f'(s(n))(y(n) - d_\mu) z(n-i_1)$$

…

$$w_{i_1 \ldots i_K}(n) = w_{i_1 \ldots i_K}(n) - \mu \frac{\partial E(n)}{\partial w_{i_1 \ldots i_K}(n)}$$

$$\frac{\partial E(n)}{\partial w_{i_1 \ldots i_K}(n)} = f'(s(n))(y(n) - d_\mu) z(n-i_1) \ldots z(n-i_K)$$

7. Проверка условия завершения

Если $n \bmod P > 0$, то $\mu = \mu + 1$, $n = n + 1$, переход к 4.

Если $n \bmod P = 0$ и $\frac{1}{P} \sum_{s=1}^{P} (y(n-P+s) - d_s) > \varepsilon$, то $n = n + 1$, переход к 2.

Если $n \bmod P = 0$ и $\frac{1}{P} \sum_{s=1}^{P} (y(n-P+s) - d_s) < \varepsilon$, то завершиться.

Функционирование ИНС

1. Инициализация

$n = 1$.

$z(n-v) = 0$, $v \in \overline{1, M}$.

2. Прогноз

$z(n) = x_1$.

$y(n) = f(s(n))$,

$$s(n) = \sum_{i_1=0}^{M} w_{i_1} z(n-i_1) + \ldots + \sum_{i_1=0}^{M} \ldots \sum_{i_K=0}^{M} w_{i_1 \ldots i_K} z(n-i_1) \ldots z(n-i_K)$$

Достоинства

1. Является универсальным аппроксиматором. Обеспечивает глобальную аппроксимацию нелинейного отображения входного сигнала в выходной.

2. Обеспечивает хорошее качество обобщения.

3. Количество классов может быть больше двух (в выходном слое может быть больше одного нейрона).

4. В отличие от статических ИНС, позволяет осуществлять адаптивную фильтрацию, прогноз, адаптивное управление, параметрическую идентификацию модели, классификацию нестационарных сигналов.

Недостатки

1. Обучение происходит медленнее, чем в случае MLP, RBFNN, PNN, сети Хемминга, SOM, CPNN.

2. Отсутствует автоматическое определение числа уровней и величины временной задержки.

3. В отличие от методов обучения SVM, градиентные методы обучения VNN выбирают положение разделяющей гиперплоскости произвольным образом.

4. Обучение не сводится к задаче квадратичного программирования в выпуклой области, имеющей единственное решение.

5. В отличие от ART, не решает проблему пластичности-стабильности.

6. В отличие от NARMANN хуже решает задачи адаптивной фильтрации, прогноза, адаптивного управления, параметрической идентификации модели, классификации нестационарных сигналов.

5.3. Нейросеть с задержкой по времени

Нейросеть с задержкой по времени (TDNN) [41,42] является нерекуррентной динамической трехслойной ИНС. Классы разделяются гиперплоскостями. На рис. 5.3 изображена TDNN для классификации в случае трех классов. TDNN является особым классом многослойного персептрона.

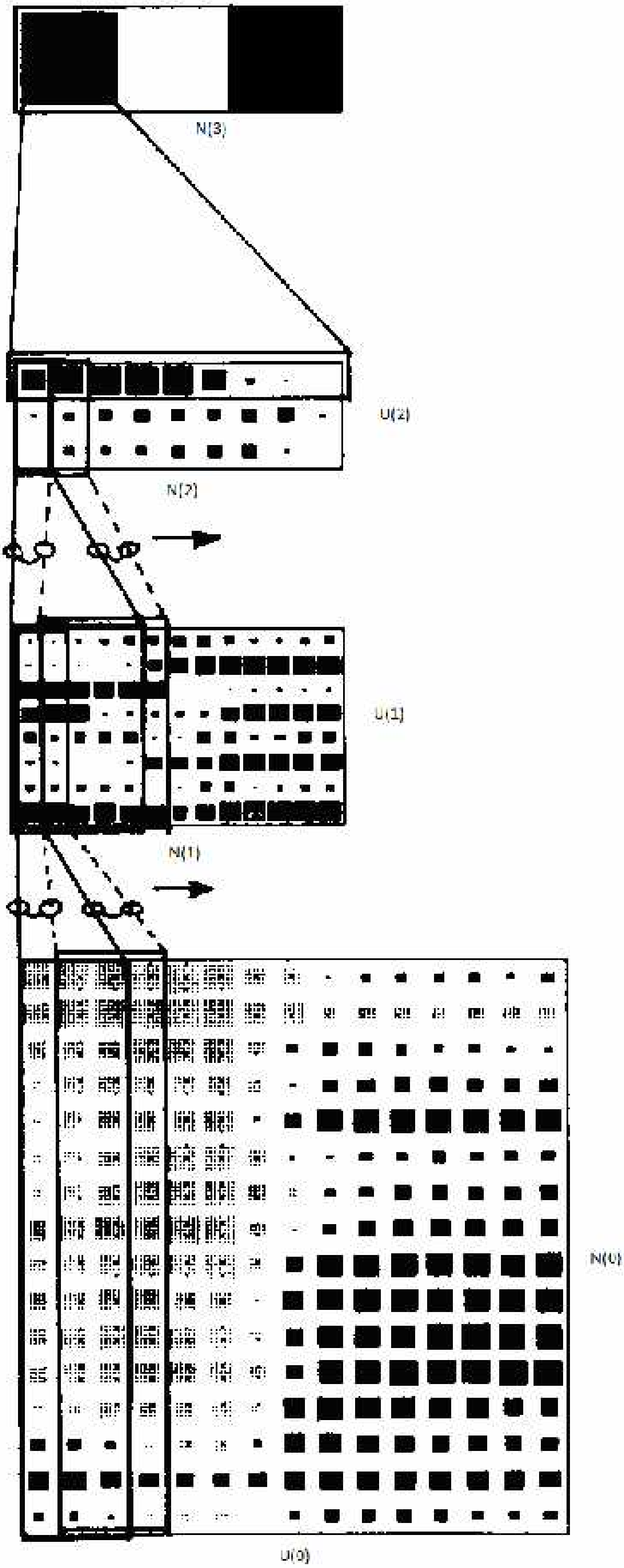


Рис. 5.3. Нейросеть с задержкой по времени (TDNN)

Для TDNN используется обучение на основе коррекции ошибок (обучение с учителем), при этом чаще всего применяется алгоритм обратного распространения (BP). Это итеративный градиентный алгоритм обучения, который обеспечивает минимизацию среднеквадратичной ошибки.

Обучение ИНС (алгоритм обратного распространения)

1. Номер итерации обучения $n=1$, инициализация посредством равномерного распределения на интервале (0,1) или [-0.5, 0.5] порогов $b_{iml}^{(1)}(n)$, $b_{jml}^{(2)}(n)$, $b_{i1i}^{(3)}(n)$ и весов $w_{ijml}^{(k)}(n)$, $i\in\overline{1,U^{(k)}}, j\in\overline{1,N^{(k)}}, m\in\overline{1,U^{(k+1)}}, l\in\overline{1,N^{(k+1)}}, k\in\overline{1,3}$, где $N^{(k)}$ – количество нейронов в одном модуле k-го слоя, $U^{(k)}$ – количество модулей k-го слоя.

2. Задается обучающее множество $\{(\mathbf{x}_\mu,\mathbf{d}_\mu)\,|\,\mathbf{x}_\mu\in R^{U^{(0)}\times N^{(0)}},\mathbf{d}_\mu\in\{0,1\}^{N^{(3)}}\}$, $\mu\in\overline{1,P}$, где $\mathbf{x}_\mu$ – μ-я обучающая входная матрица, $\mathbf{d}_\mu$ – μ-й обучающий выходной вектор, $N^{(0)}$ – количество нейронов в одном модуле входного слоя, $U^{(0)}$ – количество модулей (обучающих входных векторов), $N^{(3)}$ – количество нейронов выходного слоя, P – мощность обучающего множества. Номер текущей пары из обучающего множества $\mu=1$.

3. Вычисление выходного сигнала k-го слоя (прямой ход)

$$\mathbf{y}^{(0)}(n)=\mathbf{x}_\mu,$$

$$y_{ml}^{(1)}(n)=f^{(1)}(s_{ml}^{(1)}(n)),\qquad s_{ml}^{(1)}(n)=\sum_{i=l}^{l+M^{(0)}}\sum_{j=0}^{N^{(0)}}w_{ijml}^{(1)}(n)y_{ij}^{(0)}(n),$$

$m\in\overline{1,U^{(1)}}, l\in\overline{1,N^{(1)}}$.

$$y_{ml}^{(2)}(n)=f^{(2)}(s_{ml}^{(2)}(n)),\qquad s_{ml}^{(2)}(n)=\sum_{j=l}^{l+M^{(1)}}\sum_{i=0}^{U^{(1)}}w_{ijml}^{(2)}(n)y_{ij}^{(1)}(n),$$

$m\in\overline{1,U^{(2)}}, l\in\overline{1,N^{(2)}}$.

$$y_{1i}^{(3)}(n)=f^{(3)}(s_{1i}^{(3)}(n)),\ s_{1i}^{(3)}(n)=\sum_{j=0}^{N^{(2)}}w_{ij1i}^{(3)}(n)y_{ij}^{(2)}(n),\ i\in\overline{1,N^{(3)}},$$

где $N^{(k)}$ – число нейронов в *k*-м слое, *k* – номер слоя, $w_{ijml}^{(k)}(n)$ – вес связи от *j*-го нейрона *i*-го модуля к *l*-му нейрону *m*-го модуля на *k*-м скрытом слое в момент времени n, $y_{ml}^{(k)}(n)$ – выход *j*-го нейрона *m* -го модуля на *k*-м слое, $f^{(k)}$ – функция активация нейронов *k*-го слоя, $M^{(k)}$ – задержка в k-м слое, $M^{(0)}+N^{(1)}=U^{(0)}, M^{(1)}+N^{(2)}=N^{(1)}$, $U^{(2)}=N^{(3)}$.

Считается, что $w_{i0ml}^{(1)}(n)=b_{iml}^{(1)}(n)$, $y_{i0}^{(1)}(n)=1$,
$w_{0jml}^{(2)}(n)=b_{jml}^{(2)}(n)$, $y_{0j}^{(1)}(n)=1$, $w_{i01i}^{(3)}(n)=b_{i1i}^{(3)}(n)$, $y_{i0}^{(1)}(n)=1$.

4. Вычисление энергии ошибки ИНС

$$E(n)=\frac{1}{2}\sum_{j=1}^{N^{(3)}} e_j^2(n),\ e_j(n)=y_j^{(3)}(n)-d_{\mu j}.$$

5. Настройка синаптических весов (обратный ход) на основе обобщенного дельта правила

$$\frac{\partial E(n)}{\partial w_{ij1i}^{(3)}(n)}=y_{ij}^{(2)}(n)g_{1i}^{(3)}(n),$$

$$\frac{\partial E(n)}{\partial w_{ijml}^{(2)}(n)}=y_{ij}^{(1)}(n)g_{ml}^{(2)}(n),$$

$$\frac{\partial E(n)}{\partial w_{ijml}^{(1)}(n)}=y_{ij}^{(0)}(n)g_{ml}^{(1)}(n),$$

$$g_{1i}^{(3)}(n)=f'^{(3)}(s_{1i}^{(3)}(n))(y_{1i}^{(3)}(n)-d_{\mu i})$$

$$g_{ml}^{(2)}(n)=f'^{(2)}(s_{ml}^{(2)}(n))\sum_{i=1}^{N^{(3)}} w_{ml1i}^{(3)}(n)g_{1i}^{(3)}(n)$$

$$g_{ml}^{(1)}(n)=f'^{(1)}(s_{ml}^{(1)}(n))\sum_{i=1}^{U^{(2)}}\sum_{j=\max\{1,l-M^{(1)}+1\}}^{\min\{N^{(2)},l\}} w_{mlij}^{(2)}(n)g_{ij}^{(2)}(n)$$

6. Проверка условия завершения

Если $n \bmod P>0$, то $\mu=\mu+1$, $n=n+1$, переход к 3.

Если $n \bmod P=0$ и $\frac{1}{P}\sum_{s=1}^{P}E(n-P+s)>\varepsilon$, то $n=n+1$, переход к 2

Если $n \bmod P = 0$ и $\frac{1}{P}\sum_{s=1}^{P} E(n-P+s) < \varepsilon$, то завершиться.

Функционирование ИНС

$\mathbf{y}^{(0)} = \mathbf{x}$,

$$y_{ml}^{(1)} = f^{(1)}(b_{iml}^{(1)} + \sum_{i=l}^{l+M^{(0)}} \sum_{j=1}^{N^{(0)}} w_{ijml}^{(1)} y_{ij}^{(0)}),\ m \in \overline{1, U^{(1)}}, l \in \overline{1, N^{(1)}}.$$

$$y_{ml}^{(2)} = f^{(2)}(b_{jml}^{(1)} + \sum_{j=l}^{l+M^{(1)}} \sum_{i=1}^{U^{(1)}} w_{ijml}^{(2)} y_{ij}^{(1)}),\ m \in \overline{1, U^{(2)}}, l \in \overline{1, N^{(2)}}.$$

$$y_{1i}^{(3)} = f^{(3)}(b_{i1i}^{(1)} + \sum_{j=1}^{N^{(2)}} w_{ij1i}^{(3)} y_{ij}^{(2)}),\ i \in \overline{1, N^{(3)}}.$$

$$i^* = \arg\max_i y_{1i}^{(3)},\ i \in \overline{1, N^{(3)}}.$$

Достоинства

1. Используется для классификации образцов.

2. Обеспечивает хорошее качество обобщения (т.е. хорошо работает с тестовыми данными, которые в процессе обучения не предъявлялись).

3. Количество классов может быть больше двух (в выходном слое может быть больше одного нейрона).

4. Дает более высокую вероятность классификации, чем MLP.

Недостатки

1. Обучение происходит медленнее, чем в случае MLP, RBFNN, PNN, сети Хемминга, SOM, CPNN.

2. Отсутствует автоматическое определение числа скрытых слоев и числа нейронов в этих слоях, величины временной задержки.

3. В отличие от методов обучения SVM, градиентные методы обучения TDNN выбирают положение разделяющей гиперплоскости произвольным образом.

4. Обучение не сводится к задаче квадратичного программирования в выпуклой области, имеющей единственное решение.

5. В отличие от ART, не решает проблему пластичности-стабильности (способность к восприятию новых образцов при сохранении старых образцов).

5.4. Сверточная нейросеть

На рис.5.4. приведена сверточная нейросеть (CNN) [43-45], которая является нерекуррентной динамической ИНС и имеет иерархическую структуру. CNN является особым классом многослойного персептрона. Она образована входным слоем, который состоит из одной рецепторной плоскости, и чередующимися сверточными слоями (соответствуют S-слоям неокогнитрона) и субдискретизирующими (субвыборочными, прореживающими) слоями (соответствуют C-слоям неокогнитрона). Сверточный слой состоит из сверточных плоскостей, Субдискретизирующий слой состоит из субдискретизирующих плоскостей. Каждая сверточная плоскость состоит из сверточных клеток, каждая субдискретизирующая плоскость состоит из субдискретизирующих клеток. Субдискретизирующий слой уменьшает размерность изображения. Сверточный слой уменьшает чувствительность к сдвигу элементы изображения. Каждая плоскость выходного слоя содержит только одну клетку и соответствует определенному классу. С клеткой плоскости клеток текущего слоя связана область связи плоскости клеток предыдущего слоя (на рис.5.5 область связи представлена квадратом, а ее клетки выделены черным цветом). Геометрически область связи обычно представляет собой квадрат. Для всех плоскостей одного слоя она имеет один и тот же размер. Все клетки одной плоскости клеток текущего слоя, связанные с областями связи плоскости клеток предыдущего слоя имеют одинаковые веса (рис.5.6). Области связи плоскости клеток субдискретизирующего слоя перекрываются (рис.5.7). Из-за этого одна клетка плоскости клеток субдискретизирующего слоя (на рис.5.8 эта клетка выделена черным цветом), входящая в разные перекрывающиеся области связи, может активизировать несколько клеток плоскости клеток сверточного слоя (на рис.5.8 активизируемые клетки выделены черным цветом).

Для CNN используется обучение на основе коррекции ошибок (обучение с учителем), при этом чаще всего применяется алгоритм обратного распространения (BP). Это итеративный градиентный алгоритм обучения, который обеспечивает минимизацию среднеквадратичной ошибки.

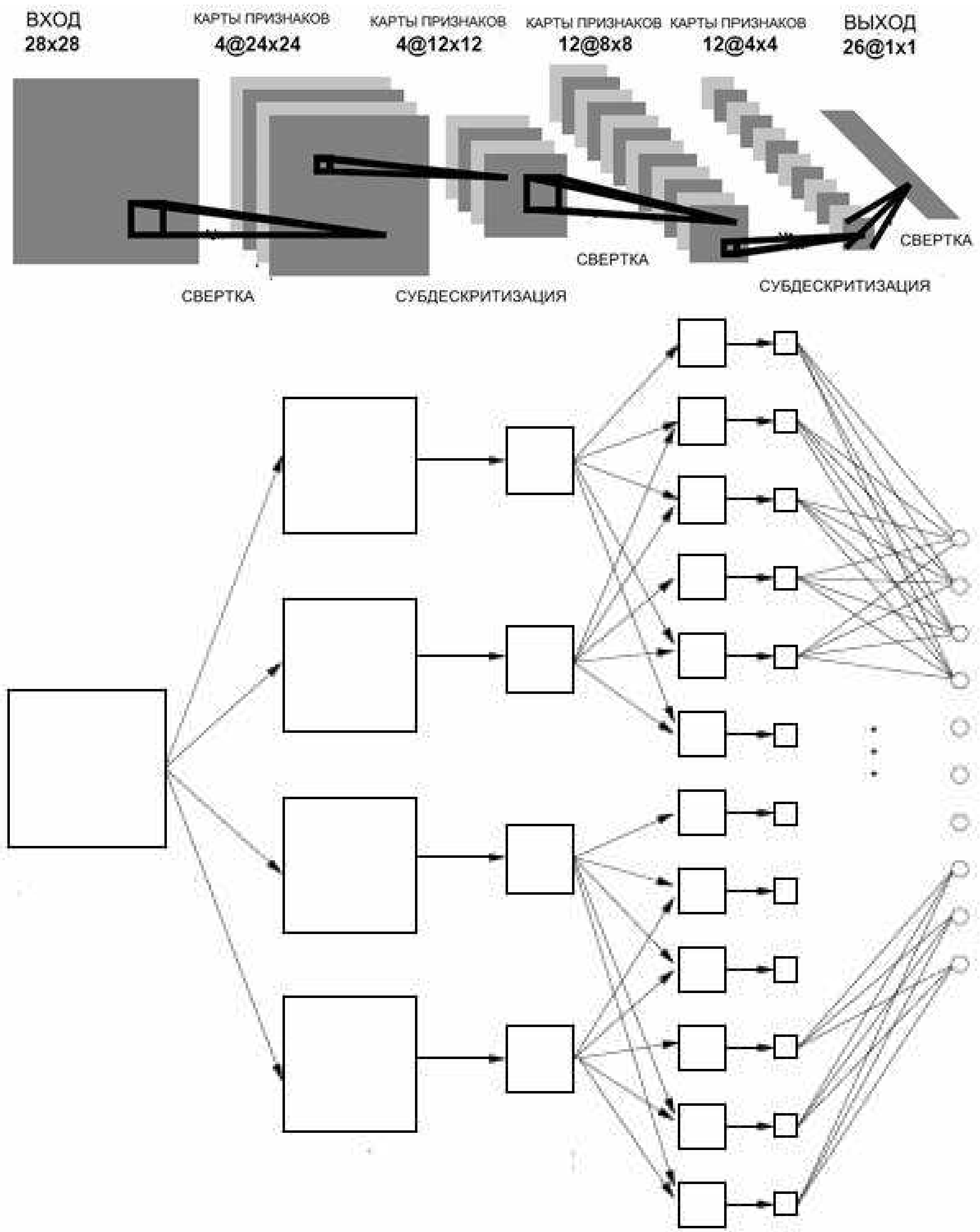


Рис.5.4. Сверточная нейронная сеть (CNN)

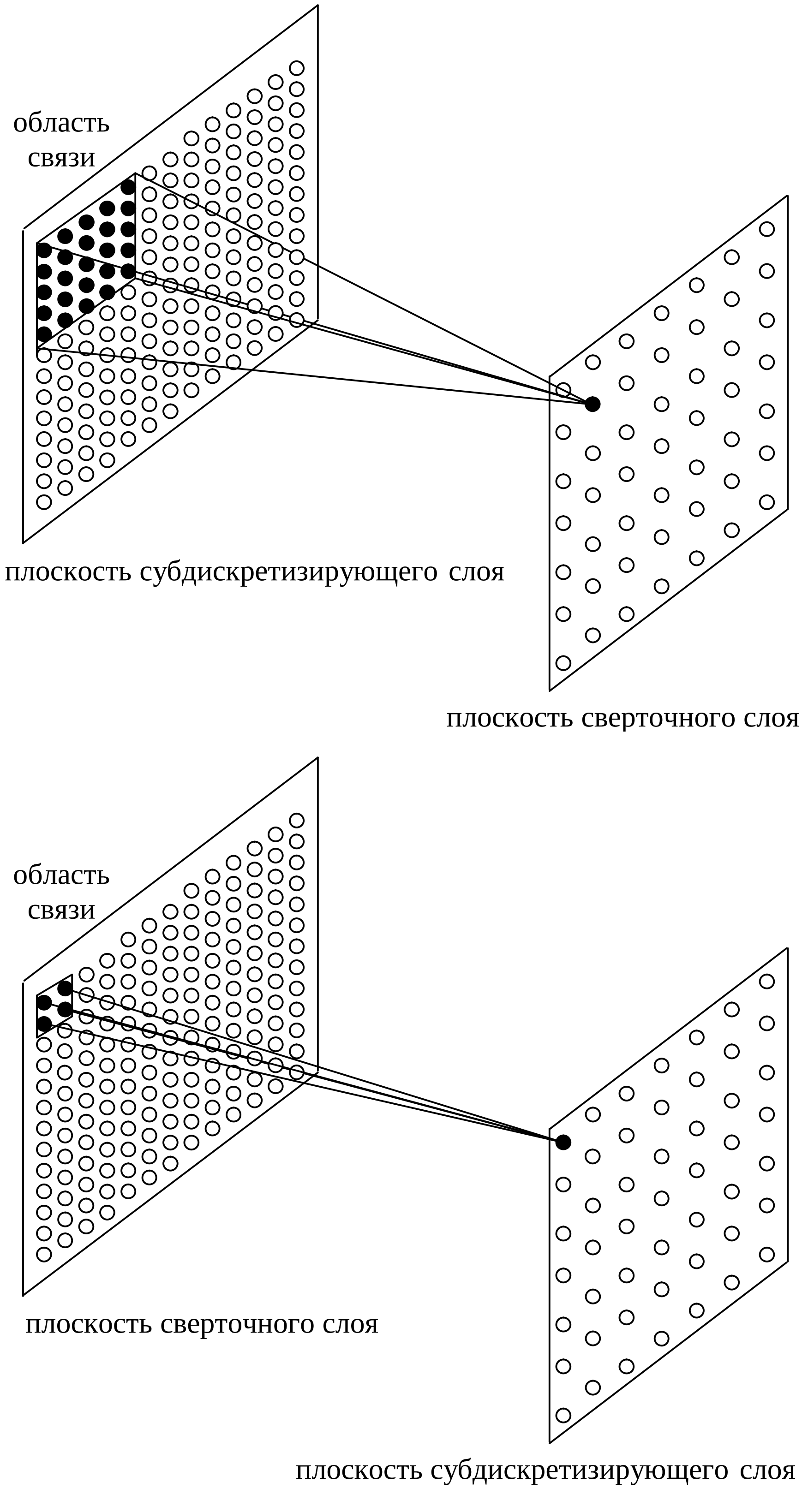


Рис. 5.5. Область связи

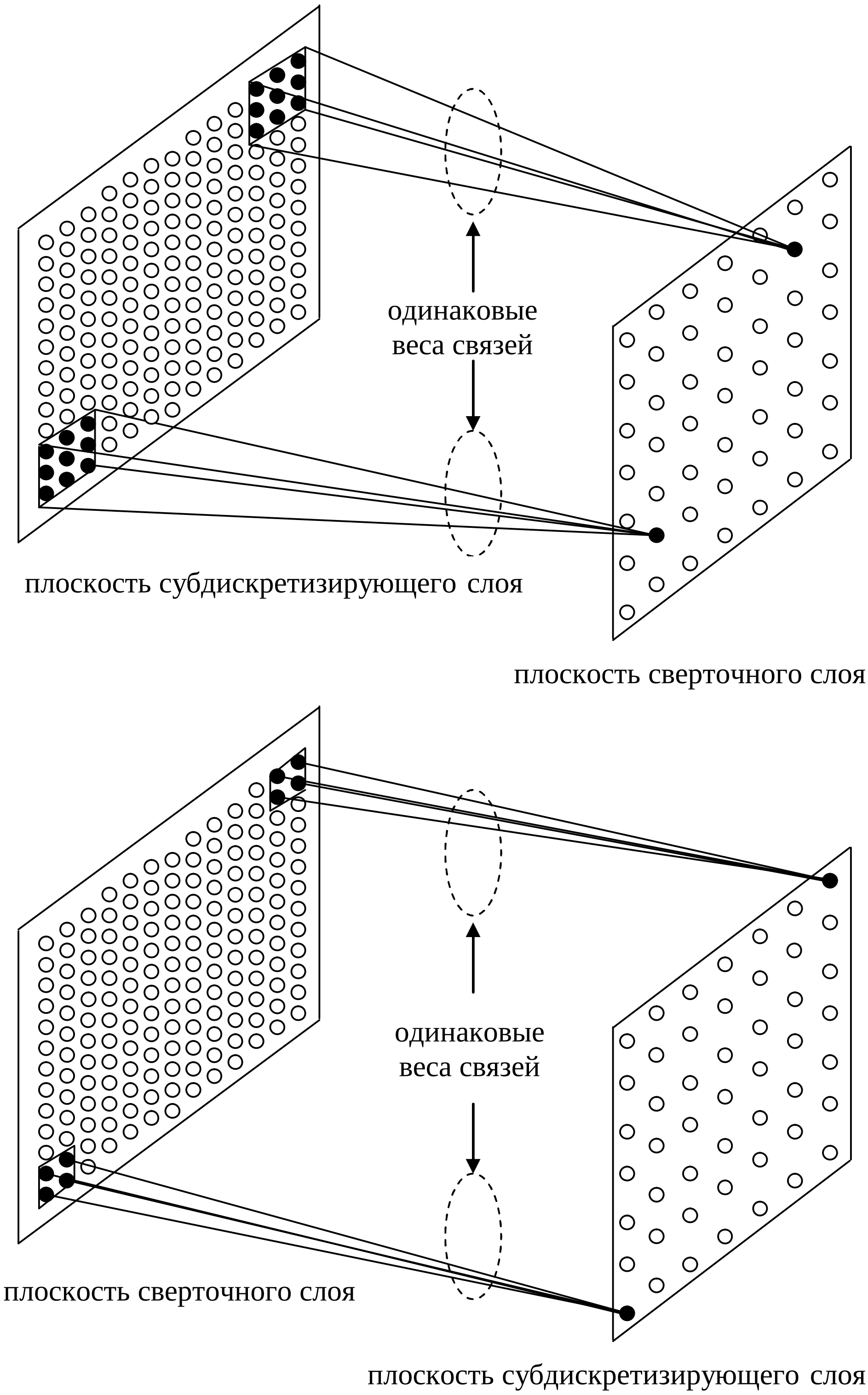


Рис. 5.6. Совпадение весов клеток плоскости клеток текущего слоя, связанных с областями связи плоскости клеток предыдущего слоя

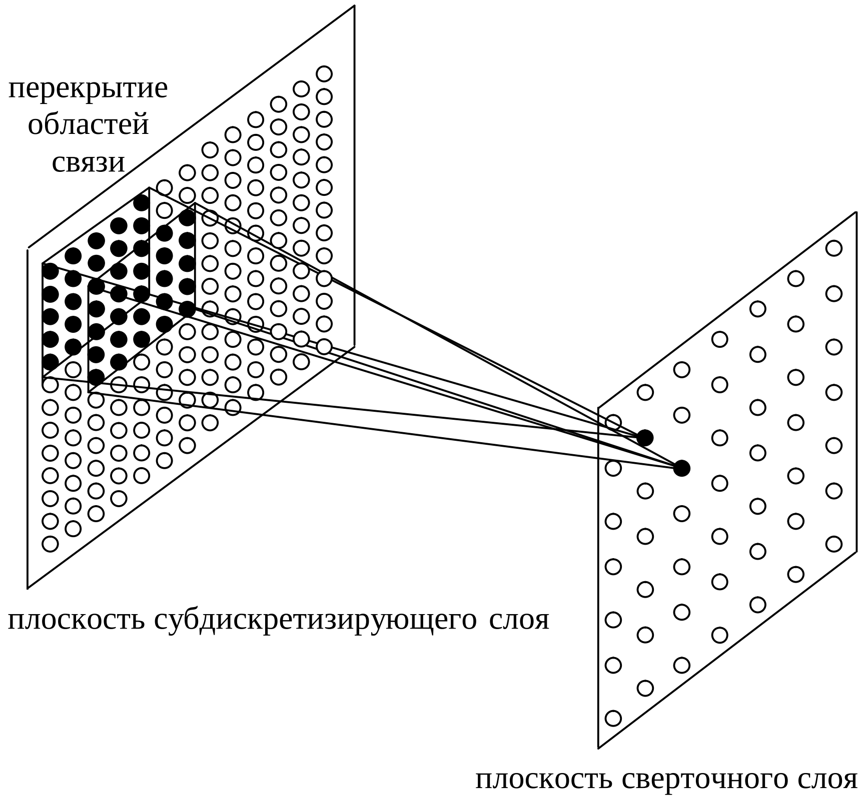


Рис. 5.7. Перекрытие областей связи

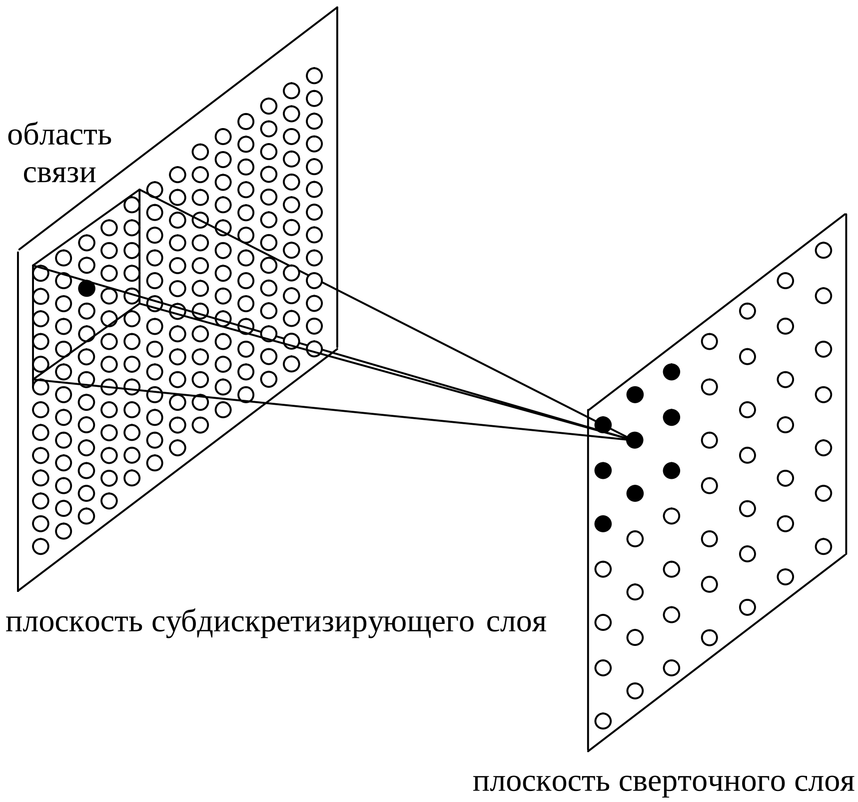


Рис. 5.8. Активизируемые клетки
(активизирующая клетка входит в разные области связи)

Обучение ИНС (алгоритм обратного распространения)

1. Номер итерации обучения $n=1$. Инициализация посредством равномерного распределения на интервале (0,1) или [-0.5, 0.5] весов $b_{s_l}(n,k)$, $w_{s_l}(n,i,k)$, $i \in \overline{1,K_{c_l}}$, $k \in \overline{1,K_{s_l}}$, $b_{c_l}(n,i)$, $w_{c_l}(n,\nu,k,i)$, $\nu \in D_{l-1}$, $k \in \overline{1,K_{s_{l-1}}}$, $i \in \overline{1,K_{c_l}}$, $w_o(n,\nu,k,i)$, $\nu \in D_L$, $k \in \overline{1,K_{s_L}}$, $i \in \overline{1,K_o}$, $l \in \overline{1,L}$, где i – номер плоскости клеток сверточного слоя C_l или выходного слоя O, k – номер плоскости клеток субдискретизирующего слоя S_l, ν – позиция в области связи, $\nu=(\nu_x,\nu_y)$, K_{s_l} – количество плоскостей клеток в слое S_l, K_{c_l} – количество плоскостей клеток в слое C_l, K_o – количество плоскостей клеток в выходном слое O, D_l – область связи плоскости слоя S_l, L – количество сверточных (или субдискретизирующих) слоев.

2. Задается обучающее множество $\{(\mathbf{x}_\mu,\mathbf{d}_\mu) \mid \mathbf{x}_\mu \in \{0,1\}^{N^{(0)}\times N^{(0)}}, \mathbf{d}_\mu \in \{0,1\}^{N^{(L)}}\}$, $\mu \in \overline{1,P}$, где $\mathbf{x}_\mu$ – μ-я обучающая входная матрица, $\mathbf{d}_\mu$ – μ-й обучающий выходной вектор, P – мощность обучающего множества. Номер текущей пары из обучающего множества $\mu=1$.

3. Вычисление выходного сигнала сверточной клетки для первой ступени

$$u_{c_1}(n,m,i) = f_{c_1}(h_{c_1}(n,m,i)),\ m \in \{1,...,N_{c_1}\}^2,\ i \in \overline{1,K_{c_1}},$$

$$h_{c_1}(n,m,i) = b_{c_1}(n,i) + \sum_{\nu \in D_1} w_{c_1}(n,\nu,1,i)x_\mu(m+\nu),$$

где m – позиция клетки в k-й плоскости клеток сверточного слоя C_l.

4. $l=1$.

5. Вычисление выходного сигнала субдискретизирующей клетки (уменьшение масштаба в два раза)

$$u_{s_l}(n,m,k) = f_{s_l}(h_{s_l}(n,m,k)),\ m \in \{1,...,N_{s_l}\}^2,\ k \in \overline{1,K_{s_l}},$$

$$h_{s_l}(n,m,k) = b_{s_l}(n,k) + w_{s_l}(n,k,k)\frac{1}{4}\sum_{\upsilon \in \{0,1\}^2} u_{c_l}(n,2m+\upsilon,k)$$

6. Вычисление выходного сигнала сверточной клетки

$$u_{c_l}(n,m,i) = f_{c_l}(h_{c_l}(n,m,i)),\ m \in \{1,...,N_{c_l}\}^2,\ i \in \overline{1,K_{c_l}},$$

$$h_{c_l}(n,m,i) = b_{c_l}(n,i) + \sum_{k=1}^{K_{s_l}} \sum_{\nu \in D_l} w_{c_l}(n,\nu,k,i) u_{s_l}(n,m+\nu,k)$$

7. Если $l \le L$, то $l = l+1$, переход к 5.

8. Вычисление выходного сигнала выходного слоя

$u_o(n,i) = f_o(h_o(n,i))$, $i \in \overline{1,K_o}$,

$$h_o(n,i) = b_o(n,i) + \sum_{k=1}^{K_{s_L}} \sum_{\nu \in D_L} w_o(n,\nu,k,i) u_{s_L}(n,(1,1)+\nu,k)$$

9. Вычисление энергии ошибки ИНС

$$E(n) = \frac{1}{2}\sum_{i=1}^{K_o} e_i^2(n),\ e_i(n) = u_o(n,i) - d_{\mu i},$$

10. Настройка синаптических весов на основе обобщенного дельта правила (обратный ход)

$$b_o(n+1,i) = b_o(n,i) - \eta \frac{\partial E(n)}{\partial b_o(n,i)},$$

$$w_o(n+1,\nu,k,i) = w_o(n,\nu,k,i) - \eta \frac{\partial E(n)}{\partial w_o(n,\nu,k,i)},$$

$$b_{c_l}(n+1,i) = b_{c_l}(n,i) - \eta \frac{\partial E(n)}{\partial b_{c_l}(n,i)},$$

$$w_{c_l}(n+1,\nu,k,i) = w_{c_l}(n,\nu,k,i) - \eta \frac{\partial E(n)}{\partial w_{c_l}(n,\nu,k,i)},$$

$$b_{s_l}(n+1,k) = b_{s_l}(n,k) - \eta \frac{\partial E(n)}{\partial b_{s_l}(n,k)},$$

$$w_{s_l}(n+1,i,k) = w_{s_l}(n,i,k) - \eta \frac{\partial E(n)}{\partial w_{s_l}(n,i,k)},$$

$$\frac{\partial E(n)}{\partial b_o(n,i)} = g_o(n,i),$$

$$\frac{\partial E(n)}{\partial w_o(n,\nu,k,i)} = u_{s_L}(n,(1,1)+\nu,k) g_o(n,i),$$

$$\frac{\partial E(n)}{\partial b_{c_l}(n,i)} = \sum_m g_{c_l}(n,m,i),\ m \in \{1,\ldots,N_{s_{l-1}}\}^2,$$

$$\frac{\partial E(n)}{\partial w_{c_l}(n,\nu,k,i)} = \sum_m u_{s_{l-1}}(n, m+\nu, k) g_{c_l}(n,m,i),\ m \in \{1,...,N_{c_l}\}^2,$$

$$\frac{\partial E(n)}{\partial b_{s_l}(n,k)} = \sum_m g_{s_l}(n,m,k),\ m \in \{1,...,N_{c_l}\}^2,$$

$$\frac{\partial E(n)}{\partial w_{s_l}(n,i,k)} = \begin{cases} \frac{1}{4}\sum_m \sum_{\upsilon\in\{0,1\}^2} u_{c_l}(n, 2m+\upsilon, i) g_{s_l}(n,m,k), & i = k \\ 0, & i \neq k \end{cases},$$

$m \in \{1,...,N_{s_l}\}^2$,

$$g_o(n,i) = e_i(n) f_o'(h_o(n,i)),$$

$$g_{c_l}(n,m,i) = f_{c_l}'(h_{c_l}(n,m,i)) \sum_{k=1}^{K_{s_l}} w_{s_l}(n,i,k) g_{s_l}(n,[m/2],k),$$

$$g_{s_l}(n,m,k) = \begin{cases} f_{s_L}'(h_{s_L}(n,m,k)) \sum_{i=1}^{K_o} \sum_{\nu\in D_L} w_o(n,\nu,k,i) g_o(n,i), & l = L \\ f_{s_l}'(h_{s_l}(n,m,k)) \sum_{i=1}^{K_{c_{l+1}}} \sum_{\nu\in D_l} w_{c_{l+1}}(n,\nu,k,i) g_{c_{l+1}}(n,m,i), & l < L \end{cases},$$

11. Проверка условия завершения

Если $n \bmod P > 0$, то $\mu = \mu + 1$, $n = n + 1$, переход к 3.

Если $n \bmod P = 0$ и $\frac{1}{P}\sum_{s=1}^{P} E(n - P + s) > \varepsilon$, то $n = n + 1$, переход к 2.

Если $n \bmod P = 0$ и $\frac{1}{P}\sum_{s=1}^{P} E(n - P + s) < \varepsilon$, то завершиться.

Функционирование ИНС

1. $u_{c_1}(m,i) = f_{c_1}(h_{c_1}(m,i))$, $m \in \{1,...,N_{c_1}\}^2$, $i \in \overline{1, K_{c_1}}$,

$$h_{c_1}(m,i) = b_{c_1}(i) + \sum_{\nu\in D_1} w_{c_1}(\nu,1,i) x(m+\nu)$$

2. $u_{s_l}(m,k) = f_{s_l}(h_{s_l}(m,k))$, $m \in \{1,...,N_{s_l}\}^2$, $k \in \overline{1, K_{s_l}}$,

$$h_{s_l}(m,k) = b_{s_l}(k) + w_{s_l}(k,k)\frac{1}{4}\sum_{\upsilon\in\{0,1\}^2} u_{c_l}(2m+\upsilon, k)$$

3. $u_{c_l}(m,i) = f_{c_l}(h_{c_l}(m,i))$, $m \in \{1,...,N_{c_l}\}^2$, $i \in \overline{1, K_{c_l}}$,

$$h_{c_l}(m,i) = b_{c_l}(i) + \sum_{k=1}^{K_{s_l}} \sum_{\nu \in D_l} w_{c_l}(\neg, k, i) u_{s_l}(m + \neg, k)$$

4. Если $l \le L$, то $l = l + 1$, переход к 2.

5. $u_o(i) = f_o(h_o(i))$, $i \in \overline{1, K_o}$,

$$h_o(i) = b_o(i) + \sum_{k=1}^{K_{s_L}} \sum_{\nu \in D_L} w_o(\nu, k, i) u_{s_L}((1,1) + \nu, k)$$

Достоинства

1. Используется для классификации образцов.

2. В отличие от других ИНС (кроме неокогнитрона), адаптирован к различным искажениям образцов (сдвиг, масштабирование, поворот, неполны образ, наличие шума), т.е. обеспечивает хорошее качество обобщения.

3. Количество классов может быть больше двух (в выходном слое может быть больше одной клетки).

4. Ориентирован на моделирование зрительной системы человека.

5. В отличие от других ИНС (кроме когнитрона и неокогнитрона) представляет изображение матрицей.

6. В отличие от когнитрона и неокогнитрона, веса определяются с помощью алгоритма обратного распространения.

Недостатки

1. Большое число клеток в ступенях приводит к более медленному функционированию CNN по сравнению с другими ИНС.

2. Отсутствует автоматическое определение числа ступеней, количества и размеров плоскостей в каждом слое.

3. В отличие от ART, не решает проблему пластичности-стабильности.

4. Большое число клеток в ступенях приводит к более медленному обучению по сравнению с другими ИНС.

5.5. Когнитрон

На рис. 5.9 приведен когнитрон [46,47], который является нерекуррентной динамической ИНС и имеет иерархическую структуру. Все слои, кроме входного слоя, состоят из одинакового количества клеток. Количество слоев не менее 3. Каждый слой имеет одинаковое количество возбуждающих (нейронов Фукушимы) и тормозящих клеток (от 8x8 до 128x128). Результатом распознавания является номер клетки, победившей во всем последнем слое.

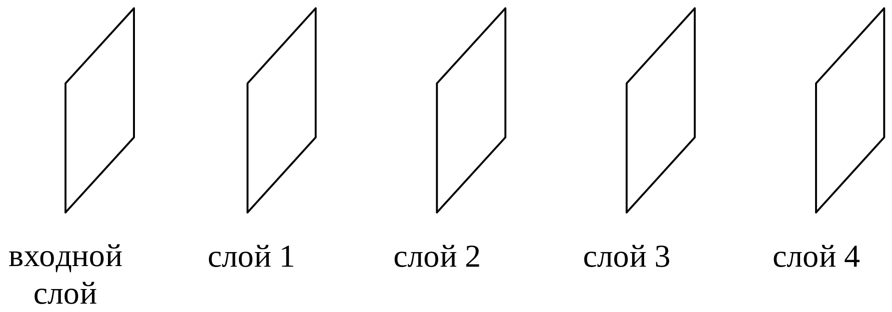


Рис. 5.9. Когнитрон

В каждом текущем слое имеются пересекающиеся области конкуренции (в каждой из них выбирается только одна клетка-победитель) (рис.5.10). Область конкуренции представляет собой множество клеток этого слоя (например, в виде квадрата, равностороннего ромба, круга).

Область связи клетки текущего слоя представляет собой множество клеток предыдущего слоя (например, в виде квадрата, равностороннего ромба, круга). На рис.5.11 область связи представлена квадратом, а ее клетки выделены черным цветом. Области связи клеток всех слоев имеют один и тот же размер. Области связи клеток текущего слоя перекрываются (рис.5.12). Из-за этого одна клетка предыдущего слоя (на рис.5.13 эта клетка выделена черным цветом), входящая в разные перекрывающиеся области связи, может активизировать несколько клеток текущего слоя (на рис.5.13 активизируемые клетки выделены черным цветом).

Для когнитрона используют конкурентное обучение (обучение без учителя).

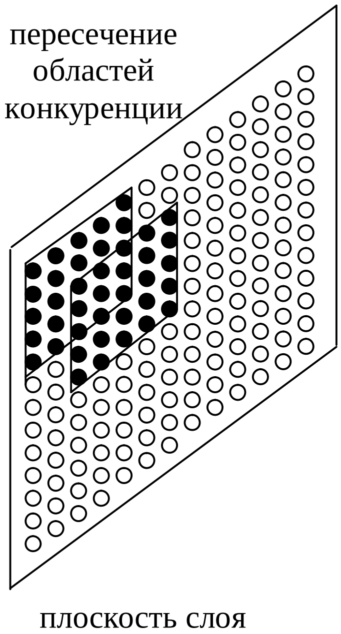


Рис. 5.10. Пересечение областей конкуренции

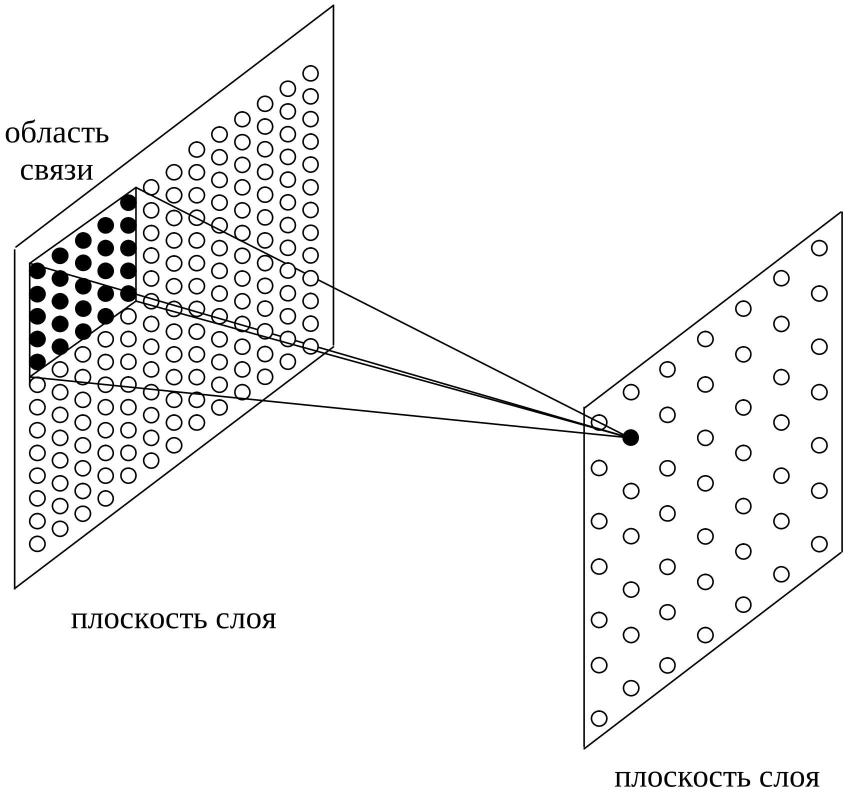


Рис. 5.11. Область связи

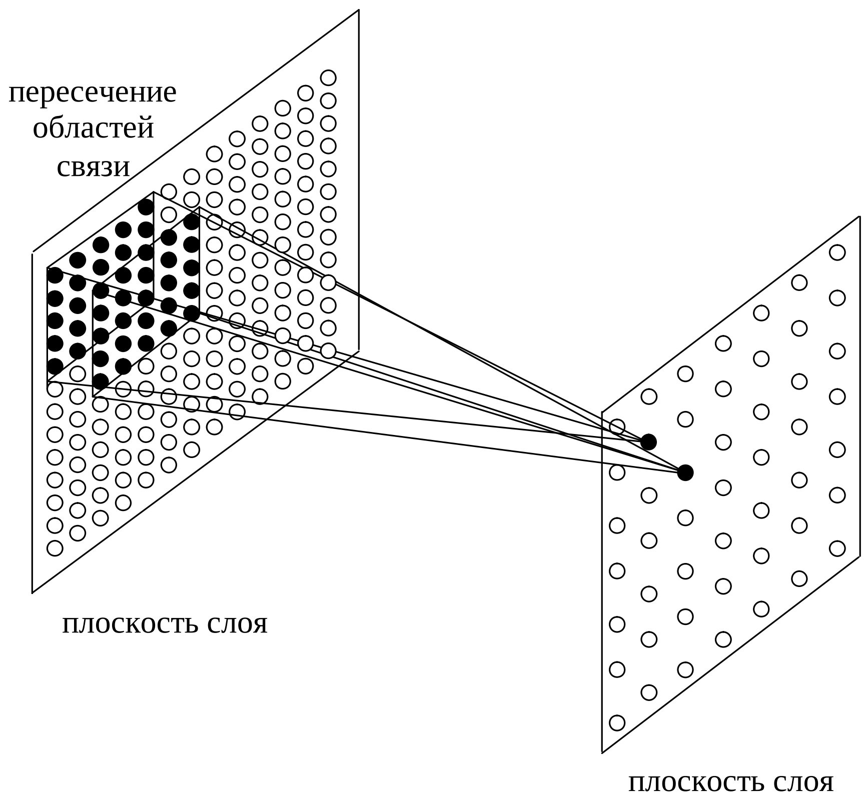


Рис. 5.12. Пересечение областей связи

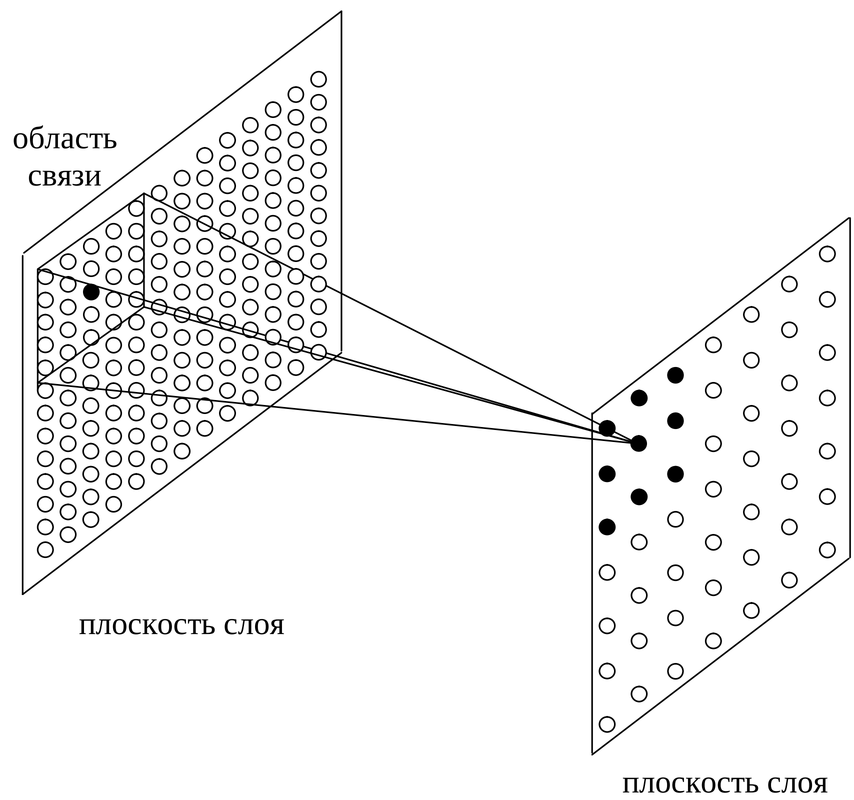


Рис. 5.13. Активизируемые клетки
(активизирующая клетка входит в разные области связи)

Обучение ИНС

1. Заданы веса $c_l(\nu)$, $\nu \in A_l$, где $c_l(\nu) \ge 0$, $\sum_{\nu \in A_l} c_l(\nu) = 1$. Заданы веса $d_l(\nu)$, $\nu \in D_l$, где $d_l(\nu) \ge 0$, $\sum_{\nu \in D_l} d_l(\nu) = 1$. Инициализация посредством равномерного распределения на интервале (0,1) или [-0.5, 0.5] весов $a_l(\nu, n)$, $\nu \in A_l$, $b_l(\nu)$, $\nu \in B_l$, $l \in \overline{1, L}$, где ν – позиция в области связи, $\nu = (\nu_x, \nu_y)$, A_l – область связи возбуждающих клеток l-го слоя, состоящая из клеток l-1-го слоя, L – количество слоев, D_l – область латерального торможения (например, в виде квадрата, равностороннего ромба, круга), состоящая из клеток l-го слоя. Задается обучающее множество $\{\mathbf{x}_\mu \mid \mathbf{x}_\mu \in \{0,1\}^{N^{(0)} \times N^{(0)}}\}$, $\mu \in \overline{1, P}$, где $\mathbf{x}_\mu$ – μ-я обучающая входная матрица, P – мощность обучающего множества.

2. $l = 1$.

3. Номер матрицы из обучающего множества $\mu = 1$.

4. Если $l = 1$, то $u_0(m) = x_\mu(m)$, $m = (m_x, m_y)$, $m_x, m_y \in \overline{1, N^{(0)}}$.

5. Вычисление выходного сигнала возбуждающей клетки

$$\hat{u}_l(n) = \varphi\left(\frac{1 + \sum_{\nu \in A_l} a_l(\nu, n) u_{l-1}(\nu + n)}{1 + b_l(n) v_{l-1}(n)} - 1 \right),$$

$$u_l(n) = \varphi\left(\frac{1 + \hat{u}_l(n)}{1 + \sum_{\nu \in D_l} d_l(\nu) \hat{u}_l(n + \nu)} - 1 \right),$$

$$\varphi(x) = \begin{cases} x, & x \ge 0 \\ 0, & x < 0 \end{cases},$$

где n – позиция клетки в l-м слое, $n = (n_x, n_y)$, $n_x, n_y \in \overline{1, N^{(l)}}$.

6. Вычисление выходного сигнала тормозящей клетки

$$v_l(n) = \sum_{\nu \in A_l} c_l(\nu) u_l(n + \nu),\ n \in \overline{1, N^{(l)} \times N^{(l)}}.$$

7. Настройка синаптических весов на основе правила когнитрона

$$a_l(\neg, n) = a_l(\neg, n) + \Delta a_l(\neg, n),\ \nu \in A_l,\ n \in \overline{1, N^{(l)} \times N^{(l)}},$$

$$b_l(n) = b_l(n) + \Delta b_l(n),\ n \in \overline{1, N^{(l)} \times N^{(l)}},$$

$$\Delta a_l(\nu, n) = \begin{cases} \eta_1 c_{l-1}(\nu) u_{l-1}(n+\nu)\delta_l(n), & u_l(n) > 0 \\ \eta_2 c_{l-1}(\nu) u_{l-1}(n+\nu)\delta_l(n), & u_l(n) = 0 \end{cases},\ \nu \in A_l,$$

$$\Delta b_l(n) = \begin{cases} \eta_1 \dfrac{\sum\limits_{\nu \in A_l} c_{l-1}(\nu) u_{l-1}^2(n+\nu)}{2 v_{l-1}(\nu)} \delta_l(n), & u_l(n) > 0, \\ \eta_2 v_{l-1}(\nu)\delta_l(n), & u_l(n) = 0 \end{cases}$$

$$\delta_l(n) = \begin{cases} 1, & \forall \nu \in E_l \ \ u_l(n) \ge u_l(n+\nu) \\ 0, & \textit{иначе} \end{cases},$$

где η_1, η_2 – параметры, определяющие скорость обучения (при больших η_1, η_2 обучение происходит быстрее, но увеличивается опасность получить неверное решение), $0 < \eta_2 < \eta_1$,

E_l – область конкуренции клеток l-го слоя, состоящая из нейронов l-го слоя.

8. Проверка условия завершения

Если $\mu > P$, то $l = l + 1$, иначе $\mu = \mu + 1$, переход к 5.

Если $l > L$, то останов, иначе переход к шагу 3.

Обычно в качестве $c_l(\nu)$ и $d_l(\nu)$ используется двухмерная функция Гаусса.

Функционирование ИНС

1. $u_0(m) = x(m),\ m = (m_x, m_y),\ m_x, m_y \in \overline{1, N^{(0)}}$.

2. $$\widehat{u}_l(n) = \varphi\left(\frac{1 + \sum\limits_{\nu \in A_l} a_l(\nu, n) u_{l-1}(\nu + n)}{1 + b_l(n) v_{l-1}(n)} - 1 \right),$$

$$u_l(n) = \varphi\left(\frac{1 + \widehat{u}_l(n)}{1 + \sum\limits_{\nu \in D_l} d_l(\nu) \widehat{u}_l(n+\nu)} - 1 \right),$$

$$\varphi(x) = \begin{cases} x, & x \ge 0 \\ 0, & x < 0 \end{cases},\ n = (n_x, n_y),\ n_x, n_y \in \overline{1, N^{(l)}}.$$

3. $v_l(n) = \sum\limits_{\nu \in A_l} c_l(\nu) u_l(n+\nu),\ n = (n_x, n_y),\ n_x, n_y \in \overline{1, N^{(l)}}$.

Достоинства

1. Используется для классификации образцов.

2. Количество классов может быть больше двух (в выходном слое может быть больше одной клетки).

3. Ориентирован на моделирование зрительной системы человека.

4. В отличие от других ИНС (кроме CNN и неокогнитрона) представляет изображение матрицей.

Недостатки

1. Большое число нейронов в слоях приводит к более медленному функционированию когнитрона по сравнению с другими ИНС.

2. Отсутствует автоматическое определение числа слоев.

3. В отличие от ART, не решает проблему пластичности-стабильности.

4. Большое число нейронов в слоях приводит к более медленному обучению по сравнению с другими ИНС.

5. В отличие от неокогнитрона, не адаптирован к различным искажениям образцов.

5.6. Неокогнитрон

На рис.5.14 приведен неокогнитрон [32,48], который является нерекуррентной динамической ИНС и имеет иерархическую структуру. Он образован входным слоем, который состоит из одной рецепторной плоскости (размер от 8x8 до 128x128), и ступенями (не менее 3), каждая из которых состоит S-слоя, C-слоя, V-слоя, которые состоят из S-плоскостей, C-плоскостей и одной V-плоскости соответственно. На рис.5.14 эти плоскости представлены прямоугольниками внутри слоев. Каждая плоскость состоит из S-клеток (простых клеток или нейронов Фукушимы), C-клеток (сложных клеток) и V-клеток (тормозящих клеток) соответственно. Количество клеток в плоскостях уменьшается от ступени к ступени. Каждая плоскость выходного слоя C_4 содержит только одну клетку и соответствует определенному классу. Увеличение количества плоскостей увеличивает вероятность распознавания, но увеличивает вычислительную сложность обучения и функционирования.

S-слой обеспечивает выделение элементов изображения. C-слой обеспечивает устойчивость (уменьшает чувствительность) к искажению элементов изображения, выделенного на S-слое. V-слой обеспечивает получение информации о средней активности C-слоя.

Каждой плоскости текущего S-слоя соответствует своя область конкуренции (в ней выбирается только одна клетка-победитель), которая представляет собой множество клеток этой плоскости (например, в виде квадрата, равностороннего ромба, круга). Области конкуренции плоскостей текущего S-слоя пересекаются.

Область связи клетки плоскости клеток текущего слоя представляет собой множество клеток плоскости предыдущего слоя (например, в виде квадрата, равностороннего ромба, круга). На рис. 5.15 область связи представлена квадратом, а ее клетки выделены черным цветом. Области связи клеток всех плоскостей текущего слоя имеют один и тот же размер. Если предыдущим слоем является V-слой, то область связи состоит только из одной клетки. Все клетки одной плоскости клеток текущего слоя, которые связаны с клетками предыдущего слоя, входящими в области связи этих клеток, имеют одинаковые веса. На рис.5.16 обе клетки имеют одинаковые веса.

Области связи клеток плоскости клеток текущего слоя пересекаются (рис.5.17). Из-за этого одна клетка предыдущего слоя (на рис.5.18 выделена черным цветом), входящая в разные перекрывающиеся области связи, может активизировать несколько клеток плоскости клеток текущего слоя (на рис.5.18 активизируемые клетки выделены черным цветом).

На рис.5.19 приведен пример ступенчатого распознавания. Первая ступень неокогнитрона распознает простые элементы изображения (например, отрезки под разным углом), а ее S-плоскости и C-плоскости реагируют на конкретный элемент изображения (например, на конкретное начертание отрезка). Следующие ступени распознают комбинации простых изображений. Последняя ступень распознает все изображение (например, букву или цифру).

Для неокогнитрона используют конкурентное обучение (обучение без учителя).

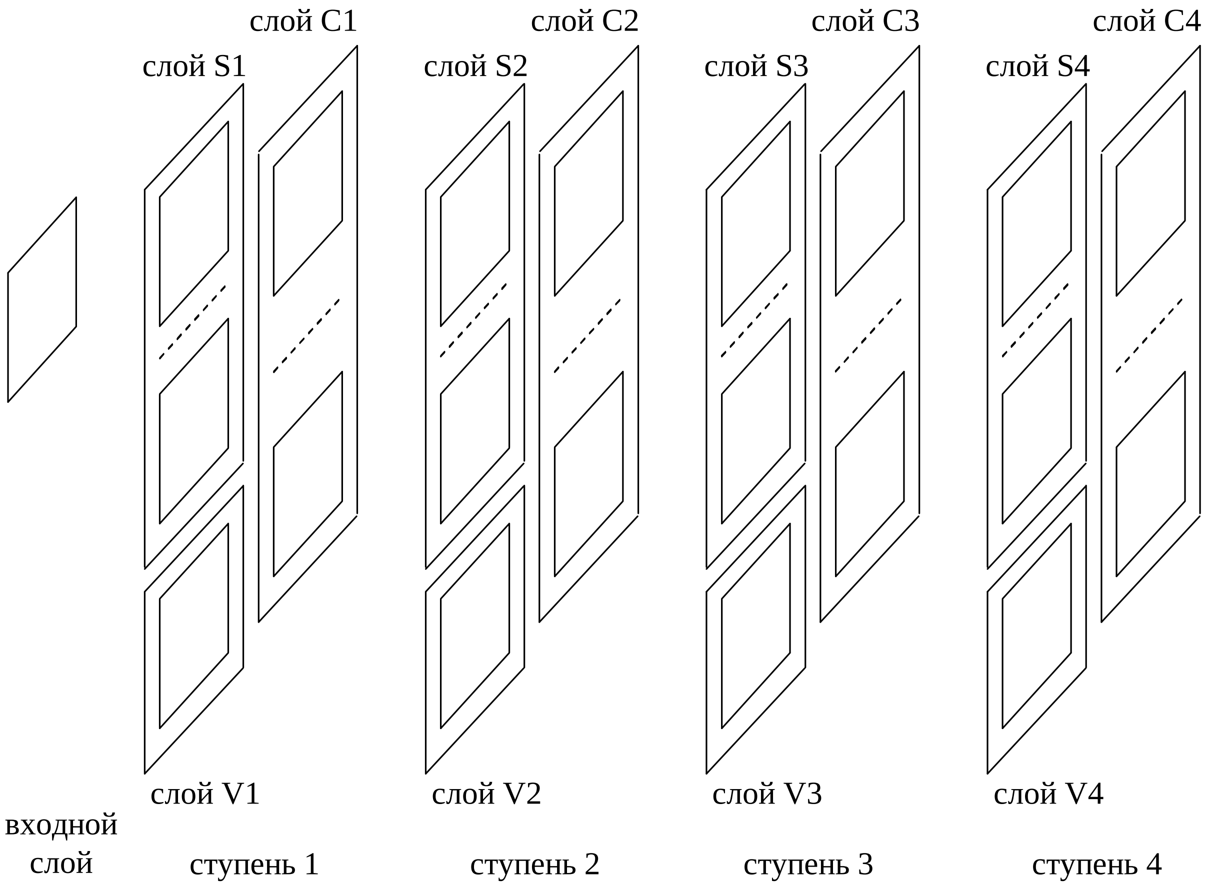


Рис.5.14. Неокогнитрон

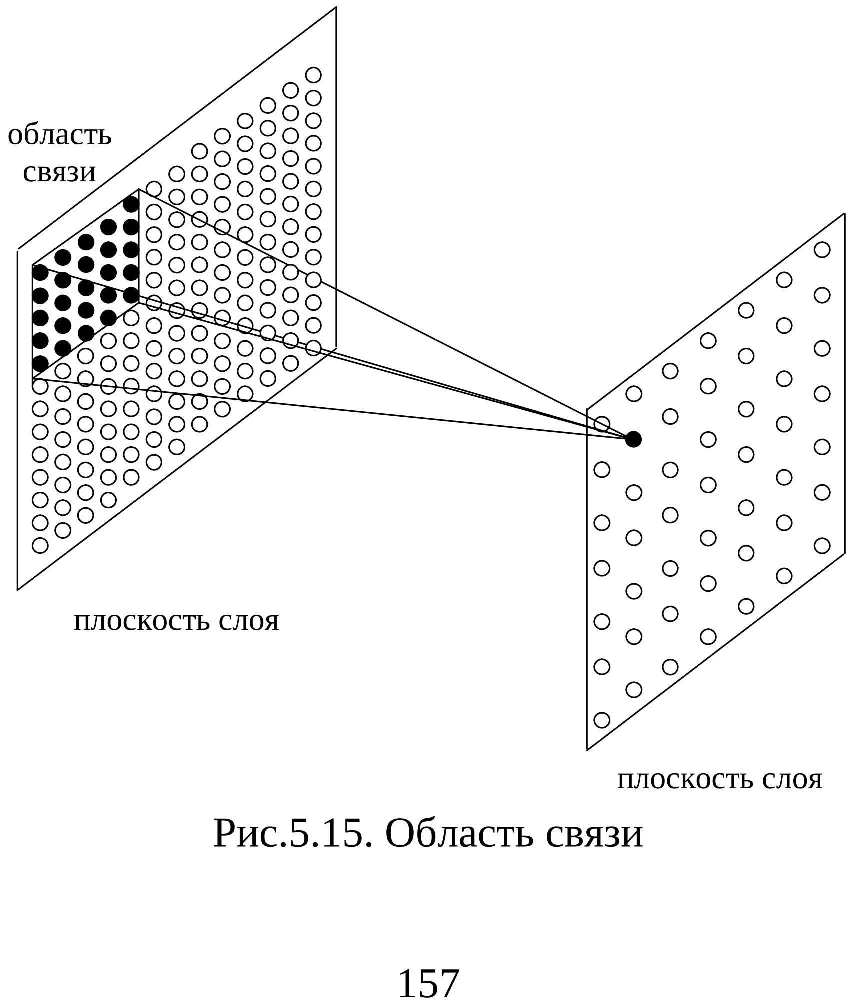


Рис.5.15. Область связи

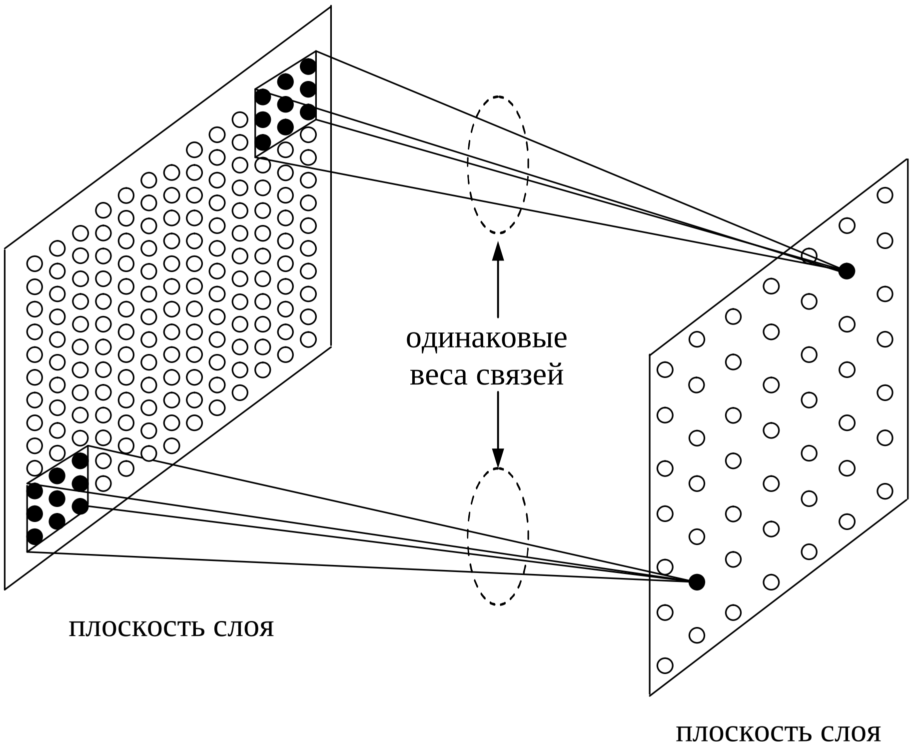


Рис.5.16. Совпадение весов клеток плоскости клеток текущего слоя, связанных с областями связи плоскости клеток предыдущего слоя

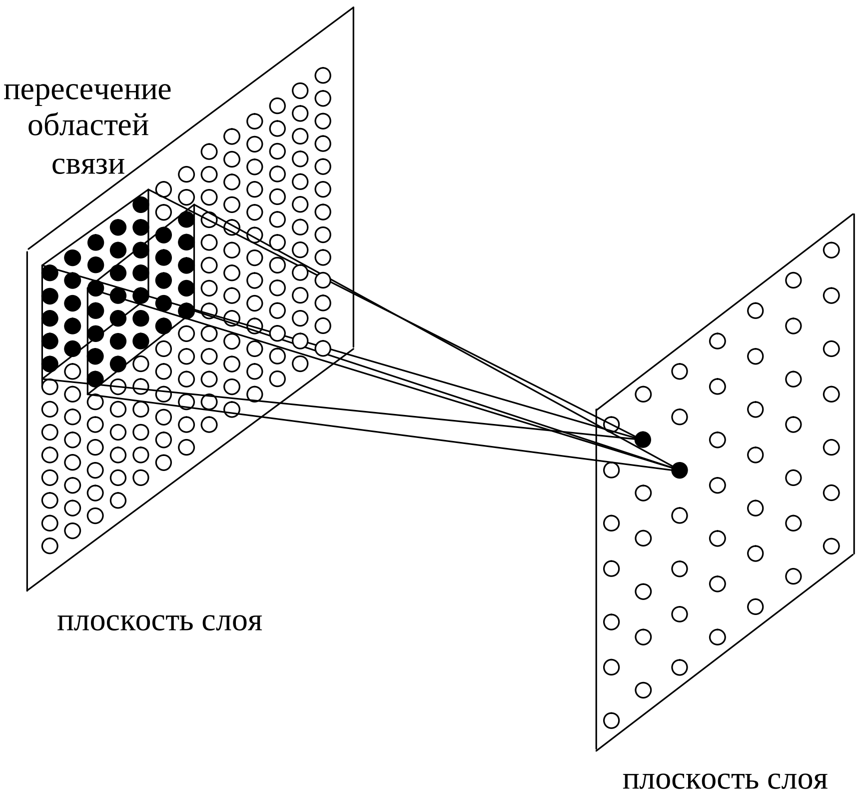


Рис.5.17. Пересечение областей связи

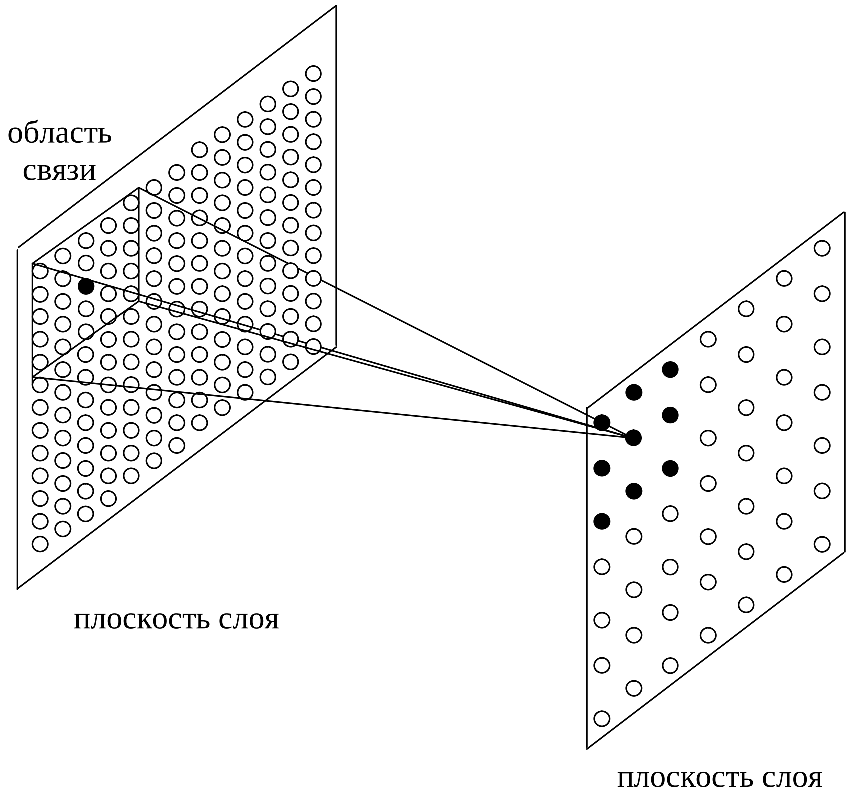


Рис.5.18. Активизируемые клетки
(активизирующая клетка входит в разные области связи)

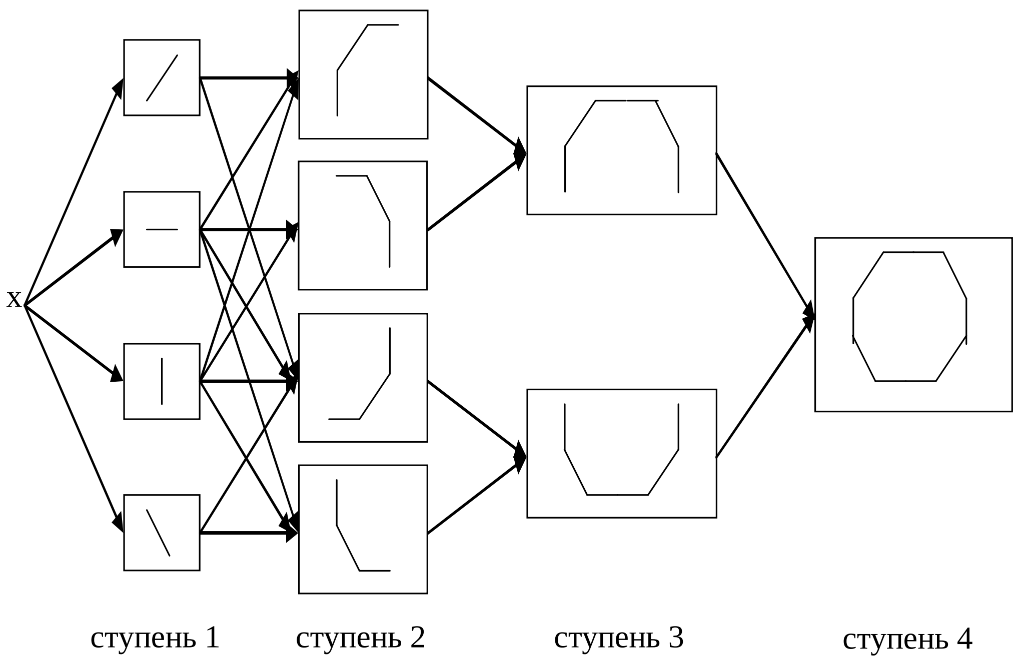


Рис. 5.19. Ступенчатое распознавание изображения

Обучение ИНС

1. Заданы веса $c_{v_l}(\nu)$, $v \in A_l$, где $c_{v_l}(\nu) \ge 0$, $\sum_{i=1}^{K_{c_{l-1}}} \sum_{v \in A_l} c_{v_l}(\nu) = 1$. Заданы веса $d_{c_l}(\nu)$, $v \in D_l$, где $d_{c_l}(\nu) \ge 0$, $\sum_{k=1}^{K_{s_l}} \sum_{v \in D_l} d_{c_l}(\nu) = 1$. Задан порог r_l для клеток слоя S_l, $r_l > 0$. Инициализация посредством равномерного распределения на интервале (0,1) или [-0.5, 0.5] весов $b_{s_l}(k)$, $a_{s_l}(\nu, i, k)$, $v \in A_l$, $i \in \overline{1, K_{c_{l-1}}}$, $k \in \overline{1, K_{s_l}}$, $l \in \overline{1, L}$, где i – номер плоскости клеток слоя C_l, k – номер плоскости клеток слоя S_l, ν – позиция в области связи, $\nu = (\nu_x, \nu_y)$, K_{s_l} – количество плоскостей клеток в слое S_l, K_{c_l} – количество плоскостей клеток в слое C_l, A_l – область связи клетки плоскости клеток слоя V_l или S_l, состоящая из клеток плоскости клеток слоя C_{l-1}, D_l – область связи клетки плоскости клеток слоя C_l, состоящая из клеток плоскости клеток слоя S_l, L – количество ступеней.

Задается функция связи двух плоскостей $j_l(k, i)$, $k \in \overline{1, K_{s_l}}$, $i \in \overline{1, K_{c_l}}$, причем, если k-я плоскость клеток слоя S_l связана с i-й плоскостью клеток слоя C_l, то $j(k, i) = 1$, иначе $j(k, i) = 0$.

Задается обучающее множество $\{\mathbf{x}_\mu \mid \mathbf{x}_\mu \in \{0,1\}^{N^{(0)} \times N^{(0)}}\}$, $\mu \in \overline{1, P}$, где $\mathbf{x}_\mu$ – μ-я обучающая входная матрица, P – мощность обучающего множества.

2. $l = 1$.

3. Номер матрицы из обучающего множества $\mu = 1$.

4. $u_{c_0}(m, 1) = x_\mu(m)$, $m = (m_x, m_y)$, $m_x, m_y \in \overline{1, N_{c_0}}$, $K_{c_0} = 1$, $q = 1$.

5. Вычисление выходного сигнала V-клетки

$$u_{v_l}(n) = \sqrt{\sum_{i=1}^{K_{c_{l-1}}} \sum_{v \in A_l} c_{v_l}(\nu) u_{c_{l-1}}^2(n + \nu, i)},$$

где n – позиция клетки в k-й плоскости клеток слоя S_l, $n = (n_x, n_y)$, $n_x, n_y \in \overline{1, N_{s_l}}$.

6. Вычисление выходного сигнала S-клетки

$$u_{s_l}(n,k)=r_l\varphi\left(\frac{1+\sum_{i=1}^{K_{c_{l-1}}}\sum_{\nu\in A_l}a_{s_l}(\nu,i,k)u_{c_{l-1}}(n+\nu,i)}{1+\frac{r_l}{r_l+1}b_{s_l}(k)u_{v_l}(n)}-1\right),$$

$\varphi(x)=\begin{cases}x, & x\geq 0\\ 0, & x<0\end{cases}$, $n=(n_x,n_y)$, $n_x,n_y\in\overline{1,N_{s_l}}$, $k\in\overline{1,K_{s_l}}$.

7. Вычисление выходного сигнала C-клетки

$$u_{c_l}(n,i)=\psi\left(\sum_{k=1}^{K_{s_l}}j_l(k,i)\sum_{\nu\in D_l}d_{c_l}(\nu)u_{s_l}(n+\nu,k)\right),$$

$\psi(x)=\dfrac{\varphi(x)}{1+\varphi(x)}$, $n=(n_x,n_y)$, $n_x,n_y\in\overline{1,N_{c_l}}$, $i\in\overline{1,K_{c_l}}$.

8. Если $q<l$, то $q=q+1$, переход к 5.

9. В каждой k-й плоскости слоя S_l в своей области конкуренции E_{lk} выбирается клетка-победитель n_k^*

$$n_k^*=\arg\max_{n\in E_{lk}}u_{s_l}(n,k),\ k\in\overline{1,K_{s_l}}.$$

10. Настройка синаптических весов на основе правила неокогнитрона

$$a_{s_l}(\nu,i,k)=a_{s_l}(\nu,i,k)+\eta_l c_{v_l}(\nu)u_{c_{l-1}}(n_k^*+\nu,i),$$

$\nu\in A_l$, $k\in\overline{1,K_{s_l}}$, $i\in\overline{1,K_{c_{l-1}}}$,

$$b_l(k)=b_l(k)+\eta_l u_{v_l}(n_k^*),\ k\in\overline{1,K_{s_l}},$$

где η_l – параметр, определяющий скорость обучения (при большом η_l обучение происходит быстрее, но увеличивается опасность получить неверное решение), $0<\eta_l$.

11. Проверка условия завершения

Если $\mu>P$, то $l=l+1$, иначе $\mu=\mu+1$, переход к 4.

Если $l>L$, то останов, иначе переход к шагу 3.

В качестве $c_l(\nu)$ и $d_l(\nu)$ может использоваться двухмерная функция Гаусса.

Функционирование ИНС

1. $u_{c_0}(m,1) = x(m)$, $m = (m_x, m_y)$, $m_x, m_y \in \overline{1, N_{c_0}}$.

2. $$u_{v_l}(n) = \sqrt{\sum_{i=1}^{K_{c_{l-1}}} \sum_{v \in A_l} c_{v_l}(\nu) u_{c_{l-1}}^2 (n+\nu, i)}, \qquad n = (n_x, n_y),$$

$n_x, n_y \in \overline{1, N_{s_l}}$.

3. $$u_{s_l}(n,k) = r_l \varphi\left(\frac{1 + \sum_{i=1}^{K_{c_{l-1}}} \sum_{v \in A_l} a_{s_l}(\nu, i, k) u_{c_{l-1}}(n+\nu, i)}{1 + \frac{r_l}{r_l + 1} b_{s_l}(k) u_{v_l}(n)} - 1 \right),$$

$\varphi(x) = \begin{cases} x, & x \ge 0 \\ 0, & x < 0 \end{cases}$, $n = (n_x, n_y)$, $n_x, n_y \in \overline{1, N_{s_l}}$, $k \in \overline{1, K_{s_l}}$.

4. $$u_{c_l}(n,i) = \psi\left(\sum_{k=1}^{K_{s_l}} j_l(k,i) \sum_{v \in D_l} d_{c_l}(\nu) u_{s_l}(n+\nu, k) \right),$$

$\psi(x) = \dfrac{\varphi(x)}{1 + \varphi(x)}$, $n = (n_x, n_y)$, $n_x, n_y \in \overline{1, N_{c_l}}$, $i \in \overline{1, K_{c_l}}$.

Достоинства

1. Используется для классификации образцов.

2. В отличие от других ИНС (в том числе и когнитрона), адаптирован к различным искажениям образцов (сдвиг, масштабирование, поворот, неполны образ, наличие шума), т.е. обеспечивает хорошее качество обобщения.

3. Количество классов может быть больше двух (в выходном слое может быть больше одной клетки).

4. Ориентирован на моделирование зрительной системы человека.

5. В отличие от других ИНС (кроме CNN и когнитрона) представляет изображение матрицей.

Недостатки

1. Большое число плоскостей в ступенях приводит к более медленному функционированию неокогнитрона по сравнению с другими ИНС.

2. Отсутствует автоматическое определение числа ступеней, количества и размеров плоскостей в каждом слое каждой ступени.

3. В отличие от ART, не решает проблему пластичности-стабильности.

4. Большое число плоскостей в ступенях приводит к более медленному обучению неокогнитрона по сравнению с другими ИНС.

5.7. Спайковая (импульсная) нейросеть

На рис. 5.20 приведена спайковая (импульсная) нейросеть [49,50], которая является нерекуррентной динамической двухслойной ИНС. Классы разделяются гиперплоскостями.

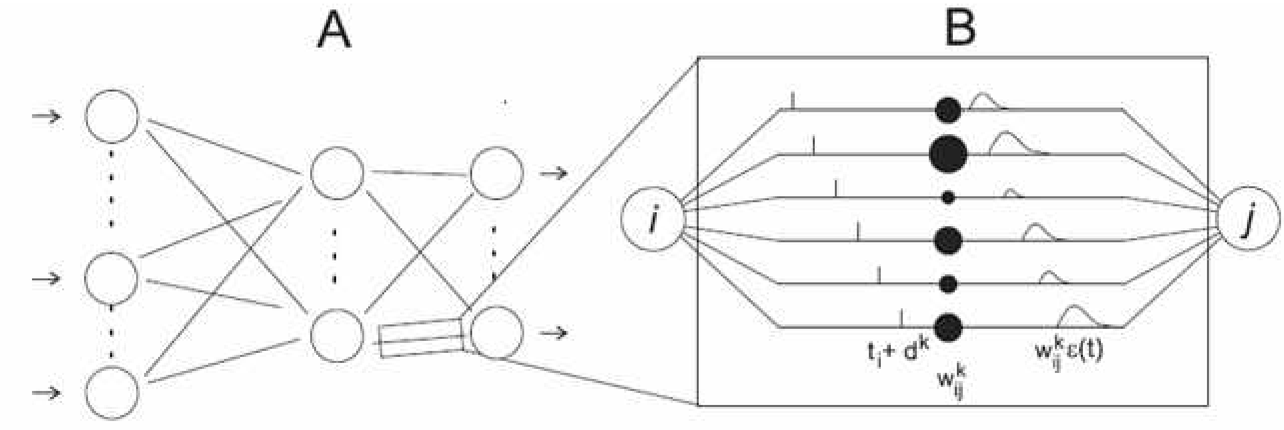


Рис. 5.20. Спайковая нейросеть (A) и
связь, содержащая совокупность задержанных субсинапсов (B)

Обычно используются два алгоритма обучения:

– обучение на основе коррекции ошибок (обучение с учителем), при этом чаще всего применяется алгоритм обратного распространения (BP). Это итеративный градиентный алгоритм обучения, который обеспечивает минимизацию среднеквадратичной ошибки;

– обучение Хебба (обучение без учителя), при этом чаще всего применяется алгоритм STDP.

Обучение ИНС (алгоритм обратного распространения)

1. Номер итерации обучения $n = 1$, инициализация посредством равномерного распределения на интервале (0,1) или [-0.5, 0.5] весов между скрытым и входным слоем $w_{imj}(n)$, задание множества допустимых весов $A = \{(i, j)\}$, задание задержки для m-го субсинапса синаптического (скрытого) нейрона d_m (разница между

срабатыванием предсинаптического (входного) и постсинаптического (выходного) нейронов), $i \in \overline{1, N^{(1)}}, m \in \overline{1, M}, j \in \overline{1, N^{(2)}}$, задание шага квантования по времени Δt, где $N^{(0)}$ – количество нейронов входного слоя, $N^{(1)}$ – количество нейронов скрытого слоя, M – количество субсинапсов в каждом синаптическом нейроне, $N^{(2)}$ – количество нейронов выходного слоя.

2. Задается обучающее множество $\{(\mathbf{t}_\mu^x, \mathbf{t}_\mu^d) \mid \mathbf{t}_\mu^x \in R^{N^{(0)}}, \mathbf{t}_\mu^d \in R^{N^{(2)}}\}$, $\mu \in \overline{1, P}$, где $\mathbf{t}_\mu^x$ – μ-й обучающий входной вектор (вектор времен срабатываний предсинаптических нейронов), $\mathbf{t}_\mu^d$ – μ-й обучающий выходной вектор (вектор желаемых времен срабатываний постсинаптических нейронов), P – мощность обучающего множества. Номер текущей пары из обучающего множества $\mu = 1$.

3. Вычисление времени срабатывания для каждого j-го выходного нейрона (прямой ход)

3.1. $t_i^x(n) = t_{\mu i}^x$, $t = 0$.

3.2. $y_j^{(2)}(t) = \sum_{i=1}^{N^{(0)}} \sum_{m=1}^{M} w_{imj}(n) y_{im}^{(1)}(t)$,

$$y_{im}^{(1)}(t) = \varepsilon(t - t_i^x(n) - d_m), \ \varepsilon(s) = \frac{s}{\tau}\exp\left(1 - \frac{s}{\tau}\right).$$

3.3. Если $y_j^{(2)}(t) \geq \theta$, то $t_j^y(n) = t$, переход к 4, иначе $t = t + \Delta t$, переход к 3.2.

4. Вычисление энергии ошибки ИНС

$$E(n) = \frac{1}{2}\sum_{j=1}^{N^{(2)}} e_j^2(n), \ e_j(n) = t_j^y(n) - t_{\mu j}^d,$$

5. Настройка синаптических весов на основе обобщенного дельта правила (обратный ход)

$$w_{imj}(n+1) = \begin{cases} w_{imj}(n) - \eta \dfrac{\partial E(n)}{\partial w_{imj}(n)}, & (i,j) \in A \\ w_{imj}(n), & (i,j) \notin A \end{cases},$$

$i \in \overline{1, N^{(0)}}, m \in \overline{1, M}, j \in \overline{1, N^{(2)}}$,

$$\frac{\partial E(n)}{\partial w_{imj}(n)} = y_{im}^{(1)}(t_j^y(n))\delta_j(n),$$

$$\delta_j(n) = \frac{(t_j^y(n) - t_{\mu j}^d)}{\sum_{i=1}^{N^{(0)}} \sum_{m=1}^{M} w_{imj}(n) \frac{\partial y_{im}^{(1)}(t_j^y(n))}{\partial t_j^y(n)}},$$

$$\frac{\partial y_{im}^{(1)}(t_j^y(n))}{\partial t_j^y n)} = \varepsilon(t_j^y(n) - t_i^x(n) - d_m)\left(\frac{1}{t_j^y(n) - t_i^x(n) - d_m} - \frac{1}{\tau}\right).$$

6. Проверка условия завершения

Если $n \bmod P > 0$, то $\mu = \mu + 1$, $n = n + 1$, переход к 3.

Если $n \bmod P = 0$ и $\frac{1}{P}\sum_{s=1}^{P} E(n - P + s) > \varepsilon$, то $n = n + 1$, переход к 2.

Если $n \bmod P = 0$ и $\frac{1}{P}\sum_{s=1}^{P} E(n - P + s) < \varepsilon$, то завершиться.

Обучение ИНС (алгоритм STDP)

1. Номер итерации обучения $n = 1$, инициализация посредством равномерного распределения на интервале (0,1) или [-0.5, 0.5] весов между скрытым и входным слоем $w_{imj}(n)$, задание множества допустимых весов $A = \{(i, j)\}$, задание задержки для m-го субсинапса синаптического (скрытого) нейрона d_m (разница между срабатыванием предсинаптического (входного) и постсинаптического (выходного) нейронов), $i \in \overline{1, N^{(1)}}, m \in \overline{1, M}$, $j \in \overline{1, N^{(2)}}$, задание шага квантования по времени Δt, где $N^{(0)}$ – количество нейронов входного слоя, $N^{(1)}$ – количество нейронов скрытого слоя, M – количество субсинапсов в каждом синаптическом нейроне, $N^{(2)}$ – количество нейронов выходного слоя.

2. Задается обучающее множество $\{\mathbf{t}_\mu^x \mid \mathbf{t}_\mu^x \in R^{N^{(0)}}\}$, $\mu \in \overline{1, P}$, где $\mathbf{t}_\mu^x$ – μ-й обучающий входной вектор (вектор времен срабатываний предсинаптических нейронов), P – мощность обучающего множества. Номер вектора из обучающего множества $\mu = 1$.

3. Вычисление времени срабатывания для каждого j-го выходного нейрона

3.1. $t_i^x(n) = t_{\mu i}^x$, $t = 0$.

3.2. $y_j^{(2)}(t) = \sum_{i=1}^{N^{(0)}} \sum_{m=1}^{M} w_{imj}(n) y_{im}^{(1)}(t)$,

$y_{im}^{(1)}(t) = \varepsilon(t - t_i^x(n) - d_m)$, $\varepsilon(s) = \frac{s}{\tau}\exp\left(1 - \frac{s}{\tau}\right)$.

3.3. Если $y_j^{(2)}(t) \ge \theta$, то $t_j^y(n) = t$, переход к 4, иначе $t = t + \Delta t$, переход к 3.2.

5. Настройка синаптических весов на основе правила STDP

$$w_{imj}(n+1) = \begin{cases} w_{imj}(n) - \eta\phi(t_j^y(n), t_i^x(n)), & (i,j) \in A \\ w_{imj}(n), & (i,j) \notin A \end{cases},$$

$i \in \overline{1, N^{(0)}}$, $m \in \overline{1, M}$, $j \in \overline{1, N^{(2)}}$,

$$\phi(t_j^y(n), t_i^x(n)) = \begin{cases} A^+ \exp\left(\frac{t_j^y(n) - t_i^x(n) - d_m}{\tau^+}\right), & t_j^y(n) - t_i^x(n) - d_m < 0 \\ -A^- \exp\left(-\frac{t_j^y(n) - t_i^x(n) - d_m}{\tau^-}\right), & t_j^y(n) - t_i^x(n) - d_m \ge 0 \end{cases},$$

где A^-, A^+, τ^-, τ^+ – положительные параметры.

6. Проверка условия завершения

Если $n \bmod P > 0$, то $\mu = \mu + 1$, $n = n + 1$, переход к 3.

Если $n \bmod P = 0$ и $\frac{1}{P}\sum_{s=1}^{P} E(n - P + s) > \varepsilon$, то $n = n + 1$, переход к 2.

Если $n \bmod P = 0$ и $\frac{1}{P}\sum_{s=1}^{P} E(n - P + s) < \varepsilon$, то завершиться.

Функционирование ИНС

1. $j = 1$.

2. $t = 0$.

3. $y_j^{(2)}(t) = \sum_{i=1}^{N^{(0)}} \sum_{m=1}^{M} w_{imj} y_{im}^{(1)}(t)$,

$$y_{im}^{(1)}(t) = \varepsilon(t - t_i^x - d_m),\ \varepsilon(s) = \frac{s}{\tau}\exp\left(1 - \frac{s}{\tau}\right).$$

4. Если $y_j^{(2)}(t) \ge \theta$, то $t_j^y = t$, переход к 3, иначе $t = t + \Delta t$, переход к 5.

5. Если $j < N^{(2)}$, то $j = j + 1$, переход к 2.

Достоинства

1. Является универсальным аппроксиматором. Обеспечивает глобальную аппроксимацию нелинейного отображения входного сигнала в выходной.
2. Обеспечивает хорошее качество обобщения.
3. Автоматически определяется количество скрытых слоев (равно одному).
4. Количество классов может быть больше двух (в выходном слое может быть больше одного нейрона).
5. В отличие от статических ИНС, позволяет осуществлять адаптивную фильтрацию, прогноз, адаптивное управление, параметрическую идентификацию модели, классификацию нестационарных сигналов.

Недостатки

1. Обучение происходит медленнее, чем в случае MLP, RBFNN, PNN, сети Хемминга, SOM, CPNN.
2. Отсутствует автоматическое определение числа субсинапсов и величины задержки для них.
3. В отличие от методов обучения SVM, градиентные методы обучения спайковой нейросети выбирают положение разделяющей гиперплоскости произвольным образом.
4. Обучение не сводится к задаче квадратичного программирования в выпуклой области, имеющей единственное решение.
5. В отличие от ART, не решает проблему пластичности-стабильности.

ЧАСТЬ 3
РЕКУРРЕНТНЫЕ ИСКУССТВЕННЫЕ НЕЙРОННЫЕ СЕТИ

ГЛАВА 6
РЕКУРРЕНТНЫЕ ИСКУССТВЕННЫЕ НЕЙРОННЫЕ СЕТИ БЕЗ АССОЦИАТИВНОЙ ПАМЯТИ

6.1. Нейросеть Джордана

На рис. 6.1 приведена нейросеть Джордана (JNN) [51,52], которая является рекуррентной двухслойной ИНС и построена на базе MLP. Классы разделяются гиперплоскостями.

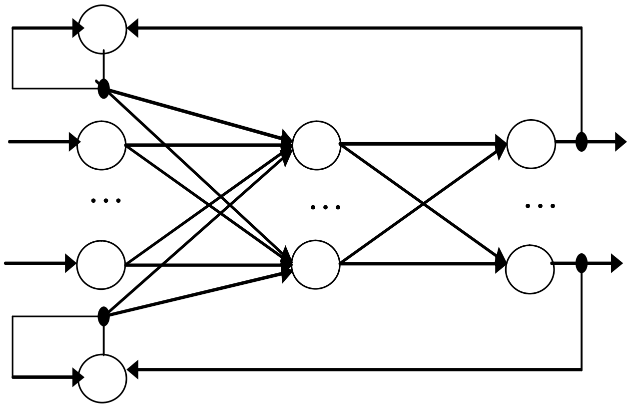


Рис. 6.1. Нейросеть Джордана (JNN)

Для JNN используется обучение на основе коррекции ошибок (обучение с учителем), при этом чаще всего применяется алгоритм обратного распространения (BP). Это итеративный градиентный алгоритм обучения, который обеспечивает минимизацию среднеквадратичной ошибки.

Обучение ИНС (алгоритм обратного распространения)

1. Номер итерации обучения $n = 1$, инициализация посредством равномерного распределения на интервале (0,1) или [-0.5, 0.5] смещений (порогов) $b_j^{(k)}(n)$ и весов $w_{ij}^{(k)}(n)$, $i \in \overline{1, N^{(k-1)}}$, $j \in \overline{1, N^{(k)}}, k \in \overline{1,2}$, где $N^{(k)}$ – количество нейронов в k-м слое.

2. Задается обучающее множество $\{(\mathbf{x}_\mu, \mathbf{d}_\mu) \mid \mathbf{x}_\mu \in R^{N^{(0)}}, \mathbf{d}_\mu \in R^{N^{(2)}}\}$, $\mu \in \overline{1, P}$, где $\mathbf{x}_\mu$ – μ-й обучающий входной вектор, $\mathbf{d}_\mu$ – μ-й обучающий выходной вектор, $N^{(0)}$ – количество нейронов входного слоя, $N^{(2)}$ – количество нейронов

выходного слоя, P – мощность обучающего множества. Номер текущей пары из обучающего множества $\mu = 1$.

3. Начальное вычисление выходного сигнала для второго слоя

$y_i^{(2)}(n-1) = 0$, $i \in \overline{1, N^{(0)}}$.

$z_i\ (n-1) = 0$, $i \in \overline{1, N^{(2)}}$.

4. Вычисление выходного сигнала для каждого слоя (прямой ход)

$y_i^{(0)}(n) = x_{\mu i}$,

$y_j^{(1)}(n) = f^{(1)}(s_j^{(1)}(n))$,

$$s_j^{(1)}(n) = \sum_{i=0}^{N^{(0)}} w_{ij}^{(1)}(n) y_i^{(0)}(n) + \sum_{i=N^{(0)}+1}^{N^{(0)}+N^{(2)}} w_{ij}^{(1)}(n) z_{i-N^{(0)}}(n),\ j \in \overline{1, N^{(1)}},$$

$$z_{i-N^{(0)}}(n) = y_{i-N^{(0)}}^{(2)}(n-1) + \gamma z_{i-N^{(0)}}(n-1),$$

$$y_j^{(2)}(n) = f^{(2)}(s_j^{(2)}(n)),\ s_j^{(2)}(n) = \sum_{i=0}^{N^{(1)}} w_{ij}^{(2)}(n) y_i^{(1)}(n),\ j \in \overline{1, N^{(2)}},$$

где $N^{(k)}$ – число нейронов в k-м слое, k – номер слоя, $w_{ij}^{(k)}(n)$ – вес связи от i-го нейрона к j-му нейрону на k-м слое в момент времени n, $y_j^{(k)}(n)$ – выход j-го нейрона на k-м слое, $f^{(k)}$ – функция активация нейронов k-го слоя, γ – параметр, $0 < \gamma < 1$.

Считается, что $w_{0j}^{(k)}(n) = b_j^{(k)}(n), y_0^{(k-1)}(n) = 1$.

5. Вычисление энергии ошибки ИНС

$$E(n) = \frac{1}{2} \sum_{j=1}^{N^{(2)}} e_j^2(n),\ e_j(n) = y_j^{(L)}(n) - d_{\mu j},$$

6. Настройка синаптических весов на основе обобщенного дельта правила (обратный ход)

$$w_{ij}^{(k)}(n+1) = w_{ij}^{(k)}(n) - \eta \frac{\partial E(n)}{\partial w_{ij}^{(k)}(n)},$$

где η – параметр, определяющий скорость обучения (при большом η обучение происходит быстрее, но увеличивается опасность получить неверное решение), $0 < \eta < 1$.

$$\frac{\partial E(n)}{\partial w_{ij}^{(2)}(n)} = y_i^{(1)}(n) g_j^{(2)}(n),\ i \in \overline{0, N^{(1)}},\ j \in \overline{1, N^{(2)}},$$

$$\frac{\partial E(n)}{\partial w_{ij}^{(1)}(n)} = \begin{cases} z_{i-N^{(0)}}^{(0)}(n) g_j^{(1)}(n), & i > N^{(0)} \\ y_i^{(0)}(n) g_j^{(1)}(n), & i \le N^{(0)} \end{cases}, \ i \in \overline{0, N^{(0)} + N^{(2)}}, \ j \in \overline{1, N^{(1)}},$$

$$g_j^{(k)}(n) = \begin{cases} f'^{(2)}(s_j^{(2)}(n))(y_j^{(2)}(n) - d_{\mu j}), & k = 2 \\ f'^{(1)}(s_j^{(1)}(n)) \sum_{l=1}^{N^{(2)}} w_{jl}^{(2)}(n) g_l^{(2)}(n), & k = 1 \end{cases}$$

7. Проверка условия завершения

Если $n \bmod P > 0$, то $\mu = \mu + 1$, $n = n + 1$, переход к 4.

Если $n \bmod P = 0$ и $\frac{1}{P}\sum_{s=1}^{P} E(n - P + s) > \varepsilon$, то $n = n + 1$, переход к 2.

Если $n \bmod P = 0$ и $\frac{1}{P}\sum_{s=1}^{P} E(n - P + s) < \varepsilon$, то завершиться.

Функционирование ИНС

1. Инициализация

$n = 1$.

$y_i^{(2)}(n-1) = 0$, $i \in \overline{1, N^{(0)}}$.

$z_i\ (n-1) = 0$, $i \in \overline{1, N^{(2)}}$.

2. Прогноз

$y_i^{(0)}(n) = x_i$,

$y_j^{(1)}(n) = f^{(1)}(s_j^{(1)}(n))$,

$$s_j^{(1)}(n) = b_j^{(1)} + \sum_{i=1}^{N^{(0)}} w_{ij}^{(1)} y_i^{(0)}(n) + \sum_{i=N^{(0)}+1}^{N^{(0)}+N^{(2)}} w_{ij}^{(1)} z_{i-N^{(0)}}^{(0)}(n),\ j \in \overline{1, N^{(1)}},$$

$$z_{i-N^{(0)}}^{(0)}(n) = y_{i-N^{(0)}}^{(2)}(n-1) + \gamma z_{i-N^{(0)}}^{(0)}(n-1),$$

$$y_j^{(2)}(n) = f^{(2)}(s_j^{(2)}(n)),\ s_j^{(2)}(n) = b_j^{(2)} + \sum_{i=1}^{N^{(1)}} w_{ij}^{(2)} y_i^{(1)}(n),\ j \in \overline{1, N^{(2)}}.$$

Достоинства

1. Является универсальным аппроксиматором. Обеспечивает глобальную аппроксимацию нелинейного отображения входного сигнала в выходной.

2. Обеспечивает хорошее качество обобщения.

3. Автоматически определяется количество скрытых слоев (равно одному).

4. Количество классов может быть больше двух (в выходном слое может быть больше одного нейрона).

5. В отличие от статических ИНС, позволяет осуществлять адаптивную фильтрацию, прогноз, адаптивное управление, параметрическую идентификацию модели, классификацию нестационарных сигналов.

Недостатки

1. Обучение происходит медленнее, чем в случае MLP, RBFNN, PNN, сети Хемминга, SOM, CPNN.

2. Отсутствует автоматическое определение числа нейронов в скрытом слое.

3. В отличие от методов обучения SVM, градиентные методы обучения JNN выбирают положение разделяющей гиперплоскости произвольным образом.

4. Обучение не сводится к задаче квадратичного программирования в выпуклой области, имеющей единственное решение.

5. В отличие от ART, не решает проблему пластичности-стабильности.

6. В отличие от RMLP хуже решает задачи адаптивной фильтрации, прогноза, адаптивного управления, параметрической идентификации модели, классификации нестационарных сигналов.

6.2. Нейросеть Элмана первого порядка

На рис. 6.2 приведена нейросеть Элмана (ENN) первого порядка или простая рекуррентная сеть (SRN) [53, 54], которая является рекуррентной двухслойной ИНС и построена на базе MLP. Классы разделяются гиперплоскостями.

Для ENN первого порядка используется обучение на основе коррекции ошибок (обучение с учителем), при этом чаще всего применяется алгоритм обратного распространения (BP). Это итеративный градиентный алгоритм обучения, который обеспечивает минимизацию среднеквадратичной ошибки.

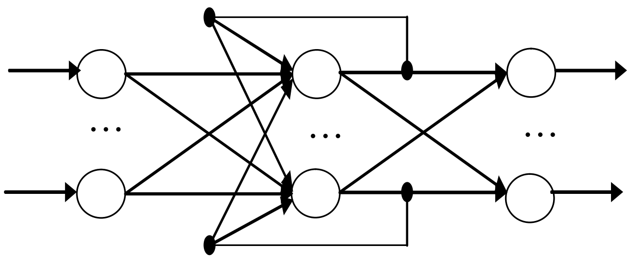


Рис. 6.2. Нейросеть Элмана (ENN) первого порядка

Обучение ИНС (алгоритм обратного распространения)

1. Номер итерации обучения $n=1$, инициализация посредством равномерного распределения на интервале (0,1) или [-0.5, 0.5] смещений (порогов) $b_j^{(k)}(n)$ и весов $w_{ij}^{(k)}(n)$, $i \in \overline{1, N^{(k-1)}}$, $j \in \overline{1, N^{(k)}}, k \in \overline{1,2}$, где $N^{(k)}$ – количество нейронов в k-м слое.

2. Задается обучающее множество $\{(\mathbf{x}_\mu, \mathbf{d}_\mu) \mid \mathbf{x}_\mu \in R^{N^{(0)}}, \mathbf{d}_\mu \in R^{N^{(2)}}\}$, $\mu \in \overline{1, P}$, где $\mathbf{x}_\mu$ – μ-й обучающий входной вектор, $\mathbf{d}_\mu$ – μ-й обучающий выходной вектор, $N^{(0)}$ – количество нейронов входного слоя, $N^{(2)}$ – количество нейронов выходного слоя, P – мощность обучающего множества. Номер текущей пары из обучающего множества $\mu = 1$.

3. Начальное вычисление выходного сигнала для первого слоя

$y_i^{(1)}(n-1) = 0$, $i \in \overline{1, N^{(1)}}$.

4. Вычисление выходного сигнала для каждого слоя (прямой ход)

$y_i^{(0)}(n) = x_{\mu i}$,

$y_j^{(1)}(n) = f^{(1)}(s_j^{(1)}(n))$,

$$s_j^{(1)}(n) = \sum_{i=0}^{N^{(0)}} w_{ij}^{(1)}(n) y_i^{(0)}(n) + \sum_{i=N^{(0)}+1}^{N^{(0)}+N^{(1)}} w_{ij}^{(1)}(n) y_{i-N^{(0)}}^{(1)}(n-1),\ j \in \overline{1, N^{(1)}},$$

$$y_j^{(2)}(n) = f^{(2)}(s_j^{(2)}(n)),\ s_j^{(2)}(n) = \sum_{i=0}^{N^{(1)}} w_{ij}^{(2)}(n) y_i^{(1)}(n),\ j \in \overline{1, N^{(2)}},$$

где $N^{(k)}$ – число нейронов в k-м слое, k – номер слоя, $w_{ij}^{(k)}(n)$ – вес связи от i-го нейрона к j-му нейрону на k-м слое в момент времени n,

$y_j^{(k)}(n)$ – выход *j*-го нейрона на *k*-м слое, $f^{(k)}$ – функция активация нейронов *k*-го слоя.

Считается, что $w_{0j}^{(k)}(n) = b_j^{(k)}(n), y_0^{(k-1)}(n) = 1$.

5. Вычисление энергии ошибки ИНС

$$E(n) = \frac{1}{2}\sum_{j=1}^{N^{(2)}} e_j^2(n),\ e_j(n) = y_j^{(2)}(n) - d_{\mu j},$$

6. Настройка синаптических весов на основе обобщенного дельта правила (обратный ход)

$$w_{ij}^{(k)}(n+1) = w_{ij}^{(k)}(n) - \eta \frac{\partial E(n)}{\partial w_{ij}^{(k)}(n)},$$

где η – параметр, определяющий скорость обучения (при большом η обучение происходит быстрее, но увеличивается опасность получить неверное решение), $0 < \eta < 1$.

$$\frac{\partial E(n)}{\partial w_{ij}^{(2)}(n)} = y_i^{(1)}(n) g_j^{(2)}(n),\ i \in \overline{0, N^{(1)}},\ j \in \overline{1, N^{(2)}},$$

$$\frac{\partial E(n)}{\partial w_{ij}^{(1)}(n)} = \begin{cases} y_{i-N^{(0)}}^{(1)}(n-1) g_j^{(1)}(n), & i > N^{(0)} \\ y_i^{(0)}(n) g_j^{(1)}(n), & i \le N^{(0)} \end{cases},$$

$i \in \overline{0, N^{(0)} + N^{(1)}}$, $j \in \overline{1, N^{(1)}}$,

$$g_j^{(k)}(n) = \begin{cases} f'^{(2)}(s_j^{(2)}(n))(y_j^{(2)}(n) - d_{\mu j}), & k = 2 \\ f'^{(1)}(s_j^{(1)}(n)) \sum_{l=1}^{N^{(2)}} w_{jl}^{(2)}(n) g_l^{(2)}(n), & k = 1 \end{cases}$$

7. Проверка условия завершения

Если $n \bmod P > 0$, то $\mu = \mu + 1$, $n = n + 1$, переход к 4.

Если $n \bmod P = 0$ и $\frac{1}{P}\sum_{s=1}^{P} E(n - P + s) > \varepsilon$, то $n = n + 1$, переход к 2.

Если $n \bmod P = 0$ и $\frac{1}{P}\sum_{s=1}^{P} E(n - P + s) < \varepsilon$, то завершиться.

Функционирование ИНС

1. Инициализация

$n = 1$.

$y_i^{(1)}(n-1)=0$, $i \in \overline{1, N^{(1)}}$.

2. Прогноз

$y_i^{(0)}(n)=x_i$,

$y_j^{(1)}(n)=f^{(1)}(s_j^{(1)}(n))$,

$$s_j^{(1)}(n)=b_j^{(1)}+\sum_{i=1}^{N^{(0)}} w_{ij}^{(1)} y_i^{(0)}(n)+\sum_{i=N^{(0)}+1}^{N^{(0)}+N^{(1)}} w_{ij}^{(1)} y_{i-N^{(0)}}^{(1)}(n-1),\ j \in \overline{1, N^{(1)}},$$

$$y_j^{(2)}(n)=f^{(2)}(s_j^{(2)}(n)),\ s_j^{(2)}(n)=b_j^{(2)}+\sum_{i=1}^{N^{(1)}} w_{ij}^{(2)} y_i^{(1)}(n),\ j \in \overline{1, N^{(2)}}$$

Достоинства

1. Является универсальным аппроксиматором. Обеспечивает глобальную аппроксимацию нелинейного отображения входного сигнала в выходной.

2. Обеспечивает хорошее качество обобщения.

3. Автоматически определяется количество скрытых слоев (равно одному).

4. Количество классов может быть больше двух (в выходном слое может быть больше одного нейрона).

5. В отличие от статических ИНС, позволяет осуществлять адаптивную фильтрацию, прогноз, адаптивное управление, параметрическую идентификацию модели, классификацию нестационарных сигналов.

Недостатки

1. Обучение происходит медленнее, чем в случае MLP, RBFNN, PNN, сети Хемминга, SOM, CPNN.

2. Отсутствует автоматическое определение числа нейронов в скрытом слое.

3. В отличие от методов обучения SVM, градиентные методы обучения ENN первого порядка выбирают положение разделяющей гиперплоскости произвольным образом.

4. Обучение не сводится к задаче квадратичного программирования в выпуклой области, имеющей единственное решение.

5. В отличие от ART, не решает проблему пластичности-стабильности (способность к восприятию новых образцов при сохранении старых образцов).

6. В отличие от RMLP, LSTM, ESN, BRNN хуже решает задачи адаптивной фильтрации, прогноза, адаптивного управления, параметрической идентификации модели, классификации нестационарных сигналов.

6.3. Нейросеть Элмана второго порядка

На рис. 6.3 приведена нейросеть Элмана (ENN) второго порядка [2], которая является рекуррентной двухслойной ИНС и построена на базе MLP. Классы разделяются гиперплоскостями. Нейроны скрытого слоя являются нейронами второго порядка. Полносвязная ENN второго порядка может использоваться для представления и обучения детерминированных конечных автоматов.

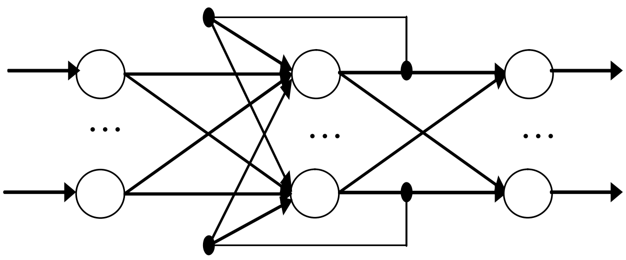


Рис. 6.3. Нейросеть Элмана (ENN) второго порядка

Для ENN второго порядка используется обучение на основе коррекции ошибок (обучение с учителем), при этом чаще всего применяется алгоритм обратного распространения (BP). Это итеративный градиентный алгоритм обучения, который обеспечивает минимизацию среднеквадратичной ошибки.

Обучение ИНС (алгоритм обратного распространения)

1. Номер итерации обучения $n=1$, инициализация посредством равномерного распределения на интервале (0,1) или [-0.5, 0.5] смещений (порогов) $b_j^{(2)}(n)$ и весов $w_{imj}^{(1)}(n), i \in \overline{1, N^{(0)}}, m, j \in \overline{1, N^{(1)}}$ и $w_{ij}^{(2)}(n)$, $i \in \overline{1, N^{(1)}}, j \in \overline{1, N^{(2)}}$, где $N^{(k)}$ – количество нейронов в k-м слое.

2. Задается обучающее множество $\{(\mathbf{x}_\mu, \mathbf{d}_\mu) \mid \mathbf{x}_\mu \in R^{N^{(0)}}, \mathbf{d}_\mu \in R^{N^{(2)}}\}$, $\mu \in \overline{1, P}$, где $\mathbf{x}_\mu$ – μ-й обучающий входной вектор, $\mathbf{d}_\mu$ – μ-й обучающий выходной вектор, $N^{(0)}$ – количество нейронов входного слоя, $N^{(2)}$ – количество нейронов выходного слоя, P – мощность обучающего множества. Номер текущей пары из обучающего множества $\mu = 1$.

3. Начальное вычисление выходного сигнала для первого слоя

$$y_i^{(1)}(n-1) = 0,\ i \in \overline{1, N^{(1)}}.$$

4. Вычисление выходного сигнала для каждого слоя (прямой ход)

$$y_i^{(0)}(n) = x_{\mu i},$$

$$y_j^{(1)}(n) = f^{(1)}(s_j^{1}(n)),$$

$$s_j^{(1)}(n) = \sum_{i=1}^{N^{(0)}} \sum_{m=1}^{N^{(1)}} w_{imj}^{(1)}(n) y_i^{(0)}(n) y_m^{(1)}(n-1),\ j \in \overline{1, N^{(1)}},$$

$$y_j^{(2)}(n) = f^{(2)}(s_j^{(2)}(n)),$$

$$s_j^{(2)}(n) = \sum_{m=0}^{N^{(2)}} w_{ij}^{(2)}(n) y_i^{(1)}(n),\ j \in \overline{1, N^{(2)}},$$

где $N^{(k)}$ – число нейронов в k-м слое, k – номер слоя, $w_{imj}^{(1)}(n)$ – вес связи от i-го и m-го нейронов к j-му нейрону на первом слое в момент времени n, $w_{ij}^{(2)}(n)$ – вес связи от i-го нейрона к j-му нейрону на втором слое в момент времени n, $y_j^{(k)}(n)$ – выход j-го нейрона на k-м слое, $f^{(k)}$ – функция активация нейронов k-го слоя.

Считается, что $w_{0j}^{(2)}(n) = b_j^{(2)}(n), y_0^{(1)}(n) = 1$.

5. Вычисление энергии ошибки ИНС

$$E(n) = \frac{1}{2} \sum_{j=1}^{N^{(2)}} e_j^2(n),\ e_j(n) = y_j^{(2)}(n) - d_{\mu j}.$$

6. Настройка синаптических весов на основе обобщенного дельта правила (обратный ход)

$$w_{ij}^{(2)}(n+1) = w_{ij}^{(2)}(n) - \eta \frac{\partial E(n)}{\partial w_{ij}^{(2)}(n)},$$

$$w_{imj}^{(1)}(n+1) = w_{imj}^{(1)}(n) - \eta \frac{\partial E(n)}{\partial w_{imj}^{(1)}(n)},$$

где η – параметр, определяющий скорость обучения (при большом η обучение происходит быстрее, но увеличивается опасность получить неверное решение), $0 < \eta < 1$.

$$\frac{\partial E(n)}{\partial w_{ij}^{(2)}(n)} = y_i^{(1)}(n) g_j^{(2)}(n),\ i \in \overline{0, N^{(1)}},\ j \in \overline{1, N^{(2)}},$$

$$\frac{\partial E(n)}{\partial w_{imj}^{(1)}(n)} = y_i^{(0)}(n) y_m^{(1)}(n-1) g_j^{(1)}(n),\ i \in \overline{1, N^{(0)}},\ m, j \in \overline{1, N^{(1)}},$$

$$g_j^{(2)}(n) = f'^{(2)}(s_j^{(2)}(n))(y_j^{(2)}(n) - d_{\mu j}),$$

$$g_j^{(1)}(n) = f'^{(1)}(s_j^{(1)}(n)) \sum_{l=1}^{N^{(2)}} w_{jl}^{(2)}(n) g_l^{(2)}(n)$$

7. Проверка условия завершения

Если $n \bmod P > 0$, то $\mu = \mu + 1$, $n = n + 1$, переход к 4.

Если $n \bmod P = 0$ и $\frac{1}{P}\sum_{s=1}^{P} E(n - P + s) > \varepsilon$, то $n = n + 1$, переход к 2.

Если $n \bmod P = 0$ и $\frac{1}{P}\sum_{s=1}^{P} E(n - P + s) < \varepsilon$, то завершиться.

Функционирование ИНС

1. Инициализация

$n = 1$.

$y_i^{(1)}(n-1) = 0$, $i \in \overline{1, N^{(1)}}$.

2. Прогноз

$y_i^{(0)}(n) = x_i$,

$y_j^{(1)}(n) = f^{(1)}(s_j^{1)}(n))$,

$$s_j^{(1)}(n) = \sum_{i=1}^{N^{(0)}} \sum_{m=1}^{N^{(1)}} w_{imj}^{(1)} y_i^{(0)}(n) y_m^{(1)}(n-1),\ j \in \overline{1, N^{(1)}},$$

$$y_j^{(2)}(n) = f^{(2)}(s_j^{(2)}(n)),\ s_j^{(2)}(n) = b_j^{(2)} + \sum_{m=1}^{N^{(2)}} w_{ij}^{(2)} y_i^{(1)}(n),\ j \in \overline{1, N^{(2)}}$$

Достоинства

1. Является универсальным аппроксиматором. Обеспечивает глобальную аппроксимацию нелинейного отображения входного сигнала в выходной.

2. Обеспечивает хорошее качество обобщения.

3. Количество классов может быть больше двух (в выходном слое может быть больше одного нейрона).

4. В отличие от статических ИНС, позволяет осуществлять адаптивную фильтрацию, прогноз, адаптивное управление, параметрическую идентификацию модели, классификацию нестационарных сигналов.

Недостатки

1. Обучение происходит медленнее, чем в случае MLP, RBFNN, PNN, сети Хемминга, SOM, CPNN.

2. Отсутствует автоматическое определение числа нейронов в этих слоях.

3. В отличие от методов обучения SVM, градиентные методы обучения ENN второго порядка выбирают положение разделяющей гиперплоскости произвольным образом.

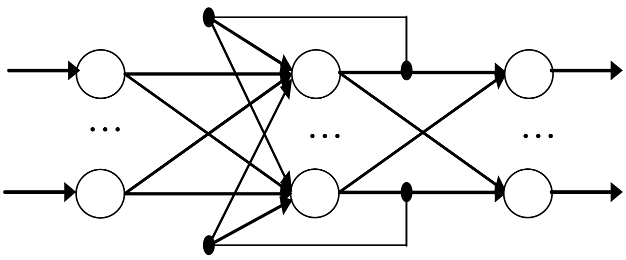
4. Обучение не сводится к задаче квадратичного программирования в выпуклой области, имеющей единственное решение.

5. В отличие от ART, не решает проблему пластичности-стабильности.

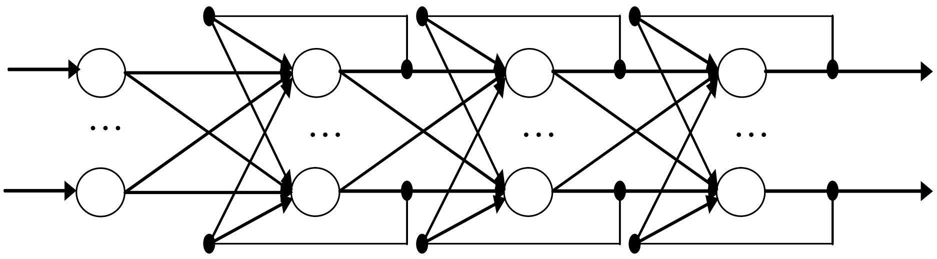
6. В отличие от RMLP, LSTM, ESN, BRNN хуже решает задачи адаптивной фильтрации, прогноза, адаптивного управления, параметрической идентификации модели, классификации нестационарных сигналов.

6.4. Рекуррентный многослойный персептрон

Рекуррентный многослойный персептрон (RMLP) [2] является рекуррентной ИНС и построен на базе MLP. Классы разделяются гиперплоскостями. На рис. 6.4 приведены примеры двух RMLP с одним и двумя скрытыми слоями соответственно. В общем случае количество скрытых слоев не менее одного.



а)



б)

Рис. 6.4. Рекуррентный многослойный персептрон (RMLP)

Обычно используются два вида алгоритмов обучения с учителем:

1. Алгоритмы на основе фильтра Калмана (например, алгоритм глобального расширенного фильтра Калмана (GEKF)).

2. Алгоритмы на основе градиента:

– алгоритм рекуррентного обучения в реальном времени (RTRL);

– алгоритм обратного распространения во времени (BPTT).

Алгоритмы на основе фильтра Калмана обладают более быстрой сходимостью и точностью, чем алгоритмы на основе градиента, но имеют большую вычислительную сложность. Алгоритм RTRL настраивает веса нейронов только одного рекуррентного скрытого слоя (для этого алгоритма подходит только сеть а) рис.6.4), а алгоритм на основе фильтра Калмана настраивает веса нейронов только одного (скрытого или выходного) рекуррентного слоя. В отличие от них, алгоритм BPTT может применяться к нейронам всех слоев.

Обучение одного рекуррентного слоя ИНС (глобальный алгоритм расширенной фильтрации Калмана)

1. Номер итерации обучения $n = 1$, номер обучаемого рекуррентного слоя l $(1 \le l \le L)$, $y_i^{(k)}(n-1) = 0$, $k \in \overline{1, L}$, $i \in \overline{1, N^{(k)}}$, инициализация посредством равномерного распределения на интервале (0,1) или [-0.5, 0.5] смещений (порогов) $b_j^{(k)}(n)$ и весов

$w_{ij}^{(k)}(n)$, $i \in \overline{1, N^{(k-1)}}, j \in \overline{1, N^{(k)}}, k \in \overline{1, L}$, где $N^{(k)}$ – количество нейронов в k-м слое, L – количество слоев.

2. Задается обучающее множество. Если требуемый рекуррентный слой является скрытым ($1 \le l < L$), то задается $\{\mathbf{x}_\mu \mid \mathbf{x}_\mu \in R^{N^{(0)}}\}$, $\mu \in \overline{1, P}$. Если требуемый рекуррентный слой является выходным ($l = L$), то задается $\{(\mathbf{x}_\mu, \mathbf{d}_\mu) \mid \mathbf{x}_\mu \in R^{N^{(0)}}, \mathbf{d}_\mu \in R^{N^{(L)}}\}$, $\mu \in \overline{1, P}$, где $\mathbf{x}_\mu$ – μ-й обучающий входной вектор, $\mathbf{d}_\mu$ – μ-й обучающий выходной вектор, $N^{(0)}$ – количество нейронов входного слоя, $N^{(L)}$ – количество нейронов выходного слоя, P – мощность обучающего множества. Номер текущей пары из обучающего множества $\mu = 1$.

3. Задается матрица ковариации ошибок $\mathbf{P}(0) = \delta \mathbf{I}$ размерности $W \times W$ (обычно $\delta = 1000$ или $\delta = 100$), матрица ковариации погрешности измерения $\mathbf{R}(0) = \eta_1(0)\mathbf{I}$ размерности $N^{(l)} \times N^{(l)}$ (обычно $\eta_1(0) = 1000$ или $\eta_1(0) = 100$), матрица ковариации шума процесса $\mathbf{Q}(0) = \eta_2(0)\mathbf{I}$ размерности $W \times W$ (обычно $\eta_2(0) = 10^{-1}$ или $\eta_2(0) = 10^{-2}$), где $W = (N^{(l-1)} + N^{(l)} + 1)N^{(l)}$ – число весов нейронов требуемого l-го рекуррентного слоя.

4. Формирование матрицы $\mathbf{C}(n)$ размерности $N^{(l)} \times W$ соответствуют частные производные выходов рекуррентных нейронов ИНС в виде

$$y_i^{(0)}(n) = x_{\mu i},$$

$$y_j^{(k)}(n) = f^{(k)}(s_j^{(k)}(n)),\ j \in \overline{1, N^{(k)}},$$

$$s_j^{(k)}(n) = \sum_{i=0}^{N^{(k-1)}} w_{ij}^{(k)}(n) y_i^{(k-1)}(n) + \sum_{i=N^{(k-1)}+1}^{N^{(k-1)}+N^{(k)}} w_{ij}^{(k)}(n) y_{i-N^{(k-1)}}^{(k)}(n-1),$$

$\mathbf{C}(n) = \left[\dfrac{\partial y_k^{(l)}(n)}{\partial w_{ij}^{(l)}(n)} \right]$, $i \in \overline{0, N^{(l-1)} + N^{(l)}}$, $k, j \in \overline{1, N^{(l)}}$, причем если $k \ne j$, то $\dfrac{\partial y_k^{(l)}(n)}{\partial w_{ij}^{(l)}(n)} = 0$.

где $N^{(k)}$ – число нейронов в k-м слое, k – номер слоя, $w_{ij}^{(k)}(n)$ – вес связи от i-го нейрона к j-му нейрону на k-м слое в момент времени n, $y_j^{(k)}(n)$ – выход j-го нейрона на k-м слое, $f^{(k)}$ – функция активация нейронов k-го слоя.

Считается, что $w_{0j}^{(k)}(n) = b_j^{(k)}(n), y_0^{(k-1)}(n) = 1$.

5. Вычисление вектора ошибки $\mathbf{e}(n)$ длиной $N^{(l)}$. Если требуемый рекуррентный слой является скрытым ($1 \le l < L$), то $\mathbf{e}(n) = \left(1 - \left(y_1^{(l)}(n)\right)^2, ..., 1 - \left(y_{N^{(l)}}^{(l)}(n)\right)^2\right)^T$. Если требуемый рекуррентный слой является выходным ($l = L$), то $\mathbf{e}(n) = \left(y_1^{(L)}(n) - d_{\mu 1}, ..., y_{N^{(L)}}^{(L)}(n) - d_{\mu N^{(L)}}\right)^T$

6. Формирование вектора весов нейронов требуемого рекуррентного слоя $\mathbf{w}(n)$ длиной W, т.е. $\mathbf{w}(n) = \left(w_{01}^{(l)}(n) ... w_{N^{(l-1)}+N^{(l)}, N^{(l)}}^{(l)}(n)\right)^T$.

7. Вычисление коэффициента усиления Калмана, представленного матрицей $\mathbf{\Gamma}(n)$ размерности $W \times N^{(l)}$

$$\mathbf{\Gamma}(n) = \mathbf{P}(n)\mathbf{C}^T(n)[\mathbf{C}(n)\mathbf{P}(n)\mathbf{C}^T(n) + \mathbf{R}(n)]^{-1}.$$

8. Настройка вектора синаптических весов $\mathbf{w}(n+1)$ на основе правила GEKF

$$\mathbf{w}(n+1) = \mathbf{w}(n) + \mathbf{\Gamma}(n)\mathbf{e}(n).$$

9. Настройка матрицы ковариации ошибок $\mathbf{P}(n+1)$

$$\mathbf{P}(n+1) = \mathbf{P}(n) - \mathbf{\Gamma}(n)\mathbf{C}(n)\mathbf{P}(n) + \mathbf{Q}(n).$$

10. Настройка матрицы ковариации погрешности измерения $\mathbf{R}(n+1)$

$$\mathbf{R}(n+1) = \eta_1(n)\mathbf{R}(n)$$

где $\eta_1(n)$ – параметр обучения, определяющий скорость уменьшения погрешности измерения, обычно уменьшается до 1.

11. Настройка матрицы ковариации шума процесса $\mathbf{Q}(n+1)$

$$\mathbf{Q}(n+1) = \eta_2(n)\mathbf{Q}(n)$$

где $\eta_2(n)$ – параметр обучения, определяющий скорость уменьшения шума процесса, обычно уменьшается до 10^{-6}.

12. Проверка условия завершения

Если $n < P$, то $\mu = \mu + 1$, $n = n + 1$, переход к 4.

Обучение одного рекуррентного скрытого слоя ИНС (алгоритм рекуррентного обучения в реальном времени)

1. Номер итерации обучения $n = 1$, $y_i^{(k)}(n-1) = 0$, $k \in \overline{1,2}$, $i \in \overline{1, N^{(k)}}$, инициализация посредством равномерного распределения на интервале (0,1) или [-0.5, 0.5] смещений (порогов) $b_j^{(k)}(n)$ и весов $w_{ij}^{(k)}(n)$, $i \in \overline{1, N^{(k-1)}}, j \in \overline{1, N^{(k)}}, k \in \overline{1,2}$, где $N^{(k)}$ – количество нейронов в k-м слое.

2. Задается обучающее множество $\{(\mathbf{x}_\mu, \mathbf{d}_\mu) \mid \mathbf{x}_\mu \in R^{N^{(0)}}, \mathbf{d}_\mu \in R^{N^{(2)}}\}$, $\mu \in \overline{1, P}$, где $\mathbf{x}_\mu$ – μ-й обучающий входной вектор, $\mathbf{d}_\mu$ – μ-й обучающий выходной вектор, $N^{(0)}$ – количество нейронов входного слоя, $N^{(2)}$ – количество нейронов выходного слоя, P – мощность обучающего множества. Номер текущей пары из обучающего множества $\mu = 1$.

3. Задаются нулевые матрицы частных производных $\mathbf{C}_j(n)$ размерности $N^{(1)} \times (N^{(0)} + N^{(1)} + 1)$, $j \in \overline{1, N^{(1)}}$.

4. Вычисление выходного сигнала для каждого слоя

$$y_i^{(0)}(n) = x_{\mu i},$$

$$y_j^{(1)}(n) = f^{(1)}(s_j^{(1)}(n)), \; j \in \overline{1, N^{(1)}},$$

$$s_j^{(1)}(n) = b_j^{(1)} + \sum_{i=1}^{N^{(0)}} w_{ij}^{(1)} y_i^{(0)}(n) + \sum_{i=N^{(0)}+1}^{N^{(0)}+N^{(1)}} w_{ij}^{(1)} y_{i-N^{(0)}}^{(1)}(n-1),$$

$$y_j^{(2)}(n) = f^{(2)}(s_j^{(2)}(n)), \; s_j^{(2)}(n) = \sum_{i=1}^{N^{(1)}} w_{ij}^{(2)} y_i^{(1)}(n), \; j \in \overline{1, N^{(2)}},$$

где $N^{(k)}$ – число нейронов в k-м слое, k – номер слоя, $w_{ij}^{(k)}(n)$ – вес связи от i-го нейрона к j-му нейрону на k-м слое в момент времени n, $y_j^{(k)}(n)$ – выход j-го нейрона на k-м слое, $f^{(k)}$ – функция активация нейронов k-го слоя.

Считается, что $w_{0j}^{(k)}(n) = b_j^{(k)}(n), y_0^{(k-1)}(n) = 1$.

5. Формирование диагональных матриц $\mathbf{U}_j(n)$ размерности $N^{(1)} \times N^{(1)}$ с единственной ненулевой j-й строкой

$$\mathbf{U}_j(n) = \begin{bmatrix} 0 & \ldots & 0 & 0 & \ldots & 0 \\ \ldots & \ldots & \ldots & \ldots & \ldots & \ldots \\ y_1^{(1)}(n-1) & \ldots & y_{N^{(k)}}^{(1)}(n-1) & y_0^{(0)}(n) & \ldots & y_{N^{(k-1)}}^{(0)}(n) \\ \ldots & \ldots & \ldots & \ldots & \ldots & \ldots \\ 0 & \ldots & 0 & 0 & \ldots & 0 \end{bmatrix},$$

$j \in \overline{1, N^{(1)}}$.

6. Формирование диагональной матрицы $\mathbf{\Phi}$ размерности $N^{(1)} \times N^{(1)}$

$$\mathbf{\Phi}(n) = \begin{bmatrix} f'^{(1)}(s_1^{(1)}(n)) & \ldots & 0 & \ldots & 0 \\ \ldots & \ldots & \ldots & \ldots & \ldots \\ 0 & \ldots & f'^{(1)}(s_j^{(1)}(n)) & \ldots & 0 \\ \ldots & \ldots & \ldots & \ldots & \ldots \\ 0 & \ldots & 0 & \ldots & f'^{(1)}(s_{N^{(l)}}^{(1)}(n)) \end{bmatrix}$$

7. Формирование вектора весов нейронов требуемого рекуррентного скрытого слоя $\mathbf{w}_j(n)$ длиной $N^{(0)} + N^{(1)} + 1$, т.е. $\mathbf{w}_j(n) = \left(w_{0j}^{(1)}(n) \ldots w_{N^{(0)}+N^{(1)},j}^{(1)}(n) \right)^T$.

8. Формирование матрицы весов обратных связей рекуррентного скрытого слоя $\mathbf{W}_r(n)$ размерности $N^{(1)} \times N^{(1)}$

$$\mathbf{W}_r(n) = \begin{bmatrix} w_{N^{(0)}+1,1}^{(1)}(n) & \ldots & w_{i,1}^{(1)}(n) & \ldots & w_{N^{(0)}+N^{(1)},1}^{(1)}(n) \\ \ldots & \ldots & \ldots & \ldots & \ldots \\ w_{N^{(0)}+1,j}^{(1)}(n) & \ldots & w_{i,j}^{(1)}(n) & \ldots & w_{N^{(0)}+N^{(1)},j}^{(1)}(n) \\ \ldots & \ldots & \ldots & \ldots & \ldots \\ w_{N^{(0)}+1,N^{(1)}}^{(1)}(n) & \ldots & w_{i,N^{(1)}}^{(1)}(n) & \ldots & w_{N^{(0)}+N^{(1)},N^{(1)}}^{(1)}(n) \end{bmatrix}$$

9. Формирование матрицы весов выходного слоя $\mathbf{W}^{(2)}(n)$ размерности $N^{(2)} \times N^{(2)}$

$$\mathbf{W}^{(2)}(n)=\begin{bmatrix} w_{1,1}^{(2)}(n) & \dots & w_{i,1}^{(2)}(n) & \dots & w_{N^{(1)},1}^{(2)}(n) \\ \dots & \dots & \dots & \dots & \dots \\ w_{1,j}^{(2)}(n) & \dots & w_{i,j}^{(2)}(n) & \dots & w_{N^{(1)},j}^{(2)}(n) \\ \dots & \dots & \dots & \dots & \dots \\ w_{1,N^{(2)}}^{(2)}(n) & \dots & w_{i,N^{(2)}}^{(2)}(n) & \dots & w_{N^{(1)},N^{(2)}}^{(2)}(n) \end{bmatrix}$$

10. Вычисление матриц частных производных $\mathbf{C}_j(n)$

$$\mathbf{C}_j(n+1)=\mathbf{\Phi}(n)(\mathbf{W}_r(n)\mathbf{C}_j(n)+\mathbf{U}_j(n)),\ j\in\overline{1,N^{(1)}}.$$

11. Вычисление вектора ошибки $\mathbf{e}(n)$ длиной $N^{(2)}$

$$\mathbf{e}(n)=\left(y_1^{(2)}(n)-d_{\mu 1},...,y_{N^{(2)}}^{(2)}(n)-d_{\mu N^{(2)}}\right)^T$$

12. Настройка вектора синаптических весов нейронов требуемого рекуррентного слоя $\mathbf{w}_j(n)$ на основе правила RTRL

$$\mathbf{w}_j(n+1)=\mathbf{w}_j(n)+\eta(\mathbf{W}^{(2)}\mathbf{C}_j(n))\mathbf{e}(n),\ j\in\overline{1,N^{(1)}}.$$

где η – параметр, определяющий скорость обучения (при большом η обучение происходит быстрее, но увеличивается опасность получить неверное решение), $0<\eta<1$.

13. Проверка условия завершения

Если $n<P$, то $\mu=\mu+1$, $n=n+1$, переход к 4.

Обучение всей ИНС (алгоритм обратного распространения во времени)

1. Номер итерации обучения $n=1$, инициализация посредством равномерного распределения на интервале (0,1) или [-0.5, 0.5] смещений (порогов) $b_j^{(k)}(n)$ и весов $w_{ij}^{(k)}(n)$, $i\in\overline{1,N^{(k-1)}}$, $j\in\overline{1,N^{(k)}}$, $k\in\overline{1,L}$, где $N^{(k)}$ – количество нейронов в k-м слое, L – количество слоев.

2. Задается обучающее множество $\{(\mathbf{x}_\mu,\mathbf{d}_\mu)\,|\,\mathbf{x}_\mu\in R^{N^{(0)}},\mathbf{d}_\mu\in R^{N^{(L)}}\}$, $\mu\in\overline{1,P}$, где $\mathbf{x}_\mu$ – μ-й обучающий входной вектор, $\mathbf{d}_\mu$ – μ-й обучающий выходной вектор, $N^{(0)}$ – количество нейронов входного слоя, $N^{(L)}$ – количество нейронов выходного слоя, P – мощность обучающего множества. Номер текущей пары из обучающего множества $\mu=1$.

3. Начальное вычисление выходного сигнала для первого слоя

$y_i^{(k)}(n-1)=0$, $i\in\overline{1,N^{(k)}}$, $k\in\overline{1,L}$.

4. Вычисление выходного сигнала для каждого слоя (прямой ход)

$$y_i^{(0)}(n)=x_{\mu i},$$

$$y_j^{(k)}(n)=f^{(k)}(s_j^{(k)}(n)),\ j\in\overline{1,N^{(k)}},$$

$$s_j^{(k)}(n)=\sum_{i=0}^{N^{(k-1)}} w_{ij}^{(k)}(n)y_i^{(k-1)}(n)+\sum_{i=N^{(k-1)}+1}^{N^{(k-1)}+N^{(k)}} w_{ij}^{(k)}(n)y_{i-N^{(k-1)}}^{(k)}(n-1),$$

где $N^{(k)}$ – число нейронов в k-м слое, k – номер слоя, $w_{ij}^{(k)}(n)$ – вес связи от i-го нейрона к j-му нейрону на k-м слое в момент времени n, $y_j^{(k)}(n)$ – выход j-го нейрона на k-м слое, $f^{(k)}$ – функция активация нейронов k-го слоя.

Считается, что $w_{0j}^{(k)}(n)=b_j^{(k)}(n), y_0^{(k-1)}(n)=1$.

5. Вычисление энергии ошибки ИНС

$$E(n)=\frac{1}{2}\sum_{j=1}^{N^{(L)}} e_j^2(n),\ e_j(n)=y_j^{(L)}(n)-d_{\mu j}.$$

6. Настройка синаптических весов на основе обобщенного дельта правила (обратный ход)

$$w_{ij}^{(k)}(n+1)=w_{ij}^{(k)}(n)-\eta\frac{\partial E(n)}{\partial w_{ij}^{(k)}(n)},$$

где η – параметр, определяющий скорость обучения (при большом η обучение происходит быстрее, но увеличивается опасность получить неверное решение), $0<\eta<1$.

$$\frac{\partial E(n)}{\partial w_{ij}^{(k)}(n)}=\begin{cases} y_{i-N^{(k-1)}}^{(k)}(n-1)g_j^{(k)}(n), & i>N^{(k-1)} \\ y_i^{(k-1)}(n)g_j^{(k)}(n), & i\le N^{(k-1)} \end{cases},$$

$i\in\overline{0,N^{(k-1)}+N^{(k)}}$, $j\in\overline{1,N^{(k)}}$,

$$g_j^{(k)}(n)=\begin{cases} f'^{(L)}(s_j^{(L)}(n))(y_j^{(L)}(n)-d_{\mu j}), & k=L \\ f'^{(k)}(s_j^{(k)}(n))\sum_{l=1}^{N^{(k+1)}} w_{jl}^{(k+1)}(n)g_l^{(k+1)}(n), & k<L \end{cases}$$

7. Проверка условия завершения

Если $n \bmod P>0$, то $\mu=\mu+1$, $n=n+1$, переход к 4.

Если $n \bmod P = 0$ и $\frac{1}{P}\sum_{s=1}^{P} E(n-P+s) > \varepsilon$, то $n = n+1$, переход к 2.

Если $n \bmod P = 0$ и $\frac{1}{P}\sum_{s=1}^{P} E(n-P+s) < \varepsilon$, то завершиться.

Функционирование ИНС

1. Инициализация

$n = 1$.

$y_i^{(k)}(n-1) = 0$, $i \in \overline{1, N^{(k)}}$, $k \in \overline{1, L}$.

2. Прогноз

$y_i^{(0)}(n) = x_i$,

$y_j^{(k)}(n) = f^{(k)}(s_j^{(k)}(n))$,

$$s_j^{(k)}(n) = b_j^{(k)} + \sum_{i=1}^{N^{(k-1)}} w_{ij}^{(k)} y_i^{(k-1)}(n) + \sum_{i=N^{(k-1)}+1}^{N^{(k-1)}+N^{(k)}} w_{ij}^{(k)} y_{i-N^{(k-1)}}^{(k)}(n-1),$$

$j \in \overline{1, N^{(k)}}, k \in \overline{1, L}$.

Достоинства

1. Является универсальным аппроксиматором. Обеспечивает глобальную аппроксимацию нелинейного отображения входного сигнала в выходной.

2. Обеспечивает хорошее качество обобщения (т.е. хорошо работает с тестовыми данными, которые в процессе обучения не предъявлялись).

3. Количество классов может быть больше двух (в выходном слое может быть больше одного нейрона).

4. В отличие от статических ИНС, позволяет осуществлять адаптивную фильтрацию, прогноз, адаптивное управление, параметрическую идентификацию модели, классификацию нестационарных сигналов.

Недостатки

1. Обучение происходит медленнее, чем в случае MLP, RBFNN, PNN, сети Хемминга, SOM, CPNN.

2. Отсутствует автоматическое определение числа скрытых слоев и числа нейронов в этих слоях.

3. В отличие от методов обучения SVM, градиентные методы обучения RMLP выбирают положение разделяющей гиперплоскости произвольным образом.

4. Обучение не сводится к задаче квадратичного программирования в выпуклой области, имеющей единственное решение.

5. В отличие от ART, не решает проблему пластичности-стабильности.

6.5. Нейросеть нелинейной авторегрессии-скользящего среднего

Нейросеть нелинейной авторегрессии-скользящего среднего (NARMANN) [39] является рекуррентной двухслойной ИНС. На рис.6.5 скрытый слой изображен прямоугольником, за основу бралась модель нелинейной авторегрессии-скользящего среднего NARMA(5,3). NARMANN является особым классом многослойного персептрона.

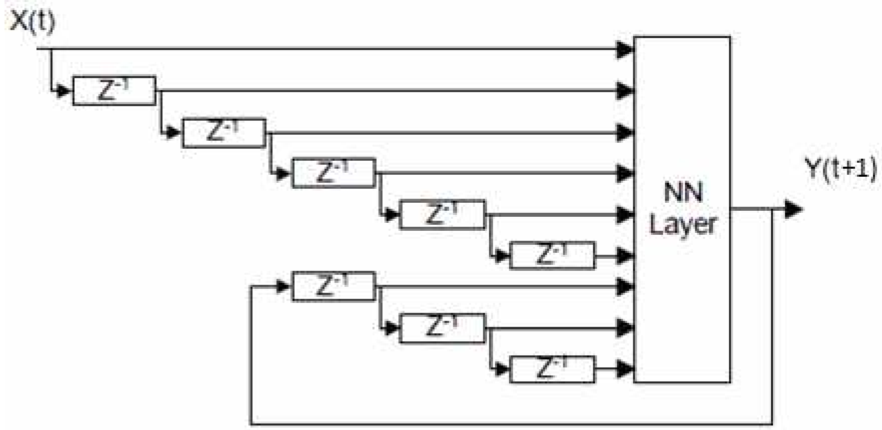


Рис. 6.5. Нейросеть нелинейной авторегрессии-скользящего среднего (NARMANN)

Для NARMANN используется обучение на основе коррекции ошибок (обучение с учителем), при этом чаще всего применяется алгоритм обратного распространения (BP). Это итеративный градиентный алгоритм обучения, который обеспечивает минимизацию среднеквадратичной ошибки.

Обучение ИНС (алгоритм обратного распространения)

1. Номер итерации обучения $n = 1$, инициализация посредством равномерного распределения на интервале (0,1) или [-0.5, 0.5] смещений (порогов) $b^{(1)}(n), b^{(2)}(n)$ и весов $w_{lj}^{(1)}(n)$,

$l \in \overline{0, M^{(0)} + M^{(2)}}$, $j \in \overline{1, N^{(1)}}$, $w_i^{(2)}(n)$, $i \in \overline{1, N^{(1)}}$, где $N^{(1)}$ – количество нейронов в первом слое, $M^{(k)}$ – задержка в k-ом слое.

2. Задается обучающее множество $\{(x_\mu, d_\mu) \mid x_\mu \in R, d_\mu \in R\}$, $\mu \in \overline{1, P}$, где x_μ – μ-е обучающее входное значение, d_μ – μ-е обучающее выходное значение, P – мощность обучающего множества. Номер текущей пары из обучающего множества $\mu = 1$

3. Начальное вычисление выходного сигнала для входного и выходного слоя

$M = \max\{M^{(0)}, M^{(2)}\}$.

$y^{(0)}(n - v) = 0$, $v \in \overline{1, M}$.

$y^{(2)}(n - v) = 0$, $v \in \overline{1, M}$.

4. Вычисление выходного сигнала для каждого слоя (прямой ход)

$y^{(0)}(n) = x_\mu$,

$y_j^{(1)}(n) = f^{(1)}(s_j^{(1)}(n))$, $j \in \overline{1, N^{(1)}}$,

$$s_j^{(1)}(n) = b_j^{(1)}(n) + \sum_{l=0}^{M^{(0)}} w_{lj}^{(1)}(n) y^{(0)}(n - l) +$$

$$+ \sum_{l=M^{(0)}+1}^{M^{(0)}+M^{(2)}} w_{lj}^{(1)}(n)(y^{(0)}(n - (l - M^{(0)})) - y^{(2)}(n - (l - M^{(0)}))),$$

$$y^{(2)}(n) = f^{(2)}(s^{(2)}(n)),\ s^{(2)}(n) = \sum_{i=0}^{N^{(1)}} w_i^{(2)}(n) y_i^{(1)}(n),$$

где $N^{(1)}$ – число нейронов в первом слое, $w_{lj}^{(1)}(n)$ – вес связи от входного нейрона в момент времени $n - l$ к *j*-му нейрону в первом слое в момент времени n либо вес связи от выходного нейрона в момент времени $n - (l - M^{(0)})$ к *j*-му нейрону в первом слое в момент времени n, $w_i^{(2)}(n)$ – вес связи от i-го нейрона к выходному нейрону в момент времени n, $y_j^{(1)}(n)$ – выход *j*-го нейрона в первом слое, $y^{(2)}(n)$ – выход нейрона во втором слое, $f^{(k)}$ – функция активация нейронов *k*-го слоя.

Считается, что $w_0^{(2)}(n) = b^{(2)}(n), y_0^{(1)}(n) = 1$.

5. Вычисление энергии ошибки ИНС

$E(n)=\frac{1}{2}e^2(n)$, $e(n)=y^{(2)}(n)-d_\mu$.

6. Настройка синаптических весов на основе обобщенного дельта правила (обратный ход)

$$w_i^{(2)}(n+1)=w_i^{(2)}(n)-\eta\frac{\partial E(n)}{\partial w_i^{(2)}(n)},$$

$$w_{lj}^{(1)}(n+1)=w_{lj}^{(1)}(n)-\eta\frac{\partial E(n)}{\partial w_{lj}^{(1)}(n)},$$

где η – параметр, определяющий скорость обучения (при большом η обучение происходит быстрее, но увеличивается опасность получить неверное решение), $0<\eta<1$.

$$\frac{\partial E(n)}{\partial w_i^{(2)}(n)}=y_i^{(1)}(n)g^{(2)}(n),\ i\in\overline{0,N^{(1)}},$$

$$\frac{\partial E(n)}{\partial w_{lj}^{(1)}(n)}=\begin{cases}y^{(0)}(n-l)g_j^{(1)}(n), & l\le M\\ (y^{(0)}(n-(l-M^{(0)}))-y^{(2)}(n-(l-M^{(0)})))g_j^{(1)}(n), & l>M\end{cases},$$

$l\in\overline{0,M^{(0)}+M^{(2)}}$, $j\in\overline{1,N^{(1)}}$,

$$\frac{\partial E(n)}{\partial b_j^{(1)}(n)}=g_j^{(1)}(n),\ j\in\overline{1,N^{(1)}},$$

$$g^{(2)}(n)=f'^{(2)}(s^{(2)}(n))(y^{(2)}(n)-d_\mu),$$

$$g_j^{(1)}(n)=f'^{(1)}(s_j^{(1)}(n))w_j^{(2)}(n)g^{(2)}(n).$$

7. Проверка условия завершения

Если $n \bmod P>0$, то, , переход к 4.

Если $n \bmod P=0$ и $\frac{1}{P}\sum_{s=1}^{P}E(n-P+s)>\varepsilon$, то $n=n+1$, переход к 2.

Если $n \bmod P=0$ и $\frac{1}{P}\sum_{s=1}^{P}E(n-P+s)<\varepsilon$, то завершиться.

Функционирование ИНС

1. Инициализация

$n=1$.

$M=\max\{M^{(0)},M^{(2)}\}$.

$y^{(0)}(n-v)=0,\ v\in\overline{1,M}$.

$y^{(2)}(n-v)=0,\ v\in\overline{1,M}$.

2. Прогноз

$y(n)=x_1$.

$y_j^{(1)}(n)=f^{(1)}(s_j^{(1)}(n)),\ j\in\overline{1,N^{(1)}}$,

$$s_j^{(1)}(n)=b_j^{(1)}+\sum_{l=0}^{M^{(0)}} w_{lj}^{(1)}y^{(0)}(n-l)+$$

$$+\sum_{l=M^{(0)}+1}^{M^{(0)}+M^{(2)}} w_{lj}^{(1)}(y^{(0)}(n-(l-M^{(0)}))-y^{(2)}(n-(l-M^{(0)}))).$$

$$y^{(2)}(n)=f^{(2)}(b^{(2)}+\sum_{i=1}^{N^{(1)}} w_i^{(2)}y_i^{(1)}(n)).$$

Достоинства

1. Является универсальным аппроксиматором. Обеспечивает глобальную аппроксимацию нелинейного отображения входного сигнала в выходной.

2. Обеспечивает хорошее качество обобщения.

3. Автоматически определяется количество скрытых слоев (равно одному).

4. В отличие от статических ИНС позволяет осуществлять адаптивную фильтрацию, прогноз, адаптивное управление, параметрическую идентификацию модели, классификацию нестационарных сигналов.

5. В отличие от NARNN лучше решает задачи адаптивной фильтрации, прогноза, адаптивного управления, параметрической идентификации модели, классификации нестационарных сигналов.

Недостатки

1. Обучение происходит медленнее, чем в случае MLP, RBFNN, PNN, сети Хемминга, SOM, CPNN.

2. Отсутствует автоматическое определение числа нейронов в скрытом слое.

3. В отличие от методов обучения SVM, градиентные методы обучения NARMANN выбирают положение разделяющей гиперплоскости произвольным образом.

4. Обучение не сводится к задаче квадратичного программирования в выпуклой области, имеющей единственное решение.

5. В отличие от ART, не решает проблему пластичности-стабильности.

6. Может быть только два класса, что не всегда подходит для задач классификации.

7. В отличие от LSTM не работает с нестационарными последовательностями.

8. В отличие от NARNN обучается медленнее.

6.6. Долгократкосрочная память

Рассмотрим традиционную и модифицированную долгократкосрочную память.

6.6.1. Традиционная долгократкосрочная память

На рис.6.6 приведена традиционная долгократкосрочная память (LSTM) [55, 56], которая является рекуррентной двухслойной ИНС. Вместо скрытого нейрона используется блок памяти, который содержит одну или более ячеек (рис.6.7), и связан с входным и выходным шлюзами (КИХ-фильтры).

Обучение традиционной LSTM происходит на основе коррекции ошибок (обучение с учителем) в виде комбинации алгоритмов обратного распространения во времени (BPTT) и рекуррентного обучения в реальном времени (RTRL).

Обучение ИНС (комбинация алгоритмов обратного распространения во времени и рекуррентного обучения в реальном времени)

1. Номер итерации обучения $n=2$, инициализация посредством равномерного распределения на интервале (0,1) или [-0.5, 0.5] весов $w_{ij}^{in-gin}(n)$, $w_{ij}^{in-gout}(n)$, $w_{ij}^{in-c}(n)$, $i\in\overline{1,N^{(0)}}$, $j\in\overline{1,N^{(1)}}$, $w_{ij}^{c-out}(n)$, $i\in\overline{1,N^{(1)}}$, $j\in\overline{1,N^{(2)}}$, где $N^{(0)}$ – количество нейронов входного слоя, $N^{(1)}$ – количество нейронов скрытого слоя, $N^{(2)}$ – количество нейронов выходного слоя, S_j – количество ячеек в j-м блоке памяти.

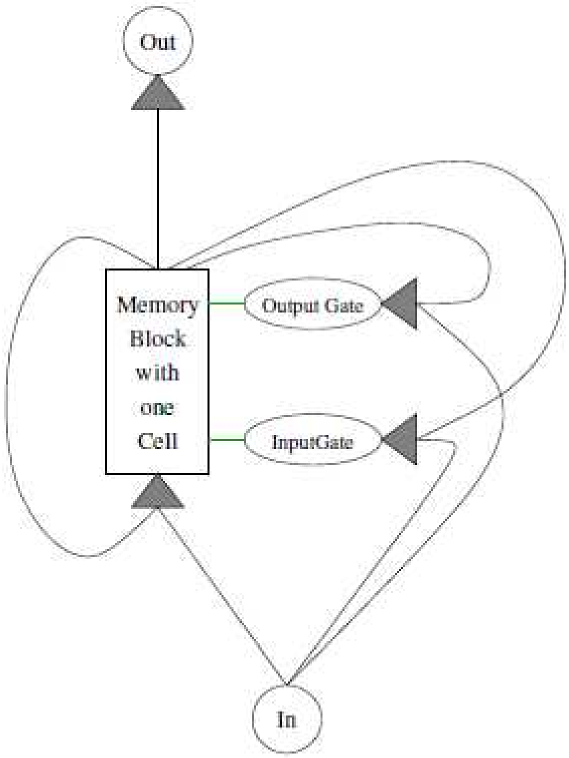


Рис.6.6. Традиционная долгократкосрочная память (LSTM)

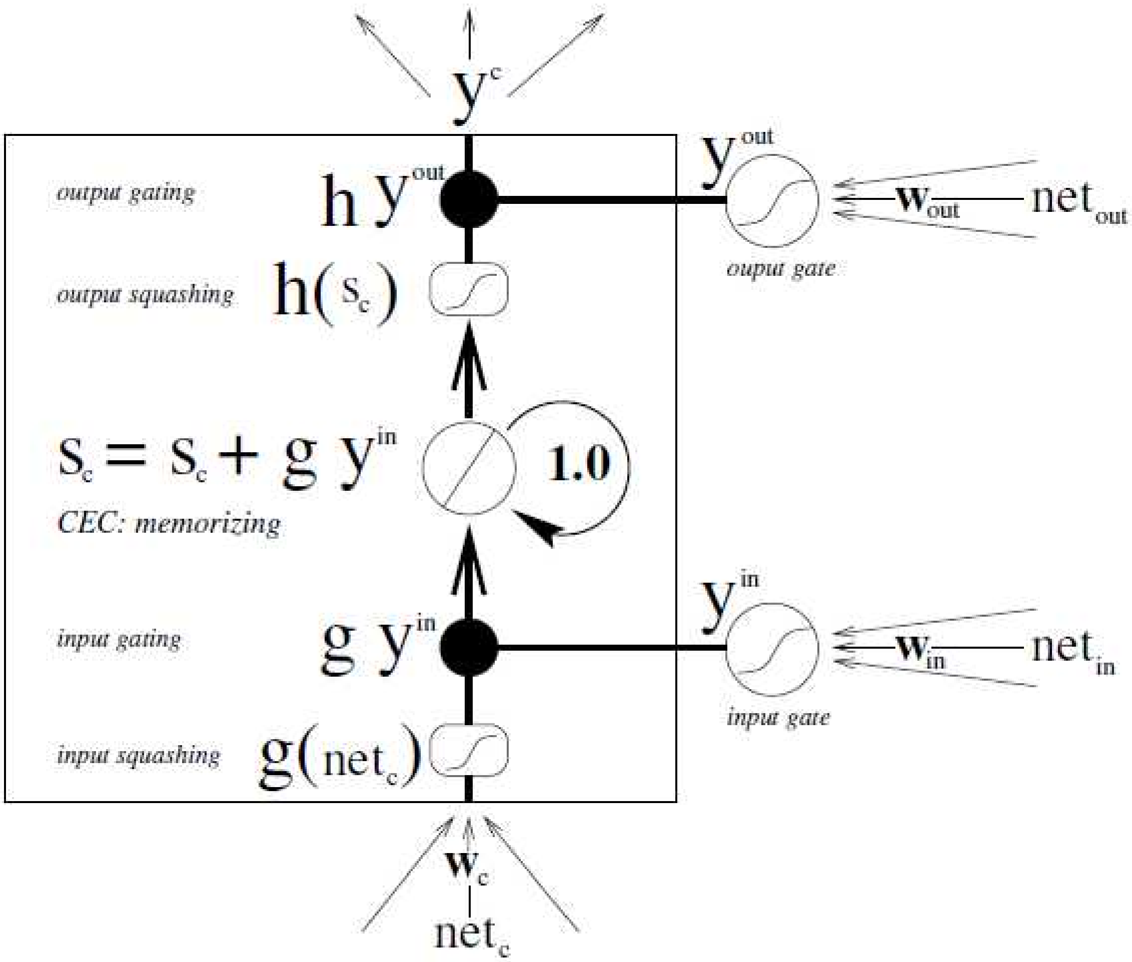


Рис.6.7. Структура ячейки памяти

2. Задается обучающее множество $\{(\mathbf{x}_\mu, \mathbf{d}_\mu) \mid \mathbf{x}_\mu \in R^{N^{(0)}}, \mathbf{d}_\mu \in R^{N^{(2)}}\}$, $\mu \in \overline{1,P}$, где $\mathbf{x}_\mu$ – μ-й обучающий входной вектор, $\mathbf{d}_\mu$ – μ-й обучающий выходной вектор, P – мощность обучающего множества. Номер текущей пары из обучающего множества $\mu = 2$.

3. Начальное вычисление выходного сигнала ячейки

$s_{jv}^{c}(n-1) = 0$, $v \in \overline{1,S_j}$, $j \in \overline{1,N^{(1)}}$.

4. Вычисление выходного сигнала для каждого слоя (прямой ход)

$y_i^{in}(n-1) = x_{\mu-1,i}$,

$y_j^{gin}(n) = f(net_j^{gin}(n))$, $j \in \overline{1,N^{(1)}}$,

$$f(s) = \frac{1}{1+e^{-s}},\ net_j^{gin}(n) = \sum_{i=1}^{N^{(0)}} w_{ij}^{in-gin}(n) y_i^{in}(n-1),$$

$$net_{jv}^{c}(n) = \sum_{i=1}^{N^{(0)}} w_{ijv}^{in-c}(n) y_i^{in}(n-1),\ v \in \overline{1,S_j},\ j \in \overline{1,N^{(1)}},$$

$s_{jv}^{c}(n) = s_{jv}^{c}(n-1) + y_j^{gin}(n) g(net_{jv}^{c}(n))$, $v \in \overline{1,S_j}$, $j \in \overline{1,N^{(1)}}$,

$g(s) = 2\tanh(s)$,

$y_j^{gout}(n) = f(net_j^{gout}(n))$, $j \in \overline{1,N^{(1)}}$,

$$f(s) = \frac{1}{1+e^{-s}},\ net_j^{gout}(n) = \sum_{i=1}^{N^{(0)}} w_{ij}^{in-gout}(n) y_i^{in}(n-1),$$

$y_{jv}^{c}(n) = y_j^{gout}(n) h(s_{jv}^{c}(n))$, $v \in \overline{1,S_j}$, $j \in \overline{1,N^{(1)}}$,

$h(s) = \tanh(s)$,

$y_j^{out}(n) = f(net_j^{out}(n))$, $j \in \overline{1,N^{(2)}}$,

$$f(s) = \frac{1}{1+e^{-s}},\ net_j^{out}(n) = \sum_{i=1}^{N^{(1)}} \sum_{v=1}^{S_i} w_{ivj}^{c-out}(n) y_{iv}^{c}(n-1).$$

5. Вычисление энергии ошибки ИНС

$$E(n) = \frac{1}{2} \sum_{i=1}^{N^{(2)}} e_i^2(n),\ e_i(n) = y_i^{out}(n) - d_{\mu-1,i}.$$

6. Настройка синаптических весов на основе обобщенного дельта правила (обратный ход)

$$w_{ij}^{c-out}(n+1)=w_{ij}^{c-out}(n)-\eta\frac{\partial E(n)}{\partial w_{ij}^{c-out}(n)},\ i\in\overline{1,N^{(1)}},\ j\in\overline{1,N^{(2)}},$$

$$w_{ij}^{in-gout}(n+1)=w_{ij}^{in-gout}(n)-\eta\frac{\partial E(n)}{\partial w_{ij}^{in-gout}(n)},\ i\in\overline{1,N^{(0)}},\ j\in\overline{1,N^{(1)}},$$

$$w_{ijv}^{in-c}(n+1)=w_{ijv}^{in-c}(n)-\eta e_{jv}^{c}(n)\frac{\partial s_{jv}^{c}(n)}{\partial w_{ijv}^{in-c}(n)},\qquad i\in\overline{1,N^{(0)}},$$

$v\in\overline{1,S_j}$, $j\in\overline{1,N^{(1)}}$,

$$w_{ij}^{in-gin}(n+1)=w_{ij}^{in-gin}(n)-\eta\sum_{v=1}^{S_j}e_{jv}^{c}(n)\frac{\partial s_{jv}^{c}(n)}{\partial w_{ij}^{in-gin}(n)},\qquad i\in\overline{1,N^{(0)}},$$

$v\in\overline{1,S_j}$, $j\in\overline{1,N^{(1)}}$,

где η – параметр, определяющий скорость обучения (при большом η обучение происходит быстрее, но увеличивается опасность получить неверное решение), $0<\eta<1$.

$$\frac{\partial E(n)}{\partial w_{ivj}^{c-out}(n)}=y_{iv}^{c}(n-1)\delta_{j}^{out}(n),$$

$$\frac{\partial E(n)}{\partial w_{ij}^{in-gout}(n)}=y_{i}^{in}(n-1)\delta_{j}^{gout}(n),$$

$$\frac{\partial s_{jv}^{c}(n)}{\partial w_{ijv}^{in-c}(n)}=\begin{cases}\dfrac{\partial s_{jv}^{c}(n-1)}{\partial w_{ijv}^{in-c}(n-1)}+y_{i}^{in}(n-1)y_{j}^{gin}(n)g'(net_{jv}^{c}(n)), & n>2\\ y_{i}^{in}(n-1)y_{j}^{gin}(n)g'(net_{jv}^{c}(n)), & n=2\end{cases},$$

$$\frac{\partial s_{jv}^{c}(n)}{\partial w_{ij}^{in-gin}(n)}=\begin{cases}\dfrac{\partial s_{jv}^{c}(n-1)}{\partial w_{ij}^{in-gin}(n-1)}+y_{j}^{in}(n-1)g(net_{jv}^{c}(n))f'(net_{j}^{gin}(n)), & n>2\\ y_{j}^{in}(n-1)g(net_{jv}^{c}(n))f'(net_{j}^{gin}(n)), & n=2\end{cases},$$

$$e_{jv}^{c}(n)=y_{j}^{out}(n)h'(s_{jv}^{c}(n))\sum_{l=1}^{N^{(2)}}w_{jvl}^{c}(n)\delta_{l}^{out}(n),$$

$$\delta_{j}^{out}(n)=f'(net_{j}^{out}(n))(y_{j}^{out}(n)-d_{\mu-1,j}),$$

$$\delta_j^{gout}(n) = f'(net_j^{gout}(n))\sum_{v=1}^{S_j} h(s_{jv}^c(n)) \sum_{l=1}^{N^{(2)}} w_{jvl}^c(n)\delta_l^{out}(n).$$

7. Проверка условия завершения

Если $(n-1) \bmod P > 0$, то $\mu = \mu + 1$, $n = n+1$, переход к 4.

Если $(n-1) \bmod P = 0$ и $\frac{1}{P}\sum_{s=1}^{P} E(n-P+s) > \varepsilon$, то $n = n+1$, переход к 2.

Если $(n-1) \bmod P = 0$ и $\frac{1}{P}\sum_{s=1}^{P} E(n-P+s) < \varepsilon$, то завершиться.

Функционирование ИНС

1. Инициализация

$n = 2$.

$s_{jv}^c(n-1) = 0$, $v \in \overline{1, S_j}$, $j \in \overline{1, N^{(1)}}$.

2. Прогноз

$y_i^{in}(n-1) = x_i$,

$$y_j^{gin}(n) = f\left(\sum_{i=1}^{N^{(0)}} w_{ij}^{in-gin} y_i^{in}(n-1)\right), \ j \in \overline{1, N^{(1)}},$$

$$net_{jv}^c(n) = \sum_{i=1}^{N^{(0)}} w_{ijv}^{in-c} y_i^{in}(n-1), \ v \in \overline{1, S_j}, \ j \in \overline{1, N^{(1)}},$$

$$s_{jv}^c(n) = s_{jv}^c(n-1) + y_j^{gin}(n) g(net_{jv}^c(n)), \ v \in \overline{1, S_j}, \ j \in \overline{1, N^{(1)}},$$

$$y_j^{gout}(n) = f\left(\sum_{i=1}^{N^{(0)}} w_{ij}^{in-gout} y_i^{in}(n-1)\right), \ j \in \overline{1, N^{(1)}},$$

$$y_{jv}^c(n) = y_j^{gout}(n) h(s_{jv}^c(n)), \ v \in \overline{1, S_j}, \ j \in \overline{1, N^{(1)}},$$

$$y_j^{out}(n) = f\left(\sum_{i=1}^{N^{(1)}} \sum_{v=1}^{S_i} w_{ivj}^{c-out} y_{iv}^c(n-1)\right), \ j \in \overline{1, N^{(2)}}.$$

Достоинства

1. Является универсальным аппроксиматором. Обеспечивает глобальную аппроксимацию нелинейного отображения входного сигнала в выходной.

2. Обеспечивает хорошее качество обобщения.

3. Автоматически определяется количество скрытых слоев (равно одному).

4. В отличие от статических ИНС позволяет осуществлять адаптивную фильтрацию, прогноз, адаптивное управление, параметрическую идентификацию модели, класификацию нестационарных сигналов.

5. В отличие от SRN (ENN), JNN, NARNN и NARMANN позволяет работать с долгосрочными нестационарными последовательностями.

Недостатки

1. Обучение происходит медленнее, чем в случае MLP, RBFNN, PNN, сети Хемминга, SOM, CPNN.

2. Отсутствует автоматическое определение числа нейронов (блоков памяти) в скрытом слое и числа ячеек в каждом блоке памяти.

3. В отличие от методов обучения SVM, градиентные методы обучения LSTM выбирают положение разделяющей гиперплоскости произвольным образом.

4. Обучение не сводится к задаче квадратичного программирования в выпуклой области, имеющей единственное решение.

5. В отличие от ART, не решает проблему пластичности-стабильности.

6.6.2. Расширенная долгократкосрочная память со шлюзом забывания

На рис.6.8 приведена расширенная долгократкосрочная память (LSTM) со шлюзом забывания [56], которая является рекуррентной двухслойной ИНС. Вместо скрытого нейрона используется блок памяти, который содержит одну или более ячеек (рис.6.9), и связан с входным, выходным и забывания шлюзами (КИХ-фильтры).

Шлюз забывания заменяет константу 1 и используется для сброса блока памяти, чтобы предотвратить неограниченный рост значения в ячейке памяти.

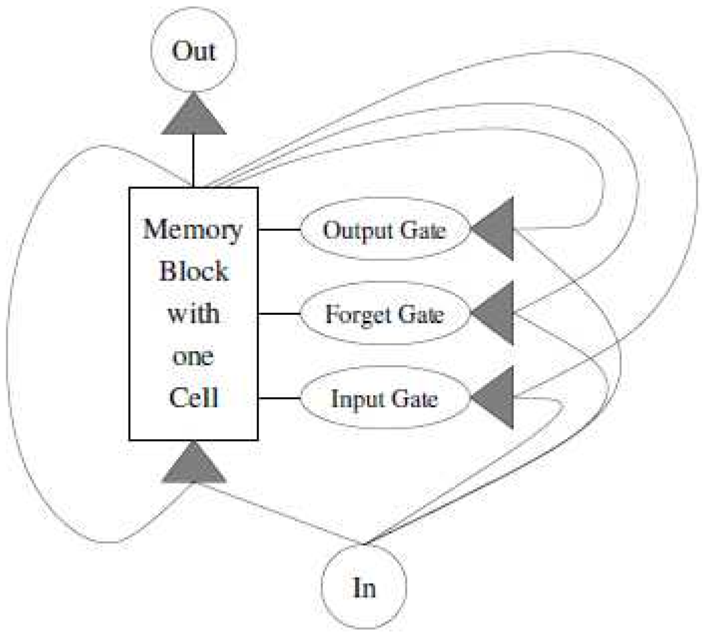


Рис.6.8. Расширенная долгократкосрочная память (LSTM) со шлюзом забывания

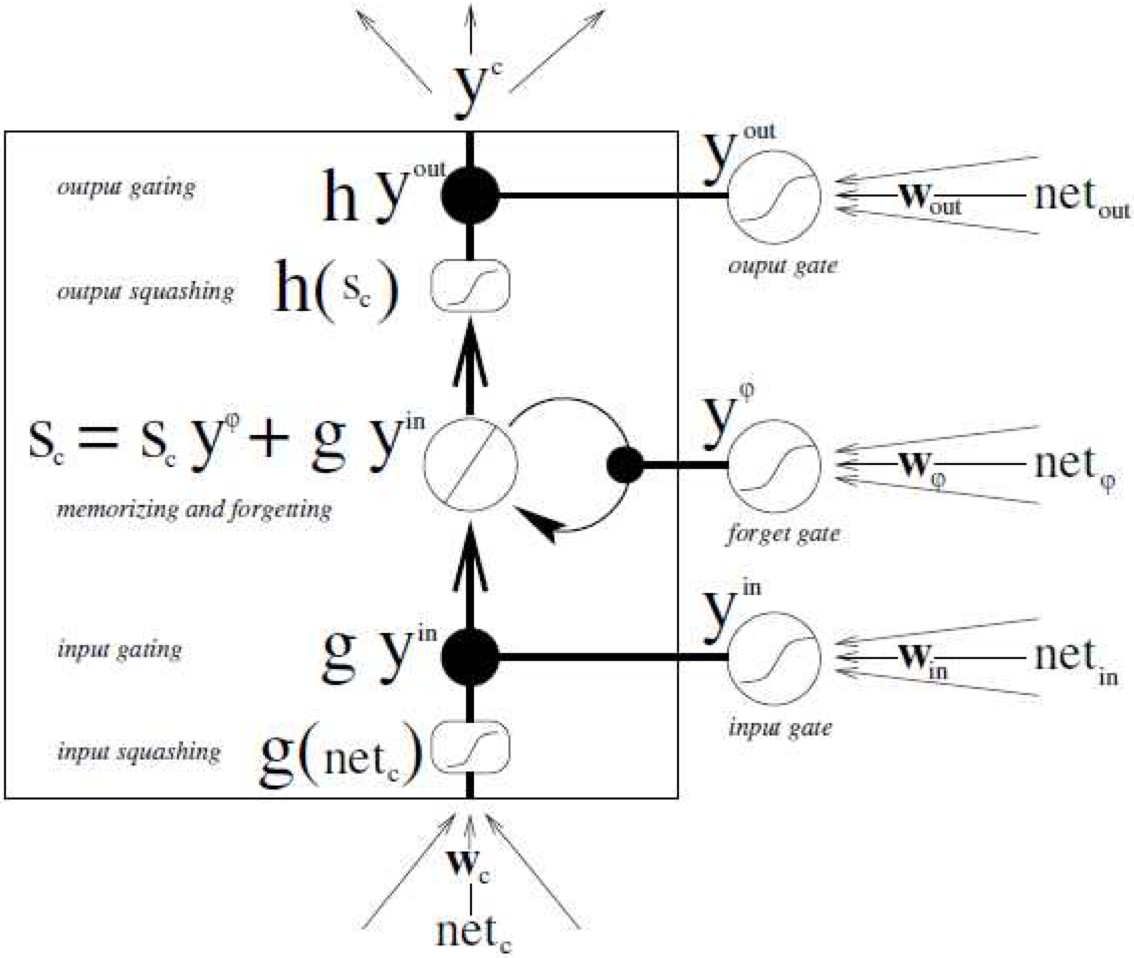


Рис.6.9. Структура ячейки памяти

Обучение расширенной LSTM со шлюзом забывания происходит на основе коррекции ошибок (обучение с учителем) в виде комбинации алгоритмов обратного распространения во времени (BPTT) и рекуррентного обучения в реальном времени (RTRL).

Обучение ИНС (комбинация алгоритмов обратного распространения во времени и рекуррентного обучения в реальном времени)

1. Номер итерации обучения $n=2$, инициализация посредством равномерного распределения на интервале (0,1) или [-0.5, 0.5] весов $w_{ij}^{in-gin}(n)$, $w_{ij}^{in-g\varphi}(n)$, $w_{ij}^{in-gout}(n)$, $w_{ij}^{in-c}(n)$, $i\in\overline{1,N^{(0)}}$, $j\in\overline{1,N^{(1)}}$, $w_{ij}^{c-out}(n)$, $i\in\overline{1,N^{(1)}}$, $j\in\overline{1,N^{(2)}}$, где $N^{(0)}$ – количество нейронов входного слоя, $N^{(1)}$ – количество нейронов скрытого слоя, $N^{(2)}$ – количество нейронов выходного слоя, S_j – количество ячеек в j-м блоке памяти.

2. Задается обучающее множество $\{(\mathbf{x}_\mu,\mathbf{d}_\mu)\,|\,\mathbf{x}_\mu\in R^{N^{(0)}},\mathbf{d}_\mu\in R^{N^{(2)}}\}$, $\mu\in\overline{1,P}$, где $\mathbf{x}_\mu$ – μ-й обучающий входной вектор, $\mathbf{d}_\mu$ – μ-й обучающий выходной вектор, P – мощность обучающего множества. Номер текущей пары из обучающего множества $\mu=2$.

3. Начальное вычисление выходного сигнала ячейки

$$s_{jv}^{c}(n-1)=0,\ v\in\overline{1,S_j},\ j\in\overline{1,N^{(1)}}.$$

4. Вычисление выходного сигнала для каждого слоя (прямой ход)

$$y_i^{in}(n-1)=x_{\mu-1,i},$$

$$y_j^{gin}(n)=f(net_j^{gin}(n)),\ j\in\overline{1,N^{(1)}},$$

$$f(s)=\frac{1}{1+e^{-s}},\ net_j^{gin}(n)=\sum_{i=1}^{N^{(0)}}w_{ij}^{in-gin}(n)y_i^{in}(n-1),$$

$$y_j^{g\varphi}(n)=f(net_j^{g\varphi}(n)),\ j\in\overline{1,N^{(1)}},$$

$$f(s)=\frac{1}{1+e^{-s}},\ net_j^{g\varphi}(n)=\sum_{i=1}^{N^{(0)}}w_{ij}^{in-g\varphi}(n)y_i^{in}(n-1),$$

$$net_{jv}^{c}(n)=\sum_{i=1}^{N^{(0)}} w_{ijv}^{in-c}(n)y_i^{in}(n-1),\ v\in\overline{1,S_j},\ j\in\overline{1,N^{(1)}},$$

$$s_{jv}^{c}(n)=y_j^{g\varphi}(n)s_{jv}^{c}(n-1)+y_j^{gin}(n)g(net_{jv}^{c}(n)),\ v\in\overline{1,S_j},\ j\in\overline{1,N^{(1)}},$$

$$g(s)=2\tanh(s),$$

$$y_j^{gout}(n)=f(net_j^{gout}(n)),\ \ j\in\overline{1,N^{(1)}},$$

$$f(s)=\frac{1}{1+e^{-s}},\ net_j^{gout}(n)=\sum_{i=1}^{N^{(0)}} w_{ij}^{in-gout}(n)y_i^{in}(n-1),$$

$$y_{jv}^{c}(n)=y_j^{gout}(n)h(s_{jv}^{c}(n)),\ v\in\overline{1,S_j},\ j\in\overline{1,N^{(1)}},$$

$$h(s)=\tanh(s),$$

$$y_j^{out}(n)=f(net_j^{out}(n)),\ \ j\in\overline{1,N^{(2)}},$$

$$f(s)=\frac{1}{1+e^{-s}},\ net_j^{out}(n)=\sum_{i=1}^{N^{(1)}}\sum_{v=1}^{S_i} w_{ivj}^{c-out}(n)y_{iv}^{c}(n-1).$$

5. Вычисление энергии ошибки ИНС

$$E(n)=\frac{1}{2}\sum_{i=1}^{N^{(2)}} e_i^2(n),\ e_i(n)=y_i^{out}(n)-d_{\mu-1,i}.$$

6. Настройка синаптических весов на основе обобщенного дельта правила (обратный ход)

$$w_{ij}^{c-out}(n+1)=w_{ij}^{c-out}(n)-\eta\frac{\partial E(n)}{\partial w_{ij}^{c-out}(n)},\ i\in\overline{1,N^{(1)}},\ \ j\in\overline{1,N^{(2)}},$$

$$w_{ij}^{in-gout}(n+1)=w_{ij}^{in-gout}(n)-\eta\frac{\partial E(n)}{\partial w_{ij}^{in-gout}(n)},\ i\in\overline{1,N^{(0)}},\ \ j\in\overline{1,N^{(1)}},$$

$$w_{ijv}^{in-c}(n+1)=w_{ijv}^{in-c}(n)-\eta e_{jv}^{c}(n)\frac{\partial s_{jv}^{c}(n)}{\partial w_{ijv}^{in-c}(n)},\qquad i\in\overline{1,N^{(0)}},$$

$v\in\overline{1,S_j},\ j\in\overline{1,N^{(1)}}$,

$$w_{ij}^{in-gin}(n+1)=w_{ij}^{in-gin}(n)-\eta\sum_{v=1}^{S_j} e_{jv}^{c}(n)\frac{\partial s_{jv}^{c}(n)}{\partial w_{ij}^{in-gin}(n)},\qquad i\in\overline{1,N^{(0)}},$$

$v\in\overline{1,S_j},\ j\in\overline{1,N^{(1)}}$,

$$w_{ij}^{in-g\varphi}(n+1) = w_{ij}^{in-g\varphi}(n) - \eta \sum_{v=1}^{S_j} e_{jv}^{c}(n) \frac{\partial s_{jv}^{c}(n)}{\partial w_{ij}^{in-g\varphi}(n)}, \qquad i \in \overline{1, N^{(0)}},$$

$v \in \overline{1, S_j}$, $j \in \overline{1, N^{(1)}}$,

где η – параметр, определяющий скорость обучения (при большом η обучение происходит быстрее, но увеличивается опасность получить неверное решение), $0 < \eta < 1$.

$$\frac{\partial E(n)}{\partial w_{ivj}^{c-out}(n)} = y_{iv}^{c}(n-1)\delta_j^{out}(n),$$

$$\frac{\partial E(n)}{\partial w_{ij}^{in-gout}(n)} = y_i^{in}(n-1)\delta_j^{gout}(n),$$

$$\frac{\partial s_{jv}^{c}(n)}{\partial w_{ijv}^{in-c}(n)} = \begin{cases} \dfrac{\partial s_{jv}^{c}(n-1)}{\partial w_{ijv}^{in-c}(n-1)} y_j^{g\varphi}(n) + y_i^{in}(n-1) y_j^{gin}(n) g'(net_{jv}^{c}(n)), & n > 2 \\ y_i^{in}(n-1) y_j^{gin}(n) g'(net_{jv}^{c}(n)), & n = 2 \end{cases},$$

$$\frac{\partial s_{jv}^{c}(n)}{\partial w_{ij}^{in-gin}(n)} = \begin{cases} \dfrac{\partial s_{jv}^{c}(n-1)}{\partial w_{ij}^{in-gin}(n-1)} y_j^{g\varphi}(n) + \\ + y_j^{in}(n-1) g(net_{jv}^{c}(n)) f'(net_j^{gin}(n)), & n > 2 \\ y_j^{in}(n-1) g(net_{jv}^{c}(n)) f'(net_j^{gin}(n)), & n = 2 \end{cases},$$

$$\frac{\partial s_{jv}^{c}(n)}{\partial w_{ij}^{in-g\varphi}(n)} = \begin{cases} \dfrac{\partial s_{jv}^{c}(n-1)}{\partial w_{ij}^{in-g\varphi}(n-1)} y_j^{g\varphi}(n) + \\ + y_j^{in}(n-1) s_{jv}^{c}(n-1) f'(net_j^{\varphi}(n)), & n > 2 \\ y_j^{in}(n-1) s_{jv}^{c}(n-1) f'(net_j^{\varphi}(n)), & n = 2 \end{cases},$$

$$e_{jv}^{c}(n) = y_j^{out}(n) h'(s_{jv}^{c}(n)) \sum_{l=1}^{N^{(2)}} w_{jvl}^{c}(n) \delta_l^{out}(n),$$

$$\delta_j^{out}(n) = f'(net_j^{out}(n))(y_j^{out}(n) - d_{\mu-1,j}),$$

$$\delta_j^{gout}(n) = f'(net_j^{gout}(n)) \sum_{v=1}^{S_j} h(s_{jv}^{c}(n)) \sum_{l=1}^{N^{(2)}} w_{jvl}^{c}(n) \delta_l^{out}(n).$$

7. Проверка условия завершения

Если $(n-1) \bmod P > 0$, то $\mu = \mu + 1$, $n = n + 1$, переход к 4.

Если $(n-1) \bmod P = 0$ и $\frac{1}{P}\sum_{s=1}^{P} E(n-P+s) > \varepsilon$, то $n = n + 1$, переход к 2.

Если $(n-1) \bmod P = 0$ и $\frac{1}{P}\sum_{s=1}^{P} E(n-P+s) < \varepsilon$, то завершиться.

Функционирование ИНС

1. Инициализация

$n = 2$.

$s_{jv}^{c}(n-1) = 0$, $v \in \overline{1, S_j}$, $j \in \overline{1, N^{(1)}}$.

2. Прогноз

$y_i^{in}(n-1) = x_i$,

$$y_j^{gin}(n) = f\left(\sum_{i=1}^{N^{(0)}} w_{ij}^{in-gin} y_i^{in}(n-1)\right), \; j \in \overline{1, N^{(1)}},$$

$$y_j^{g\varphi}(n) = f\left(\sum_{i=1}^{N^{(0)}} w_{ij}^{g\varphi} y_i^{in}(n-1)\right), \; j \in \overline{1, N^{(1)}},$$

$$net_{jv}^{c}(n) = \sum_{i=1}^{N^{(0)}} w_{ijv}^{in-c} y_i^{in}(n-1), \; v \in \overline{1, S_j}, \; j \in \overline{1, N^{(1)}},$$

$$s_{jv}^{c}(n) = y_j^{\varphi}(n) s_{jv}^{c}(n-1) + y_j^{gin}(n) g(net_{jv}^{c}(n)), \; v \in \overline{1, S_j}, \; j \in \overline{1, N^{(1)}},$$

$$y_j^{gout}(n) = f\left(\sum_{i=1}^{N^{(0)}} w_{ij}^{in-gout} y_i^{in}(n-1)\right), \; j \in \overline{1, N^{(1)}},$$

$$y_{jv}^{c}(n) = y_j^{gout}(n) h(s_{jv}^{c}(n)), \; v \in \overline{1, S_j}, \; j \in \overline{1, N^{(1)}},$$

$$y_j^{out}(n) = f\left(\sum_{i=1}^{N^{(1)}} \sum_{v=1}^{S_i} w_{ivj}^{c-out} y_{iv}^{c}(n-1)\right), \; j \in \overline{1, N^{(2)}}.$$

Достоинства

1. Является универсальным аппроксиматором. Обеспечивает глобальную аппроксимацию нелинейного отображения входного сигнала в выходной.

2. Обеспечивает хорошее качество обобщения.

3. Автоматически определяется количество скрытых слоев (равно одному).

4. В отличие от статических ИНС позволяет осуществлять адаптивную фильтрацию, прогноз, адаптивное управление, параметрическую идентификацию модели, классификацию нестационарных сигналов.

5. В отличие от SRN (ENN), JNN, NARNN и NARMANN позволяет работать с долгосрочными нестационарными последовательностями.

6. В отличие от традиционной LSTM лучше решает задачи адаптивной фильтрации, прогноза, адаптивного управления, параметрической идентификации модели, классификации нестационарных сигналов.

Недостатки

1. Обучение происходит медленнее, чем в случае MLP, RBFNN, PNN, сети Хемминга, SOM, CPNN.

2. Отсутствует автоматическое определение числа нейронов (блоков памяти) в скрытом слое и числа ячеек в каждом блоке памяти.

3. В отличие от методов обучения SVM, градиентные методы обучения LSTM выбирают положение разделяющей гиперплоскости произвольным образом.

4. Обучение не сводится к задаче квадратичного программирования в выпуклой области, имеющей единственное решение.

5. В отличие от ART, не решает проблему пластичности-стабильности.

6.6.3. Расширенная долгократкосрочная память со шлюзом забывания и связями между ячейкой и шлюзами

На рис.6.10 приведена расширенная долгократкосрочная память (LSTM) со шлюзом забывания и связями между ячейкой и шлюзами [56], которая является рекуррентной двухслойной ИНС.

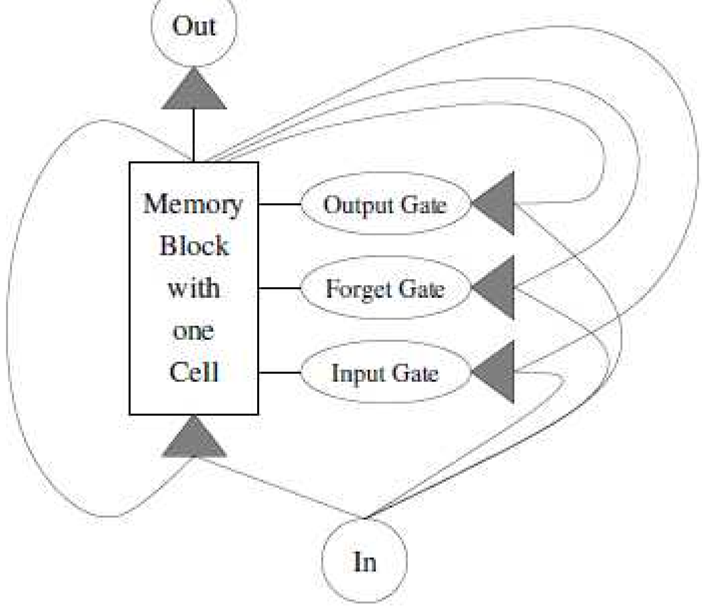


Рис.6.10. Расширенная долгая краткосрочная память (LSTM) со шлюзом забывания и связями между ячейкой и шлюзами

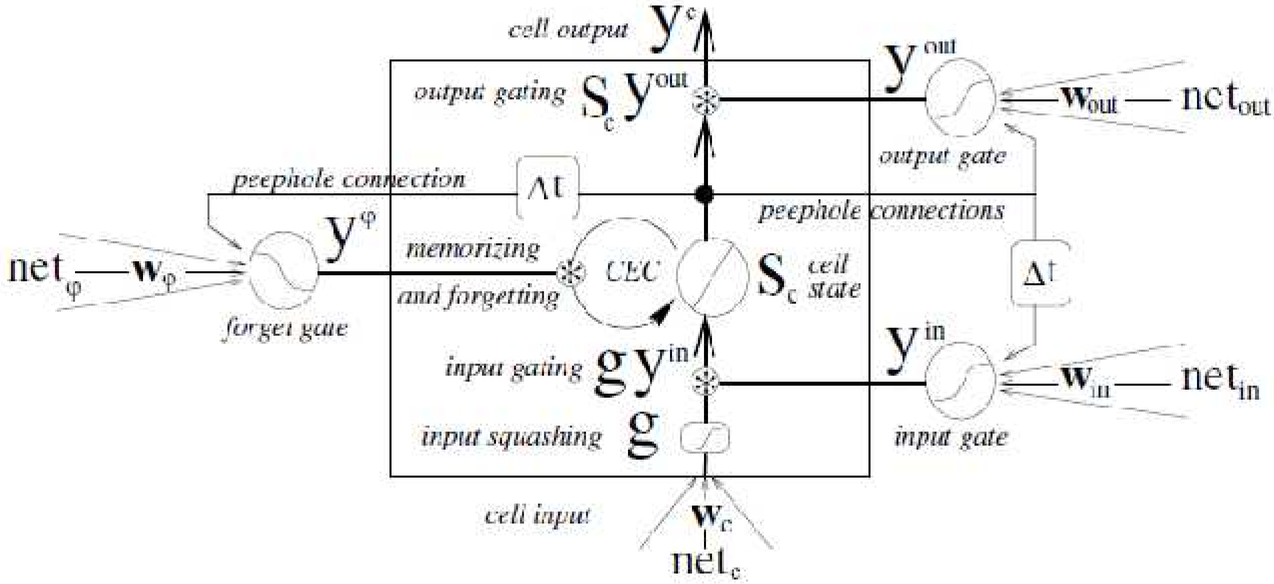


Рис.6.11. Структура ячейки памяти

Вместо скрытого нейрона используется блок памяти, который содержит одну или более ячеек (рис.6.11), и связан с входным, выходным и забывания шлюзами (КИХ-фильтры).

Связи между ячейкой и шлюзами позволят шлюзам учитывать состояние ячейки.

Обучение расширенной LSTM со шлюзом забывания и связями между ячейкой и шлюзами происходит на основе коррекции ошибок (обучение с учителем) в виде комбинации алгоритмов обратного распространения во времени (BPTT) и рекуррентного обучения в реальном времени (RTRL).

Обучение ИНС (комбинация алгоритмов обратного распространения во времени и рекуррентного обучения в реальном времени)

1. Номер итерации обучения $n = 2$, инициализация посредством равномерного распределения на интервале (0,1) или [-0.5, 0.5] весов $w_{ij}^{in-gin}(n)$, $w_{ij}^{in-g\varphi}(n)$, $w_{ij}^{in-gout}(n)$, $w_{ij}^{in-c}(n)$, $i \in \overline{1, N^{(0)}}$, $j \in \overline{1, N^{(1)}}$, $w_{ij}^{c-gin}(n)$, $w_{ij}^{c-g\varphi}(n)$, $w_{ij}^{c-gout}(n)$, $i \in \overline{1, N^{(1)}}$, $j \in \overline{1, N^{(1)}}$, $w_{ij}^{c-out}(n)$, $i \in \overline{1, N^{(1)}}$, $j \in \overline{1, N^{(2)}}$, где $N^{(0)}$ – количество нейронов входного слоя, $N^{(1)}$ – количество нейронов скрытого слоя, $N^{(2)}$ – количество нейронов выходного слоя, S_j – количество ячеек в j-м блоке памяти.

2. Задается обучающее множество $\{(\mathbf{x}_\mu, \mathbf{d}_\mu) \mid \mathbf{x}_\mu \in R^{N^{(0)}}, \mathbf{d}_\mu \in R^{N^{(2)}}\}$, $\mu \in \overline{1, P}$, где $\mathbf{x}_\mu$ – μ-й обучающий входной вектор, $\mathbf{d}_\mu$ – μ-й обучающий выходной вектор, P – мощность обучающего множества. Номер текущей пары из обучающего множества $\mu = 2$.

3. Начальное вычисление выходного сигнала ячейки

$$s_{jv}^{c}(n-1) = 0,\ v \in \overline{1, S_j},\ j \in \overline{1, N^{(1)}}.$$

4. Вычисление выходного сигнала для каждого слоя (прямой ход)

$$y_i^{in}(n-1) = x_{\mu-1,i},$$

$$y_j^{gin}(n) = f(net_j^{gin}(n)),\ j \in \overline{1, N^{(1)}},$$

$$net_j^{gin}(n) = \sum_{i=1}^{N^{(0)}} w_{ij}^{in-gin}(n) y_i^{in}(n-1) + \sum_{v=1}^{S_j} w_{jvj}^{c-gin}(n) s_{jv}^{c}(n-1),$$

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

$$y_j^{g\varphi}(n)=f(net_j^{g\varphi}(n)),\ j\in\overline{1,N^{(1)}},$$

$$net_j^{g\varphi}(n)=\sum_{i=1}^{N^{(0)}} w_{ij}^{in-g\varphi}(n)y_i^{in}(n-1)+\sum_{v=1}^{S_j} w_{jvj}^{c-g\varphi}(n)s_{jv}^{c}(n-1),$$

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

$$net_{jv}^{c}(n)=\sum_{i=1}^{N^{(0)}} w_{ijv}^{in-c}(n)y_i^{in}(n-1),\ v\in\overline{1,S_j},\ j\in\overline{1,N^{(1)}},$$

$$s_{jv}^{c}(n)=y_j^{g\varphi}(n)s_{jv}^{c}(n-1)+y_j^{gin}(n)g(net_{jv}^{c}(n)),\ v\in\overline{1,S_j},\ j\in\overline{1,N^{(1)}},$$

$$g(s)=2\tanh(s),$$

$$y_j^{gout}(n)=f(net_j^{gout}(n),\ j\in\overline{1,N^{(1)}},$$

$$net_j^{gout}(n)=\sum_{i=1}^{N^{(0)}} w_{ij}^{in-gout}(n)y_i^{in}(n-1)+\sum_{v=1}^{S_j} w_{jvj}^{c-gout}(n)s_{jv}^{c}(n-1),$$

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

$$y_{jv}^{c}(n)=y_j^{gout}(n)s_{jv}^{c}(n),\ v\in\overline{1,S_j},\ j\in\overline{1,N^{(1)}},$$

$$y_j^{out}(n)=f(net_j^{out}(n)),\ j\in\overline{1,N^{(2)}},$$

$$f(s)=\frac{1}{1+e^{-s}},\ net_j^{out}(n)=\sum_{i=1}^{N^{(1)}}\sum_{v=1}^{S_i} w_{ivj}^{c-out}(n)y_{iv}^{c}(n-1).$$

5. Вычисление энергии ошибки ИНС

$$E(n)=\frac{1}{2}\sum_{i=1}^{N^{(2)}} e_i^2(n),\ e_i(n)=y_i^{out}(n)-d_{\mu-1,i}.$$

6. Настройка синаптических весов на основе обобщенного дельта правила (обратный ход)

$$w_{ij}^{c-out}(n+1)=w_{ij}^{c-out}(n)-\eta\frac{\partial E(n)}{\partial w_{ij}^{c-out}(n)},\ i\in\overline{1,N^{(1)}},\ j\in\overline{1,N^{(2)}},$$

$$w_{ij}^{in-gout}(n+1)=w_{ij}^{in-gout}(n)-\eta\frac{\partial E(n)}{\partial w_{ij}^{in-gout}(n)},\ i\in\overline{1,N^{(0)}},\ j\in\overline{1,N^{(1)}},$$

$$w_{ijv}^{in-c}(n+1)=w_{ijv}^{in-c}(n)-\eta e_{jv}^{c}(n)\frac{\partial s_{jv}^{c}(n)}{\partial w_{ijv}^{in-c}(n)}, \qquad i\in\overline{1,N^{(0)}},$$

$v\in\overline{1,S_j}$, $j\in\overline{1,N^{(1)}}$,

$$w_{ij}^{in-gin}(n+1)=w_{ij}^{in-gin}(n)-\eta\sum_{v=1}^{S_j}e_{jv}^{c}(n)\frac{\partial s_{jv}^{c}(n)}{\partial w_{ij}^{in-gin}(n)}, \qquad i\in\overline{1,N^{(0)}},$$

$v\in\overline{1,S_j}$, $j\in\overline{1,N^{(1)}}$,

$$w_{ij}^{in-g\varphi}(n+1)=w_{ij}^{in-g\varphi}(n)-\eta\sum_{v=1}^{S_j}e_{jv}^{c}(n)\frac{\partial s_{jv}^{c}(n)}{\partial w_{ij}^{in-g\varphi}(n)}, \qquad i\in\overline{1,N^{(0)}},$$

$v\in\overline{1,S_j}$, $j\in\overline{1,N^{(1)}}$,

$$w_{jvj}^{c-gout}(n+1)=w_{jvj}^{c-gout}(n)-\eta\frac{\partial E(n)}{\partial w_{ivj}^{c-gout}(n)},\ v\in\overline{1,S_j},\ j\in\overline{1,N^{(1)}},$$

$$w_{jvj}^{c-gin}(n+1)=w_{jvj}^{c-gin}(n)-\eta\sum_{z=1}^{S_j}e_{jz}^{c}(n)\frac{\partial s_{jv}^{c}(n)}{\partial w_{jzj}^{c-gin}(n)},$$

$v\in\overline{1,S_j}$, $j\in\overline{1,N^{(1)}}$,

$$w_{jvj}^{c-g\varphi}(n+1)=w_{jvj}^{c-g\varphi}(n)-\eta\sum_{z=1}^{S_j}e_{jz}^{c}(n)\frac{\partial s_{jv}^{c}(n)}{\partial w_{jzj}^{c-g\varphi}(n)},\ v\in\overline{1,S_j},\ j\in\overline{1,N^{(1)}},$$

где η – параметр, определяющий скорость обучения (при большом η обучение происходит быстрее, но увеличивается опасность получить неверное решение), $0<\eta<1$.

$$\frac{\partial E(n)}{\partial w_{ivj}^{c-out}(n)}=y_{iv}^{c}(n-1)\delta_j^{out}(n),$$

$$\frac{\partial E(n)}{\partial w_{ij}^{in-gout}(n)}=y_i^{in}(n-1)\delta_j^{gout}(n),$$

$$\frac{\partial s_{jv}^{c}(n)}{\partial w_{ijv}^{in-c}(n)}=\begin{cases}\dfrac{\partial s_{jv}^{c}(n-1)}{\partial w_{ijv}^{in-c}(n-1)}y_j^{g\varphi}(n)+y_i^{(0)}(n-1)y_j^{in}(n)g'(net_{jv}^{c}(n)), & n>2\\ y_i^{(0)}(n-1)y_j^{in}(n)g'(net_{jv}^{c}(n)), & n=2\end{cases},$$

$$\frac{\partial s_{jv}^{c}(n)}{\partial w_{ij}^{in-gin}(n)} = \begin{cases} \dfrac{\partial s_{jv}^{c}(n-1)}{\partial w_{ij}^{in-gin}(n-1)} y_j^{g\varphi}(n) + \\ + y_j^{in}(n-1) g(net_{jv}^{c}(n)) f'(net_j^{gin}(n)), & n > 2 \\ y_j^{in}(n-1) g(net_{jv}^{c}(n)) f'(net_j^{gin}(n)), & n = 2 \end{cases},$$

$$\frac{\partial s_{jv}^{c}(n)}{\partial w_{ij}^{in-g\varphi}(n)} = \begin{cases} \dfrac{\partial s_{jv}^{c}(n-1)}{\partial w_{ij}^{in-g\varphi}(n-1)} y_j^{\varphi}(n) + \\ + y_j^{in}(n-1) s_{jv}^{c}(n-1) f'(net_j^{\varphi}(n)), & n > 2 \\ y_j^{in}(n-1) s_{jv}^{c}(n-1) f'(net_j^{\varphi}(n)), & n = 2 \end{cases},$$

$$\frac{\partial E(n)}{\partial w_{jvj}^{c-gout}(n)} = s_{jv}^{c}(n-1)\delta_j^{gout}(n),$$

$$\frac{\partial s_{jv}^{c}(n)}{\partial w_{jzj}^{c-gin}(n)} = \begin{cases} \dfrac{\partial s_{jv}^{c}(n-1)}{\partial w_{izj}^{c-gin}(n-1)} y_j^{g\varphi}(n) + \\ + s_{jz}^{c}(n-1) f'(net_j^{gin}(n)) g(net_{jv}^{c}(n)), & n > 2 \\ s_{jz}^{c}(n-1) f'(net_j^{gin}(n)) g(net_{jv}^{c}(n)), & n = 2 \end{cases},$$

$$\frac{\partial s_{jv}^{c}(n)}{\partial w_{jzj}^{c-g\varphi}(n)} = \begin{cases} \dfrac{\partial s_{jv}^{c}(n-1)}{\partial w_{jzj}^{c-g\varphi}(n-1)} y_j^{\varphi}(n) + \\ + s_{jv}^{c}(n-1) s_{jz}^{c}(n-1) f'(net_j^{\varphi}(n)), & n > 2 \\ s_{jv}^{c}(n-1) s_{jz}^{c}(n-1) f'(net_j^{\varphi}(n)), & n = 2 \end{cases},$$

$$e_{jv}^{c}(n) = y_j^{gout}(n) \sum_{l=1}^{N^{(2)}} w_{jvl}^{c-out}(n)\delta_l^{out}(n),$$

$$\delta_j^{out}(n) = f'(net_j^{out}(n))(y_j^{out}(n) - d_{\mu-1,j}),$$

$$\delta_j^{gout}(n) = f'(net_j^{gout}(n)) \sum_{v=1}^{S_j} s_{jv}^{c}(n) \sum_{l=1}^{N^{(2)}} w_{jvl}^{c-out}(n)\delta_l^{out}(n).$$

7. Проверка условия завершения

Если $(n-1) \bmod P > 0$, то $\mu = \mu + 1$, $n = n + 1$, переход к 4.

Если $(n-1) \bmod P = 0$ и $\frac{1}{P}\sum_{s=1}^{P} E(n-P+s) > \varepsilon$, то $n = n+1$, переход к 2.

Если $(n-1) \bmod P = 0$ и $\frac{1}{P}\sum_{s=1}^{P} E(n-P+s) < \varepsilon$, то завершиться.

Функционирование ИНС

1. Инициализация

$n = 2$.

$s_{jv}^{c}(n-1) = 0$, $v \in \overline{1, S_j}$, $j \in \overline{1, N^{(1)}}$.

2. Прогноз

$y_i^{in}(n-1) = x_i$,

$$y_j^{gin}(n) = f\left(\sum_{i=1}^{N^{(0)}} w_{ij}^{in-gin} y_i^{in}(n-1) + \sum_{v=1}^{S_j} w_{jvj}^{c-gin} s_{jv}^{c}(n-1)\right),\ j \in \overline{1, N^{(1)}},$$

$$y_j^{g\varphi}(n) = f\left(\sum_{i=1}^{N^{(0)}} w_{ij}^{in-g\varphi} y_i^{in}(n-1) + \sum_{v=1}^{S_j} w_{jvj}^{c-g\varphi} s_{jv}^{c}(n-1)\right),\ j \in \overline{1, N^{(1)}},$$

$$net_{jv}^{c}(n) = \sum_{i=1}^{N^{(0)}} w_{ijv}^{in-c} y_i^{in}(n-1),\ v \in \overline{1, S_j},\ j \in \overline{1, N^{(1)}},$$

$$s_{jv}^{c}(n) = y_j^{g\varphi}(n) s_{jv}^{c}(n-1) + y_j^{gin}(n) g(net_{jv}^{c}(n)),\ v \in \overline{1, S_j},\ j \in \overline{1, N^{(1)}},$$

$$y_j^{gout}(n) = f\left(\sum_{i=1}^{N^{(0)}} w_{ij}^{in-gout}(n) y_i^{in}(n-1) + \sum_{v=1}^{S_j} w_{jvj}^{c-gout}(n) s_{jv}^{c}(n-1)\right),$$

$j \in \overline{1, N^{(1)}}$,

$$y_{jv}^{c}(n) = y_j^{gout}(n) s_{jv}^{c}(n),\ v \in \overline{1, S_j},\ j \in \overline{1, N^{(1)}},$$

$$y_j^{out}(n) = f\left(\sum_{i=1}^{N^{(1)}} \sum_{v=1}^{S_i} w_{ivj}^{c-out}(n) y_{iv}^{c}(n-1)\right),\ j \in \overline{1, N^{(2)}}.$$

Достоинства

1. Является универсальным аппроксиматором. Обеспечивает глобальную аппроксимацию нелинейного отображения входного сигнала в выходной.

2. Обеспечивает хорошее качество обобщения.

3. Автоматически определяется количество скрытых слоев (равно одному).

4. В отличие от статических ИНС позволяет осуществлять адаптивную фильтрацию, прогноз, адаптивное управление, параметрическую идентификацию модели, классификацию нестационарных сигналов.

5. В отличие от SRN (ENN), JNN, NARNN, NARMANN позволяет работать с долгосрочными нестационарными последовательностями.

6. В отличие от традиционной LSTM лучше решает задачи адаптивной фильтрации, прогноза, адаптивного управления, параметрической идентификации модели, классификации нестационарных сигналов.

Недостатки

1. Обучение происходит медленнее, чем в случае MLP, RBFNN, PNN, сети Хемминга, SOM, CPNN.

2. Отсутствует автоматическое определение числа нейронов (блоков памяти) в скрытом слое и числа ячеек в каждом блоке памяти.

3. В отличие от методов обучения SVM, градиентные методы обучения LSTM выбирают положение разделяющей гиперплоскости произвольным образом.

4. Обучение не сводится к задаче квадратичного программирования в выпуклой области, имеющей единственное решение.

5. В отличие от ART, не решает проблему пластичности-стабильности.

6.7. Сеть эхо состояния

На рис. 6.12 приведена сеть эхо состояния (ESN) [57], которая является неполносвязной рекуррентной двухслойной ИНС и построена на базе MLP. Классы разделяются гиперплоскостями.

Свойство эхо состояния заключается в том, что если ИНС была запущена в течение долгого времени, то текущее состояние ИНС однозначно определяется историей обучающих входа и выхода (т.е. существует эхо функция, которая отображает историю входа/выхода в текущем состоянии).

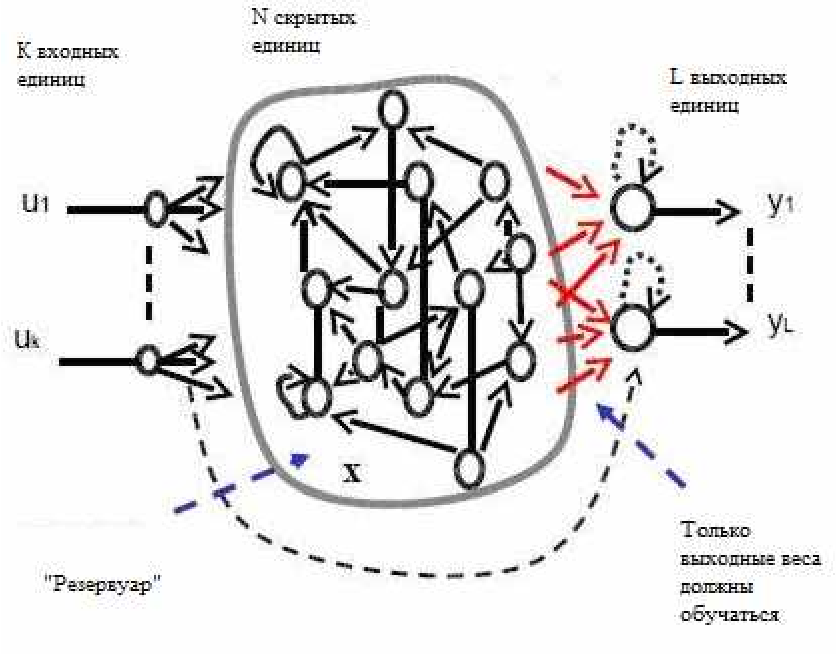


Рис. 6.12. Сеть эхо состояния (ESN)

Для ESN используется обучение на основе коррекции ошибок (обучение с учителем), при этом чаще всего применяется алгоритм обратного распространения (BP). Это итеративный градиентный алгоритм обучения, который обеспечивает минимизацию среднеквадратичной ошибки.

Обучение ИНС (алгоритм обратного распространения)

1. Номер итерации обучения $n=1$, инициализация посредством равномерного распределения на интервале (0,1) или [-0.5, 0.5] смещений (порогов) $b_j^{(k)}(n)$ и весов $w_{ij}^{(k)}(n)$, задание множества допустимых весов $A=\{(k,i,j)\}$, $i \in \overline{1, N^{(0)}+N^{(1)}+N^{(2)}}$, $j \in \overline{1, N^{(k)}}$, $k \in \overline{1,2}$, где $N^{(0)}$ – количество нейронов входного слоя, $N^{(1)}$ – количество нейронов скрытого слоя, $N^{(2)}$ – количество нейронов выходного слоя.

2. Определить веса $w_{ij}^{(1)}(n)$, $i \in \overline{N^{(0)}+1, N^{(0)}+N^{(1)}}$, $j \in \overline{1, N^{(1)}}$, следующим образом:

2.1. Составить из этих весов матрицу $W=[w_{ij}]$, $i, j \in \overline{1, N^{(1)}}$.

2.2. Определить матрицу $\widehat{W}$ в виде

$$\widehat{W} = \alpha \frac{W}{\max\limits_{j \in \overline{1, N^{(1)}}} \{|\lambda_j|\}},$$

где α – спектральный радиус матрицы $\widehat{W}$ (при большом α обучение происходит быстрее, но уменьшается долгая краткосрочная память), $0 < \alpha < 1$,

λ_j – собственные значения матрицы W.

2.3. Присвоить весам $w_{ij}^{(1)}(n)$, $i \in \overline{N^{(0)}+1, N^{(0)}+N^{(1)}}$, $j \in \overline{1, N^{(1)}}$, значения соответствующих элементов матрицы $\widehat{W}$.

3. Задается обучающее множество $\{(\mathbf{x}_\mu, \mathbf{d}_\mu) \mid \mathbf{x}_\mu \in R^{N^{(0)}}, \mathbf{d}_\mu \in R^{N^{(2)}}\}$, $\mu \in \overline{1, P}$, где $\mathbf{x}_\mu$ – μ-й обучающий входной вектор, $\mathbf{d}_\mu$ – μ-й обучающий выходной вектор, P – мощность обучающего множества. Номер текущей пары из обучающего множества $\mu = 1$.

4. Начальное вычисление выходного сигнала первого слоя

$y_i^{(1)}(n-1) = 0$, $i \in \overline{1, N^{(1)}}$.

5. Вычисление выходного сигнала для каждого слоя (прямой ход)

$y_i^{(0)}(n) = x_{\mu i}$,

$y_j^{(1)}(n) = f^{(1)}(s_j^{(1)}(n))$,

$$s_j^{(1)}(n) = \sum_{i=0}^{N^{(0)}} w_{ij}^{(1)}(n) y_i^{(0)}(n) + \sum_{i=N^{(0)}+1}^{N^{(0)}+N^{(1)}} w_{ij}^{(1)}(n) y_{i-N^{(0)}}^{(1)}(n-1) +$$

$$+ \sum_{i=N^{(0)}+N^{(1)}+1}^{N^{(0)}+N^{(1)}+N^{(2)}} w_{ij}^{(1)}(n) y_{i-(N^{(0)}+N^{(1)})}^{(2)}(n-1),\ j \in \overline{1, N^{(1)}},$$

$y_j^{(2)}(n) = f^{(2)}(s_j^{(2)}(n))$,

$$s_j^{(1)}(n) = \sum_{i=0}^{N^{(0)}} w_{ij}^{(2)}(n) y_i^{(0)}(n) + \sum_{i=N^{(0)}+1}^{N^{(0)}+N^{(1)}} w_{ij}^{(2)}(n) y_{i-N^{(0)}}^{(1)}(n) +$$

$$+ \sum_{i=N^{(0)}+N^{(1)}+1}^{N^{(0)}+N^{(1)}+N^{(2)}} w_{ij}^{(2)}(n) y_{i-(N^{(0)}+N^{(1)})}^{(2)}(n-1),\ j \in \overline{1, N^{(2)}},$$

где $N^{(k)}$ – число нейронов в *k*-м слое, *k* – номер слоя, $w_{ij}^{(k)}(n)$ – вес связи от *i*-го нейрона к *j*-му нейрону на *k*-м слое в момент времени n, $y_j^{(k)}(n)$ – выход *j*-го нейрона на *k*-м слое, $f^{(k)}$ – функция активация sнейронов *k*-го слоя (обычно $f^{(k)}(s) = \tanh(s)$ или $f^{(k)}(s) = s$).

Считается, что $w_{0j}^{(k)}(n)=b_j^{(k)}(n), y_0^{(k-1)}(n)=1$.

6. Вычисление энергии ошибки ИНС

$$E(n)=\frac{1}{2}\sum_{j=1}^{N^{(2)}} e_j^2(n),\ e_j(n)=y_j^{(2)}(n)-d_{\mu j},$$

7. Настройка синаптических весов на основе обобщенного дельта правила (обратный ход)

$$w_{ij}^{(k)}(n+1)=\begin{cases} w_{ij}^{(k)}(n)-\eta\dfrac{\partial E(n)}{\partial w_{ij}^{(k)}(n)}, & (k,i,j)\in A\wedge(k=2\vee \\ & \vee\, i\notin\overline{N^{(0)}+1,N^{(0)}+N^{(1)}}) \\ w_{ij}^{(k)}(n), & (k,i,j)\notin A\vee(k=1\wedge \\ & \wedge\, i\in\overline{N^{(0)}+1,N^{(0)}+N^{(1)}}) \end{cases},$$

$i\in\overline{1,N^{(0)}+N^{(1)}+N^{(2)}}$, $j\in\overline{1,N^{(k)}}$, $k\in\overline{1,2}$,

где η – параметр, определяющий скорость обучения (при большом η обучение происходит быстрее, но увеличивается опасность получить неверное решение), $0<\eta<1$.

$$\frac{\partial E(n)}{\partial w_{ij}^{(2)}(n)}=\begin{cases} y_i^{(2)}(n)g_j^{(2)}(n), & i\ge N^{(0)}+N^{(1)}+1 \\ y_i^{(1)}(n)g_j^{(2)}(n), & N^{(0)}<i<N^{(0)}+N^{(1)}+1 \\ y_i^{(0)}(n)g_j^{(2)}(n), & i\le N^{(0)} \end{cases},$$

$$\frac{\partial E(n)}{\partial w_{ij}^{(1)}(n)}=\begin{cases} y_i^{(2)}(n)g_j^{(1)}(n), & i\ge N^{(0)}+N^{(1)}+1 \\ y_i^{(0)}(n)g_j^{(1)}(n), & i\le N^{(0)} \end{cases},$$

$$g_j^{(k)}(n)=\begin{cases} f'^{(2)}(s_j^{(2)}(n))(y_j^{(2)}(n)-d_{\mu j}), & k=2 \\ f'^{(1)}(s_j^{(1)}(n))\displaystyle\sum_{l=0}^{N^{(0)}+N^{(1)}+N^{(2)}} w_{jl}^{(2)}(n)g_l^{(2)}(n), & k=1 \end{cases}.$$

8. Проверка условия завершения

Если $n \bmod P>0$, то $\mu=\mu+1$, $n=n+1$, переход к 5.

Если $n \bmod P=0$ и $\dfrac{1}{P}\sum_{s=1}^{P}E(n-P+s)>\varepsilon$, то $n=n+1$, переход к 3.

Если $n \bmod P=0$ и $\dfrac{1}{P}\sum_{s=1}^{P}E(n-P+s)<\varepsilon$, то завершиться.

Функционирование ИНС

1. Инициализация

$n=1$.

$y_i^{(1)}(n-1)=0$, $i\in\overline{1,N^{(1)}}$.

2. Прогноз

$y_i^{(0)}(n)=x_i$,

$y_j^{(1)}(n)=f^{(1)}(s_j^{(1)}(n))$,

$$s_j^{(1)}(n)=\sum_{i=0}^{N^{(0)}} w_{ij}^{(1)} y_i^{(0)}(n)+\sum_{i=N^{(0)}+1}^{N^{(0)}+N^{(1)}} w_{ij}^{(1)} y_{i-N^{(0)}}^{(1)}(n-1)+$$

$$+\sum_{i=N^{(0)}+N^{(1)}+1}^{N^{(0)}+N^{(1)}+N^{(2)}} w_{ij}^{(1)} y_{i-(N^{(0)}+N^{(1)})}^{(2)}(n-1),\ j\in\overline{1,N^{(1)}},$$

$y_j^{(2)}(n)=f^{(2)}(s_j^{(2)}(n))$,

$$s_j^{(1)}(n)=\sum_{i=0}^{N^{(0)}} w_{ij}^{(2)} y_i^{(0)}(n)+\sum_{i=N^{(0)}+1}^{N^{(0)}+N^{(1)}} w_{ij}^{(2)} y_{i-N^{(0)}}^{(1)}(n)+$$

$$+\sum_{i=N^{(0)}+N^{(1)}+1}^{N^{(0)}+N^{(1)}+N^{(2)}} w_{ij}^{(2)} y_{i-(N^{(0)}+N^{(1)})}^{(2)}(n-1),\ j\in\overline{1,N^{(2)}}.$$

Достоинства

1. Является универсальным аппроксиматором. Обеспечивает глобальную аппроксимацию нелинейного отображения входного сигнала в выходной.

2. Обеспечивает хорошее качество обобщения.

3. Автоматически определяется количество скрытых слоев (равно одному).

4. В отличие от статических ИНС позволяет осуществлять адаптивную фильтрацию, прогноз, адаптивное управление, параметрическую идентификацию модели, классификацию нестационарных сигналов.

5. В отличие от SRN (ENN), JNN, NARNN, NARMANN позволяет работать с долгосрочными нестационарными последовательностями.

Недостатки

1. Обучение происходит медленнее, чем в случае MLP, RBFNN, PNN, сети Хемминга, SOM, CPNN.

2. Отсутствует автоматическое определение числа нейронов в скрытом слое.

3. В отличие от методов обучения SVM, градиентные методы обучения ESN выбирают положение разделяющей гиперплоскости произвольным образом.

4. Обучение не сводится к задаче квадратичного программирования в выпуклой области, имеющей единственное решение.

5. В отличие от ART, не решает проблему пластичности-стабильности.

6. Отсутствует автоматическое определение множества допустимых весов.

6.8. Двунаправленная рекуррентная нейросеть

На рис. 6.13 приведена двунаправленная рекуррентная нейросеть (BRNN) [58], которая является рекуррентной двухслойной ИНС и построена на базе SRN (ENN). Классы разделяются гиперплоскостями.

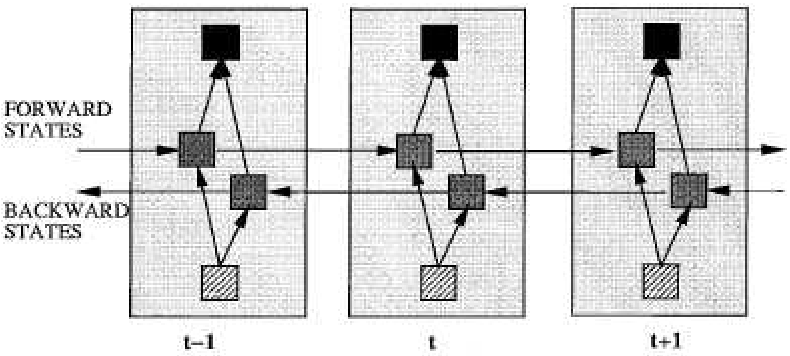


Рис. 6.13. Двунаправленная рекуррентная нейросеть (BRNN)

Для BRNN используется обучение на основе коррекции ошибок (обучение с учителем), при этом чаще всего применяется алгоритм обратного распространения (BP). Это итеративный градиентный алгоритм обучения, который обеспечивает минимизацию среднеквадратичной ошибки.

Обучение ИНС (алгоритм обратного распространения)

1. Номер итерации обучения $n=1$, инициализация посредством равномерного распределения на интервале (0,1) или [-0.5, 0.5] смещений (порогов) $b1_j^{(1)}(n)$, $b2_j^{(1)}(n)$, $b_j^{(2)}(n)$ и весов $w1_{ij}^{(1)}(n)$, $w2_{ij}^{(1)}(n)$, $i\in\overline{1,N^{(0)}}$, $j\in\overline{1,N^{(1)}}$, $w1_{ij}^{(2)}(n)$, $w2_{ij}^{(2)}(n)$, $i\in\overline{1,N^{(1)}}$, $j\in\overline{1,N^{(2)}}$, где $N^{(0)}$ – количество нейронов входного слоя, $N^{(1)}$ – количество нейронов скрытого слоя, $N^{(2)}$ – количество нейронов выходного слоя.

2. Задается обучающее множество $\{(\mathbf{x}_\mu,\mathbf{d}_\mu)\,|\,\mathbf{x}_\mu\in R^{N^{(0)}},\mathbf{d}_\mu\in R^{N^{(L)}}\}$, $\mu\in\overline{1,P}$, где $\mathbf{x}_\mu$ – μ-й обучающий входной вектор, $\mathbf{d}_\mu$ – μ-й обучающий выходной вектор, P – мощность обучающего множества. Номер текущей пары из обучающего множества $\mu=1$. $x_{0i}=0$, $x_{P+1,i}=0$, $i\in\overline{1,N^{(0)}}$.

3. Вычисление выходного сигнала для каждого слоя (прямой ход)

$$y_i^{(0)}(n)=x_{\mu i},\ i\in\overline{1,N^{(0)}},$$

$$y1_i^{(1)}(n-1)=x_{\mu-1,i},\ i\in\overline{1,N^{(0)}},$$

$$y1_i^{(1)}(n+1)=x_{\mu+1,i},\ i\in\overline{1,N^{(0)}},$$

$$y1_j^{(1)}(n)=f^{(1)}(s1_j^{(1)}(n)),\ j\in\overline{1,N^{(1)}},$$

$$s1_j^{(1)}(n)=\sum_{i=0}^{N^{(0)}}w1_{ij}^{(1)}(n)y_i^{(0)}(n)+\sum_{i=N^{(0)}+1}^{N^{(0)}+N^{(1)}}w1_{ij}^{(1)}(n)y1_{i-N^{(0)}}^{(1)}(n-1),$$

$$y2_j^{(1)}(n)=f^{(1)}(s2_j^{(1)}(n)),\ j\in\overline{1,N^{(1)}},$$

$$s2_j^{(1)}(n)=\sum_{i=0}^{N^{(0)}}w2_{ij}^{(1)}(n)y_i^{(0)}(n)+\sum_{i=N^{(0)}+1}^{N^{(0)}+N^{(1)}}w2_{ij}^{(1)}(n)y2_{i-N^{(0)}}^{(1)}(n+1),$$

$$y_j^{(2)}(n)=f^{(2)}(s_j^{(2)}(n)),\ j\in\overline{1,N^{(2)}},$$

$$s_j^{(2)}(n)=\sum_{i=0}^{N^{(1)}}w1_{ij}^{(2)}y1_i^{(1)}(n)+\sum_{i=1}^{N^{(1)}}w2_{ij}^{(2)}y2_i^{(1)}(n).$$

где $N^{(k)}$ – число нейронов в k-м слое, k – номер слоя, $w1_{ij}^{(k)}(n)$, $w1_{ij}^{(2)}(n)$ – вес связи от i-го нейрона к j-му нейрону на k-м слое в момент времени n, $y_j^{(k)}(n)$ – выход j-го нейрона на k-м слое, $f^{(k)}$ – функция активация нейронов k-го слоя (обычно $f^{(1)}(s) = \tanh(s)$, $f^{(2)}(s) = \mathrm{sigm}(s)$ или $f^{(2)}(s) = \mathrm{softmax}(s)$).

Считается, что $w1_{0j}^{(1)}(n) = b1_j^{(1)}(n), y_0^{(0)}(n) = 1$, $w2_{0j}^{(1)}(n) = b2_j^{(1)}(n), y_0^{(0)}(n) = 1$, $w1_{0j}^{(2)}(n) = b_j^{(2)}(n), y1_0^{(1)}(n) = 1$.

4. Вычисление энергии ошибки ИНС

$$E(n) = \frac{1}{2}\sum_{j=1}^{N^{(2)}} e_j^2(n),\; e_j(n) = y_j^{(2)}(n) - d_{\mu j},$$

5. Настройка синаптических весов на основе обобщенного дельта правила (обратный ход)

$$w_{ij}^{(k)}(n+1) = w_{ij}^{(k)}(n) - \eta\frac{\partial E(n)}{\partial w_{ij}^{(k)}(n)},$$

где η – параметр, определяющий скорость обучения (при большом η обучение происходит быстрее, но увеличивается опасность получить неверное решение), $0 < \eta < 1$.

$$\frac{\partial E(n)}{\partial w1_{ij}^{(2)}(n)} = y1_i^{(1)}(n)g_j^{(2)}(n),\; i \in \overline{0, N^{(1)}},\; j \in \overline{1, N^{(2)}},$$

$$\frac{\partial E(n)}{\partial w2_{ij}^{(2)}(n)} = y2_i^{(1)}(n)g_j^{(2)}(n),\; i \in \overline{0, N^{(1)}},\; j \in \overline{1, N^{(2)}},$$

$$\frac{\partial E(n)}{\partial w1_{ij}^{(1)}(n)} = \begin{cases} y1_{i-N^{(0)}}^{(1)}(n-1)g1_j^{(1)}(n), & i > N^{(0)} \\ y_i^{(0)}(n)g1_j^{(1)}(n), & i \le N^{(0)} \end{cases},$$

$$\frac{\partial E(n)}{\partial w2_{ij}^{(1)}(n)} = \begin{cases} y2_{i-N^{(0)}}^{(1)}(n+1)g2_j^{(1)}(n), & i > N^{(0)} \\ y_i^{(0)}(n)g2_j^{(1)}(n), & i \le N^{(0)} \end{cases},$$

$i \in \overline{0, N^{(0)} + N^{(1)}}$, $j \in \overline{1, N^{(1)}}$,

$$g_j^{(2)}(n) = f'^{(2)}(s_j^{(2)}(n))(y_j^{(2)}(n) - d_{\mu j})$$

$$g1_j^{(1)}(n) = f'^{(1)}(s1_j^{(1)}(n))\sum_{l=0}^{N^{(2)}} w1_{jl}^{(2)}(n)g_l^{(2)}(n)$$

$$g2_j^{(1)}(n) = f'^{(1)}(s2_j^{(1)}(n)) \sum_{l=0}^{N^{(2)}} w2_{jl}^{(2)}(n) g_l^{(2)}(n)$$

6. Проверка условия завершения

Если $n \bmod P > 0$, то $\mu = \mu + 1$, $n = n + 1$, переход к 3.

Если $n \bmod P = 0$ и $\frac{1}{P}\sum_{s=1}^{P} E(n - P + s) > \varepsilon$, то $n = n + 1$, переход к 2.

Если $n \bmod P = 0$ и $\frac{1}{P}\sum_{s=1}^{P} E(n - P + s) < \varepsilon$, то завершиться.

Функционирование ИНС

1. Инициализация

$n = 1$.

$y1_i^{(1)}(n-1) = 0$, $i \in \overline{1, N^{(0)}}$.

$y1_i^{(1)}(n+1) = 0$, $i \in \overline{1, N^{(0)}}$.

2. Прогноз

$y_i^{(0)}(n) = x_i$, $i \in \overline{1, N^{(0)}}$,

$y1_j^{(1)}(n) = f^{(1)}(s1_j^{(1)}(n))$, $j \in \overline{1, N^{(1)}}$,

$$s1_j^{(1)}(n) = b1_j^{(1)} + \sum_{i=1}^{N^{(0)}} w1_{ij}^{(1)} y_i^{(0)}(n) + \sum_{i=N^{(0)}+1}^{N^{(0)}+N^{(1)}} w1_{ij}^{(1)} y1_{i-N^{(0)}}^{(1)}(n-1),$$

$y2_j^{(1)}(n) = f^{(1)}(s2_j^{(1)}(n))$, $j \in \overline{1, N^{(1)}}$,

$$s2_j^{(1)}(n) = b2_j^{(1)} + \sum_{i=1}^{N^{(0)}} w2_{ij}^{(1)} y_i^{(0)}(n) + \sum_{i=N^{(0)}+1}^{N^{(0)}+N^{(1)}} w2_{ij}^{(1)} y2_{i-N^{(0)}}^{(1)}(n+1),$$

$y_j^{(2)}(n) = f^{(2)}(s_j^{(2)}(n))$, $j \in \overline{1, N^{(2)}}$,

$$s_j^{(2)}(n) = b_j^{(2)} + \sum_{i=1}^{N^{(1)}} w1_{ij}^{(2)} y1_i^{(1)}(n) + \sum_{i=1}^{N^{(1)}} w2_{ij}^{(2)} y2_i^{(1)}(n).$$

Достоинства

1. Является универсальным аппроксиматором. Обеспечивает глобальную аппроксимацию нелинейного отображения входного сигнала в выходной.

2. Обеспечивает хорошее качество обобщения.

3. Автоматически определяется количество скрытых слоев (равно одному).

4. Количество классов может быть больше двух (в выходном слое может быть больше одного нейрона).

5. В отличие от статических ИНС, позволяет осуществлять адаптивную фильтрацию, прогноз, адаптивное управление, параметрическую идентификацию модели, классификацию нестационарных сигналов.

6. Дает более точный прогноз, чем ENN (SRN), поскольку анализирует последовательность как слева, так и справа от текущего элемента.

Недостатки

1. Обучение происходит медленнее, чем в случае MLP, RBFNN, PNN, сети Хемминга, SOM, CPNN.

2. Отсутствует автоматическое определение числа нейронов в скрытом слое.

3. В отличие от методов обучения SVM, градиентные методы обучения BRNN выбирают положение разделяющей гиперплоскости произвольным образом.

4. Обучение не сводится к задаче квадратичного программирования в выпуклой области, имеющей единственное решение.

5. В отличие от ART, не решает проблему пластичности-стабильности.

6. В отличие от LSTM и ESN не позволяет работать с долгосрочными нестационарными последовательностями.

6.13. Рекуррентная каскадно-корреляционная нейросеть

На рис. 6.14 приведена рекуррентная каскадно-корреляционная нейросеть (RCCNN) [26], который является рекуррентной статической многослойной ИНС, содержащей один или более скрытых слоев и выходной слой. Классы разделяются гиперплоскостями.

Для RCCNN используется обучение на основе коррекции ошибок (обучение с учителем), при этом применяется алгоритм обратного распространения (BP) или аналогичные ему, но более быстрые алгоритмы. Это итеративные градиентные алгоритмы обучения, которые обеспечивают минимизацию среднеквадратичной ошибки.

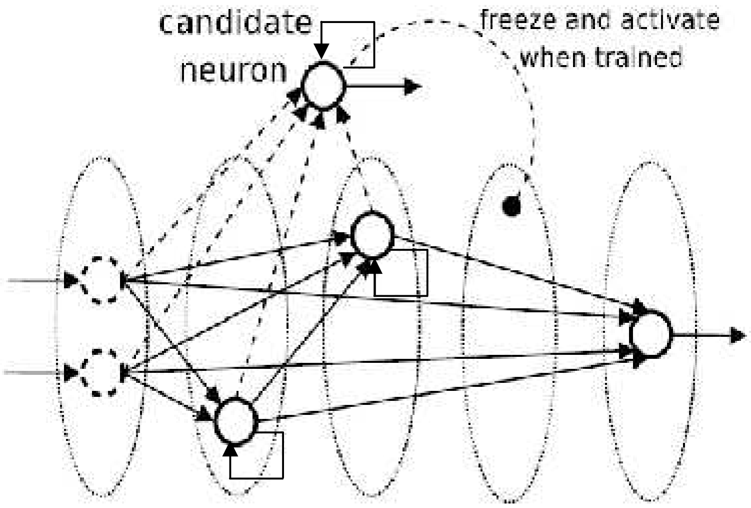


Рис. 6.14. Рекуррентная каскадно-корреляционная нейросеть (RCCNN)

На начальном этапе формируется ИНС, состоящая только из входного и выходного слоев. Происходит обучение ИНС. Если ИНС дает небольшую ошибку, то скрытые слои не добавляются и формирование ИНС завершается. Если ИНС дает большую ошибку, то создается новый слой. Процесс создания нового слоя заключается в следующем. По обученной ИНС вычисляются ошибки каждого нейрона выходного слоя и средняя ошибка нейрона выходного слоя. Затем входной слой соединяется с новым скрытым слоем. Настраиваются только веса между нейронами входного слоя и единственным рекурсивным нейроном нового слоя, используя ковариацию между выходом нового нейрона и ошибкой каждого нейрона выходного слоя. Затем выходной слой соединяется с новым скрытым слоем. Настраиваются только веса между единственным рекурсивным нейроном нового слоя и нейронами выходного слоя. Процесс создания новых слоев с единственным рекурсивным нейроном продолжается, пока ИНС дает большую ошибку. Для более точной работы ИНС может параллельно использоваться не один рекурсивный нейрон, а несколько рекурсивных нейронов-кандидатов (от 5 до 10) с разными функциями активации и начальными весами, из которых будет выбран лучший.

Обучение ИНС (алгоритм обратного распространения)

Обучение весов между входным и выходным слоем

1. Номер итерации обучения $n = 1$, инициализация посредством равномерного распределения на интервале (0,1) или [-0.5, 0.5] весов

$w_{ij}^{(0,1)}(n)$, $i \in \overline{1, N^{(0)}}$, $j \in \overline{1, N^{(1)}}$, где $N^{(k)}$ – количество нейронов в k-м слое.

2. Задается обучающее множество $\{(\mathbf{x}_\mu, \mathbf{d}_\mu) \mid \mathbf{x}_\mu \in R^{N^{(0)}}, \mathbf{d}_\mu \in R^{N^{(1)}}\}$, $\mu \in \overline{1, P}$, где $\mathbf{x}_\mu$ – μ-й обучающий входной вектор, $\mathbf{d}_\mu$ – μ-й обучающий выходной вектор, $N^{(0)}$ – количество нейронов входного слоя, $N^{(1)}$ – количество нейронов выходного слоя, P – мощность обучающего множества. Номер текущей пары из обучающего множества $\mu = 1$.

3. Вычисление выходного сигнала для выходного слоя (прямой ход)

$$y_i^{(0)}(n) = x_{\mu i},$$

$$y_j^{(1)}(n) = f^{(1)}(s_j^{(1)}(n)), s_j^{(1)}(n) = \sum_{i=1}^{N^{(0)}} w_{ij}^{(0,1)}(n) y_i^{(0)}(n), \; j \in \overline{1, N^{(1)}},$$

где $N^{(k)}$ – число нейронов в k-м слое, k – номер слоя, $w_{ij}^{(0,1)}(n)$ – вес связи от i-го нейрона входного слоя к j-му нейрону выходного слоя в момент времени n, $y_j^{(k)}(n)$ – выход j-го нейрона на k-м слое, $f^{(k)}$ – функция активация нейронов k-го слоя.

4. Вычисление энергии ошибки ИНС

$$E(n) = \frac{1}{2} \sum_{j=1}^{N^{(1)}} e_j^2(n), \; e_j(n) = y_j^{(1)}(n) - d_{\mu j}$$

5. Настройка синаптических весов на основе обобщенного дельта правила (обратный ход)

$$w_{ij}^{(0,1)}(n+1) = w_{ij}^{(0,1)}(n) - \eta \frac{\partial E(n)}{\partial w_{ij}^{(0,1)}(n)},$$

где η – параметр, определяющий скорость обучения (при большом η обучение происходит быстрее, но увеличивается опасность получить неверное решение), $0 < \eta < 1$.

$$\frac{\partial E(n)}{\partial w_{ij}^{(0,1)}(n)} = y_i^{(0)}(n) f'^{(1)}(s_j^{(1)}(n))(y_j^{(1)}(n) - d_{\mu j}), \qquad i \in \overline{1, N^{(0)}}, \; j \in \overline{1, N^{(1)}}$$

6. Проверка условия завершения

Если $n \bmod P > 0$, то $\mu = \mu + 1$, $n = n + 1$, переход к 3.

Если $n \bmod P = 0$ и $\frac{1}{P}\sum_{s=1}^{P} E(n - P + s) > \varepsilon$, то $n = n + 1$, переход к 2.

Если $n \bmod P = 0$ и $\frac{1}{P}\sum_{s=1}^{P} E(n - P + s) < \varepsilon$, то $w_{ij}^{(0,1)} = w_{ij}^{(0,1)}(n)$.

Обучение весов между нейронами слоев, предшествующих новому скрытому слою, и единственным рекурсивным нейроном нового скрытого слоя

1. Номер итерации обучения $n = 1$, начальная ковариация $\sigma_j(n-1) = 0$, $j \in \overline{1, N^{(L)}}$, инициализация посредством равномерного распределения на интервале (0,1) или [-0.5, 0.5] весов $w_{i1}^{(k,new)}(n)$, $i \in \overline{1, N^{(k)}}, k \in \overline{0, L-1}$, где $N^{(k)}$ – количество нейронов в k-м слое, L – количество слоев.

2. Задается обучающее множество $\{(\mathbf{x}_\mu, \mathbf{d}_\mu) \mid \mathbf{x}_\mu \in R^{N^{(0)}}, \mathbf{d}_\mu \in R^{N^{(L)}}\}$, $\mu \in \overline{1, P}$, где $\mathbf{x}_\mu$ – μ-й обучающий входной вектор, $\mathbf{d}_\mu$ – μ-й обучающий выходной вектор, $N^{(0)}$ – количество нейронов входного слоя, $N^{(L)}$ – количество нейронов выходного слоя, P – мощность обучающего множества. Номер текущей пары из обучающего множества $\mu = 1$.

3. Начальное вычисление выходного сигнала для первого слоя

$y_{0j}^{(k)} = 0$, $j \in \overline{1, N^{(k)}}, k \in \overline{1, L-1}$.

$y_{01}^{(new)}(n-1) = 0$.

4. Вычисление выходного сигнала для каждого слоя

$y_{\mu i}^{(0)} = x_{\mu i}$,

$y_{\mu j}^{(k)} = f^{(k)}(s_{\mu j}^{(k)})$, $j \in \overline{1, N^{(k)}}, k \in \overline{1, L-1}$,

$$s_{\mu j}^{(k)} = \sum_{l=0}^{k-1} \sum_{i=1}^{N^{(l)}} w_{ij}^{(l,k)} y_{\mu i}^{(l)} + w_j^{(k,k)} y_{\mu-1 j}^{(l)},$$

$$y_{\mu j}^{(L)} = f^{(L)}(s_{\mu j}^{(L)}),\ s_{\mu j}^{(L)} = \sum_{l=0}^{L-1} \sum_{i=1}^{N^{(l)}} w_{ij}^{(l,L)} y_{\mu i}^{(l)},\ j \in \overline{1, N^{(L)}},$$

где $N^{(k)}$ – число нейронов в *k*-м слое, *k* – номер слоя, *L* – число слоев, $w_j^{(k,k)}(n)$ – вес связи от *j*-го нейрона *k*-го слоя к *j*-му нейрону *k*-го слоя в момент времени n, $w_{ij}^{(l,k)}$ – вес связи от *i*-го нейрона на *l*-м слое к *j*-му нейрону на *k*-м слое, $y_{\mu j}^{(k)}$ – выход *j*-го нейрона на *k*-м слое для μ-го обучающего входного вектора, $f^{(k)}$ – функция активация нейронов *k*-го слоя.

5. Вычисление ошибки ИНС для каждого выходного нейрона

$$e_{\mu j} = y_{\mu j}^{(1)} - d_{\mu j}$$

6. Проверка условия завершения

Если $\mu < P$, то $\mu = \mu + 1$, переход к 4.

7. Вычисление средней ошибки ИНС для каждого выходного нейрона

$$\bar{e}_j = \frac{1}{P}\sum_{s=1}^{P} e_{\mu j}$$

8. Вычисление выходного сигнала для единственного нейрона нового слоя

$$y_{\mu 1}^{(new)}(n) = f^{(new)}(s_{\mu 1}^{(new)}(n)),$$

$$s_{\mu 1}^{(new)}(n) = \sum_{l=0}^{L-1}\sum_{i=1}^{N^{(l)}} w_{i1}^{(l,new)}(n) y_{\mu i}^{(l)} + w_1^{(new,new)}(n) y_{\mu-1,1}^{(new)}(n-1),\ \mu \in \overline{1,P},$$

где $N^{(k)}$ – число нейронов в *k*-м слое, *k* – номер слоя, *L* – число слоев, $w_1^{(new,new)}(n)$ – вес связи от единственного нейрона на новом слое к единственному нейрону на новом слое в момент времени n, $w_{i1}^{(l,new)}(n)$ – вес связи от *i*-го нейрона на *l*-м слое к единственному нейрону на новом слое в момент времени n, $y_j^{(k)}$ – выход *j*-го нейрона на *k*-м слое, $f^{(k)}$ – функция активация нейронов *k*-го слоя.

9. Вычисление среднего выходного сигнала для единственного нейрона нового слоя

$$\bar{y}_1^{(new)}(n) = \frac{1}{P}\sum_{s=1}^{P} y_{\mu 1}^{(new)}(n)$$

10. Вычисление ковариации между выходными сигналами единственного нейрона на новом слое и нейронами входного слоя

$$\sigma_j(n) = \frac{1}{P}\sum_{\mu=1}^{P}(y_{\mu 1}^{(new)}(n) - \bar{y}_1^{(new)}(n))(e_{\mu j} - \bar{e}_j), \; j \in \overline{1, N^{(L)}}.$$

11. Настройка синаптических весов на основе расширенного обобщенного дельта правила (обратный ход)

$$w_{i1}^{(k,new)}(n+1) = w_{i1}^{(k,new)}(n) - \eta\frac{\partial S(n)}{\partial w_{i1}^{(k,new)}(n)},$$

$$w_1^{(new,new)}(n+1) = w_1^{(new,new)}(n) - \eta\frac{\partial S(n)}{\partial w_1^{(new,new)}(n)},$$

где η – параметр, определяющий скорость обучения (при большом η обучение происходит быстрее, но увеличивается опасность получить неверное решение), $0 < \eta < 1$.

$$\frac{\partial S(n)}{\partial w_{i1}^{(k,new)}(n)} = \sum_{\mu=1}^{P}\sum_{j=1}^{L} y_{\mu i}^{(k)} f'^{(new)}(s_{\mu 1}^{(new)}(n))\sigma_j(n)(e_{\mu j} - \bar{e}_j),$$

$i \in \overline{1, N^{(k)}}, k \in \overline{0, L-1}$.

$$\frac{\partial S(n)}{\partial w_1^{(new,new)}(n)} = \sum_{\mu=1}^{P} y_{\mu-1,j}^{(new)}(n-1) f'^{(new)}(s_{\mu 1}^{(new)}(n))\sigma_j(n)(e_{\mu j} - \bar{e}_j).$$

12. Проверка условия завершения

Если $\left|\sum_{j=1}^{L}\sigma_j(n-1) - \sum_{j=1}^{L}\sigma_j(n)\right| > \varepsilon$, то $n = n+1$, переход к 8.

Если $\left|\sum_{j=1}^{L}\sigma_j(n-1) - \sum_{j=1}^{L}\sigma_j(n)\right| < \varepsilon$, то $w_{i1}^{(k,new)} = w_{i1}^{(k,new)}(n)$, $w_1^{(new,new)} = w_1^{(new,new)}(n)$, $y_{\mu 1}^{(new)} = y_{\mu 1}^{(new)}(n)$.

Обучение весов между единственным рекурсивным нейроном нового скрытого слоя и нейронами выходного слоя

1. Номер итерации обучения $n = 1$, инициализация посредством равномерного распределения на интервале (0,1) или [-0.5, 0.5] весов $w_{1j}^{(new,L)}(n)$, $j \in \overline{1, N^{(L)}}$.

2. Номер текущей пары из обучающего множества $\mu = 1$.

3. Вычисление выходного сигнала для выходного слоя (прямой ход)

$$y_j^{(L)}(n) = f^{(L)}(s_j^{(L)}(n)),$$

$$s_j^{(L)}(n) = w_{1j}^{(new,L)}(n) y_{\mu 1}^{(new)} + \sum_{l=0}^{L-1} \sum_{i=1}^{N^{(l)}} w_{ij}^{(l,k)} y_{\mu i}^{(l)}, \ j \in \overline{1, N^{(L)}},$$

где $N^{(k)}$ – число нейронов в k-м слое, k – номер слоя, $w_{ij}^{(l,k)}$ – вес связи от i-го нейрона на l-м слое к j-му нейрону на k-м слое, $y_j^{(k)}(n)$ – выход j-го нейрона на k-м слое, $f^{(k)}$ – функция активация нейронов k-го слоя.

4. Вычисление энергии ошибки ИНС

$$E(n) = \frac{1}{2} \sum_{j=1}^{N^{(L)}} e_j^2(n), \ e_j(n) = y_j^{(L)}(n) - d_{\mu j}$$

5. Настройка синаптических весов на основе обобщенного дельта правила (обратный ход)

$$w_{1j}^{(new,L)}(n+1) = w_{1j}^{(new,L)}(n) - \eta \frac{\partial E(n)}{\partial w_{1j}^{(new,L)}(n)},$$

где η – параметр, определяющий скорость обучения (при большом η обучение происходит быстрее, но увеличивается опасность получить неверное решение), $0 < \eta < 1$.

$$\frac{\partial E(n)}{\partial w_{1j}^{(new,L)}(n)} = y_1^{(new)}(n) f'^{(L)}(s_j^{(L)}(n))(y_j^{(L)}(n) - d_{\mu j}), \ j \in \overline{1, N^{(L)}}.$$

6. Проверка условия завершения

Если $n \bmod P > 0$, то $\mu = \mu + 1$, $n = n + 1$, переход к 3.

Если $n \bmod P = 0$ и $\frac{1}{P} \sum_{s=1}^{P} E(n - P + s) > \varepsilon$, то $n = n + 1$, переход к 2.

Если $n \bmod P = 0$ и $\frac{1}{P} \sum_{s=1}^{P} E(n - P + s) < \varepsilon$, то $w_{1j}^{(new,L)} = w_{1j}^{(new,L)}(n)$.

Функционирование ИНС

1. Инициализация

$n = 1$.

$y_j^{(k)}(n-1) = 0, \ j \in \overline{1, N^{(k)}}, k \in \overline{1, L-1}$.

2. Прогноз

$$y_i^{(0)}(n) = x_i,$$

$$y_j^{(k)}(n) = f^{(k)}(s_j^{(k)}(n)),\ j \in \overline{1, N^{(k)}}, k \in \overline{1, L-1},$$

$$s_j^{(k)}(n) = \sum_{l=0}^{k-1} \sum_{i=1}^{N^{(l)}} w_{ij}^{(l,k)} y_i^{(l)}(n) + w_j^{(k,k)} y_j^{(k)}(n-1).$$

$$y_j^{(L)}(n) = f^{(L)}(s_j^{(L)}(n)),\ s_j^{(L)}(n) = \sum_{l=0}^{L-1} \sum_{i=1}^{N^{(l)}} w_{ij}^{(l,L)} y_i^{(l)}(n),\ j \in \overline{1, N^{(L)}}.$$

Достоинства

1. Является универсальным аппроксиматором. Обеспечивает глобальную аппроксимацию нелинейного отображения входного сигнала в выходной.

2. Обеспечивает хорошее качество обобщения.

3. Количество классов может быть больше двух (в выходном слое может быть больше одного нейрона).

4. Автоматически определяет число скрытых слоев и число нейронов в этих слоях (равно одному).

5. В отличие от статических ИНС, позволяет осуществлять адаптивную фильтрацию, прогноз, адаптивное управление, параметрическую идентификацию модели, классификацию нестационарных сигналов.

Недостатки

1. Обучение происходит медленнее, чем в случае MLP, RBFNN, PNN, сети Хемминга, SOM, CPNN.

3. В отличие от методов обучения SVM, градиентные методы обучения ENN первого порядка выбирают положение разделяющей гиперплоскости произвольным образом.

4. Обучение не сводится к задаче квадратичного программирования в выпуклой области, имеющей единственное решение.

5. В отличие от ART, не решает проблему пластичности-стабильности (способность к восприятию новых образцов при сохранении старых образцов).

6. В отличие от RMLP, LSTM, ESN, BRNN хуже решает задачи адаптивной фильтрации, прогноза, адаптивного управления,

параметрической идентификации модели, классификации нестационарных сигналов.

6.14. MaxNet

На рис. 6.15 приведена MaxNet [6] (для случая четырех нейронов во входном и выходном слоях), которая является полносвязной рекуррентной однослойной ИНС.

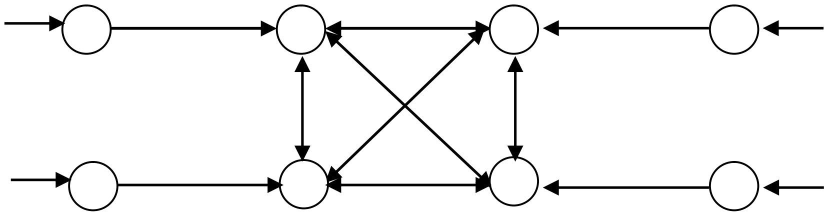


Рис. 6.15. MaxNet

Суть работы MaxNet заключается в следующем. MaxNet начинает циклические вычисления с использованием обратных связей, во время которых выходы ИНС, соответствующие классам, пытаются взаимно подавить конкурирующие выходные сигналы. В результате этого процесса активизируется выход нейрона-победителя, который соответствует максимальному значению. Номер выигравшего нейрона определяет номер элемента с максимальным значением.

Для обучения MaxNet используется одношаговое обучение (обучение с учителем).

Обучение ИНС

Производится инициализация весов связей $w_{ij} = \begin{cases} 1, & i = j \\ -\varepsilon, & i \neq j \end{cases}$, $i \in \overline{1,N}$, $j \in \overline{1,N}$, где $\varepsilon \in (0, 1/N)$

Функционирование ИНС

1. $y_j(1) = x_j$, $j \in \overline{1,N^{(2)}}$, $n = 1$.

2. $y_j(n+1) = f\left(\sum_{i=1}^{N} w_{ij} y_i(n)\right)$, $f(s) = \begin{cases} s & s \le 0 \\ 0, & s < 0 \end{cases}$, $j \in \overline{1,N}$.

3. Если более одного ненулевого $y_j(n+1)$, то $n = n+1$, переход на 2, иначе выдать номер класса $j^* = \arg\max_j y_j(n+1)$.

Достоинства

1. Не требует определения количества скрытых слоев (отсутствуют).

2. Не требует определения количества нейронов скрытого слоя (отсутствуют).

Недостатки

Обычно самостоятельно не используется (например, выходной слой MaxNet соответствует выходному слою сети Хемминга).

6.15. Mexican Hat

На рис. 6.16 приведена Mexican Hat [6] (для случая четырех нейронов во входном и выходном слоях), которая является рекуррентной неполносвязной однослойной ИНС.

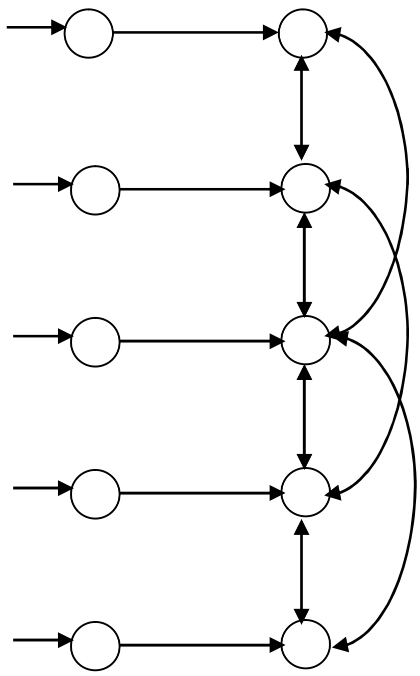


Рис. 6.16. Mexican Hat

Суть работы Mexican Hat заключается в следующем. Mexican Hat начинает циклические вычисления с использованием обратных связей, во время которых выходы ИНС, соответствующие классам, пытаются взаимно подавить конкурирующие выходные сигналы. В результате этого процесса активизируется выход нейрона-победителя, который соответствует максимальному значению. Номер выигравшего нейрона определяет номер элемента с максимальным значением.

Для обучения Mexican Hat используется одношаговое обучение (обучение с учителем).

Обучение ИНС

Производится инициализация весов связей $w_i = \begin{cases} c_1, & |i| \in \{0,..,r_1\} \\ c_2, & |i| \in \{r_1+1,...,r_2\} \end{cases}$, $i \in \overline{1,N}$, где $c_1 > 0, c_2 < 0$. Задается значение $x_{\max}$, $n_{\max}$.

Функционирование ИНС

1. $y_j(1) = x_j$, $j \in \overline{1,N^{(2)}}$, $n = 1$.

2. $y_i(n+1) = f\left(\sum_{k=-r_2}^{r_2} w_{i+k} y_{i+k}(n)\right)$, $f(s) = \begin{cases} 0, & s < 0 \\ s & 0 \le s \le x_{\max} \\ x_{\max}, & x > x_{\max} \end{cases}$, $i \in \overline{1,N}$.

3. Если $n < n_{\max}$, то $n = n+1$, переход на 2, иначе выдать номер класса $j^* = \arg\max_j y_j(n+1)$.

Достоинства

1. Не требует определения количества скрытых слоев (отсутствуют).

2. Не требует определения количества нейронов скрытого слоя (отсутствуют).

Недостатки

Обычно самостоятельно не используется.

ГЛАВА 7
РЕКУРРЕНТНЫЕ ИСКУССТВЕННЫЕ НЕЙРОННЫЕ СЕТИ С АССОЦИАТИВНОЙ ПАМЯТЬЮ

7.1. Рекуррентная автоассоциативная память

На рис. 7.1 приведена рекуррентная автоассоциативная память (RAAM) [59], которая является рекуррентной двухслойной ИНС и построена на базе ENN (SRN). Классы разделяются гиперплоскостями. В отличие от ENN (SRN) число входных и выходных нейронов совпадает. Если входные и выходные нейроны разделены на две равные части (на вход и выход подаются два образца), тогда скрытый слой представляет собой конкатенацию обоих образцов.

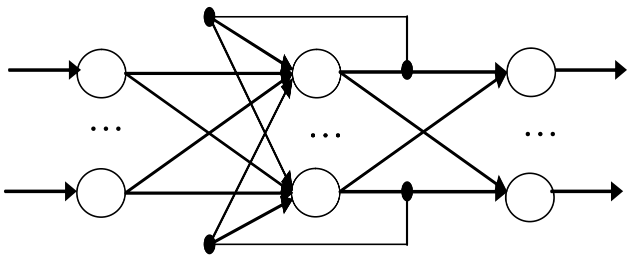


Рис. 7.1. Рекуррентная автоассоциативная память (RAAM)

RAAM реализует автоассоциативную память (ее элемент представлен парой образцов $(\mathbf{m}_x, \mathbf{m}_y)$, $\mathbf{m}_y = \mathbf{m}_x$) и восстанавливает запомненный образец $\mathbf{m}_y$ по ключевому образцу $\mathbf{m}_x$, соответствующему входному вектору $\mathbf{x}$.

Важнейшим свойством сети RAAM является то, что одна и та же ИНС с одними и теми же весами связей может хранить и воспроизводить несколько различных запомненных образцов.

Для RAAM используется обучение на основе коррекции ошибок (обучение с учителем), при этом чаще всего применяется алгоритм обратного распространения (BP). Это итеративный градиентный алгоритм обучения, который обеспечивает минимизацию среднеквадратичной ошибки.

Обучение ИНС (алгоритм обратного распространения)

1. Номер итерации обучения $n = 1$, инициализация посредством равномерного распределения на интервале (0,1) или [-0.5, 0.5]

смещений (порогов) $b_j^{(k)}(n)$ и весов $w_{ij}^{(k)}(n)$, $i \in \overline{1, N^{(k-1)}}$, $j \in \overline{1, N^{(k)}}, k \in \overline{1,2}$, где $N^{(k)}$ – количество нейронов в k-м слое.

2. Задается обучающее множество $\{\mathbf{x}_\mu \mid \mathbf{x}_\mu \in R^{N^{(0)}}\}$, $\mu \in \overline{1, P}$, где $\mathbf{x}_\mu$ – μ-й обучающий входной вектор, $N^{(0)}$ – количество нейронов входного слоя, $N^{(2)}$ – количество нейронов выходного слоя $N^{(0)} = N^{(2)}$, P – мощность обучающего множества. Номер вектора из обучающего множества $\mu = 1$.

3. Начальное вычисление выходного сигнала для первого слоя

$y_i^{(1)}(n-1) = 0$, $i \in \overline{1, N^{(1)}}$.

4. Вычисление выходного сигнала для каждого слоя (прямой ход)

$y_i^{(0)}(n) = x_{\mu i}$,

$y_j^{(1)}(n) = f^{(1)}(s_j^{(1)}(n))$,

$$s_j^{(1)}(n) = \sum_{i=0}^{N^{(0)}} w_{ij}^{(1)}(n) y_i^{(0)}(n) + \sum_{i=N^{(0)}+1}^{N^{(0)}+N^{(1)}} w_{ij}^{(1)}(n) y_{i-N^{(0)}}^{(1)}(n-1),\ j \in \overline{1, N^{(1)}},$$

$$y_j^{(2)}(n) = f^{(2)}(s_j^{(2)}(n)),\ s_j^{(2)}(n) = \sum_{i=0}^{N^{(1)}} w_{ij}^{(2)}(n) y_i^{(1)}(n),\ j \in \overline{1, N^{(2)}},$$

где $N^{(k)}$ – число нейронов в *k*-м слое, *k* – номер слоя, $w_{ij}^{(k)}(n)$ – вес связи от *i*-го нейрона к *j*-му нейрону на *k*-м слое в момент времени n, $y_j^{(k)}(n)$ – выход *j*-го нейрона на *k*-м слое, $f^{(k)}$ – функция активация нейронов *k*-го слоя.

Считается, что $w_{0j}^{(k)}(n) = b_j^{(k)}(n), y_0^{(k-1)}(n) = 1$.

5. Вычисление энергии ошибки ИНС

$$E(n) = \frac{1}{2} \sum_{j=1}^{N^{(2)}} e_j^2(n),\ e_j(n) = y_j^{(2)}(n) - x_{\mu j},$$

6. Настройка синаптических весов на основе обобщенного дельта правила (обратный ход)

$$w_{ij}^{(k)}(n+1) = w_{ij}^{(k)}(n) - \eta \frac{\partial E(n)}{\partial w_{ij}^{(k)}(n)},$$

где η – параметр, определяющий скорость обучения (при большом η обучение происходит быстрее, но увеличивается опасность получить неверное решение), $0 < \eta < 1$.

$$\frac{\partial E(n)}{\partial w_{ij}^{(2)}(n)} = y_i^{(1)}(n) g_j^{(2)}(n),\ i \in \overline{0, N^{(1)}},\ j \in \overline{1, N^{(2)}},$$

$$\frac{\partial E(n)}{\partial w_{ij}^{(1)}(n)} = \begin{cases} y_{i-N^{(0)}}^{(1)}(n-1) g_j^{(1)}(n), & i > N^{(0)} \\ y_i^{(0)}(n) g_j^{(1)}(n), & i \le N^{(0)} \end{cases},$$

$i \in \overline{0, N^{(0)} + N^{(1)}}$, $j \in \overline{1, N^{(1)}}$,

$$g_j^{(k)}(n) = \begin{cases} f'^{(2)}(s_j^{(2)}(n))(y_j^{(2)}(n) - x_{\mu j}), & k = 2 \\ f'^{(1)}(s_j^{(1)}(n)) \sum_{l=1}^{N^{(2)}} w_{jl}^{(2)}(n) g_l^{(2)}(n), & k = 1 \end{cases}$$

7. Проверка условия завершения

Если $n \bmod P > 0$, то $\mu = \mu + 1$, $n = n + 1$, переход к 3.

Если $n \bmod P = 0$ и $\frac{1}{P}\sum_{s=1}^{P} E(n - P + s) > \varepsilon$, то $n = n + 1$, переход к 2.

Если $n \bmod P = 0$ и $\frac{1}{P}\sum_{s=1}^{P} E(n - P + s) < \varepsilon$, то завершиться.

Функционирование ИНС

1. Инициализация

$n = 1$.

$y_i^{(1)}(n-1) = 0$, $i \in \overline{1, N^{(1)}}$.

2. Прогноз

$y_i^{(0)}(n) = x_i$,

$y_j^{(1)}(n) = f^{(1)}(s_j^{(1)}(n))$,

$$s_j^{(1)}(n) = b_j^{(1)} + \sum_{i=1}^{N^{(0)}} w_{ij}^{(1)} y_i^{(0)}(n) + \sum_{i=N^{(0)}+1}^{N^{(0)}+N^{(1)}} w_{ij}^{(1)} y_{i-N^{(0)}}^{(1)}(n-1),\ j \in \overline{1, N^{(1)}},$$

$$y_j^{(2)}(n) = f^{(2)}(s_j^{(2)}(n)),\ s_j^{(2)}(n) = b_j^{(2)} + \sum_{i=1}^{N^{(1)}} w_{ij}^{(2)} y_i^{(1)}(n),\ j \in \overline{1, N^{(2)}}$$

Достоинства

1. Используется для восстановления образцов.
2. Обеспечивает хорошее качество обобщения.
3. Автоматически определяется количество скрытых слоев (равно одному).
4. Количество классов может быть больше двух.
5. В отличие от сети DHNN, GM и BAM сеть RAAM работает с вещественными данными.

Недостатки

1. Отсутствует автоматическое определение числа нейронов в скрытом слое.
2. В отличие от ART, не решает проблему пластичности-стабильности.
3. В отличие от SOM требует больше объемов вычислений и занимаемой памяти.
4. В отличие от BM не гарантирует нахождение глобального минимума.

7.2. Нейросеть Хопфилда

На рис. 7.2 приведена нейросеть Хопфилда (HNN) [60,61], которая является рекуррентной однослойной ИНС.

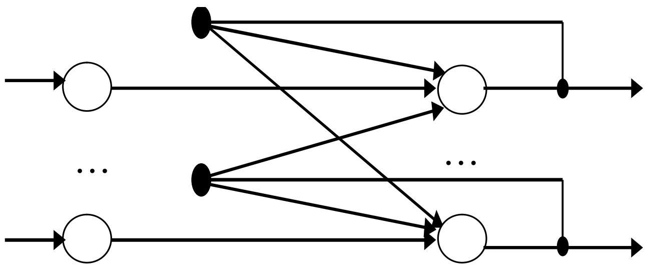


Рис. 7.2. Нейросеть Хопфилда (HNN)

HNN реализует автоассоциативную память (ее элемент представлен парой образцов $(\mathbf{m}_x, \mathbf{m}_y)$, $\mathbf{m}_y = \mathbf{m}_x$) и восстанавливает запомненный образец $\mathbf{m}_y$ по ключевому образцу $\mathbf{m}_x$, соответствующему входному вектору $\mathbf{x}$. Емкость памяти (число пар $(\mathbf{m}_x, \mathbf{m}_y)$) составляет $\dfrac{N^{(0)}}{2\ln N^{(0)}}$.

Важнейшим свойством HNN является то, что одна и та же ИНС с одними и теми же весами связей может хранить и воспроизводить несколько запомненных образцов.

Существует дискретная сеть Хопфилда (DHNN) и непрерывная сеть Хопфилда (CHNN).

Для обучения DHNN и CHNN используется комбинация обучения Хебба (обучение с учителем) и одношагового обучения (обучение с учителем).

7.2.1. Дискретная сеть Хопфилда

Обучение ИНС (фаза запоминания)

Задается обучающее множество $\{\mathbf{x}_\mu \mid \mathbf{x}_\mu \in \{-1,1\}^{N^{(0)}}\}$, $\mu \in \overline{1,P}$, P – мощность обучающего множества. Производится инициализация весов связей выходного слоя $w_{ij} = \frac{1}{N^{(0)}} \sum_{\mu=1}^{P} x_{\mu i} x_{\mu j}$, $i, j \in \overline{1, N^{(0)}}$.

Замечание. В случае бинарных данных $x_{\mu j} = 2x_{\mu j} - 1$.

Замечание. Полученная матрица весов $\mathbf{W}$ должна быть:

1. Симметричной, $w_{ij} = w_{ji}$.

2. С нулевой главной диагональю, $w_{ii} = 0$.

Функционирование ИНС (фаза восстановления)

1. $y_j(1) = x_j$ (если $\mathbf{x}$ биполярный) или $y_j(1) = 2x_j - 1$ (если $\mathbf{x}$ бинарный), $n = 1$.

2. $y_j(n+1) = \mathrm{sgn}(\sum_{i=1}^{N^{(0)}} w_{ij} y_i(n))$, $j \in \overline{1, N^{(0)}}$,

3. Если $\sum_{j=1}^{N^{(0)}} | y_j(n+1) - y_j(n) | > 0$, то $n = n+1$, переход на шаг 2

Результатом является образец $\mathbf{y}$.

7.2.2. Непрерывная сеть Хопфилда

Обучение ИНС (фаза запоминания)

Задается обучающее множество $\{\mathbf{x}_\mu \mid \mathbf{x}_\mu \in (-1,1)^{N^{(0)}}\}$ или $\{\mathbf{x}_\mu \mid \mathbf{x}_\mu \in (0,1)^{N^{(0)}}\}$, $\mu \in \overline{1,P}$, P – мощность обучающего множества.

Производится инициализация весов связей выходного слоя $w_{ij} = \frac{1}{N^{(0)}} \sum_{\mu=1}^{P} x_{\mu i} x_{\mu j}$, $i, j \in \overline{1, N^{(0)}}$.

Функционирование ИНС (фаза восстановления)

1. $y_j(0) = x_j$, $t = 0$.

2. $y_j(t + \Delta t) = y_j(t) + \Delta t(-y_j(t) + b_j + \sum_{i=1}^{N^{(0)}} w_{ij} \varphi(y_i(t)))$, $j \in \overline{1, N^{(0)}}$,

$\varphi(s) = \tanh(\alpha s)$ для $y_j \in (-1,1)$, $\varphi(s) = \frac{1}{1 + \exp(-\alpha s)}$ для $y_j \in (0,1)$,

где α – константа, $0 < \Delta t < 1$.

3. Если $\sum_{j=1}^{N^{(0)}} | y_j(t + \Delta t) - y_j(t) | > \delta$, то $t = t + \Delta t$, переход на шаг 2

Результатом является образец $\mathbf{y}$.

Использование непрерывной нейросети Хопфилда для задачи поиска оптимального маршрута

Для задачи поиска оптимального маршрута непрерывная нейронная сеть Хопфилда состоит из N^2 нейронов, расположенных в виде двумерной матрицы переездов $N \times N$. Строка матрицы соответствует городу, столбец матрицы соответствует порядку посещения.

Процедура поиска оптимального маршрута

1. $y_{pq}(0) = 0.5 + \min + (\max - \min) * rand()$, $p, q \in \overline{1, N}$,

где $rand()$ – функция, возвращающая случайное число,

$\min = -0.01$, $\max = 0.01$.

$v_{pq}(0) = \varphi(y_{pq}(0))$, $p, q \in \overline{1, N}$,

$\varphi(s) = \frac{1}{2}(1 + \tanh(\alpha s))$ или $\varphi(s) = \frac{1}{1 + e^{-\alpha x}}$,

$d_{\max} = \max_{(x_1, x_2)} d(x_1, x_2)$,

где $d(x_1, x_2)$ – расстояние между городами x_1 и x_2 (например, расстояние Евклида),

φ – функция активации,

α – положительная константа.

Задаётся максимальное количество итераций $T_{\max}$ и порог ε.

$t = 0$

2. $y_{x_2 j}(t+1) = y_{x_2 j}(t) + \Delta t(s_{x_2 j}(t+1) - y_{x_2 j}(t) + b_{x_2 j})$, $x_2, j \in \overline{1, N}$,

$$s_{x_2 j}(t+1) = -A\sum_{\substack{i=1\\ i\neq j}}^{N} v_{x_2 i}(t) - B\sum_{\substack{x_1=1\\ x_1\neq x_2}}^{N} v_{x_1 j}(t) - C\sum_{x_1=1}^{N}\sum_{i=1}^{N} v_{x_1 i}(t) -$$

$$- D\sum_{x_1=1}^{N}\frac{d(x_1, x_2)}{d_{\max}}(v_{x_1, j+1}(t) + v_{x_1, j-1}(t)),$$

$$v_{x_1, i+1}(t) = \begin{cases} v_{x_1, i+1}(t), & i < N \\ v_{x_1 1}(t), & i = N \end{cases}, \quad v_{x_1, i-1}(t) = \begin{cases} v_{x_1, i-1}(t), & i > 1 \\ v_{x_1 N}(t), & i = 1 \end{cases},$$

где A, B, C, D – положительные константы,

$b_{x_2 j}$ – порог, $b_{x_2 j} = CN$,

Δt – шаг квантования, $\Delta t = 1/T_{\max}$.

3. $v_{x_2 j}(t+1) = \varphi(y_{x_2 j}(t+1))$, $x_2, j \in \overline{1, N}$

4. Если $\sum_{x_2=1}^{N}\sum_{j=1}^{N}| v_{x_2 j}(t+1) - v_{x_2 j}(t) | > \varepsilon$ или $t < T_{\max}$, то $t = t + 1$, переход на шаг 2.

5. $\tilde{v}_{x_2 j} = f(v_{x_2 j}(t+1))$, $f(v_{x_2 j}(t+1)) = \begin{cases} 0, & v_{x_2 j}(t+1) < 0.5 \\ 1, & v_{x_2 j}(t+1) > 0.5 \end{cases}$

6. Если $\sum_{x_2=1}^{N}\tilde{v}_{x_2 j} \neq 1$ или $\sum_{j=1}^{N}\tilde{v}_{x_2 j} \neq 1$, $x_2, j \in \overline{1, N}$, то переход к 1.

Результатом является матрица переездов $[\tilde{v}_{x_2 j}]$,

$$\tilde{v}_{x_2 j} = \begin{cases} 1, & \textit{город } x_2 \textit{ посещен } j-\textit{м} \\ 0, & \textit{город } x_2 \textit{ не посещен } j-\textit{м} \end{cases}$$

Достоинства

1. Используется для восстановления образцов.
2. Обеспечивает хорошее качество обобщения.
3. Не требует определения количества скрытых слоев.
4. Не требует определения количества нейронов скрытого слоя.
5. Количество классов может быть больше двух.

Недостатки

1. В отличие от ART, не решает проблему пластичности-стабильности.

2. В отличие от сети Хемминга требует больше объемов вычислений и занимаемой памяти.

3. DHNN работает только с биполярными или бинарными данными.

4. Емкость памяти ограничена.

5. В отличие от SBM не гарантирует нахождение глобального минимума.

7.3. Машина Гаусса

На рис. 7.3 приведена машина Гаусса (GM) [62,63], которая является рекуррентной однослойной ИНС.

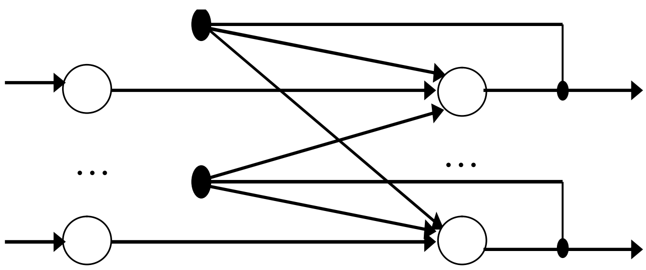


Рис. 7.3. Машина Гаусса (GM)

GM реализует автоассоциативную память (ее элемент представлен парой образцов $(\mathbf{m}_x, \mathbf{m}_y)$, $\mathbf{m}_y = \mathbf{m}_x$) и восстанавливает запомненный образец $\mathbf{m}_y$ по ключевому образцу $\mathbf{m}_x$, соответствующему входному вектору $\mathbf{x}$. Емкость памяти (число пар $(\mathbf{m}_x, \mathbf{m}_y)$) составляет $\frac{N^{(0)}}{2\ln N^{(0)}}$.

Важнейшим свойством GM является то, что одна и та же ИНС с одними и теми же весами связей может хранить и воспроизводить несколько запомненных образцов.

Для обучения GM используется комбинация обучения Хебба (обучение с учителем) и одношагового обучения (обучение с учителем).

Обучение ИНС (фаза запоминания)

Задается обучающее множество $\{\mathbf{x}_\mu \mid \mathbf{x}_\mu \in \{0,1\}^{N^{(0)}}\}$, $\mu \in \overline{1,P}$, P – мощность обучающего множества. Производится инициализация весов связей выходного слоя $w_{ij} = \frac{1}{N^{(0)}} \sum_{\mu=1}^{P} x_{\mu i} x_{\mu j}$, $i \in \overline{1, N^{(0)}}$, $j \in \overline{1, N^{(0)}}$.

Замечание. Полученная матрица весов $\mathbf{W}$ должна быть:

1. Симметричной, $w_{ij} = w_{ji}$.
2. С нулевой главной диагональю, $w_{ii} = 0$.

Функционирование ИНС (фаза восстановления)

1. Задание параметров нечеткости $A_{\max}$ и температуры $T_{\max}$.
$y_j(1) = x_j$, $T(1) = T_{\max}$, $A(1) = A_{\max}$, $n = 1$.

2. $s_j(n+1) = \sum_{i=1}^{N^{(0)}} w_{ij} y_i(n) + noise(y_j(n))$, $j \in \overline{1, N^{(0)}}$,

$noise(y_j(n)) = \frac{1}{\sqrt{2\pi}\sigma(n)} \exp - \frac{(s_j(n))^2}{2(\sigma(n))^2}$ – шум в виде функции Гаусса с нулевым средним, $\sigma(n) = T(n)\sqrt{8\pi^{-1}}$.

3. $a_j(n+1) = s_j(n+1) - \frac{a_j(n+1) - a_j(n)}{\beta}$, $j \in \overline{1, N^{(0)}}$, $0 < \beta < 1$.

4. $y_j(n+1) = \frac{1}{1 + \exp\left(-\frac{a_j(n)}{A(n)}\right)}$, $j \in \overline{1, N^{(0)}}$.

5. $T(n) = \frac{T_{\max}}{1 + n\tau}$, $A(n) = \frac{A_{\max}}{1 + n\alpha}$,

где параметры α, τ определяются скорость убывания $A_{\max}, T_{\max}$.

6. $\sum_{l=1}^{N^{(0)}} | y_l(n+1) - y_l(n) | > \varepsilon$, то $n = n+1$, переход на шаг 2.

Результатом является образец $\mathbf{y}$.

Замечание. При больших n значения $A(n)$ стремятся к нулю и значения $y_j(n+1)$ стремятся к бинарным. При больших n значения $T(n)$ стремятся к нулю и значения $noise(y_j(n))$ стремятся к нулю.

Достоинства

1. Используется для восстановления образцов.
2. Обеспечивает хорошее качество обобщения.
3. Не требует определения количества скрытых слоев.
4. Не требует определения количества нейронов скрытого слоя.
5. Количество классов может быть больше двух.
6. В отличие от HNN гарантирует нахождение глобального минимума.
7. Учитывает шум для бинарных данных.

Недостатки

1. В отличие от ART, не решает проблему пластичности-стабильности.
2. В отличие от сети Хемминга требует больше объемов вычислений и занимаемой памяти.
3. Образцы являются только бинарными данными.
4. Емкость памяти ограничена.

7.4. Двунаправленная ассоциативная память

На рис. 5.3 приведена двунаправленная ассоциативная память (BAM) [64], которая является рекуррентной двухслойной ИНС.

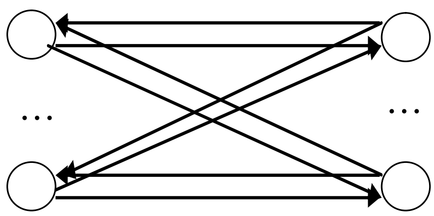


Рис. 5.3. Двунаправленная ассоциативная память (BAM)

В отличие от сети HNN реализует гетероассоциативную память (ее элемент представлен парой образцов $(\mathbf{m}_x, \mathbf{m}_y)$, $\mathbf{m}_y \neq \mathbf{m}_x$) и автоассоциативную память (ее элемент представлен парой образцов $(\mathbf{m}_x, \mathbf{m}_y)$, $\mathbf{m}_y = \mathbf{m}_x$) и восстанавливает запомненный образец $\mathbf{m}_y$ по ключевому образцу $\mathbf{m}_x$, соответствующему входному вектору $\mathbf{x}$. Емкость памяти (число пар $(\mathbf{m}_x, \mathbf{m}_y)$) составляет $\frac{N^{(0)}}{2\ln N^{(0)}}$.

Важнейшим свойством ВАМ является то, что одна и та же ИНС с одними и теми же весами связей может хранить и воспроизводить несколько различных запомненных образцов.

Для обучения ВАМ используется комбинация обучения Хебба (обучение с учителем) и одношагового обучения (обучение с учителем).

Обучение ИНС (фаза запоминания)

Задается обучающее множество $\{(\mathbf{x}_\mu, \mathbf{y}_\mu) \mid \mathbf{x}_\mu \in \{-1,1\}^{N^{(1)}}, \mathbf{y}_\mu \in \{-1,1\}^{N^{(2)}}\}$, $\mu \in \overline{1,P}$, P – мощность обучающего множества. Производится инициализация весов прямых и обратных связей $w_{ij} = \sum_{\mu=1}^{P} x_{\mu i} y_{\mu j}$, $v_{ji} = \sum_{\mu=1}^{P} x_{\mu i} y_{\mu j}$, $i \in \overline{1, N^{(1)}}$, $j \in \overline{1, N^{(2)}}$.

Замечание. В случае бинарных данных $x_{\mu j} = 2x_{\mu j} - 1$, $y_{\mu j} = 2y_{\mu j} - 1$.

Функционирование ИНС (фаза восстановления)

1. $z_j(1) = x_j$ (если $\mathbf{x}_\mu$ биполярный) или $z_j(1) = 2x_j - 1$ (если $\mathbf{x}_\mu$ бинарный), $n = 1$.

2. $y_j(n+1) = \mathrm{sgn}(\sum_{i=1}^{N^{(1)}} w_{ij} z_i(n))$, $j \in \overline{1, N^{(2)}}$.

2. $z_i(n+1) = \mathrm{sgn}(\sum_{j=1}^{N^{(2)}} v_{ji} y_j(n+1))$, $i \in \overline{1, N^{(1)}}$.

3. Если $\sum_{i=1}^{N^{(1)}} | z_i(n+1) - z_i(n) | > 0$ или $\sum_{j=1}^{N^{(2)}} | y_j(n+1) - y_j(n) | > 0$, то $n = n + 1$, переход на шаг 2.

Результатом является образец $\mathbf{y}$ и образец $\mathbf{z}$.

Достоинства

1. Используется для восстановления образцов.
2. Обеспечивает хорошее качество обобщения.
3. Не требует определения количества скрытых слоев.
4. Не требует определения количества нейронов скрытого слоя.
5. Количество классов может быть больше двух.
6. В отличие от HNN, GM, BSB и сети Хемминга реализует оба вида памяти.

Недостатки

1. В отличие от ART, не решает проблему пластичности-стабильности.
2. В отличие от сети Хемминга требует больше объемов вычислений и занимаемой памяти.
3. Работает только с биполярными или бинарными данными.
4. Емкость памяти ограничена.

7.5. Модель состояния мозга

На рис. 7.5 приведена модель состояния мозга (BSB) [65,66], которая является рекуррентной однослойной ИНС.

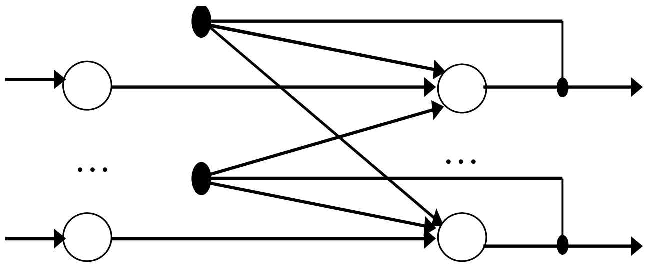


Рис. 7.5. Модель состояния мозга (BSB)

Сеть BSB реализует автоассоциативную память (ее элемент представлен парой образцов $(\mathbf{m}_x, \mathbf{m}_y)$, $\mathbf{m}_y = \mathbf{m}_x$) и восстанавливает запомненный образец $\mathbf{m}_y$ по ключевому образцу $\mathbf{m}_x$, соответствующему входному вектору $\mathbf{x}$.

Важнейшим свойством сети BSB является то, что одна и та же ИНС с одними и теми же весами связей может хранить и воспроизводить несколько запомненных образцов.

Для BSB, не использующей коэффициент обратной связи, применяется комбинация обучения Хебба (обучение с учителем) и одношагового обучения (обучение с учителем).

Для BSB, использующей коэффициент обратной связи, применяется обучение на основе коррекции ошибок (обучение с учителем).

7.5.1. Модель состояния мозга, не использующая коэффициент обратной связи

Обучение ИНС (фаза запоминания)

Задается обучающее множество $\{\mathbf{x}_\mu \mid \mathbf{x}_\mu \in [-1,1]^{N^{(0)}}\}$, $\mu \in \overline{1,P}$, P – мощность обучающего множества. Производится инициализация весов связей выходного слоя $w_{ij} = \frac{1}{N^{(0)}} \sum_{\mu=1}^{P} x_{\mu i} x_{\mu j}$, $i \in \overline{1,N^{(0)}}$, $j \in \overline{1,N^{(0)}}$.

Замечание. Полученная матрица весов $\mathbf{W}$ должна быть:

1. Симметричной, $w_{ij} = w_{ji}$.

2. С нулевой главной диагональю, $w_{ii} = 0$.

Функционирование ИНС (фаза восстановления)

1. $y_j(1) = x_j$, $n = 1$.

2. $y_j(n+1) = f(s_j(n))$, $s_j(n) = y_i(n) + \sum_{i=1}^{N^{(0)}} w_{ij} y_i(n)$,

$$f(s_j(n)) = \begin{cases} 1, & s_j(n) > 1 \\ s_j(n), & -1 \le s_j(n) \le 1, \quad j \in \overline{1,N^{(0)}}. \\ -1, & s_j(n) < -1 \end{cases}$$

3. Если $\sum_{j=1}^{N^{(0)}} | y_j(n+1) - y_j(n) | > 0$, то $n = n+1$, переход на шаг 2.

Результатом является образец $\mathbf{y}$.

7.5.2. Модель состояния мозга, использующая коэффициент обратной связи

Обучение ИНС (фаза запоминания)

1. Номер итерации обучения $n = 0$, инициализация посредством равномерного распределения на интервале (0,1) или [-0.5, 0.5] весов $w_{ij}(n)$, $i \in \overline{1,N^{(0)}}$, $j \in \overline{1,N^{(0)}}$, где $N^{(0)}$ – количество нейронов входного слоя и выходного.

Задается обучающее множество $\{\mathbf{x}_\mu \mid \mathbf{x}_\mu \in [-1,1]^{N^{(0)}}\}$, $\mu \in \overline{1,P}$, где $\mathbf{x}_\mu$ – μ-й обучающий входной вектор, P – мощность обучающего множества. Номер вектора из обучающего множества $\mu = 1$.

2. Настройка синаптических весов на основе правила BSB

$$y_j(n+1) = \sum_{i=1}^{N^{(0)}} w_{ij}(n) x_{\mu i} \ ,$$

$$w_{ij}(n+1) = w_{ij}(n) + \eta(x_{\mu j} - y_j(n+1)) x_{\mu i} ,$$

где η – параметр, определяющий скорость обучения (при большом η обучение происходит быстрее, но увеличивается опасность получить неверное решение), $0 < \eta < 1$.

3. Проверка условия завершения

Если $n \bmod P > 0$, то $\mu = \mu + 1$, $n = n + 1$, переход к 2.

Если $n \bmod P = 0$ и $\frac{1}{P}\sum_{\mu=1}^{P}\sum_{j=1}^{N^{(0)}} | d_{\mu j} - y_j(n+1) | > \varepsilon$, то $n = n + 1$, переход к 2.

Если $n \bmod P = 0$ и $\frac{1}{P}\sum_{\mu=1}^{P}\sum_{i=1}^{N^{(0)}} | d_{\mu j} - y_j(n+1) | < \varepsilon$, то завершиться.

Замечание. Полученная матрица весов $\mathbf{W}$ должна быть:

1. Симметричной, $w_{ij} = w_{ji}$.

2. Строго диагонально-доминантной, $w_{jj} \geq \sum_{i \neq j} | w_{ji} | + const$.

3. Положительно полуопределенной (наименьшее собственное значение неотрицательно).

Функционирование ИНС (фаза восстановления)

1. $y_j(1) = x_j$, $n = 1$.

2. $y_j(n+1) = f(s_j(n))$, $s_j(n) = y_j(n) + \gamma \sum_{i=1}^{N^{(0)}} w_{ij} y_i(n)$,

$$f(s_j(n)) = \begin{cases} 1, & s_j(n) > 1 \\ s_j(n), & -1 \leq s_j(n) \leq 1, \ j \in \overline{1, N^{(0)}} \\ -1, & s_j(n) < -1 \end{cases},$$

где γ - коэффициент обратной связи, $\gamma > 0$.

3. Если $\sum_{j=1}^{N^{(0)}} | y_j(n+1) - y_j(n) | > 0$, то $n = n+1$, переход на шаг 2.

Результатом является образец $\mathbf{y}$.

Достоинства

1. Используется для восстановления образцов.

2. Используется для кластеризации. Поскольку выходной вектор всегда ограничен $N^{(0)}$-мерным единичным гиперкубом, то углы этого гиперкуба рассматриваются как кластеры.

3. Обеспечивает хорошее качество обобщения.

4. Не требует определения количества скрытых слоев.

5. Не требует определения количества нейронов скрытого слоя.

6. Количество классов может быть больше двух.

7. В отличие от DHNN, GM и BAM работает с вещественными данными.

Недостатки

1. В отличие от ART, не решает проблему пластичности-стабильности.

2. В отличие от сети Хемминга требует больше объемов вычислений и занимаемой памяти.

3. Количество классов (кластеров) ограничено количеством вершин гиперкуба.

4. BSB, не использующая коэффициент обратной связи, дает более низкий результат, чем BSB, использующая коэффициент обратной связи.

7.6. Нейросеть Хемминга

На рис. 7.6 приведена нейросеть Хемминга [67,68], которая является рекуррентной двухслойной ИНС. Выходной слой соответствует выходному слою MaxNet.

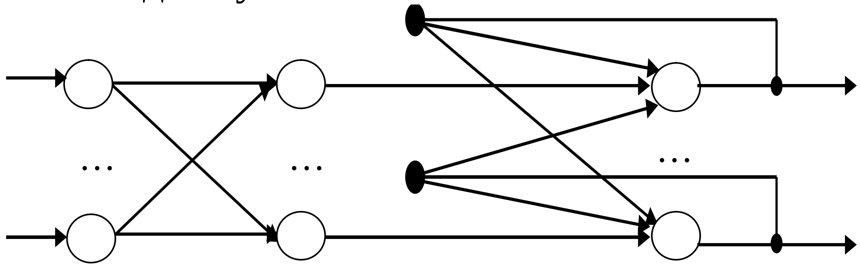


Рис. 7.6. Нейросеть Хемминга

Сеть Хемминга реализует автоассоциативную память (ее элемент представлен парой образцов $(\mathbf{m}_x, \mathbf{m}_y)$, $\mathbf{m}_y = \mathbf{m}_x$) и восстанавливает запомненный образец $\mathbf{m}_y$ по ключевому образцу $\mathbf{m}_x$, соответствующему входному вектору $\mathbf{x}$.

Емкость памяти (число пар $(\mathbf{m}_x, \mathbf{m}_y)$) составляет $2^{\beta N^{(0)}}$, $\beta \le 1$, что больше емкости сети DHNN, GM и BAM.

Суть работы сети Хемминга заключается в следующем. После начальной оценки расстояния Хемминга между входным вектором и запомненными при обучении образцами, сеть Хемминга начинает циклические вычисления с использованием обратных связей, во время которых выходы ИНС, соответствующие классам, пытаются взаимно подавить конкурирующие выходные сигналы. В результате этого процесса активизируется выход нейрона-победителя, который соответствует запомненному образцу, ближайшему к входному вектору. Номер выигравшего нейрона определяет номер запомненного образца. Роль первого слоя весьма условна: сеть Хемминга использует его веса только один раз, после чего этот слой в вычислениях не участвует.

Существует бинарная и вещественная сеть Хемминга.

Для сети Хемминга используется одношаговое обучение (обучение с учителем).

7.6.1. Бинарная нейросеть Хемминга

Обучение ИНС (фаза запоминания)

Задается обучающее множество $\{\mathbf{x}_i \mid \mathbf{x}_i \in \{0,1\}^{N^{(0)}}\}$, $i \in \overline{1, N^{(1)}}$. Производится инициализация смещения (порога) $b_j^{(1)} = N^{(0)}/2$, $j \in \overline{1, N^{(1)}}$, весов связей первого слоя $w_{ij}^{(1)} = x_{ij}/2$, $i \in \overline{1, N^{(0)}}$, $j \in \overline{1, N^{(1)}}$.

Замечание. В случае биполярных данных $x_{ij} = 2x_{ij} - 1$.

Функционирование ИНС (фаза восстановления)

1. $z_i = x_i$ (если $\mathbf{x}$ бинарный) или $z_i = 2x_i - 1$ (если $\mathbf{x}$ биполярный).

2. $y_j^{(1)} = b_j^{(1)} + \sum_{i=1}^{N^{(0)}} w_{ij}^{(1)} z_i$, $j \in \overline{1, N^{(2)}}$

(соответствует расстоянию Хемминга $\sum_{i=1}^{N^{(0)}} | w_{ij}^{(1)} - z_i \ |$).

3. $y_j^{(2)}(1) = y_j^{(1)}$, $j \in \overline{1, N^{(2)}}$, $n = 1$.

4. $$y_j^{(2)}(n+1) = f(y_j^{(2)}(n) - \varepsilon \sum_{i=1}^{N^{(2)}} y_i^{(2)}(n)), \qquad f(s) = \begin{cases} 0 & s \le 0 \\ 1, & s > 0 \end{cases},$$

$j \in \overline{1, N^{(2)}}$, $0 < \varepsilon < 1$.

4. Если $\sum_{j=1}^{N^{(2)}} | y_j^{(2)}(n+1) - y_j^{(2)}(n) | > 0$, то $n = n+1$, переход на 4, иначе выдать номер класса $j^* = \arg\max_j y_j^{(2)}(n+1)$.

Результатом является образец $(w_{1j^*}, ..., w_{N^{(1)} j^*})$.

7.6.2. Вещественная нейросеть Хемминга

Обучение ИНС (фаза запоминания)

Задается обучающее множество $\{\mathbf{x}_i \mid \mathbf{x}_i \in (0,1)^{N^{(0)}}\}$, $i \in \overline{1, N^{(1)}}$. Производится инициализация смещения (порога) $b_j^{(1)} = N^{(0)} / 2$, $j \in \overline{1, N^{(1)}}$, весов связей первого слоя $w_{ij}^{(1)} = x_{ij}$, $i \in \overline{1, N^{(0)}}$, $j \in \overline{1, N^{(1)}}$.

Функционирование ИНС (фаза восстановления)

1. $y_j^{(1)} = \sum_{i=1}^{N^{(0)}} (w_{ij}^{(1)} - x_i \ )^2$

2. $y_j^{(2)}(1) = y_j^{(1)}$, $n = 1$.

3. $$y_j^{(2)}(n+1) = f(y_j^{(2)}(n) - \varepsilon \sum_{i=1}^{N^{(2)}} y_i^{(2)}(n)), \qquad \varphi(s) = \frac{1}{1 + \exp(-\alpha s)},$$

$j \in \overline{1, N^{(2)}}$, $0 < \varepsilon < 1$.

4. Если $\sum_{j=1}^{N^{(2)}} | y_j^{(2)}(n+1) - y_j^{(2)}(n) | > \delta$, то $n = n+1$, переход на 3, иначе выдать номер класса $j^* = \arg\max_j y_j^{(2)}(n+1)$.

Результатом является образец $(w_{1j^*}, ..., w_{N^{(1)} j^*})$.

Достоинства

1. Используется для распознавания образцов.

2. Обеспечивает хорошее качество обобщения.

3. Не требует определения количества скрытых слоев (один слой).

4. Не требует определения количества нейронов скрытого слоя (совпадает с количеством нейронов второго слоя).

5. В отличие от HNN, GM, BAM и BSB требует меньше объемов вычислений и занимаемой памяти (нужно выдавать только номер класса образца).

6. Количество классов может быть больше двух.

7. В отличие от DHNN, GM и BAM вещественная сеть Хемминга работает с вещественными данными.

Недостатки

В отличие от ART, не решает проблему пластичности-стабильности.

7.7. Полная машина Больцмана

На рис. 7.7, 7.8 представлена полная машина Больцмана (FBM) [2,32,69,70], которая является рекуррентной ИНС и содержит один видимый слой и один скрытый слой.

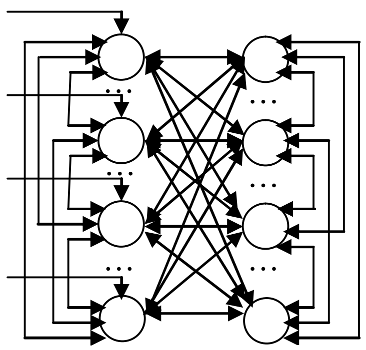


Рис. 7.7. Полная машина Больцмана (FBM) как генеративная модель

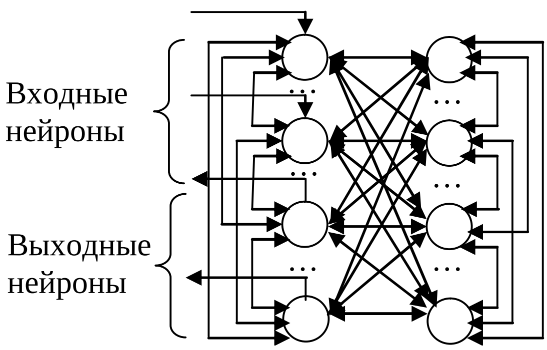


Рис. 7.8. Полная машина Больцмана (FBM) как дескриптивная модель

Генеративная FBM реализует автоассоциативную память (ее элемент представлен парой образцов $(\mathbf{m}_x, \mathbf{m}_y)$, $\mathbf{m}_y = \mathbf{m}_x$) и восстанавливает запомненный образец $\mathbf{m}_y$ по ключевому образцу $\mathbf{m}_x$, соответствующему входному вектору $\mathbf{x}$.

Дескриптивная FBM реализует гетероассоциативную память (ее элемент представлен парой образцов $(\mathbf{m}_x, \mathbf{m}_y)$, $\mathbf{m}_y \neq \mathbf{m}_x$) и восстанавливает запомненный образец $\mathbf{m}_y$ по ключевому образцу $\mathbf{m}_x$, соответствующему входному вектору $\mathbf{x}$.

Важнейшим свойством FBM является то, что одна и та же ИНС с одними и теми же весами связей может хранить и воспроизводить несколько различных запомненных образцов.

Для генеративной стохастической FBM используется стохастическое обучение (обучение без учителя). Для дескриптивной стохастической FBM используется стохастическое обучение (обучение с учителем). Для дескриптивной детерминированной FBM используется обучение Хебба (обучение с учителем).

FBM работает в двух фазах – положительной и отрицательной.

Для генеративной стохастической FBM в положительной фазе фиксируются видимые нейроны, а ИНС функционирует до тех пор, пока не установятся скрытые нейроны. В отрицательной фазе вначале фиксируются обученные в положительной фазе скрытые нейроны, а ИНС функционирует до тех пор, пока не установятся видимые нейроны, после чего фиксируются обученные в отрицательной фазе видимые нейроны, а ИНС функционирует до тех пор, пока не установятся скрытые нейроны.

Для дескриптивной стохастической и детерминированной FBM в положительной фазе фиксируются видимые входные и выходные нейроны, а ИНС функционирует до тех пор, пока не установятся скрытые нейроны. В отрицательной фазе вначале фиксируются видимые входные нейроны и обученные в положительной фазе скрытые нейроны, а ИНС функционирует до тех пор, пока не установятся видимые выходные нейроны, после чего фиксируются видимые входные нейроны и обученные в отрицательной фазе видимые выходные нейроны, а ИНС функционирует до тех пор, пока не установятся скрытые нейроны.

Компонентами стохастической FBM являются стохастические нейроны, состояние которых описывается на основе распределения Бернулли в виде

$$x_j = \begin{cases} 1, & \text{с вероятностью } P_j \\ 0, & \text{с вероятностью } 1 - P_j \end{cases}.$$

Величина изменения состояния j-го стохастического нейрона определена в виде

$$\Delta x_j = \hat{x}_j - x_j,$$

где $\hat{x}_j$ – состояние, противоположное состоянию x_j, т.е. $\hat{x}_j = 1 - x_j$.

Вероятность перехода j-го стохастического нейрона из текущего состояния x_j в новое состояние $\hat{x}_j$ определяется на основе имитации отжига в виде

$$P_j = \frac{1}{1 + \exp\left(-\frac{\Delta E_j}{T}\right)},$$

где ΔE_j – приращение энергии сети при изменении состояния j-го нейрона, T – температура.

Компонентами детерминированной FBM являются детерминированные нейроны, $< x_j > \in (0,1)$.

7.7.1. Генеративная стохастическая полная машина Больцмана

Обучение ИНС

1. Номер итерации обучения $n = 1$, инициализация посредством равномерного распределения на интервале (0,1) или [-0.5, 0.5] весов $w_{ij}(n)$, $w_{ii}(n) = 0$, $w_{ij}(n) = w_{ji}(n)$, $i, j \in \overline{1, N^v + N^h}$. Задание параметра P_0, температурных параметров $T_{\max}, T_{\min}$.

2. Инициализация бинарного вектора состояний скрытых нейронов $\mathbf{x}^h = (x_1^h, ..., x_{N^h}^h)$. Задается обучающее множество $\{\mathbf{x}_\mu^v \mid \mathbf{x}_\mu^v \in \{0,1\}^{N^v}\}$, $\mu \in \overline{1, P}$, где $\mathbf{x}_\mu^v$ – μ-й обучающий вектор состояний видимых нейронов, P – мощность обучающего множества.

Положительная фаза (шаги 3-9)

3. $\mu = 1$.

4. $k = 1$, $T(k) = T_{\max}$.

$$\mathbf{x} = (x_{\mu 1}^v, ..., x_{\mu N^v}^v, x_1^h, ..., x_{N^h}^h).$$

5. Квантование Гиббса для скрытых нейронов ($j \in \overline{N^v+1, N^v+N^h}$)

$\Delta x_j = \hat{x}_j - x_j$.

$\Delta E_j = -\Delta x_j \sum_{i=1}^{N^v+N^h} w_{ij}(n) x_i$.

$P_j = \dfrac{1}{1+\exp\left(-\dfrac{\Delta E_j}{T(k)}\right)}$.

Если $\Delta E_j \le 0$, то $x_j = \hat{x}_j$.

Если $\Delta E_j > 0$ и $P_j \ge P_0$, то $x_j = \hat{x}_j$.

6. Если имеются скрытые нейроны, которые изменили состояние, то переход к 5.

7. Уменьшение температурного коэффициента в соответствии с методом моделирования отжига

$T(k+1) = \beta T(k)$,

где β – коэффициент охлаждения, $0 < \beta < 1$.

8. Если $T(k) > T_{\min}$, то $k = k+1$, переход к 5.

9. $\mathbf{x}1_\mu = \mathbf{x}$. Если $\mu < P$, то $\mu = \mu + 1$, переход к 4.

Отрицательная фаза (шаги 10-18)

10. μ=1.

11. $k = 1$, $T(k) = T_{\max}$.

$\mathbf{x} = \mathbf{x}1_\mu$.

12. Квантование Гиббса для видимых нейронов ($j \in \overline{1, N^v}$)

$\Delta x_j = \hat{x}_j - x_j$.

$\Delta E_j = -\Delta x_j \sum_{i=1}^{N^v+N^h} w_{ij}(n) x_i$.

$P_j = \dfrac{1}{1+\exp\left(-\dfrac{\Delta E_j}{T(k)}\right)}$.

Если $\Delta E_j \le 0$, то $x_j = \hat{x}_j$.

Если $\Delta E_j > 0$ и $P_j \ge P_0$, то $x_j = \hat{x}_j$.

13. Если имеются видимые нейроны, которые изменили состояние, то переход к 12.

14. Квантование Гиббса для скрытых нейронов ($j \in \overline{N^v+1, N^v+N^h}$)

$\Delta x_j = \hat{x}_j - x_j$.

$$\Delta E_j = -\Delta x_j \sum_{i=1}^{N^v+N^h} w_{ij}(n) x_i \ .$$

$$P_j = \frac{1}{1+\exp\left(-\frac{\Delta E_j}{T(k)}\right)}.$$

Если $\Delta E_j \le 0$, то $x_j = \hat{x}_j$.

Если $\Delta E_j > 0$ и $P_j \ge P_0$, то $x_j = \hat{x}_j$.

15. Если имеются скрытые нейроны, которые изменили состояние, то переход к 14.

16. Уменьшение температурного коэффициента в соответствии с методом моделирования отжига

$T(k+1) = \beta T(k)$,

где β – коэффициент охлаждения, $0 < \beta < 1$.

17. Если $T(k) > T_{\min}$, то $k = k+1$, переход к 12.

18. $\mathbf{x}2_\mu = \mathbf{x}$. Если $\mu < P$, то $\mu = \mu + 1$, переход к 11.

19. Настройка синаптических весов на основе стохастического правила Больцмана

$$w_{ij}(n) = w_{ij}(n) + \eta(\rho_{ij}^+ - \rho_{ij}^-),\ i, j \in \overline{1, N^v + N^h},$$

$$\rho_{ij}^+ = \frac{1}{P}\sum_{\mu=1}^{P} \varphi(x1_{\mu i}, x1_{\mu j}),\ \rho_{ij}^- = \frac{1}{P}\sum_{\mu=1}^{P} \varphi(x2_{\mu i}, x2_{\mu j}),$$

$$\varphi(x_i, x_j) = \begin{cases} 1, & x_i = x_j \\ 0, & x_i \ne x_j \end{cases}.$$

20. Если $\frac{1}{P}\sum_{\mu=1}^{P}\sum_{i=1}^{N^v} | x1_{\mu i}^v - x2_{\mu i}^v | > \varepsilon$, то $n = n+1$, переход к 2.

Замечание. Полученная матрица весов $\mathbf{W}$ должна быть:

1. Симметричной, $w_{ij} = w_{ji}$.

2. С нулевой главной диагональю, $w_{ii} = 0$.

Функционирование ИНС

1. Инициализация бинарного вектора состояний скрытых нейронов $\mathbf{x}^h = (x_1^h, ..., x_{N^h}^h)$.

Положительная фаза (шаги 2-7)

2. $k = 1$, $T(k) = T_{\max}$.

$\mathbf{x} = (x_1^v, ..., x_{N^v}^v, x_1^h, ..., x_{N^h}^h)$.

3. Квантование Гиббса для скрытых нейронов ($j \in \overline{N^v + 1, N^v + N^h}$)

$\Delta x_j = \hat{x}_j - x_j$.

$$\Delta E_j = -\Delta x_j \sum_{i=1}^{N^v + N^h} w_{ij} x_i \ .$$

$$P_j = \frac{1}{1 + \exp\left(-\frac{\Delta E_j}{T(k)}\right)}.$$

Если $\Delta E_j \le 0$, то $x_j = \hat{x}_j$.

Если $\Delta E_j > 0$ и $P_j \ge P_0$, то $x_j = \hat{x}_j$.

4. Если имеются скрытые нейроны, которые изменили состояние, то переход к 3.

5. Уменьшение температурного коэффициента в соответствии с методом моделирования отжига

$T(k+1) = \beta T(k)$,

где β – коэффициент охлаждения, $0 < \beta < 1$.

6. Если $T(k) > T_{\min}$, то $k = k + 1$, переход к 3.

7. $\mathbf{x}1 = \mathbf{x}$.

Отрицательная фаза (шаги 8-13)

8. $k = 1$, $T(k) = T_{\max}$.

$\mathbf{x} = \mathbf{x}1$.

9. Квантование Гиббса для видимых нейронов ($j \in \overline{1, N^v}$)

$\Delta x_j = \hat{x}_j - x_j$.

$$\Delta E_j = -\Delta x_j \sum_{i=1}^{N^v} w_{ij}(n) x_i \ ,$$

$$P_j = \frac{1}{1+\exp\left(-\frac{\Delta E_j}{T(k)}\right)}.$$

Если $\Delta E_j \le 0$, то $x_j = \hat{x}_j$.

Если $\Delta E_j > 0$ и $P_j \ge P_0$, то $x_j = \hat{x}_j$.

10. Если имеются видимые нейроны, которые изменили состояние, то переход к 9.

11. Уменьшение температурного коэффициента в соответствии с методом моделирования отжига

$$T(k+1) = \beta T(k),$$

где β – коэффициент охлаждения, $0 < \beta < 1$.

12. Если $T(k) > T_{\min}$, то $k = k+1$, переход к 9.

13. $\mathbf{x}^v = (x_1,...,x_{N^v})$.

Результатом является образец $\mathbf{x}^v$.

Замечание. Для ускорения процедуры обучения вместо BM используют **машину Коши (CM)**. В этом случае распределение Больцмана меняется на распределение Коши, т.е. $P_j = \frac{1}{1+\exp\left(-\frac{\Delta E_j}{T(k)}\right)}$ меняется на $P_j = \frac{1}{\pi}\cdot\frac{T(k)}{T^2(k)+(\Delta x_j)^2}$, где $\Delta x_j = \hat{x}_j - x_j$.

Замечание. Для ускорения процедуры обучения для машины Больцмана $T(k) = \frac{T_{\max}}{1+\ln(k+1)}$, для машины Коши $T(k) = \frac{T_{\max}}{1+k}$.

Замечание. Если состояние нейронов перестает изменяться, то это означает, что система достигла температурного равновесия, т.е. энергия сети $E(\mathbf{x}) = -\sum_{i=1}^{N^v+N^h}\sum_{j=1}^{N^v+N^h} w_{ij}x_i x_j \to \min$.

Замечание. В случае биполярных нейронов, т.е. $x_j \in \{-1,1\}$, $E(\mathbf{x}) = -\sum_{i=1}^{N^v+N^h}\sum_{j=1}^{N^v+N^h} w_{ij}x_i x_j$ заменяется на

$E(\mathbf{x}) = -\frac{1}{2}\sum_{i=1}^{N^v+N^h}\sum_{j=1}^{N^v+N^h} w_{ij}x_i x_j$, $\Delta E_j = -\Delta x_j \sum_{i=1}^{N^v+N^h} w_{ij}x_i$ заменяется на $\Delta E_j = -\frac{1}{2}\Delta x_j \sum_{i=1}^{N^v+N^h} w_{ij}x_i$, $\hat{x}_j = 1 - x_j$ заменяется на $\hat{x}_j = -x_j$.

7.7.2. Дескриптивная стохастическая полная машина Больцмана

Обучение ИНС

1. Номер итерации обучения $n=1$, инициализация посредством равномерного распределения на интервале (0,1) или [-0.5, 0.5] весов $w_{ij}(n)$, $w_{ii}(n)=0$, $w_{ij}(n)=w_{ji}(n)$, $i,j \in \overline{1, N^{in}+N^{out}+N^h}$. Задание параметра P_0, температурных параметров $T_{\max}, T_{\min}$.

2. Инициализация бинарного вектора состояний скрытых нейронов $\mathbf{x}^h = (x_1^h,...,x_{N^h}^h)$. Задается обучающее множество $\{(\mathbf{x}_\mu^{in}, \mathbf{x}_\mu^{out}) \mid \mathbf{x}_\mu^{in} \in \{0,1\}^{N^{in}}, \mathbf{x}_\mu^{out} \in \{0,1\}^{N^{out}}\}$, $\mu \in \overline{1,P}$, где $\mathbf{x}_\mu^{in}$ – μ-й обучающий вектор состояний видимых входных нейронов, $\mathbf{x}_\mu^{out}$ – μ-й обучающий вектор состояний видимых выходных нейронов, P – мощность обучающего множества.

Положительная фаза (шаги 3-9)

3. $\mu=1$.

4. $k=1$, $T(k)=T_{\max}$.

$\mathbf{x} = (x_{\mu 1}^{in},...,x_{\mu N^{in}}^{in}, x_{\mu 1}^{out},...,x_{\mu N^{out}}^{out}, x_1^h,...,x_{N^h}^h)$.

5. Квантование Гиббса для скрытых нейронов ($j \in \overline{N^{in}+N^{out}+1, N^{in}+N^{out}+N^h}$)

$\Delta x_j = \hat{x}_j - x_j$.

$$\Delta E_j = -\Delta x_j \sum_{i=1}^{N^{in}+N^{out}+N^h} w_{ij}(n)x_i .$$

$$P_j = \frac{1}{1+\exp\left(-\frac{\Delta E_j}{T(k)}\right)}.$$

Если $\Delta E_j \le 0$, то $x_j = \hat{x}_j$.

Если $\Delta E_j > 0$ и $P_j \ge P_0$, то $x_j = \hat{x}_j$.

6. Если имеются скрытые нейроны, которые изменили состояние, то переход к 5.

7. Уменьшение температурного коэффициента в соответствии с методом моделирования отжига

$$T(k+1) = \beta T(k),$$

где β – коэффициент охлаждения, $0 < \beta < 1$.

8. Если $T(k) > T_{\min}$, то $k = k+1$, переход к 5.

9. $\mathbf{x}1_\mu = \mathbf{x}$. Если $\mu < P$, то $\mu = \mu + 1$, переход к 4.

Отрицательная фаза (шаги 10-18)

10. μ=1.

11. $k = 1$, $T(k) = T_{\max}$.

$\mathbf{x} = \mathbf{x}1_\mu$.

12. Квантование Гиббса для видимых выходных нейронов ($j \in \overline{N^{in}+1, N^{in}+N^{out}}$)

$$\Delta x_j = \hat{x}_j - x_j.$$

$$\Delta E_j = -\Delta x_j \sum_{i=1}^{N^{in}+N^{out}+N^h} w_{ij}(n) x_i .$$

$$P_j = \frac{1}{1 + \exp\left(-\frac{\Delta E_j}{T(k)}\right)}.$$

Если $\Delta E_j \le 0$, то $x_j = \hat{x}_j$.

Если $\Delta E_j > 0$ и $P_j \ge P_0$, то $x_j = \hat{x}_j$.

13. Если имеются видимые выходные нейроны, которые изменили состояние, то переход к 12.

14. Квантование Гиббса для скрытых нейронов ($j \in \overline{N^{in}+N^{out}+1, N^{in}+N^{out}+N^h}$)

$$\Delta x_j = \hat{x}_j - x_j.$$

$$\Delta E_j = -\Delta x_j \sum_{i=1}^{N^{in}+N^{out}+N^h} w_{ij}(n) x_i .$$

$$P_j = \frac{1}{1 + \exp\left(-\frac{\Delta E_j}{T(k)}\right)}.$$

Если $\Delta E_j \le 0$, то $x_j = \widehat{x}_j$.

Если $\Delta E_j > 0$ и $P_j \ge P_0$, то $x_j = \widehat{x}_j$.

15. Если имеются скрытые нейроны, которые изменили состояние, то переход к 14.

16. Уменьшение температурного коэффициента в соответствии с методом моделирования отжига

$$T(k+1) = \beta T(k),$$

где β – коэффициент охлаждения, $0 < \beta < 1$.

17. Если $T(k) > T_{\min}$, то $k = k+1$, переход к 12.

18. $\mathbf{x}2_\mu = \mathbf{x}$. Если $\mu < P$, то $\mu = \mu + 1$, переход к 11.

19. Настройка синаптических весов на основе стохастического правила Больцмана

$$w_{ij}(n) = w_{ij}(n) + \eta(\rho_{ij}^{+} - \rho_{ij}^{-}),\ i, j \in \overline{1, N^{in} + N^{out} + N^{h}},$$

$$\rho_{ij}^{+} = \frac{1}{P}\sum_{\mu=1}^{P} \varphi(x1_{\mu i}, x1_{\mu j}),\ \rho_{ij}^{-} = \frac{1}{P}\sum_{\mu=1}^{P} \varphi(x2_{\mu i}, x2_{\mu j}),$$

$$\varphi(x_i, x_j) = \begin{cases} 1, & x_i = x_j \\ 0, & x_i \ne x_j \end{cases}.$$

20. Если $\frac{1}{P}\sum_{\mu=1}^{P} \sum_{i=N^{in}+1}^{N^{in}+N^{out}} | x1_{\mu i} - x2_{\mu i} | > \varepsilon$, то $n = n+1$, переход к 2.

Функционирование ИНС

1. Инициализация бинарного вектора состояний скрытых нейронов $\mathbf{x}^h = (x_1^h, ..., x_{N^h}^h)$ и видимых выходных нейронов $\mathbf{x}^{out} = (x_1^{out}, ..., x_{N^{out}}^{out})$,

Положительная фаза (шаги 2-7)

2. $k = 1$, $T(k) = T_{\max}$.

$$\mathbf{x} = (x_1^{in}, ..., x_{N^{in}}^{in}, x_1^{out}, ..., x_{N^{out}}^{out}, x_1^{h}, ..., x_{N^{h}}^{h}).$$

3. Квантование Гиббса для скрытых нейронов ($j \in \overline{N^{in} + N^{out} + 1, N^{in} + N^{out} + N^{h}}$)

$\Delta x_j = \widehat{x}_j - x_j$.

$\Delta E_j = -\Delta x_j \sum_{i=1}^{N^{in}+N^{out}+N^h} w_{ij} x_i$.

$P_j = \dfrac{1}{1+\exp\left(-\dfrac{\Delta E_j}{T(k)}\right)}$.

Если $\Delta E_j \le 0$, то $x_j = \widehat{x}_j$.

Если $\Delta E_j > 0$ и $P_j \ge P_0$, то $x_j = \widehat{x}_j$.

4. Если имеются скрытые нейроны, которые изменили состояние, то переход к 3.

5. Уменьшение температурного коэффициента в соответствии с методом моделирования отжига

$T(k+1) = \beta T(k)$,

где β – коэффициент охлаждения, $0 < \beta < 1$.

6. Если $T(k) > T_{\min}$, то $k = k+1$, переход к 3.

7. $\mathbf{x}1 = \mathbf{x}$.

Отрицательная фаза (шаги 8-13)

8. $k=1$, $T(k) = T_{\max}$.

$\mathbf{x} = \mathbf{x}1$.

9. Квантование Гиббса для видимых выходных нейронов ($j \in \overline{N^{in}+1, N^{in}+N^{out}}$)

$\Delta x_j = \widehat{x}_j - x_j$.

$\Delta E_j = -\Delta x_j \sum_{i=1}^{N^{in}+N^{out}+N^h} w_{ij} x_i$.

$P_j = \dfrac{1}{1+\exp\left(-\dfrac{\Delta E_j}{T(k)}\right)}$.

Если $\Delta E_j \le 0$, то $x_j = \widehat{x}_j$.

Если $\Delta E_j > 0$ и $P_j \ge P_0$, то $x_j = \widehat{x}_j$.

10. Если имеются видимые выходные нейроны, которые изменили состояние, то переход к 9.

11. Уменьшение температурного коэффициента в соответствии с методом моделирования отжига

$T(k+1) = \beta T(k)$,

где β – коэффициент охлаждения, $0 < \beta < 1$.

12. Если $T(k) > T_{\min}$, то $k = k+1$, переход к 9.

13. $\mathbf{x}^{out} = (x_{N^{in}+1}, ..., x_{N^{in}+N^{out}})$.

Результатом является образец $\mathbf{x}^{out}$.

7.7.3. Дескриптивная детерминированная полная машина Больцмана

Обучение ИНС

1. Номер итерации обучения $n = 1$, инициализация посредством равномерного распределения на интервале (0,1) или [-0.5, 0.5] весов $w_{ij}(n)$, $w_{ii}(n) = 0$, $w_{ij}(n) = w_{ji}(n)$, $i, j \in \overline{1, N^{in} + N^{out} + N^h}$. Задание параметра P_0, температурных параметров $T_{\max}, T_{\min}$.

2. Инициализация посредством равномерного распределения на интервале (0,1) вещественного вектора состояний скрытых нейронов $< \mathbf{x}^h > = (< x_1^h >, ..., < x_{N^h}^h >)$. Задается обучающее множество $\{(< \mathbf{x}_\mu^{in} >, < \mathbf{x}_\mu^{out} >) \,|\, < \mathbf{x}_\mu^{in} > \in (0,1)^{N^{in}}, < \mathbf{x}_\mu^{out} > \in (0,1)^{N^{out}}\}$, $\mu \in \overline{1, P}$, где $< \mathbf{x}_\mu^{in} >$ – μ-й обучающий вектор состояний видимых входных нейронов, $< \mathbf{x}_\mu^{out} >$ – μ-й обучающий вектор состояний видимых выходных нейронов, P – мощность обучающего множества.

Положительная фаза (шаги 3-8)

3. μ=1.

4. $k = 1$, $T(k) = T_{\max}$.

$$< \mathbf{x} > = (< x_{\mu 1}^{in} >, ..., < x_{\mu N^{in}}^{in} >, < x_{\mu 1}^{out} >, ..., < x_{\mu N^{out}}^{out} >, < x_1^h >, ..., < x_{N^h}^h >).$$

5. Вычисление среднего значения состояний скрытых нейронов

$$< x_j > = \frac{1}{1 + \exp\left(-\frac{1}{T(k)} \sum_{i=1}^{N^{in}+N^{out}+N^h} w_{ij}(n) < x_i > \right)},$$

$j \in \overline{N^{in} + N^{out} + 1, N^{in} + N^{out} + N^h}$.

6. Уменьшение температурного коэффициента в соответствии с методом моделирования отжига

$T(k+1) = \beta T(k)$,

где β – коэффициент охлаждения, $0 < \beta < 1$.

7. Если $T(k) > T_{\min}$, то $k = k+1$, переход к 5.

8. $<\mathbf{x}1_\mu> = <\mathbf{x}>$. Если $\mu < P$, то $\mu = \mu + 1$, переход к 4.

Отрицательная фаза (шаги 9-15)

9. $\mu = 1$.

10. $k = 1$, $T(k) = T_{\max}$.

$\mathbf{x} = \mathbf{x}1_\mu$.

11. Вычисление среднего значения состояний видимых выходных нейронов

$$< x_j > = \frac{1}{1 + \exp\left(-\frac{1}{T(k)} \sum_{i=1}^{N^{in}+N^{out}+N^h} w_{ij}(n) < x_i > \right)},$$

$j \in \overline{N^{in} + 1, N^{in} + N^{out}}$.

12. Вычисление среднего значения состояний скрытых нейронов

$$< x_j > = \frac{1}{1 + \exp\left(-\frac{1}{T(k)} \sum_{i=1}^{N^{in}+N^{out}+N^h} w_{ij}(n) < x_i > \right)},$$

$j \in \overline{N^{in} + N^{out} + 1, N^{in} + N^{out} + N^h}$.

13. Уменьшение температурного коэффициента в соответствии с методом моделирования отжига

$T(k+1) = \beta T(k)$,

где β – коэффициент охлаждения, $0 < \beta < 1$.

14. Если $T(k) > T_{\min}$, то $k = k+1$, переход к 11.

15. $<\mathbf{x}2_\mu> = <\mathbf{x}>$. Если $\mu < P$, то $\mu = \mu + 1$, переход к 10.

16. Настройка синаптических весов на основе детерминированного правила Больцмана

$w_{ij}(n) = w_{ij}(n) + \eta(\rho_{ij}^+ - \rho_{ij}^-)$, $i, j \in \overline{1, N^{in} + N^{out} + N^h}$,

$$\rho_{ij}^+ = \frac{1}{P} \sum_{\mu=1}^{P} < x1_{\mu i} >< x1_{\mu j} >, \; \rho_{ij}^- = \frac{1}{P} \sum_{\mu=1}^{P} < x2_{\mu i} >< x2_{\mu j} >.$$

17. Если $\frac{1}{P}\sum_{\mu=1}^{P}\sum_{i=N^{in}+1}^{N^{in}+N^{out}} |< x1_{\mu i} > - < x2_{\mu i} >| > \varepsilon$, то $n = n+1$, переход к 2.

Функционирование ИНС

1. Инициализация посредством равномерного распределения на интервале (0,1) вещественного вектора состояний скрытых нейронов $< \mathbf{x}^h >= (< x_1^h >,...,< x_{N^h}^h >)$ и видимых выходных нейронов $< \mathbf{x}^{out} >= (< x_1^{out} >,...,< x_{N^{out}}^{out} >)$.

Положительная фаза (шаги 2-6)

2. $k = 1,\ T(k) = T_{\max}$.

$$< \mathbf{x} >= (< x_1^{in}, > ...,< x_{N^{in}}^{in} >,< x_1^{out} >,...,< x_{N^{out}}^{out} >,< x_1^h >,...,< x_{N^h}^h >).$$

3. Вычисление среднего значения состояний скрытых нейронов

$$< x_j >= \frac{1}{1+\exp\left(-\frac{1}{T(k)}\sum_{i=1}^{N^{in}+N^{out}+N^h} w_{ij} < x_i >\right)},$$

$j \in \overline{N^{in} + N^{out} + 1, N^{in} + N^{out} + N^h}$.

4. Уменьшение температурного коэффициента в соответствии с методом моделирования отжига

$$T(k+1) = \beta T(k),$$

где β – коэффициент охлаждения, $0 < \beta < 1$.

5. Если $T(k) > T_{\min}$, то $k = k+1$, переход к 3.

6. $< \mathbf{x}1 >=< \mathbf{x} >$.

Отрицательная фаза (шаги 7-11)

7. $k = 1,\ T(k) = T_{\max}$.

$< \mathbf{x} >=< \mathbf{x}1$.

8. Вычисление среднего значения состояний видимых выходных нейронов

$$< x_j >= \frac{1}{1+\exp\left(-\frac{1}{T(k)}\sum_{i=1}^{N^{in}+N^{out}+N^h} w_{ij}(n) < x_i >\right)},$$

$j \in \overline{N^{in} + 1, N^{in} + N^{out}}$.

9. Уменьшение температурного коэффициента в соответствии с методом моделирования отжига

$T(k+1) = \beta T(k)$,

где β – коэффициент охлаждения, $0 < \beta < 1$.

10. Если $T(k) > T_{\min}$, то $k = k+1$, переход к 8.

11. $< \mathbf{x}^{out} >= (< x_{N^{in}+1} >, \ldots, < x_{N^{in}+N^{out}} >)$.

Результатом является образец $< \mathbf{x}^{out} >$.

Замечание. В случае биполярных данных, т.е. $x_j \in \{-1,1\}$,
$$< x_j >= \frac{1}{1+\exp\left(-\frac{1}{T(k)}\sum_{i=1}^{N^{in}+N^{out}+N^h} w_{ij} < x_i >\right)}$$
заменяется на
$$< x_j >= \tanh\left(\frac{1}{T(k)}\sum_{i=1}^{N^{in}+N^{out}+N^h} w_{ij} < x_i >\right).$$

Достоинства

1. Используется для восстановления или классификации образцов.
2. Обеспечивает хорошее качество обобщения.
3. Количество классов может быть больше двух.
4. В отличие от DHNN гарантирует нахождение глобального минимума.
5. Распределение вероятности, представляемое нейронами, близко к распределению вероятности среды, в которой работает ИНС.
6. Автоматически определяется количество скрытых слоев (равно одному).
7. Детерминированная FBM обучается быстрее стохастической.

Недостатки

1. Отсутствует автоматическое определение числа нейронов в скрытом слое.
2. Обучение стохастической FBM происходит медленнее, чем в случае MLP.
3. В отличие от ART, не решает проблему пластичности-стабильности.
4. Стохастическая FBM работает только с биполярными или бинарными данными.

5. Детерминированная FBM является только дескриптивной моделью.

7.8. Ограниченная машина Больцмана

На рис.7.9, 7.10 представлена ограниченная машина Больцмана (RBM) [71,72], которая является рекуррентной ИНС и содержит один видимый слой и один скрытый слой.

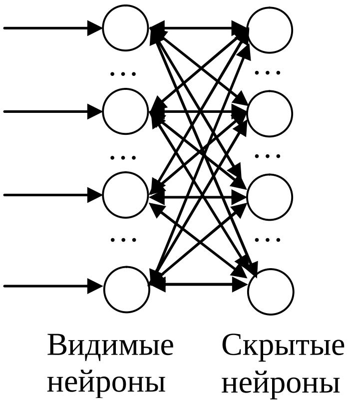


Рис. 7.9. Ограниченная машина Больцмана (RBM) как генеративная модель

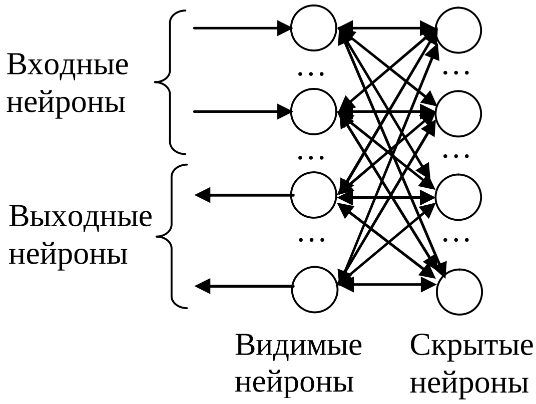


Рис. 7.10. Ограниченная машина Больцмана (RBM) как дескриптивная модель

В отличие от полной машины Больцмана (FBM) видимые нейроны не связаны между собой и скрытые нейроны не связаны между собой.

Генеративная RBM реализует автоассоциативную память (ее элемент представлен парой образцов $(\mathbf{m}_x, \mathbf{m}_y)$, $\mathbf{m}_y = \mathbf{m}_x$) и восстанавливает запомненный образец $\mathbf{m}_y$ по ключевому образцу $\mathbf{m}_x$, соответствующему входному вектору $\mathbf{x}$.

Дескриптивная RBM реализует гетероассоциативную память (ее элемент представлен парой образцов $(\mathbf{m}_x, \mathbf{m}_y)$, $\mathbf{m}_y \neq \mathbf{m}_x$) и восстанавливает запомненный образец $\mathbf{m}_y$ по ключевому образцу $\mathbf{m}_x$, соответствующему входному вектору $\mathbf{x}$.

Важнейшим свойством RBM является то, что одна и та же ИНС с одними и теми же весами связей может хранить и воспроизводить несколько различных запомненных образцов.

Для генеративной стохастической RBM используется стохастическое обучение (обучение без учителя). Для дескриптивной стохастической RBM используется стохастическое обучение

(обучение с учителем). Для дескриптивной детерминированной RBM используется обучение Хебба (обучение с учителем).

RBM работает в двух фазах – положительной и отрицательной.

Для генеративной стохастической RBM в положительной фазе фиксируются видимые нейроны, а ИНС функционирует до тех пор, пока не установятся скрытые нейроны. В отрицательной фазе вначале фиксируются обученные в положительной фазе скрытые нейроны, а ИНС функционирует до тех пор, пока не установятся видимые нейроны, после чего фиксируются обученные в отрицательной фазе видимые нейроны, а ИНС функционирует до тех пор, пока не установятся скрытые нейроны.

Для дескриптивной стохастической и детерминированной RBM в положительной фазе фиксируются видимые входные и выходные нейроны, а ИНС функционирует до тех пор, пока не установятся скрытые нейроны. В отрицательной фазе вначале фиксируются видимые входные нейроны и обученные в положительной фазе скрытые нейроны, а ИНС функционирует до тех пор, пока не установятся видимые выходные нейроны, после чего фиксируются видимые входные нейроны и обученные в отрицательной фазе видимые выходные нейроны, а ИНС функционирует до тех пор, пока не установятся скрытые нейроны.

Компонентами стохастической RBM являются стохастические нейроны, состояние которых описывается на основе распределения Бернулли в виде

$$x_j = \begin{cases} 1, & \text{с вероятностью } P_j \\ 0, & \text{с вероятностью } 1 - P_j \end{cases}.$$

Величина изменения состояния j-го стохастического нейрона определена в виде

$$\Delta x_j = \widehat{x}_j - x_j,$$

где $\widehat{x}_j$ – состояние, противоположное состоянию x_j, т.е. $\widehat{x}_j = 1 - x_j$.

Вероятность перехода j-го стохастического нейрона из текущего состояния x_j в новое состояние $\widehat{x}_j$ определяется на основе имитации отжига в виде

$$P_j = \frac{1}{1 + \exp\left(-\frac{\Delta E_j}{T}\right)},$$

где ΔE_j – приращение энергии сети при изменении состояния j-го нейрона, T – температура.

Компонентами детерминированной RBM являются детерминированные нейроны, $< x_j > \in (0,1)$.

7.8.1. Генеративная стохастическая ограниченная машина Больцмана

Обучение ИНС

1. Номер итерации обучения $n=1$, инициализация посредством равномерного распределения на интервале (0,1) или [-0.5, 0.5] весов $w_{ij}(n)$, $w_{ii}(n)=0$, $w_{ij}(n)=w_{ji}(n)$, $i \in \overline{1, N^v}$, $j \in \overline{N^v+1, N^v+N^h}$. Задание параметра P_0, температурных параметров $T_{\max}, T_{\min}$.

2. Инициализация бинарного вектора состояний скрытых нейронов $\mathbf{x}^h = (x_1^h, ..., x_{N^h}^h)$. Задается обучающее множество $\{\mathbf{x}_\mu^v \mid \mathbf{x}_\mu^v \in \{0,1\}^{N^v}\}$, $\mu \in \overline{1, P}$, где $\mathbf{x}_\mu^v$ – μ-й обучающий вектор состояний видимых нейронов, P – мощность обучающего множества.

Положительная фаза (шаги 3-9)

3. μ=1.

4. $k=1$, $T(k)=T_{\max}$.

$\mathbf{x} = (x_{\mu 1}^v, ..., x_{\mu N^v}^v, x_1^h, ..., x_{N^h}^h)$.

5. Квантование Гиббса для скрытых нейронов ($j \in \overline{N^v+1, N^v+N^h}$)

$\Delta x_j = \widehat{x}_j - x_j$.

$$\Delta E_j = -\Delta x_j \sum_{i=1}^{N^v} w_{ij}(n) x_i \ .$$

$$P_j = \frac{1}{1+\exp\left(-\frac{\Delta E_j}{T(k)}\right)}.$$

Если $\Delta E_j \le 0$, то $x_j = \widehat{x}_j$.

Если $\Delta E_j > 0$ и $P_j \ge P_0$, то $x_j = \widehat{x}_j$.

6. Если имеются скрытые нейроны, которые изменили состояние, то переход к 5.

7. Уменьшение температурного коэффициента в соответствии с методом моделирования отжига

$T(k+1) = \beta T(k)$,

где β – коэффициент охлаждения, $0 < \beta < 1$.

8. Если $T(k) > T_{\min}$, то $k = k+1$, переход к 5.

9. $\mathbf{x}1_{\mu} = \mathbf{x}$. Если $\mu < P$, то $\mu = \mu + 1$, переход к 4.

Отрицательная фаза (шаги 10-18)

10. μ=1.

11. $k = 1$, $T(k) = T_{\max}$.

$\mathbf{x} = \mathbf{x}1_{\mu}$.

12. Квантование Гиббса для видимых нейронов ($j \in \overline{1, N^v}$)

$\Delta x_j = \hat{x}_j - x_j$.

$$\Delta E_j = -\Delta x_j \sum_{i=N^v+1}^{N^v+N^h} w_{ij}(n) x_i \ .$$

$$P_j = \frac{1}{1 + \exp\left(-\frac{\Delta E_j}{T(k)}\right)}.$$

Если $\Delta E_j \le 0$, то $x_j = \hat{x}_j$.

Если $\Delta E_j > 0$ и $P_j \ge P_0$, то $x_j = \hat{x}_j$.

13. Если имеются видимые нейроны, которые изменили состояние, то переход к 12.

14. Квантование Гиббса для скрытых нейронов ($j \in \overline{N^v + 1, N^v + N^h}$)

$\Delta x_j = \hat{x}_j - x_j$.

$$\Delta E_j = -\Delta x_j \sum_{i=1}^{N^v} w_{ij}(n) x_i \ .$$

$$P_j = \frac{1}{1 + \exp\left(-\frac{\Delta E_j}{T(k)}\right)}.$$

Если $\Delta E_j \le 0$, то $x_j = \hat{x}_j$.

Если $\Delta E_j > 0$ и $P_j \ge P_0$, то $x_j = \hat{x}_j$.

15. Если имеются скрытые нейроны, которые изменили состояние, то переход к 14.

16. Уменьшение температурного коэффициента в соответствии с методом моделирования отжига

$$T(k+1) = \beta T(k),$$

где β – коэффициент охлаждения, $0 < \beta < 1$.

17. Если $T(k) > T_{\min}$, то $k = k+1$, переход к 12.

18. $\mathbf{x}2_{\mu} = \mathbf{x}$. Если $\mu < P$, то $\mu = \mu + 1$, переход к 11.

19. Настройка синаптических весов на основе стохастического правила Больцмана

$$w_{ij}(n) = w_{ij}(n) + \eta(\rho_{ij}^{+} - \rho_{ij}^{-}),\ i \in \overline{1, N^{v}},\ j \in \overline{N^{v}+1, N^{v}+N^{h}},$$

$$\rho_{ij}^{+} = \frac{1}{P}\sum_{\mu=1}^{P}\varphi(x1_{\mu i}, x1_{\mu j}),\ \rho_{ij}^{-} = \frac{1}{P}\sum_{\mu=1}^{P}\varphi(x2_{\mu i}, x2_{\mu j}),$$

$$\varphi(x_i, x_j) = \begin{cases} 1, & x_i = x_j \\ 0, & x_i \neq x_j \end{cases}.$$

20. Если $\frac{1}{P}\sum_{\mu=1}^{P}\sum_{i=1}^{N^{v}}|\, x1_{\mu i}^{v} - x2_{\mu i}^{v} \,| > \varepsilon$, то $n = n+1$, переход к 2.

Функционирование ИНС

1. Инициализация бинарного вектора состояний скрытых нейронов $\mathbf{x}^{h} = (x_1^{h}, ..., x_{N^h}^{h})$.

Положительная фаза (шаги 2-7)

2. $k = 1$, $T(k) = T_{\max}$.

$$\mathbf{x} = (x_1^{v}, ..., x_{N^v}^{v}, x_1^{h}, ..., x_{N^h}^{h}).$$

3. Квантование Гиббса для скрытых нейронов ($j \in \overline{N^{v}+1, N^{v}+N^{h}}$)

$$\Delta x_j = \widehat{x}_j - x_j.$$

$$\Delta E_j = -\Delta x_j \sum_{i=1}^{N^{v}} w_{ij} x_i .$$

$$P_j = \frac{1}{1 + \exp\left(-\frac{\Delta E_j}{T(k)}\right)}.$$

Если $\Delta E_j \le 0$, то $x_j = \hat{x}_j$.

Если $\Delta E_j > 0$ и $P_j \ge P_0$, то $x_j = \hat{x}_j$.

4. Если имеются скрытые нейроны, которые изменили состояние, то переход к 3.

5. Уменьшение температурного коэффициента в соответствии с методом моделирования отжига

$$T(k+1) = \beta T(k),$$

где β – коэффициент охлаждения, $0 < \beta < 1$.

6. Если $T(k) > T_{\min}$, то $k = k+1$, переход к 3.

7. $\mathbf{x}1 = \mathbf{x}$.

Отрицательная фаза (шаги 8-13)

8. $k = 1$, $T(k) = T_{\max}$.

$\mathbf{x} = \mathbf{x}1$.

9. Квантование Гиббса для видимых нейронов ($j \in \overline{1, N^v}$)

$$\Delta x_j = \hat{x}_j - x_j.$$

$$\Delta E_j = -\Delta x_j \sum_{i=N^v+1}^{N^v+N^h} w_{ij} x_i .$$

$$P_j = \frac{1}{1 + \exp\left(-\frac{\Delta E_j}{T(k)}\right)}.$$

Если $\Delta E_j \le 0$, то $x_j = \hat{x}_j$.

Если $\Delta E_j > 0$ и $P_j \ge P_0$, то $x_j = \hat{x}_j$.

10. Если имеются видимые нейроны, которые изменили состояние, то переход к 9.

11. Уменьшение температурного коэффициента в соответствии с методом моделирования отжига

$$T(k+1) = \beta T(k),$$

где β – коэффициент охлаждения, $0 < \beta < 1$.

12. Если $T(k) > T_{\min}$, то $k = k+1$, переход к 9.

13. $\mathbf{x}^v = (x_1, ..., x_{N^v})$.

Результатом является образец $\mathbf{x}^v$.

7.8.2. Дескриптивная стохастическая ограниченная машина Больцмана

Обучение ИНС

1. Номер итерации обучения $n=1$, инициализация посредством равномерного распределения на интервале (0,1) или [-0.5, 0.5] весов $w_{ij}(n)$, $w_{ii}(n)=0$, $w_{ij}(n)=w_{ji}(n)$, $i\in\overline{1,N^{in}+N^{out}}$, $j\in\overline{N^{in}+N^{out}+1,N^{in}+N^{out}+N^{h}}$. Задание параметра P_0, температурных параметров $T_{\max},T_{\min}$.

2. Инициализация бинарного вектора состояний скрытых нейронов $\mathbf{x}^h=(x_1^h,...,x_{N^h}^h)$. Задается обучающее множество $\{(\mathbf{x}_\mu^{in},\mathbf{x}_\mu^{out})\,|\,\mathbf{x}_\mu^{in}\in\{0,1\}^{N^{in}},\mathbf{x}_\mu^{out}\in\{0,1\}^{N^{out}}\}$, $\mu\in\overline{1,P}$, где $\mathbf{x}_\mu^{in}$ – μ-й обучающий вектор состояний видимых входных нейронов, $\mathbf{x}_\mu^{out}$ – μ-й обучающий вектор состояний видимых выходных нейронов, P – мощность обучающего множества.

Положительная фаза (шаги 3-9)

3. μ=1.

4. $k=1$, $T(k)=T_{\max}$.

$\mathbf{x}=(x_{\mu 1}^{in},...,x_{\mu N^{in}}^{in},x_{\mu 1}^{out},...,x_{\mu N^{out}}^{out},x_1^h,...,x_{N^h}^h)$.

5. Квантование Гиббса для скрытых нейронов ($j\in\overline{N^{in}+N^{out}+1,N^{in}+N^{out}+N^{h}}$)

$\Delta x_j=\hat{x}_j-x_j$.

$$\Delta E_j=-\Delta x_j\sum_{i=1}^{N^{in}+N^{out}}w_{ij}(n)x_i\ .$$

$$P_j=\frac{1}{1+\exp\left(-\dfrac{\Delta E_j}{T(k)}\right)}.$$

Если $\Delta E_j\le 0$, то $x_j=\hat{x}_j$.

Если $\Delta E_j>0$ и $P_j\ge P_0$, то $x_j=\hat{x}_j$.

6. Если имеются скрытые нейроны, которые изменили состояние, то переход к 5.

7. Уменьшение температурного коэффициента в соответствии с методом моделирования отжига

$T(k+1)=\beta T(k)$,

где β – коэффициент охлаждения, $0<\beta<1$.

8. Если $T(k)>T_{\min}$, то $k=k+1$, переход к 5.

9. $\mathbf{x}1_{\mu}=\mathbf{x}$. Если $\mu<P$, то $\mu=\mu+1$, переход к 4.

Отрицательная фаза (шаги 10-18)

10. μ=1.

11. $k=1$, $T(k)=T_{\max}$.

$\mathbf{x}=\mathbf{x}1_{\mu}$.

12. Квантование Гиббса для видимых выходных нейронов ($j\in\overline{N^{in}+1,N^{in}+N^{out}}$)

$$\Delta x_j=\hat{x}_j-x_j.$$

$$\Delta E_j=-\Delta x_j\sum_{i=N^{in}+N^{out}+1}^{N^{in}+N^{out}+N^h}w_{ij}(n)x_i\ .$$

$$P_j=\frac{1}{1+\exp\left(-\frac{\Delta E_j}{T(k)}\right)}.$$

Если $\Delta E_j\le 0$, то $x_j=\hat{x}_j$.

Если $\Delta E_j>0$ и $P_j\ge P_0$, то $x_j=\hat{x}_j$.

13. Если имеются видимые выходные нейроны, которые изменили состояние, то переход к 12.

14. Квантование Гиббса для скрытых нейронов ($j\in\overline{N^{in}+N^{out}+1,N^{in}+N^{out}+N^h}$)

$$\Delta x_j=\hat{x}_j-x_j.$$

$$\Delta E_j=-\Delta x_j\sum_{i=1}^{N^{in}+N^{out}}w_{ij}(n)x_i\ .$$

$$P_j=\frac{1}{1+\exp\left(-\frac{\Delta E_j}{T(k)}\right)}.$$

Если $\Delta E_j\le 0$, то $x_j=\hat{x}_j$.

Если $\Delta E_j > 0$ и $P_j \geq P_0$, то $x_j = \hat{x}_j$.

15. Если имеются скрытые нейроны, которые изменили состояние, то переход к 14.

16. Уменьшение температурного коэффициента в соответствии с методом моделирования отжига

$T(k+1) = \beta T(k)$,

где β – коэффициент охлаждения, $0 < \beta < 1$.

17. Если $T(k) > T_{\min}$, то $k = k+1$, переход к 12.

18. $\mathbf{x}2_\mu = \mathbf{x}$. Если $\mu < P$, то $\mu = \mu + 1$, переход к 11.

19. Настройка синаптических весов на основе стохастического правила Больцмана

$$w_{ij}(n) = w_{ij}(n) + \eta(\rho_{ij}^{+} - \rho_{ij}^{-}), \quad i \in \overline{1, N^{in} + N^{out}},$$

$$j \in \overline{N^{in} + N^{out} + 1, N^{in} + N^{out} + N^{h}},$$

$$\rho_{ij}^{+} = \frac{1}{P}\sum_{\mu=1}^{P}\varphi(x1_{\mu i}, x1_{\mu j}), \ \rho_{ij}^{-} = \frac{1}{P}\sum_{\mu=1}^{P}\varphi(x2_{\mu i}, x2_{\mu j}),$$

$$\varphi(x_i, x_j) = \begin{cases} 1, & x_i = x_j \\ 0, & x_i \neq x_j \end{cases}.$$

20. Если $\frac{1}{P}\sum_{\mu=1}^{P}\sum_{i=N^{in}+1}^{N^{in}+N^{out}} | x1_{\mu i} - x2_{\mu i} | > \varepsilon$, то $n = n+1$, переход к 2.

Функционирование ИНС

1. Инициализация бинарного вектора состояний скрытых нейронов $\mathbf{x}^h = (x_1^h, ..., x_{N^h}^h)$ и видимых выходных нейронов $\mathbf{x}^{out} = (x_1^{out}, ..., x_{N^{out}}^{out})$,

Положительная фаза (шаги 2-7)

2. $k = 1$, $T(k) = T_{\max}$.

$$\mathbf{x} = (x_1^{in}, ..., x_{N^{in}}^{in}, x_1^{out}, ..., x_{N^{out}}^{out}, x_1^{h}, ..., x_{N^h}^{h}).$$

3. Квантование Гиббса для скрытых нейронов ($j \in \overline{N^{in} + N^{out} + 1, N^{in} + N^{out} + N^{h}}$)

$$\Delta x_j = \hat{x}_j - x_j.$$

$$\Delta E_j = -\Delta x_j \sum_{i=1}^{N^{in}+N^{out}} w_{ij} x_i \ .$$

$$P_j = \frac{1}{1+\exp\left(-\frac{\Delta E_j}{T(k)}\right)}.$$

Если $\Delta E_j \le 0$, то $x_j = \hat{x}_j$.

Если $\Delta E_j > 0$ и $P_j \ge P_0$, то $x_j = \hat{x}_j$.

4. Если имеются скрытые нейроны, которые изменили состояние, то переход к 3.

5. Уменьшение температурного коэффициента в соответствии с методом моделирования отжига

$T(k+1) = \beta T(k)$,

где β – коэффициент охлаждения, $0 < \beta < 1$.

6. Если $T(k) > T_{\min}$, то $k = k+1$, переход к 3.

7. $\mathbf{x}1 = \mathbf{x}$.

Отрицательная фаза (шаги 8-13)

8. $k = 1$, $T(k) = T_{\max}$.

$\mathbf{x} = \mathbf{x}1$.

9. Квантование Гиббса для видимых выходных нейронов ($j \in \overline{N^{in}+1, N^{in}+N^{out}}$)

$\Delta x_j = \hat{x}_j - x_j$.

$$\Delta E_j = -\Delta x_j \sum_{i=N^{in}+N^{out}+1}^{N^{in}+N^{out}+N^h} w_{ij} x_i \ .$$

$$P_j = \frac{1}{1+\exp\left(-\frac{\Delta E_j}{T(k)}\right)}.$$

Если $\Delta E_j \le 0$, то $x_j = \hat{x}_j$.

Если $\Delta E_j > 0$ и $P_j \ge P_0$, то $x_j = \hat{x}_j$.

10. Если имеются видимые выходные нейроны, которые изменили состояние, то переход к 9.

11. Уменьшение температурного коэффициента в соответствии с методом моделирования отжига

$T(k+1) = \beta T(k)$,

где β – коэффициент охлаждения, $0 < \beta < 1$.

12. Если $T(k) > T_{\min}$, то $k = k+1$, переход к 9.

13. $\mathbf{x}^{out} = (x_{N^{in}+1}, ..., x_{N^{in}+N^{out}})$.

Результатом является образец $\mathbf{x}^{out}$.

7.8.3. Дескриптивная детерминированная ограниченная машина Больцмана

Обучение ИНС

1. Номер итерации обучения $n=1$, инициализация посредством равномерного распределения на интервале (0,1) или [-0.5, 0.5] весов $w_{ij}(n)$, $w_{ii}(n)=0$, $w_{ij}(n)=w_{ji}(n)$, $i \in \overline{1, N^{in}+N^{out}}$, $j \in \overline{N^{in}+N^{out}+1, N^{in}+N^{out}+N^h}$.. Задание параметра P_0, температурных параметров $T_{\max}, T_{\min}$.

2. Инициализация посредством равномерного распределения на интервале (0,1) вещественного вектора состояний скрытых нейронов $<\mathbf{x}^h> = (<x_1^h>, ..., <x_{N^h}^h>)$. Задается обучающее множество $\{(<\mathbf{x}_\mu^{in}>, <\mathbf{x}_\mu^{out}>) \,|\, <\mathbf{x}_\mu^{in}> \in (0,1)^{N^{in}}, <\mathbf{x}_\mu^{out}> \in (0,1)^{N^{out}}\}$, $\mu \in \overline{1, P}$, где $<\mathbf{x}_\mu^{in}>$ – μ-й обучающий вектор состояний видимых входных нейронов, $<\mathbf{x}_\mu^{out}>$ – μ-й обучающий вектор состояний видимых выходных нейронов, P – мощность обучающего множества.

Положительная фаза (шаги 3-8)

3. μ=1.

4. $k=1$, $T(k) = T_{\max}$.

$$<\mathbf{x}> = (<x_{\mu 1}^{in}>, ..., <x_{\mu N^{in}}^{in}>, <x_{\mu 1}^{out}>, ..., <x_{\mu N^{out}}^{out}>, <x_1^h>, ..., <x_{N^h}^h>).$$

5. Вычисление среднего значения состояний скрытых нейронов

$$<x_j> = \frac{1}{1+\exp\left(-\frac{1}{T(k)}\sum_{i=1}^{N^{in}+N^{out}} w_{ij}(n) <x_i>\right)},$$

$j \in \overline{N^{in}+N^{out}+1, N^{in}+N^{out}+N^h}$.

6. Уменьшение температурного коэффициента в соответствии с методом моделирования отжига

$T(k+1)=\beta T(k)$,

где β – коэффициент охлаждения, $0<\beta<1$.

7. Если $T(k)>T_{\min}$, то $k=k+1$, переход к 5.

8. $<\mathbf{x}1_{\mu}>=<\mathbf{x}>$. Если $\mu<P$, то $\mu=\mu+1$, переход к 4.

Отрицательная фаза (шаги 9-15)

9. $\mu=1$.

10. $k=1$, $T(k)=T_{\max}$.

$<\mathbf{x}>=<\mathbf{x}1_{\mu}>$.

11. Вычисление среднего значения состояний видимых выходных нейронов

$$<x_j>=\frac{1}{1+\exp\left(-\frac{1}{T(k)}\sum_{i=N^{in}+N^{out}+1}^{N^{in}+N^{out}+N^h} w_{ij}(n)<x_i>\right)},$$

$j\in\overline{N^{in}+1,N^{in}+N^{out}}$

12. Вычисление среднего значения состояний скрытых нейронов

$$<x_j>=\frac{1}{1+\exp\left(-\frac{1}{T(k)}\sum_{i=1}^{N^{in}+N^{out}} w_{ij}(n)<x_i>\right)},$$

$j\in\overline{N^{in}+N^{out}+1,N^{in}+N^{out}+N^h}$.

13. Уменьшение температурного коэффициента в соответствии с методом моделирования отжига

$T(k+1)=\beta T(k)$,

где β – коэффициент охлаждения, $0<\beta<1$.

14. Если $T(k)>T_{\min}$, то $k=k+1$, переход к 11.

15. $<\mathbf{x}2_{\mu}>=<\mathbf{x}>$. Если $\mu<P$, то $\mu=\mu+1$, переход к 10.

16. Настройка синаптических весов на основе детерминированного правила Больцмана

$w_{ij}(n)=w_{ij}(n)+\eta(\rho_{ij}^{+}-\rho_{ij}^{-})$, $i\in\overline{1,N^{in}+N^{out}}$, $j\in\overline{N^{in}+N^{out}+1,N^{in}+N^{out}+N^h}$,

$$\rho_{ij}^{+}=\frac{1}{P}\sum_{\mu=1}^{P}<x1_{\mu i}><x1_{\mu j}>,\ \rho_{ij}^{-}=\frac{1}{P}\sum_{\mu=1}^{P}<x2_{\mu i}><x2_{\mu j}>.$$

17. Если $\frac{1}{P}\sum_{\mu=1}^{P}\sum_{i=N^{in}+1}^{N^{in}+N^{out}}|< x1_{\mu i} > - < x2_{\mu i} >| > \varepsilon$, то $n = n+1$, переход к 2.

Функционирование ИНС

1. Инициализация посредством равномерного распределения на интервале (0,1) вещественного вектора состояний скрытых нейронов $< \mathbf{x}^h >= (< x_1^h >,...,< x_{N^h}^h >)$ и видимых выходных нейронов $< \mathbf{x}^{out} >= (< x_1^{out} >,...,< x_{N^{out}}^{out} >)$.

Положительная фаза (шаги 2-6)

2. $k = 1,\ T(k) = T_{\max}$.

$< \mathbf{x} >= (< x_1^{in} >,...,< x_{N^{in}}^{in} >,< x_1^{out} >,...,< x_{N^{out}}^{out} >,< x_1^{h} >,...,< x_{N^{h}}^{h} >)$.

3. Вычисление среднего значения состояний скрытых нейронов

$$< x_j >= \frac{1}{1+\exp\left(-\frac{1}{T(k)}\sum_{i=1}^{N^{in}+N^{out}} w_{ij} < x_i >\right)},$$

$j \in \overline{N^{in}+N^{out}+1, N^{in}+N^{out}+N^h}$.

4. Уменьшение температурного коэффициента в соответствии с методом моделирования отжига

$T(k+1) = \beta T(k)$,

где β – коэффициент охлаждения, $0 < \beta < 1$.

5. Если $T(k) > T_{\min}$, то $k = k+1$, переход к 3.

6. $< \mathbf{x}1 >=< \mathbf{x} >$.

Отрицательная фаза (шаги 7-11)

7. $k = 1,\ T(k) = T_{\max}$.

$< \mathbf{x} >=< \mathbf{x}1 >$.

8. Вычисление среднего значения состояний видимых выходных нейронов

$$< x_j >= \frac{1}{1+\exp\left(-\frac{1}{T(k)}\sum_{i=N^{in}+N^{out}+1}^{N^{in}+N^{out}+N^h} w_{ij} < x_i >\right)},$$

$j \in \overline{N^{in}+1, N^{in}+N^{out}}$

9. Уменьшение температурного коэффициента в соответствии с методом моделирования отжига

$T(k+1) = \beta T(k)$,

где β – коэффициент охлаждения, $0 < \beta < 1$.

10. Если $T(k) > T_{\min}$, то $k = k+1$, переход к 8.

11. $< \mathbf{x}^{out} >= (< x_{N^{in}+1} >, ..., < x_{N^{in}+N^{out}} >)$.

Результатом является образец $< \mathbf{x}^{out} >$.

Замечание. Существуют также полуограниченные машины Больцмана, у которых видимые нейроны связаны между собой, а скрытые нейроны не связаны между собой.

Достоинства

1. Используется для восстановления или классификации образцов.
2. Обеспечивает хорошее качество обобщения.
3. Количество классов может быть больше двух.
4. В отличие от DHNN гарантирует нахождение глобального минимума.
5. Распределение вероятности, представляемое нейронами, близко к распределению вероятности среды, в которой работает ИНС.
6. Автоматически определяется количество скрытых слоев (равно одному).
7. Обучение RBM происходит быстрее, чем BM.
8. Детерминированная RBM обучается быстрее стохастической.

Недостатки

1. Отсутствует автоматическое определение числа нейронов в скрытом слое.
2. Обучение стохастической BM происходит медленнее, чем в случае MLP.
3. В отличие от ART, не решает проблему пластичности-стабильности.
4. Стохастическая RBM работает только с биполярными или бинарными данными.
5. Детерминированная RBM является только дескриптивной моделью.

7.9. Условная ограниченная машина Больцмана

На рис.7.11, 7.12 представлена условная ограниченная машина Больцмана (CRBM) [73], которая является рекуррентной ИНс. и содержит видимый слой с временной задержкой и один скрытый слой.

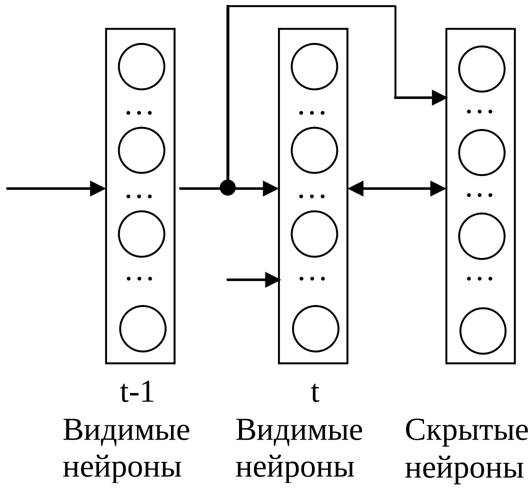


Рис. 7.11. Условная ограниченная машина Больцмана (CRBM) как генеративная модель

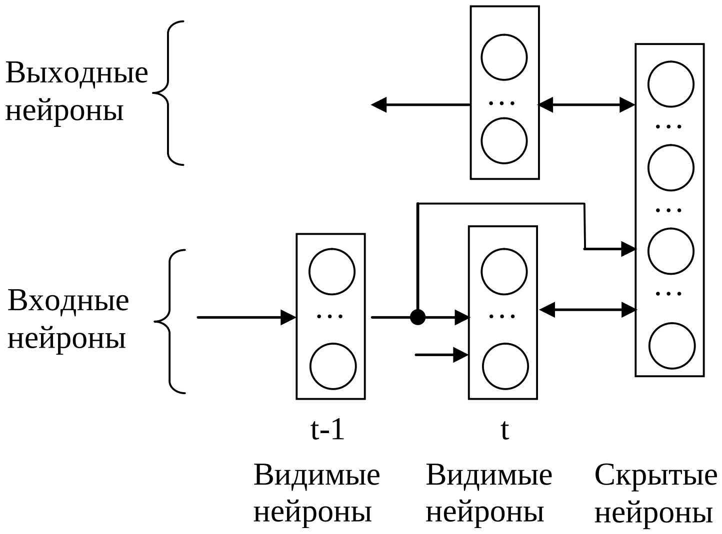


Рис. 7.12. Условная ограниченная машина Больцмана (CRBM) как дескриптивная модель

В отличие от RBM видимый слой состоит из входных и выходных видимых нейронов.

Генеративная CRBM реализует автоассоциативную память (ее элемент представлен парой образцов $(\mathbf{m}_x, \mathbf{m}_y)$, $\mathbf{m}_y = \mathbf{m}_x$) и восстанавливает запомненный образец $\mathbf{m}_y$ по ключевому образцу $\mathbf{m}_x$, соответствующему входному вектору $\mathbf{x}$.

Дескриптивная CRBM реализует гетероассоциативную память (ее элемент представлен парой образцов $(\mathbf{m}_x, \mathbf{m}_y)$, $\mathbf{m}_y \neq \mathbf{m}_x$) и восстанавливает запомненный образец $\mathbf{m}_y$ по ключевому образцу $\mathbf{m}_x$, соответствующему входному вектору $\mathbf{x}$.

Важнейшим свойством CRBM является то, что одна и та же ИНС с одними и теми же весами связей может хранить и воспроизводить несколько различных запомненных образцов.

Для генеративной стохастической CRBM используется стохастическое обучение (обучение без учителя). Для дескриптивной стохастической CRBM используется стохастическое обучение (обучение с учителем). Для дескриптивной детерминированной CRBM используется обучение Хебба (обучение с учителем).

CRBM работает в двух фазах – положительной и отрицательной.

Для генеративной стохастической CRBM в положительной фазе фиксируются видимые нейроны в моменты времени от $t - M^v$ до t, а ИНС функционирует до тех пор, пока не установятся скрытые нейроны. В отрицательной фазе вначале фиксируются видимые нейроны в моменты времени от $t - M^v$ до $t-1$ и обученные в положительной фазе скрытые нейроны, а ИНС функционирует до тех пор, пока не установятся видимые нейроны в момент времени t, после чего фиксируются видимые нейроны в моменты времени от $t - M^v$ до $t-1$ и обученные в отрицательной фазе видимые нейроны в момент времени t, а ИНС функционирует до тех пор, пока не установятся скрытые нейроны.

Для дескриптивной стохастической и детерминированной CRBM в положительной фазе фиксируются видимые входные нейроны в моменты времени от $t - M^{in}$ до t и видимые выходные нейроны в момент времени t, а ИНС функционирует до тех пор, пока не установятся скрытые нейроны. В отрицательной фазе вначале фиксируются видимые входные и выходные нейроны в моменты времени от $t - M^{in}$ до $t-1$, и обученные в положительной фазе скрытые нейроны, а ИНС функционирует до тех пор, пока не установятся видимые входные и выходные нейроны в момент времени t, после чего фиксируются видимые входные нейроны в моменты времени от $t-1$ до $t - M^{in}$, и обученные в отрицательной фазе видимые входные и выходные нейроны в момент времени t, а

ИНС функционирует до тех пор, пока не установятся скрытые нейроны.

Компонентами стохастической CRBM являются стохастические нейроны, состояние которых описывается на основе распределения Бернулли в виде

$$x_j = \begin{cases} 1, & \text{с вероятностью } P_j \\ 0, & \text{с вероятностью } 1 - P_j \end{cases}.$$

Величина изменения состояния j-го стохастического нейрона определена в виде

$$\Delta x_j = \widehat{x}_j - x_j,$$

где $\widehat{x}_j$ – состояние, противоположное состоянию x_j, т.е. $\widehat{x}_j = 1 - x_j$.

Вероятность перехода j-го стохастического нейрона из текущего состояния x_j в новое состояние $\widehat{x}_j$ определяется на основе имитации отжига в виде

$$P_j = \frac{1}{1 + \exp\left(-\frac{\Delta E_j}{T}\right)},$$

где ΔE_j – приращение энергии сети при изменении состояния j-го нейрона, T – температура.

Компонентами детерминированной CRBM являются детерминированные нейроны, $< x_j > \in (0,1)$.

7.9.1. Генеративная стохастическая условная ограниченная машина Больцмана

Обучение ИНС

1. Номер итерации обучения $n = 1$, инициализация посредством равномерного распределения на интервале (0,1) или [-0.5, 0.5] весов $w_{tij}^{v-h}(n)$, $t \in \overline{0, M^v}$, $i \in \overline{1, N^v}$, $j \in \overline{1, N^h}$, $w_{tij}^{v-v}(n)$, $t \in \overline{1, M^v}$, $i, j \in \overline{1, N^v}$, $w_{tii}^{v-h}(n) = 0$, $w_{tii}^{v-v}(n) = 0$, $w_{0ij}^{v-h}(n) = w_{0ji}^{v-h}(n)$, где M^v – задержка в видимом слое. Задание параметра P_0, температурных параметров $T_{\max}, T_{\min}$.

2. Инициализация бинарного вектора состояний скрытых нейронов $\mathbf{x}^h = (x_1^h, ..., x_{N^h}^h)$. Задается обучающее множество

$\{\mathbf{x}_\mu^v \mid \mathbf{x}_\mu^v \in \{0,1\}^{N^v}\}$, $\mu \in \overline{1,P}$, где $\mathbf{x}_\mu^v$ – μ-й обучающий вектор состояний видимых нейронов, P – мощность обучающего множества.

Положительная фаза (шаги 3-9)

3. $\mu = M^v + 1$.

4. $k = 1$, $T(k) = T_{\max}$.

$\mathbf{x}^v(\mu - t) = \mathbf{x}_{\mu-t}^v$, $t \in \overline{0, M^v}$.

5. Квантование Гиббса для скрытых нейронов ($j \in \overline{1, N^h}$) в момент времени μ

$$\Delta x_j^h(\mu) = \hat{x}_j^h(\mu) - x_j^h(\mu).$$

$$\Delta E_j = -\Delta x_j^h(\mu)\left(\sum_{t=0}^{M^v}\sum_{i=1}^{N^v} w_{tij}^{v-h}(n) x_i^v(\mu - t)\right).$$

$$P_j = \frac{1}{1 + \exp\left(-\frac{\Delta E_j}{T(k)}\right)}.$$

Если $\Delta E_j \le 0$, то $x_j^h(\mu) = \hat{x}_j^h(\mu)$.

Если $\Delta E_j > 0$ и $P_j \ge P_0$, то $x_j^h(\mu) = \hat{x}_j^h(\mu)$.

6. Если имеются скрытые нейроны, которые изменили состояние, то переход к 5.

7. Уменьшение температурного коэффициента в соответствии с методом моделирования отжига

$$T(k+1) = \beta T(k),$$

где β – коэффициент охлаждения, $0 < \beta < 1$.

8. Если $T(k) > T_{\min}$, то $k = k + 1$, переход к 5.

9. $\mathbf{x}1^v(\mu) = \mathbf{x}^v(\mu)$, $\mathbf{x}1^h(\mu) = \mathbf{x}^h(\mu)$. Если $\mu < P$, то $\mu = \mu + 1$, переход к 4.

Отрицательная фаза (шаги 10-16)

10. μ=1.

11. $k = 1$, $T(k) = T_{\max}$.

$\mathbf{x}^v(\mu - t) = \mathbf{x}_{\mu-t}^v$, $t \in \overline{0, M^v}$, $\mathbf{x}^h(\mu) = \mathbf{x}1_\mu^h(\mu)$.

12. Квантование Гиббса для видимых нейронов ($j \in \overline{1, N^v}$) в момент времени μ

$$\Delta x_j^v(\mu) = \hat{x}_j^v(\mu) - x_j^v(\mu).$$

$$\Delta E_j = -\Delta x_j^v(\mu) \left(\sum_{t=1}^{M^v} \sum_{i=1}^{N^v} w_{tij}^{v-v}(n) x_i^v(\mu - t) + \sum_{i=1}^{N^h} w_{0ij}^{v-h}(n) x_i^h(\mu) \right).$$

$$P_j = \frac{1}{1 + \exp\left(-\dfrac{\Delta E_j}{T(k)} \right)}.$$

Если $\Delta E_j \le 0$, то $x_j^v(\mu) = \hat{x}_j^v(\mu)$.

Если $\Delta E_j > 0$ и $P_j \ge P_0$, то $x_j^v(\mu) = \hat{x}_j^v(\mu)$.

13. Если имеются видимые выходные нейроны, которые изменили состояние, то переход к 12.

14. Квантование Гиббса для скрытых нейронов ($j \in \overline{1, N^h}$) в момент времени μ

$$\Delta x_j^h(\mu) = \hat{x}_j^h(\mu) - x_j^h(\mu).$$

$$\Delta E_j = -\Delta x_j^h(\mu) \left(\sum_{t=0}^{M^v} \sum_{i=1}^{N^v} w_{tij}^{v-h}(n) x_i^v(\mu - t) \right).$$

$$P_j = \frac{1}{1 + \exp\left(-\dfrac{\Delta E_j}{T(k)} \right)}.$$

Если $\Delta E_j \le 0$, то $x_j^h(\mu) = \hat{x}_j^h(\mu)$.

Если $\Delta E_j > 0$ и $P_j \ge P_0$, то $x_j^h(\mu) = \hat{x}_j^h(\mu)$.

15. Если имеются скрытые нейроны, которые изменили состояние, то переход к 14.

16. Уменьшение температурного коэффициента в соответствии с методом моделирования отжига

$$T(k+1) = \beta T(k),$$

где β – коэффициент охлаждения, $0 < \beta < 1$.

17. Если $T(k) > T_{\min}$, то $k = k + 1$, переход к 12.

18. $\mathbf{x}2^{v}(\mu)=\mathbf{x}^{v}(\mu)$, $\mathbf{x}2^{h}(\mu)=\mathbf{x}^{h}(\mu)$. Если $\mu<P$, то $\mu=\mu+1$, переход к 11.

19. Настройка синаптических весов на основе стохастического правила Больцмана

$$w_{tij}^{v-h}(n)=w_{tij}^{v-h}(n)+\eta(\rho_{tij}^{+}-\rho_{tij}^{-}),\ t\in\overline{1,M^{v}},\ i\in\overline{1,N^{v}},\ j\in\overline{1,N^{h}},$$

$$\rho_{tij}^{+}=\frac{1}{P}\sum_{\mu=1}^{P}\varphi(x_{\mu-t,i}^{v},x1_{j}^{h}(\mu)),\ \rho_{tij}^{-}=\frac{1}{P}\sum_{\mu=1}^{P}\varphi(x_{\mu-t,i}^{v},x2_{j}^{h}(\mu)),$$

$$w_{0ij}^{v-h}(n)=w_{0ij}^{v-h}(n)+\eta(\rho_{0ij}^{+}-\rho_{0ij}^{-}),\ i\in\overline{1,N^{v}},\ j\in\overline{1,N^{h}},$$

$$\rho_{0ij}^{+}=\frac{1}{P}\sum_{\mu=1}^{P}\varphi(x1_{i}^{v}(\mu),x1_{j}^{h}(\mu)),\ \rho_{0ij}^{-}=\frac{1}{P}\sum_{\mu=1}^{P}\varphi(x2_{i}^{v}(\mu),x2_{j}^{h}(\mu)),$$

$$w_{tij}^{v-v}(n)=w_{tij}^{v-v}(n)+\eta(\rho_{tij}^{+}-\rho_{tij}^{-}),\ t\in\overline{1,M^{v}},\ i,j\in\overline{1,N^{v}},$$

$$\rho_{tij}^{+}=\frac{1}{P}\sum_{\mu=1}^{P}\varphi(x_{\mu-t,i}^{v},x1_{j}^{v}(\mu)),\ \rho_{tij}^{-}=\frac{1}{P}\sum_{\mu=1}^{P}\varphi(x_{\mu-t,i}^{v},x2_{j}^{v}(\mu)),$$

$$\varphi(x_i,x_j)=\begin{cases}1, & x_i=x_j\\ 0, & x_i\ne x_j\end{cases}.$$

20. Если $\frac{1}{P}\sum_{\mu=1}^{P}\sum_{i=1}^{N^{v}}|x1_{i}^{v}(\mu)-x2_{i}^{v}(\mu)|>\varepsilon$, то $n=n+1$, переход к 2.

Функционирование ИНС

1. Инициализация бинарного вектора состояний скрытых нейронов $\mathbf{x}^{h}=(x_1^{h},...,x_{N^{h}}^{h})$.

Положительная фаза (шаги 2-7)

2. $k=1$, $T(k)=T_{\max}$.

$$\mathbf{x}^{v}(M^{v}+1-t)=\mathbf{x}_{M^{v}+1-t}^{v},\ t\in\overline{1,M^{v}},\ \mathbf{x}^{v}(M^{v}+1)=\mathbf{x}_{M^{v}+1}^{v}.$$

5. Квантование Гиббса для скрытых нейронов ($j\in\overline{1,N^{h}}$) в момент времени $M^{v}+1$

$$\Delta x_{j}^{h}(M^{v}+1)=\widehat{x}_{j}^{h}(M^{v}+1)-x_{j}^{h}(M^{v}+1).$$

$$\Delta E_{j}=-\Delta x_{j}^{h}(M^{v}+1)\left(\sum_{t=0}^{M^{v}}\sum_{i=1}^{N^{v}}w_{tij}^{v-h}(n)x_{i}^{v}(M^{v}+1-t)\right).$$

$$P_j = \frac{1}{1+\exp\left(-\frac{\Delta E_j}{T(k)}\right)}.$$

Если $\Delta E_j \le 0$, то $x_j^h(M^v+1) = \hat{x}_j^h(M^v+1)$.

Если $\Delta E_j > 0$ и $P_j \ge P_0$, то $x_j^h(M^v+1) = \hat{x}_j^h(M^v+1)$.

4. Если имеются скрытые нейроны, которые изменили состояние, то переход к 3.

5. Уменьшение температурного коэффициента в соответствии с методом моделирования отжига

$$T(k+1) = \beta T(k),$$

где β – коэффициент охлаждения, $0 < \beta < 1$.

6. Если $T(k) > T_{\min}$, то $k = k+1$, переход к 3.

7. $\mathbf{x}1^v(M^v+1) = \mathbf{x}^v(M^v+1)$, $\mathbf{x}1^h(M^v+1) = \mathbf{x}^h(M^v+1)$.

Отрицательная фаза (шаги 8-13)

8. $k = 1$, $T(k) = T_{\max}$.

$$\mathbf{x}^v(M^v+1-t) = \mathbf{x}^v_{M^v+1-t},\ t \in \overline{0, M^v},\ \mathbf{x}^h(M^v+1) = \mathbf{x}1^h_\mu(M^v+1).$$

9. Квантование Гиббса для видимых нейронов ($j \in \overline{1, N^v}$) в момент времени M^v+1

$$\Delta x_j^v(M^v+1) = (\hat{x}_j^v(M^v+1) - x_j^v(M^v+1)).$$

$$\Delta E_j = -\Delta x_j^v(M^v+1)\left(\begin{array}{l}\sum_{t=1}^{M^v}\sum_{i=1}^{N^v} w_{tij}^{v-v}(n) x_i^v(M^v+1-t) + \\ + \sum_{i=1}^{N^h} w_{0ij}^{v-h}(n) x_i^h(M^v+1)\end{array}\right).$$

$$P_j = \frac{1}{1+\exp\left(-\frac{\Delta E_j}{T(k)}\right)}.$$

Если $\Delta E_j \le 0$, то $x_j^v(M^v+1) = \hat{x}_j^v(M^v+1)$.

Если $\Delta E_j > 0$ и $P_j \ge P_0$, то $x_j^v(M^v+1) = \hat{x}_j^v(M^v+1)$.

10. Если имеются видимые выходные нейроны, которые изменили состояние, то переход к 9.

11. Уменьшение температурного коэффициента в соответствии с методом моделирования отжига

$T(k+1)=\beta T(k)$,

где β – коэффициент охлаждения, $0<\beta<1$.

12. Если $T(k)>T_{\min}$, то $k=k+1$, переход к 9.

Результатом является образец $\mathbf{x}^{v}(M^{v}+1)$.

7.9.2. Дескриптивная стохастическая условная ограниченная машина Больцмана

Обучение ИНС

1. Номер итерации обучения $n=1$, инициализация посредством равномерного распределения на интервале (0,1) или [-0.5, 0.5] весов $w_{tij}^{in-h}(n)$, $t\in\overline{0,M^{in}}$, $i\in\overline{1,N^{in}}$, $j\in\overline{1,N^{h}}$, $w_{ij}^{out-h}(n)$, $i\in\overline{1,N^{out}}$, $j\in\overline{1,N^{h}}$, $w_{tij}^{in-in}(n)$, $t\in\overline{1,M^{in}}$, $i,j\in\overline{1,N^{in}}$, $w_{tii}^{in-h}(n)=0$, $w_{ii}^{out-h}(n)=0$, $w_{tii}^{in-in}(n)=0$, $w_{0ij}^{in-h}(n)=w_{0ji}^{in-h}(n)$, $w_{ij}^{out-h}(n)=w_{ji}^{out-h}(n)$, где M^{in} – задержка для видимых входных нейронов. Задание параметра P_0, температурных параметров $T_{\max}, T_{\min}$.

2. Инициализация бинарного вектора состояний скрытых нейронов $\mathbf{x}^{h}=(x_1^{h},...,x_{N^h}^{h})$. Задается обучающее множество $\{(\mathbf{x}_\mu^{in},\mathbf{x}_\mu^{out})\,|\,\mathbf{x}_\mu^{in}\in\{0,1\}^{N^{in}},\mathbf{x}_\mu^{out}\in\{0,1\}^{N^{out}}\}$, $\mu\in\overline{1,P}$, где $\mathbf{x}_\mu^{in}$ – μ-й обучающий вектор состояний видимых входных нейронов, $\mathbf{x}_\mu^{out}$ – μ-й обучающий вектор состояний видимых выходных нейронов, P – мощность обучающего множества.

Положительная фаза (шаги 3-9)

3. $\mu=M^{in}+1$.

4. $k=1$, $T(k)=T_{\max}$.

$\mathbf{x}^{in}(\mu-t)=\mathbf{x}_{\mu-t}^{in}$, $t\in\overline{0,M^{in}}$,

$\mathbf{x}^{out}(\mu)=\mathbf{x}_\mu^{out}$.

5. Квантование Гиббса для скрытых нейронов ($j\in\overline{1,N^{h}}$) в момент времени μ

$$\Delta x_j^h(\mu) = \hat{x}_j^h(\mu) - x_j^h(\mu).$$

$$\Delta E_j = -\Delta x_j^h(\mu)\left(\sum_{t=0}^{M^{in}}\sum_{i=1}^{N^{in}} w_{tij}^{in-h}(n)x_i^{in}(\mu - t) + \sum_{i=1}^{N^{out}} w_{ij}^{out-h}(n)x_i^{out}(\mu)\right).$$

$$P_j = \frac{1}{1+\exp\left(-\frac{\Delta E_j}{T(k)}\right)}.$$

Если $\Delta E_j \le 0$, то $x_j^h(\mu) = \hat{x}_j^h(\mu)$.

Если $\Delta E_j > 0$ и $P_j \ge P_0$, то $x_j^h(\mu) = \hat{x}_j^h(\mu)$.

6. Если имеются скрытые нейроны, которые изменили состояние, то переход к 5.

7. Уменьшение температурного коэффициента в соответствии с методом моделирования отжига

$$T(k+1) = \beta T(k),$$

где β – коэффициент охлаждения, $0 < \beta < 1$.

8. Если $T(k) > T_{\min}$, то $k = k+1$, переход к 5.

9. $\mathbf{x}1^{in}(\mu) = \mathbf{x}^{in}(\mu)$, $\mathbf{x}1^{out}(\mu) = \mathbf{x}^{out}(\mu)$, $\mathbf{x}1^{h}(\mu) = \mathbf{x}^{h}(\mu)$. Если $\mu < P$, то $\mu = \mu + 1$, переход к 4.

Отрицательная фаза (шаги 10-20)

10. μ=1.

11. $k = 1$, $T(k) = T_{\max}$.

$$\mathbf{x}^{in}(\mu - t) = \mathbf{x}_{\mu - t}^{in},\ t \in \overline{0, M^{in}},$$

$$\mathbf{x}^{out}(\mu) = \mathbf{x}_{\mu}^{out},\ \mathbf{x}^{h}(\mu) = \mathbf{x}1^{h}(\mu).$$

12. Квантование Гиббса для видимых выходных нейронов ($j \in \overline{1, N^{out}}$) в момент времени μ

$$\Delta x_j^{out}(\mu) = \hat{x}_j^{out}(\mu) - x_j^{out}(\mu).$$

$$\Delta E_j = -\Delta x_j^{out}(\mu)\left(\sum_{i=1}^{N^h} w_{ij}^{out-h}(n)x_i^h(\mu)\right).$$

$$P_j = \frac{1}{1+\exp\left(-\frac{\Delta E_j}{T(k)}\right)}.$$

Если $\Delta E_j \le 0$, то $x_j^{out}(\mu) = \hat{x}_j^{out}(\mu)$.

Если $\Delta E_j > 0$ и $P_j \ge P_0$, то $x_j^{out}(\mu) = \hat{x}_j^{out}(\mu)$.

13. Если имеются видимые выходные нейроны, которые изменили состояние, то переход к 12.

14. Квантование Гиббса для видимых входных нейронов ($j \in \overline{1, N^{in}}$) в момент времени μ

$$\Delta x_j^{in}(\mu) = \hat{x}_j^{in}(\mu) - x_j^{in}(\mu).$$

$$\Delta E_j = -\Delta x_j^{in}(\mu)\left(\sum_{t=1}^{M^{in}}\sum_{i=1}^{N^{in}} w_{tij}^{in-in}(n)x_i^{in}(\mu - t) + \sum_{i=1}^{N^h} w_{0ij}^{in-h}(n)x_i^h(\mu)\right).$$

$$P_j = \frac{1}{1 + \exp\left(-\frac{\Delta E_j}{T(k)}\right)}.$$

Если $\Delta E_j \le 0$, то $x_j^{in}(\mu) = \hat{x}_j^{in}(\mu)$.

Если $\Delta E_j > 0$ и $P_j \ge P_0$, то $x_j^{in}(\mu) = \hat{x}_j^{in}(\mu)$.

15. Если имеются видимые входные нейроны, которые изменили состояние, то переход к 14.

16. Квантование Гиббса для скрытых нейронов ($j \in \overline{1, N^h}$) в момент времени μ

$$\Delta x_j^h(\mu) = \hat{x}_j^h(\mu) - x_j^h(\mu).$$

$$\Delta E_j = -\Delta x_j^h(\mu)\left(\sum_{t=0}^{M^{in}}\sum_{i=1}^{N^{in}} w_{tij}^{in-h}(n)x_i^{in}(\mu - t) + \sum_{i=1}^{N^{out}} w_{ij}^{out-h}(n)x_i^{out}(\mu)\right).$$

$$P_j = \frac{1}{1 + \exp\left(-\frac{\Delta E_j}{T(k)}\right)}.$$

Если $\Delta E_j \le 0$, то $x_j^h(\mu) = \hat{x}_j^h(\mu)$.

Если $\Delta E_j > 0$ и $P_j \ge P_0$, то $x_j^h(\mu) = \hat{x}_j^h(\mu)$.

17. Если имеются скрытые нейроны, которые изменили состояние, то переход к 16.

18. Уменьшение температурного коэффициента в соответствии с методом моделирования отжига

$T(k+1)=\beta T(k)$,

где β – коэффициент охлаждения, $0<\beta<1$.

19. Если $T(k)>T_{\min}$, то $k=k+1$, переход к 12.

20. $\mathbf{x}2^{in}(\mu)=\mathbf{x}^{in}(\mu)$, $\mathbf{x}2^{out}(\mu)=\mathbf{x}^{out}(\mu)$, $\mathbf{x}2^{h}(\mu)=\mathbf{x}^{h}(\mu)$. Если $\mu<P$, то $\mu=\mu+1$, переход к 11.

21. Настройка синаптических весов на основе стохастического правила Больцмана

$$w_{tij}^{in-h}(n)=w_{tij}^{in-h}(n)+\eta(\rho_{tij}^{+}-\rho_{tij}^{-}),\ t\in\overline{1,M^{in}},\ i\in\overline{1,N^{in}},\ j\in\overline{1,N^{h}},$$

$$\rho_{tij}^{+}=\frac{1}{P}\sum_{\mu=1}^{P}\varphi(x_{\mu-t,i}^{in},x1_j^{h}(\mu)),\ \rho_{tij}^{-}=\frac{1}{P}\sum_{\mu=1}^{P}\varphi(x_{\mu-t,i}^{in},x2_j^{h}(\mu)),$$

$$w_{0ij}^{in-h}(n)=w_{0ij}^{in-h}(n)+\eta(\rho_{0ij}^{+}-\rho_{0ij}^{-}),\ i\in\overline{1,N^{in}},\ j\in\overline{1,N^{h}},$$

$$\rho_{0ij}^{+}=\frac{1}{P}\sum_{\mu=1}^{P}\varphi(x1_i^{in}(\mu),x1_j^{h}(\mu)),\ \rho_{0ij}^{-}=\frac{1}{P}\sum_{\mu=1}^{P}\varphi(x2_i^{in}(\mu),x2_j^{h}(\mu)),$$

$$w_{ij}^{out-h}(n)=w_{ij}^{out-h}(n)+\eta(\rho_{ij}^{+}-\rho_{ij}^{-}),\ i\in\overline{1,N^{out}},\ j\in\overline{1,N^{h}},$$

$$\rho_{ij}^{+}=\frac{1}{P}\sum_{\mu=1}^{P}\varphi(x1_i^{out}(\mu),x1_j^{h}(\mu)),\ \rho_{ij}^{-}=\frac{1}{P}\sum_{\mu=1}^{P}\varphi(x2_i^{out}(\mu),x2_j^{h}(\mu)),$$

$$w_{tij}^{in-in}(n)=w_{tij}^{in-in}(n)+\eta(\rho_{tij}^{+}-\rho_{tij}^{-}),\ t\in\overline{1,M^{in}},\ i,j\in\overline{1,N^{in}},$$

$$\rho_{tij}^{+}=\frac{1}{P}\sum_{\mu=1}^{P}\varphi(x_{\mu-t,i}^{in},x1_j^{in}(\mu)),\ \rho_{tij}^{-}=\frac{1}{P}\sum_{\mu=1}^{P}\varphi(x_{\mu-t,i}^{in},x2_j^{in}(\mu)),$$

$$\varphi(x_i,x_j)=\begin{cases}1, & x_i=x_j\\ 0, & x_i\ne x_j\end{cases}.$$

20. Если $\frac{1}{P}\sum_{\mu=1}^{P}\left(\sum_{i=1}^{N^{in}}|x1_i^{in}(\mu)-x2_i^{in}(\mu)|+\sum_{i=1}^{N^{out}}|x1_i^{out}(\mu)-x2_i^{out}(\mu)|\right)>\varepsilon$, то $n=n+1$, переход к 2.

Функционирование ИНС

1. Инициализация бинарного вектора состояний скрытых нейронов $\mathbf{x}^{h}=(x_1^{h},...,x_{N^h}^{h})$ и видимых выходных нейронов $\mathbf{x}^{out}(M^{in}+1)=(x_{M^{in}+1,1}^{out},...,x_{M^{in}+1,N^{out}}^{out})$.

Положительная фаза (шаги 2-7)

2. $k=1,\ T(k)=T_{\max}$.

$\mathbf{x}^{in}(M^{in}+1-t)=\mathbf{x}^{in}_{M^{in}+1-t},\ t\in\overline{0,M^{in}}$.

5. Квантование Гиббса для скрытых нейронов ($j\in\overline{1,N^{h}}$) в момент времени $M^{in}+1$

$$\Delta x_j^h(M^v+1)=\hat{x}_j^h(M^v+1)-x_j^h(M^v+1)$$

$$\Delta E_j=-\Delta x_j^h(M^{in}+1)\left(\begin{array}{l}\sum_{t=0}^{M^{in}}\sum_{i=1}^{N^{in}}w_{tij}^{in-h}x_i^{in}(M^v+1-t)+\\+\sum_{i=1}^{N^{out}}w_{ij}^{out-h}x_i^{out}(M^v+1)\end{array}\right).$$

$$P_j=\frac{1}{1+\exp\left(-\frac{\Delta E_j}{T(k)}\right)}.$$

Если $\Delta E_j\le 0$, то $x_j^h(M^{in}+1)=\hat{x}_j^h(M^{in}+1)$.

Если $\Delta E_j>0$ и $P_j\ge P_0$, то $x_j^h(M^{in}+1)=\hat{x}_j^h(M^{in}+1)$.

4. Если имеются скрытые нейроны, которые изменили состояние, то переход к 3.

5. Уменьшение температурного коэффициента в соответствии с методом моделирования отжига

$T(k+1)=\beta T(k)$,

где β – коэффициент охлаждения, $0<\beta<1$.

6. Если $T(k)>T_{\min}$, то $k=k+1$, переход к 3.

7. $\mathbf{x}1^{in}(M^{in}+1)=\mathbf{x}^{in}(M^{in}+1)$, $\mathbf{x}1^{h}(M^{in}+1)=\mathbf{x}^{h}(M^{in}+1)$.

Отрицательная фаза (шаги 8-13)

8. $k=1,\ T(k)=T_{\max}$.

$\mathbf{x}^{in}(M^{in}+1-t)=\mathbf{x}^{in}_{M^{in}+1-t},\ t\in\overline{1,M^{in}}$,

$\mathbf{x}^{h}(M^{in}+1)=\mathbf{x}1^{h}(M^{in}+1)$.

9. Квантование Гиббса для видимых выходных нейронов ($j\in\overline{1,N^{out}}$) в момент времени $M^{in}+1$

$\Delta x_j^{out}(M^{in}+1)=\hat{x}_j^{out}(M^{in}+1)-x_j^{out}(M^{in}+1)$.

$$\Delta E_j = -\Delta x_j^{out}(M^{in}+1)\left(\sum_{i=1}^{N^h} w_{ij}^{out-h} x_i^h (M^{in}+1)\right).$$

$$P_j = \frac{1}{1+\exp\left(-\frac{\Delta E_j}{T(k)}\right)}.$$

Если $\Delta E_j \le 0$, то $x_j^{out}(M^{in}+1) = \widehat{x}_j^{out}(M^{in}+1)$.

Если $\Delta E_j > 0$ и $P_j \ge P_0$, то $x_j^{out}(M^{in}+1) = \widehat{x}_j^{out}(M^{in}+1)$.

10. Если имеются видимые выходные нейроны, которые изменили состояние, то переход к 9.

11. Уменьшение температурного коэффициента в соответствии с методом моделирования отжига

$T(k+1) = \beta T(k)$,

где β – коэффициент охлаждения, $0 < \beta < 1$.

12. Если $T(k) > T_{\min}$, то $k = k+1$, переход к 9.

Результатом является образец $\mathbf{x}^{out}(M^{out}+1)$.

7.9.3. Дескриптивная детерминированная условная ограниченная машина Больцмана

Обучение ИНС

1. Номер итерации обучения $n=1$, инициализация посредством равномерного распределения на интервале (0,1) или [-0.5, 0.5] весов $w_{tij}^{in-h}(n)$, $t \in \overline{0, M^{in}}$, $i \in \overline{1, N^{in}}$, $j \in \overline{1, N^h}$, $w_{ij}^{out-h}(n)$, $i \in \overline{1, N^{out}}$, $j \in \overline{1, N^h}$, $w_{tij}^{in-in}(n)$, $t \in \overline{1, M^{in}}$, $i, j \in \overline{1, N^{in}}$, $w_{tii}^{in-h}(n) = 0$, $w_{ii}^{out-h}(n) = 0$, $w_{tii}^{in-in}(n) = 0$, $w_{0ij}^{in-h}(n) = w_{0ji}^{in-h}(n)$, $w_{ij}^{out-h}(n) = w_{ji}^{out-h}(n)$, где M^{in} – задержка для видимых входных нейронов. Задание параметра P_0, температурных параметров $T_{\max}, T_{\min}$.

2. Инициализация посредством равномерного распределения на интервале (0,1) вещественного вектора состояний скрытых нейронов $<\mathbf{x}^h> = (<x_1^h>, ..., <x_{N^h}^h>)$. Задается обучающее множество $\{(<\mathbf{x}_\mu^{in}>, <\mathbf{x}_\mu^{out}>) | <\mathbf{x}_\mu^{in}> \in (0,1)^{N^{in}}, <\mathbf{x}_\mu^{out}> \in (0,1)^{N^{out}}\}$, $\mu \in \overline{1, P}$, где

$< \mathbf{x}_{\mu}^{in} >$ – μ-й обучающий вектор состояний видимых входных нейронов, $< \mathbf{x}_{\mu}^{out} >$ – μ-й обучающий вектор состояний видимых выходных нейронов, P – мощность обучающего множества.

Положительная фаза (шаги 3-8)

3. $\mu = M^{in} + 1$.

4. $k = 1,\ T(k) = T_{\max}$.

$< \mathbf{x}^{in}(\mu - t) >=< \mathbf{x}_{\mu-t}^{in} >,\ t \in \overline{0, M^{in}}$,

$< \mathbf{x}^{out}(\mu) >=< \mathbf{x}_{\mu}^{out} >$.

5. Вычисление среднего значения состояний скрытых нейронов в момент времени μ

$$< x_j^h(\mu) >= \frac{1}{1 + \exp\left(-\frac{1}{T(k)}\left(\sum_{t=0}^{M^{in}}\sum_{i=1}^{N^{in}} w_{tij}^{in-h}(n) < x_i^{in}(\mu - t) > + \sum_{i=1}^{N^{out}} w_{ij}^{out-h}(n) < x_i^{out}(\mu) >\right)\right)},$$

$j \in \overline{1, N^h}$.

6. Уменьшение температурного коэффициента в соответствии с методом моделирования отжига

$T(k+1) = \beta T(k)$,

где β – коэффициент охлаждения, $0 < \beta < 1$.

7. Если $T(k) > T_{\min}$, то $k = k + 1$, переход к 5.

8. $< \mathbf{x}1^{in}(\mu) >=< \mathbf{x}^{in}(\mu) >$, $< \mathbf{x}1^{out}(\mu) >=< \mathbf{x}^{out}(\mu) >$, $< \mathbf{x}1^{h}(\mu) >=< \mathbf{x}^{h}(\mu) >$. Если $\mu < P$, то $\mu = \mu + 1$, переход к 4.

Отрицательная фаза (шаги 9-16)

9. μ=1.

10. $k = 1,\ T(k) = T_{\max}$.

$< \mathbf{x}^{in}(\mu - t) >=< \mathbf{x}_{\mu-t}^{in} >,\ t \in \overline{0, M^{in}}$,

$< \mathbf{x}^{out}(\mu) >=< \mathbf{x}_{\mu}^{out} >$, $< \mathbf{x}^{h}(\mu) >=< \mathbf{x}1^{h}(\mu) >$.

11. Вычисление среднего значения состояний видимых выходных нейронов в момент времени μ

$$< x_j^{out}(\mu) >= \frac{1}{1+\exp\left(-\frac{1}{T(k)}\left(\sum_{i=1}^{N^h} w_{ij}^{out-h}(n) < x_i^h(\mu) >\right)\right)}, \; j \in \overline{1, N^{out}} .$$

12. Вычисление среднего значения состояний видимых входных нейронов в момент времени μ

$$< x_j^{in}(\mu) >= \frac{1}{1+\exp\left(-\frac{1}{T(k)}\left(\begin{array}{l}\sum_{t=1}^{M^{in}}\sum_{i=1}^{N^{in}} w_{tij}^{in-in}(n) < x_i^{in}(\mu - t) > + \\ + \sum_{i=1}^{N^h} w_{0ij}^{in-h}(n) < x_i^h(\mu) >\end{array}\right)\right)},$$

$j \in \overline{1, N^{in}}$.

13. Вычисление среднего значения состояний скрытых нейронов в момент времени μ

$$< x_j^{h}(\mu) >= \frac{1}{1+\exp\left(-\frac{1}{T(k)}\left(\begin{array}{l}\sum_{t=0}^{M^{in}}\sum_{i=1}^{N^{in}} w_{tij}^{in-h}(n) < x_i^{in}(\mu - t) > + \\ + \sum_{i=1}^{N^{out}} w_{ij}^{out-h}(n) < x_i^{out}(\mu) >\end{array}\right)\right)},$$

$j \in \overline{1, N^{h}}$.

14. Уменьшение температурного коэффициента в соответствии с методом моделирования отжига

$T(k+1) = \beta T(k)$,

где β – коэффициент охлаждения, $0 < \beta < 1$.

15. Если $T(k) > T_{\min}$, то $k = k+1$, переход к 11.

16. $< \mathbf{x}2^{in}(\mu) >=< \mathbf{x}^{in}(\mu) >$, $< \mathbf{x}2^{out}(\mu) >=< \mathbf{x}^{out}(\mu) >$, $< \mathbf{x}2^{h}(\mu) >=< \mathbf{x}^{h}(\mu) >$. Если $\mu < P$, то $\mu = \mu + 1$, переход к 10.

17. Настройка синаптических весов на основе стохастического правила Больцмана

$w_{tij}^{in-h}(n) = w_{tij}^{in-h}(n) + \eta(\rho_{tij}^{+} - \rho_{tij}^{-})$, $t \in \overline{1, M^{in}}$, $i \in \overline{1, N^{in}}$, $j \in \overline{1, N^{h}}$,

$$\rho_{tij}^{+}=\frac{1}{P}\sum_{\mu=1}^{P}< x_{\mu-t,i}^{in} >< x1_j^h(\mu)>,\ \rho_{tij}^{-}=\frac{1}{P}\sum_{\mu=1}^{P}< x_{\mu-t,i}^{in} >< x2_j^h(\mu)>,$$

$$w_{0ij}^{in-h}(n)=w_{0ij}^{in-h}(n)+\eta(\rho_{0ij}^{+}-\rho_{0ij}^{-}),\ i\in\overline{1,N^{in}},\ j\in\overline{1,N^{h}},$$

$$\rho_{0ij}^{+}=\frac{1}{P}\sum_{\mu=1}^{P}< x1_i^{in}(\mu)>< x1_j^h(\mu)>,$$

$$\rho_{0ij}^{-}=\frac{1}{P}\sum_{\mu=1}^{P}< x2_i^{in}(\mu)>< x2_j^h(\mu)>,$$

$$w_{ij}^{out-h}(n)=w_{ij}^{out-h}(n)+\eta(\rho_{ij}^{+}-\rho_{ij}^{-}),\ i\in\overline{1,N^{out}},\ j\in\overline{1,N^{h}},$$

$$\rho_{ij}^{+}=\frac{1}{P}\sum_{\mu=1}^{P}< x1_i^{out}(\mu)>< x1_j^h(\mu)>,\ \rho_{ij}^{-}=\frac{1}{P}\sum_{\mu=1}^{P}< x2_i^{out}(\mu)>< x2_j^h(\mu)>,$$

$$w_{tij}^{in-in}(n)=w_{tij}^{in-in}(n)+\eta(\rho_{tij}^{+}-\rho_{tij}^{-}),\ t\in\overline{1,M^{in}},\ i,j\in\overline{1,N^{in}},$$

$$\rho_{tij}^{+}=\frac{1}{P}\sum_{\mu=1}^{P}< x_{\mu-t,i}^{in} >< x1_j^{in}(\mu)>,\ \rho_{tij}^{-}=\frac{1}{P}\sum_{\mu=1}^{P}< x_{\mu-t,i}^{in} >< x2_j^{in}(\mu)>.$$

18. Если

$$\frac{1}{P}\sum_{\mu=1}^{P}\left(\sum_{i=1}^{N^{in}}|< x1_i^{in}(\mu)>-< x2_i^{in}(\mu)>|+\sum_{i=1}^{N^{out}}|< x1_i^{out}(\mu)>-< x2_i^{out}(\mu)>|\right)>\varepsilon,$$

то $n=n+1$, переход к 2.

Функционирование ИНС

1. Инициализация вещественного вектора средних значений состояний скрытых нейронов $<\mathbf{x}^h>=(< x_1^h>,...,< x_{N^h}^h>)$ и видимых выходных нейронов $<\mathbf{x}^{out}(M^{in}+1)>=(< x_{M^{in}+1,1}^{out}>,...,< x_{M^{in}+1,N^{out}}^{out}>)$.

Положительная фаза (шаги 2-6)

2. $k=1$, $T(k)=T_{\max}$.

$<\mathbf{x}^{in}(\mu-t)>=<\mathbf{x}_{\mu-t}^{in}>$, $t\in\overline{0,M^{in}}$,

$<\mathbf{x}^{out}(\mu)>=<\mathbf{x}_{\mu}^{out}>$.

3. Вычисление среднего значения состояний скрытых нейронов в момент времени $M^{v}+1$

$$< x_j^h(M^{in}+1) >= \frac{1}{1+\exp\left(-\frac{1}{T(k)}\left(\sum_{t=0}^{M^{in}}\sum_{i=1}^{N^{in}} w_{tij}^{in-h} < x_i^{in}(M^{in}+1-t) > + \sum_{i=1}^{N^{out}} w_{ij}^{out-h} < x_i^{out}(M^{in}+1) >\right)\right)},$$

$j \in \overline{1, N^h}$

4. Уменьшение температурного коэффициента в соответствии с методом моделирования отжига

$T(k+1) = \beta T(k)$,

где β – коэффициент охлаждения, $0 < \beta < 1$.

5. Если $T(k) > T_{\min}$, то $k = k+1$, переход к 3.

6. $< \mathbf{x}1^{in}(M^{in}+1) >=< \mathbf{x}^{in}(M^{in}+1) >$, $< \mathbf{x}1^{out}(M^{in}+1) >=< \mathbf{x}^{out}(M^{in}+1) >$, $< \mathbf{x}1^{h}(M^{in}+1) >=< \mathbf{x}^{h}(M^{in}+1) >$..

Отрицательная фаза (шаги 7-12)

7. $k = 1$, $T(k) = T_{\max}$.

$< \mathbf{x}^{in}(\mu - t) >=< \mathbf{x}_{\mu-t}^{in} >$, $t \in \overline{0, M^{in}}$,

$< \mathbf{x}^{out}(\mu) >=< \mathbf{x}_{\mu}^{out} >$,

$< \mathbf{x}^{h}(M^{in}+1) >=< \mathbf{x}1_{\mu}^{h}(M^{in}+1) >$.

8. Вычисление среднего значения состояний видимых выходных нейронов в момент времени μ

$$< x_j^{out}(\mu) >= \frac{1}{1+\exp\left(-\frac{1}{T(k)}\left(\sum_{i=1}^{N^h} w_{ij}^{out-h}(n) < x_i^h(\mu) >\right)\right)},\ j \in \overline{1, N^{out}}.$$

9. Уменьшение температурного коэффициента в соответствии с методом моделирования отжига

$T(k+1) = \beta T(k)$,

где β – коэффициент охлаждения, $0 < \beta < 1$.

10. Если $T(k) > T_{\min}$, то $k = k+1$, переход к 8.

Результатом является образец $< \mathbf{x}^{out}(M^{out}+1) >$.

Достоинства

1. Используется для восстановления или классификации образцов.
2. Обеспечивает хорошее качество обобщения.
3. Количество классов может быть больше двух.
4. В отличие от DHNN гарантирует нахождение глобального минимума.
5. Распределение вероятности, представляемое нейронами, близко к распределению вероятности среды, в которой работает ИНС.
6. Автоматически определяется количество скрытых слоев (равно одному).
7. В отличие от RBM имеется возможность работы с последовательностями.
8. Детерминированная CRBM обучается быстрее стохастической.

Недостатки

1. Отсутствует автоматическое определение числа нейронов в скрытом слое.

3. В отличие от ART, не решает проблему пластичности-стабильности.
4. Стохастическая CRBM работает только с биполярными или бинарными данными.
5. Детерминированная CRBM является только дескриптивной моделью.

7.10. Глубинная машина Больцмана

На рис.7.13, 7.14 представлена глубинная машина Больцмана (DBM) [74,75], которая является рекуррентной ИНС и содержит один видимый слой и более одного скрытого слоя (обычно два или три).

Генеративная DBM представляет собой композицию только генеративных RBM.

Дескриптивная DBM представляет собой композицию генеративных и единственной дескриптивной (самая верхняя) RBM.

Важнейшим свойством DBM является то, что одна и та же ИНС с одними и теми же весами связей может хранить и воспроизводить несколько различных запомненных образцов.

Компонентами стохастической DBM являются стохастические нейроны, состояние которых описывается на основе распределения Бернулли в виде

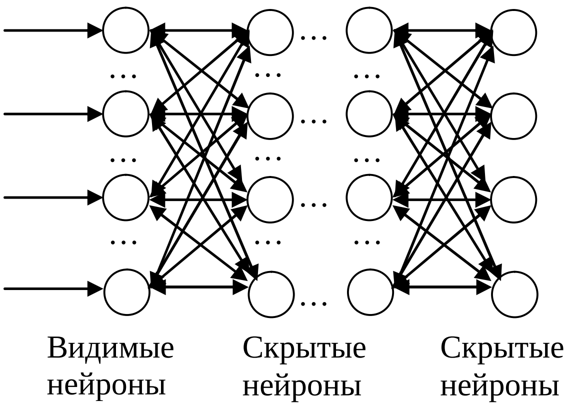


Рис. 7.13. Глубинная машина Больцмана (DBM) как генеративная модель

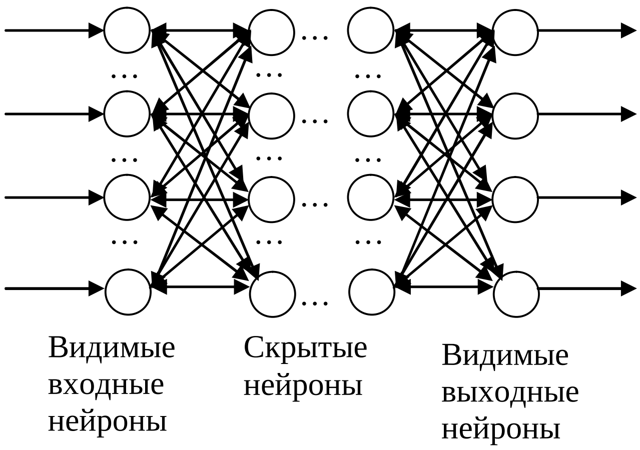


Рис. 7.14. Глубинная машина Больцмана (DBM) как дескриптивная модель

$$x_j = \begin{cases} 1, & \text{с вероятностью } P_j \\ 0, & \text{с вероятностью } 1 - P_j \end{cases}.$$

Величина изменения состояния j-го стохастического нейрона определена в виде

$$\Delta x_j = \hat{x}_j - x_j,$$

где $\hat{x}_j$ – состояние, противоположное состоянию x_j, т.е. $\hat{x}_j = 1 - x_j$.

Вероятность перехода j-го стохастического нейрона из текущего состояния x_j в новое состояние $\hat{x}_j$ определяется на основе имитации отжига в виде

$$P_j = \frac{1}{1 + \exp\left(-\frac{\Delta E_j}{T}\right)},$$

где ΔE_j – приращение энергии сети при изменении состояния j-го нейрона, T – температура.

7.10.1. Генеративная стохастическая глубинная машина Больцмана

Обучение ИНС

1. Номер итерации обучения $n = 1$, инициализация посредством равномерного распределения на интервале (0,1) или [-0.5, 0.5] весов $w_{ij}^{(l)}(n)$, $w_{ii}^{(l)}(n) = 0$, $w_{ij}^{(l)}(n) = w_{ji}^{(l)}(n)$, $i \in \overline{1, N^{(l-1)}}$, $j \in \overline{1, N^{(l)}}$, $l \in \overline{1, L}$, где $N^{(l)}$ – количество нейронов в l-м скрытом слое, L – количество

скрытых слоев. Задание параметра P_0, температурных параметров $T_{\max}, T_{\min}$, максимального количества итераций N.

2. Инициализация бинарных векторов состояний скрытых нейронов $\mathbf{x}^{(l)} = (x_1^{(l)}, ..., x_{N^{(l)}}^{(l)})$, $l \in \overline{1, L}$. Задается обучающее множество $\{\mathbf{x}_\mu^{(0)} \mid \mathbf{x}_\mu^{(0)} \in \{0,1\}^{N^{(0)}}\}$, $\mu \in \overline{1, P}$, где $\mathbf{x}_\mu^{(0)}$ – μ-й обучающий вектор состояний видимых нейронов, P – мощность обучающего множества.

Предобучение (шаги 3-7)

3. Настройка синаптических весов $w_{ij}^{(1)}$ на основе обучения генеративной ограниченной машины Больцмана (RBM), при этом видимый слой DBM рассматривается как видимый слой RBM, а первый скрытый слой DBM рассматривается как скрытый слой RBM. Связи между первым и вторым скрытыми слоями DBM не учитываются.

4. $l = 2$.

5. Выполнение положительной фазы $l-1$-й обученной RBM и получение вектора состояний нейронов $l-1$-го скрытого слоя $\mathbf{x}_\mu^{(l-1)}$, $\mu \in \overline{1, P}$.

6. Настройка синаптических весов $w_{ij}^{(l)}$ на основе обучения генеративной ограниченной машины Больцмана (RBM), при этом $l-1$-й скрытый слой DBM рассматривается как видимый слой RBM, а l-й скрытый слой DBM рассматривается как скрытый слой RBM. Связи между l-м и $l+1$-й скрытыми слоями DBM не учитываются

7. Если $l \le L$, то $l = l + 1$, переход к 5.

Вычисление средних значений состояний скрытых нейронов
(шаги 8-11)

8. $\mu = 1$.

9. Инициализировать случайным образом вероятности $m_{\mu j}^{(l)} \in (0,1)$, $l \in \overline{1, L}$, $j \in \overline{1, N^{(l)}}$.

10. Вычислить вероятности $m_{\mu j}^{(l)}$, $l \in \overline{1, L}$

$$m_{\mu j}^{(l)} = \begin{cases} \dfrac{1}{1+\exp\left(-\left(\sum\limits_{i=1}^{N^{(0)}} w_{ij}^{(1)}(n)x_{\mu i}^{(0)} + \sum\limits_{i=1}^{N^{(2)}} w_{ij}^{(2)}(n)m_{\mu i}^{(2)}\right)\right)}, & l=1 \\ \dfrac{1}{1+\exp\left(-\left(\sum\limits_{i=1}^{N^{(l-1)}} w_{ij}^{(l)}(n)m_{\mu i}^{(l-1)} + \sum\limits_{i=1}^{N^{(l+1)}} w_{ij}^{(l+1)}(n)m_{\mu i}^{(l+1)}\right)\right)}, & 1<l<L \\ \dfrac{1}{1+\exp\left(-\left(\sum\limits_{i=1}^{N^{(L-1)}} w_{ij}^{(L)}(n)m_{\mu i}^{(L-1)}\right)\right)}, & l=L \end{cases}$$

$j \in \overline{1, N^{(l)}}$.

11. Если $\mu < P$, то $\mu = \mu + 1$, переход к 9.

Проход снизу вверх (фаза распознавания) (шаги 8-33)

12. $\mu = 1$.

13. Инициализация бинарных векторов состояний скрытых нейронов $\mathbf{x}^{(l)} = (x_1^{(l)},...,x_{N^{(l)}}^{(l)})$, $l \in \overline{1, L}$.

14. $l = 1$, $\mathbf{x}1_\mu^{(0)} = \mathbf{x}_\mu^{in}$.

1. Положительная фаза (шаги 15-20)

15. $k = 1$, $T(k) = T_{\max}$.

$\mathbf{x}^{(l-1)} = \mathbf{x}1_\mu^{(l-1)}$.

16. Квантование Гиббса для скрытых нейронов l-го слоя ($j \in \overline{1, N^{(l)}}$)

$\Delta x_j^{(l)} = \widehat{x}_j^{(l)} - x_j^{(l)}$.

$$\Delta E_j = \begin{cases} -\Delta x_j^{(l)}\left(\sum\limits_{i=1}^{N^{(l-1)}} w_{ij}^{(l)}(n)x_i^{(l-1)} + \sum\limits_{i=1}^{N^{(l)}} w_{ij}^{(l+1)}(n)x_i^{(l+1)}\right), & l<L \\ -\Delta x_j^{(l)}\left(\sum\limits_{i=1}^{N^{(l-1)}} w_{ij}^{(l)}(n)x_i^{(l-1)}\right), & l=L \end{cases}.$$

$$P_j = \frac{1}{1+\exp\left(-\frac{\Delta E_j}{T(k)}\right)}.$$

Если $\Delta E_j \le 0$, то $x_j^{(l)} = \hat{x}_j^{(l)}$.

Если $\Delta E_j > 0$ и $P_j \ge P_0$, то $x_j^{(l)} = \hat{x}_j^{(l)}$.

17. Если имеются скрытые нейроны, которые изменили состояние, то переход к 16.

18. Уменьшение температурного коэффициента в соответствии с методом моделирования отжига

$$T(k+1) = \beta T(k),$$

где β – коэффициент охлаждения, $0 < \beta < 1$.

19. Если $T(k) > T_{\min}$, то $k = k+1$, переход к 16.

20. $\mathbf{x}1_\mu^{(l)} = \mathbf{x}^{(l)}$.

2. Отрицательная фаза (шаги 21-31)

21. $k = 1$, $T(k) = T_{\max}$.

$\mathbf{x}^{(l-1)} = \mathbf{x}1_\mu^{(l-1)}$, $\mathbf{x}^{(l)} = \mathbf{x}1_\mu^{(l)}$.

Если $l > 1$, то $\mathbf{x}^{(l-2)} = \mathbf{x}2_\mu^{(l-2)}$.

22. Квантование Гиббса для видимых либо скрытых нейронов $l-1$-го слоя ($j \in \overline{1, N^{(l-1)}}$)

$$\Delta x_j^{(l-1)} = \hat{x}_j^{(l-1)} - x_j^{(l-1)}.$$

$$\Delta E_j = \begin{cases} -\Delta x_j^{(l-1)} \left(\sum\limits_{i=1}^{N^{(l)}} w_{ij}^{(l)}(n) x_i^{(l)} \right), & l = 1 \\ -\Delta x_j^{(l-1)} \left(\sum\limits_{i=1}^{N^{(l)}} w_{ij}^{(l)}(n) x_i^{(l)} + \sum\limits_{i=1}^{N^{(l-2)}} w_{ij}^{(l-2)}(n) x_i^{(l-2)} \right), & 1 < l \le L \end{cases}.$$

$$P_j = \frac{1}{1+\exp\left(-\frac{\Delta E_j}{T(k)}\right)}.$$

Если $\Delta E_j \le 0$, то $x_j^{(l-1)} = \hat{x}_j^{(l-1)}$.

Если $\Delta E_j > 0$ и $P_j \ge P_0$, то $x_j^{(l-1)} = \hat{x}_j^{(l-1)}$.

23. Если имеются видимые либо скрытые нейроны, которые изменили состояние, то переход к 22.

24. Если $l < L$, то переход к 27.

25. Квантование Гиббса для скрытых нейронов L-го слоя ($j \in \overline{1, N^{(L)}}$)

$$\Delta x_j^{(L)} = \hat{x}_j^{(L)} - x_j^{(L)}.$$

$$\Delta E_j = -\Delta x_j^{(L)} \left(\sum_{i=1}^{N^{(L-1)}} w_{ij}^{(L-1)}(n) x_i^{(L-1)} \right).$$

$$P_j = \frac{1}{1 + \exp\left(-\frac{\Delta E_j}{T(k)} \right)}.$$

Если $\Delta E_j \le 0$, то $x_j^{(L)} = \hat{x}_j^{(L)}$.

Если $\Delta E_j > 0$ и $P_j \ge P_0$, то $x_j^{(L)} = \hat{x}_j^{(L)}$.

26. Если имеются скрытые нейроны, которые изменили состояние, то переход к 25.

27. Уменьшение температурного коэффициента в соответствии с методом моделирования отжига

$$T(k+1) = \beta T(k),$$

где β – коэффициент охлаждения, $0 < \beta < 1$.

28. Если $T(k) > T_{\min}$, то $k = k + 1$, переход к 22.

29. $\mathbf{x}2_\mu^{(l-1)} = \mathbf{x}^{(l-1)}$.

Если $l = L$, то $\mathbf{x}2_\mu^{(L)} = \mathbf{x}^{(L)}$.

30. Если $l < L$, то $l = l + 1$, переход к 15.

31. Если $\mu < P$, то $\mu = \mu + 1$, переход к 13.

32. Настройка синаптических весов на основе стохастического правила Больцмана

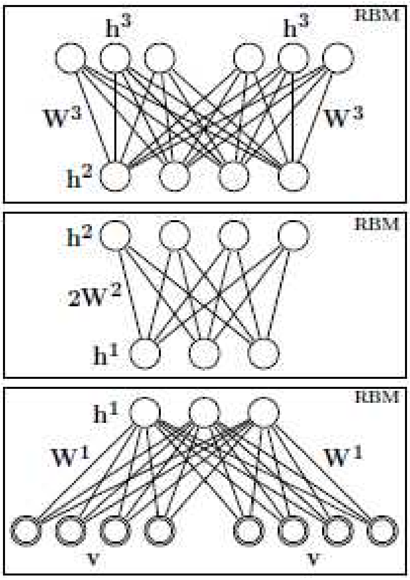
$$w_{ij}^{(l)}(n) = \begin{cases} w_{ij}^{(l)}(n) + \eta \left(\frac{1}{P} \sum_{\mu=1}^{P} x_{\mu i}^{(l-1)} m_{\mu j}^{(l)} - \frac{1}{P} \sum_{\mu=1}^{P} \varphi(x2_{\mu i}^{(l-1)}, x2_{\mu j}^{(l)}) \right), & l = 1 \\ w_{ij}^{(l)}(n) + \eta \left(\frac{1}{P} \sum_{\mu=1}^{P} m_{\mu i}^{(l-1)} m_{\mu j}^{(l)} - \frac{1}{P} \sum_{\mu=1}^{P} \varphi(x2_{\mu i}^{(l-1)}, x2_{\mu j}^{(l)}) \right), & l > 1 \end{cases}$$

, $i \in \overline{1, N^{(l-1)}}$, $j \in \overline{1, N^{(l)}}$, $l \in \overline{1, L}$,

$$\varphi(x_i, x_j) = \begin{cases} 1, & x_i = x_j \\ 0, & x_i \neq x_j \end{cases}.$$

33. Если $n < N$, то $n = n + 1$, переход к 9.

Замечание. Рекомендуется на этапе предобучения учитывать влияние верхних уровней скрытых нейронов на текущий слой скрытых нейронов. Для этого в RBM для первого слоя нейроны видимого слоя дублируются, а обучение весов $w_{ij}^{(1)}(n)$ происходит при установке равенства весов у соответствующих связей между скрытыми и видимыми нейронами. Аналогично, в RBM для последнего слоя дублируются нейроны последнего скрытого слоя, а обучение весов $w_{ij}^{(L)}(n)$ происходит при установке равенства весов у соответствующих связей между скрытыми нейронами предпоследнего и последнего слоя. Результирующие веса $w_{ij}^{(2)}(n),..., w_{ij}^{(L-1)}(n)$ устанавливаются как половина от соответствующих весов, полученных в результате обучения RBM. Пример для случая $L = 3$ показан ниже.



Функционирование ИНС

Проход снизу вверх (фаза распознавания) (шаги 1-18)

1. Инициализация бинарных векторов состояний скрытых нейронов $\mathbf{x}^{(l)} = (x_1^{(l)},..., x_{N^{(l)}}^{(l)})$, $l \in \overline{1, L}$.

2. $l = 1$, $\mathbf{x}1^{(0)} = \mathbf{x}^{in}$.

1. Положительная фаза (шаги 15-20)

3. $k=1,\ T(k)=T_{\max}$.

$\mathbf{x}^{(l-1)}=\mathbf{x}1^{(l-1)}$.

4. Квантование Гиббса для скрытых нейронов l-го слоя ($j\in\overline{1,N^{(l)}}$)

$\Delta x_j^{(l)}=\hat{x}_j^{(l)}-x_j^{(l)}$.

$$\Delta E_j=\begin{cases}-\Delta x_j^{(l)}\left(\sum_{i=1}^{N^{(l-1)}}w_{ij}^{(l)}x_i^{(l-1)}+\sum_{i=1}^{N^{(l)}}w_{ij}^{(l+1)}x_i^{(l+1)}\right), & l<L\\ -\Delta x_j^{(l)}\left(\sum_{i=1}^{N^{(l-1)}}w_{ij}^{(l)}x_i^{(l-1)}\right), & l=L\end{cases}.$$

$$P_j=\frac{1}{1+\exp\left(-\frac{\Delta E_j}{T(k)}\right)}.$$

Если $\Delta E_j\le 0$, то $x_j^{(l)}=\hat{x}_j^{(l)}$.

Если $\Delta E_j>0$ и $P_j\ge P_0$, то $x_j^{(l)}=\hat{x}_j^{(l)}$.

5. Если имеются скрытые нейроны, которые изменили состояние, то переход к 4.

6. Уменьшение температурного коэффициента в соответствии с методом моделирования отжига

$T(k+1)=\beta T(k)$,

где β – коэффициент охлаждения, $0<\beta<1$.

7. Если $T(k)>T_{\min}$, то $k=k+1$, переход к 4.

8. $\mathbf{x}1^{(l)}=\mathbf{x}^{(l)}$.

2. Отрицательная фаза (шаги 9-17)

9. $k=1,\ T(k)=T_{\max}$.

$\mathbf{x}^{(l-1)}=\mathbf{x}1^{(l-1)},\ \mathbf{x}^{(l)}=\mathbf{x}1^{(l)}$.

Если $l>1$, то $\mathbf{x}^{(l-2)}=\mathbf{x}2^{(l-2)}$.

10. Квантование Гиббса для видимых либо скрытых нейронов $l-1$-го слоя ($j\in\overline{1,N^{(l-1)}}$)

$$\Delta x_j^{(l-1)} = \hat{x}_j^{(l-1)} - x_j^{(l-1)}.$$

$$\Delta E_j = \begin{cases} -\Delta x_j^{(l-1)} \left(\sum_{i=1}^{N^{(l)}} w_{ij}^{(l)} x_i^{(l)} \right), & l = 1 \\ -\Delta x_j^{(l-1)} \left(\sum_{i=1}^{N^{(l)}} w_{ij}^{(l)} x_i^{(l)} + \sum_{i=1}^{N^{(l-2)}} w_{ij}^{(l-2)} x_i^{(l-2)} \right), & 1 < l \le L \end{cases}.$$

$$P_j = \frac{1}{1 + \exp\left(-\frac{\Delta E_j}{T(k)} \right)}.$$

Если $\Delta E_j \le 0$, то $x_j^{(l-1)} = \hat{x}_j^{(l-1)}$.

Если $\Delta E_j > 0$ и $P_j \ge P_0$, то $x_j^{(l-1)} = \hat{x}_j^{(l-1)}$.

11. Если имеются видимые либо скрытые нейроны, которые изменили состояние, то переход к 10.

12. Если $l < L$, то переход к 15.

13. Квантование Гиббса для скрытых нейронов L-го слоя ($j \in \overline{1, N^{(L)}}$)

$$\Delta x_j^{(L)} = \hat{x}_j^{(L)} - x_j^{(L)}.$$

$$\Delta E_j = -\Delta x_j^{(L)} \left(\sum_{i=1}^{N^{(L-1)}} w_{ij}^{(L-1)} x_i^{(L-1)} \right).$$

$$P_j = \frac{1}{1 + \exp\left(-\frac{\Delta E_j}{T(k)} \right)}.$$

Если $\Delta E_j \le 0$, то $x_j^{(L)} = \hat{x}_j^{(L)}$.

Если $\Delta E_j > 0$ и $P_j \ge P_0$, то $x_j^{(L)} = \hat{x}_j^{(L)}$.

14. Если имеются скрытые нейроны, которые изменили состояние, то переход к 13.

15. Уменьшение температурного коэффициента в соответствии с методом моделирования отжига

$$T(k+1) = \beta T(k),$$

где β – коэффициент охлаждения, $0 < \beta < 1$.

16. Если $T(k) > T_{\min}$, то $k = k + 1$, переход к 10.

17. $\mathbf{x}2^{(l-1)} = \mathbf{x}^{(l-1)}$.

Если $l = L$, то $\mathbf{x}2^{(L)} = \mathbf{x}^{(L)}$, $\mathbf{x}2^{(L+1)} = \mathbf{x}^{(L+1)}$.

18. Если $l < L$, то $l = l + 1$, переход к 3.

Проход сверху вниз (фаза порождения) (шаги 19-35)

19. $l = L - 1$, $\mathbf{x}3^{(L)} = \mathbf{x}2^{(L)}$.

1. Положительная фаза (шаги 20-25)

20. $k = 1$, $T(k) = T_{\max}$.

$\mathbf{x}^{(l+1)} = \mathbf{x}3^{(l+1)}$, $\mathbf{x}^{(l)} = \mathbf{x}2^{(l)}$.

Если $l > 0$, то $\mathbf{x}^{(l-1)} = \mathbf{x}2^{(l-1)}$.

21. Квантование Гиббса для скрытых либо видимых нейронов l-го слоя ($j \in \overline{1, N^{(l)}}$)

$$\Delta x_j^{(l)} = \hat{x}_j^{(l)} - x_j^{(l)}.$$

$$\Delta E_j = \begin{cases} -\Delta x_j^{(l)} \left(\sum\limits_{i=1}^{N^{(l-1)}} w_{ij}^{(l)} x_i^{(l-1)} + \sum\limits_{i=1}^{N^{(l+1)}} w_{ij}^{(l+1)} x_i^{(l+1)} \right), & l > 0 \\ -\Delta x_j^{(l)} \left(\sum\limits_{i=1}^{N^{(l+1)}} w_{ij}^{(l+1)} x_i^{(l+1)} \right), & l = 0 \end{cases}.$$

$$P_j = \frac{1}{1 + \exp\left(-\frac{\Delta E_j}{T(k)} \right)}.$$

Если $\Delta E_j \le 0$, то $x_j^{(l)} = \hat{x}_j^{(l)}$.

Если $\Delta E_j > 0$ и $P_j \ge P_0$, то $x_j^{(l)} = \hat{x}_j^{(l)}$.

22. Если имеются скрытые либо видимые нейроны, которые изменили состояние, то переход к 21.

23. Уменьшение температурного коэффициента в соответствии с методом моделирования отжига

$$T(k+1) = \beta T(k),$$

где β – коэффициент охлаждения, $0 < \beta < 1$.

24. Если $T(k) > T_{\min}$, то $k = k + 1$, переход к 21.

25. $\mathbf{x}3^{(l)} = \mathbf{x}^{(l)}$.

2. Отрицательная фаза (шаги 26-35)

26. $k = 1$, $T(k) = T_{\max}$.

$\mathbf{x}^{(l-1)} = \mathbf{x}3^{(l-1)}$, $\mathbf{x}^{(l)} = \mathbf{x}3^{(l)}$.

Если $l < L-1$, то $\mathbf{x}^{(l+2)} = \mathbf{x}4^{(l+2)}$.

27. Квантование Гиббса для скрытых нейронов $l+1$-го слоя ($j \in \overline{1, N^{(l+1)}}$)

$$\Delta x_j^{(l+1)} = \hat{x}_j^{(l+1)} - x_j^{(l+1)}.$$

$$\Delta E_j = \begin{cases} -\Delta x_j^{(l+1)} \left(\sum\limits_{i=1}^{N^{(l)}} w_{ij}^{(l+1)} x_i^{(l)} \right), & l = L-1 \\ -\Delta x_j^{(l+1)} \left(\sum\limits_{i=1}^{N^{(l)}} w_{ij}^{(l+1)} x_i^{(l)} + \sum\limits_{i=1}^{N^{(l+2)}} w_{ij}^{(l+2)} x_i^{(l+2)} \right), & 0 \le l < L-1 \end{cases}.$$

$$P_j = \frac{1}{1 + \exp\left(-\dfrac{\Delta E_j}{T(k)} \right)}.$$

Если $\Delta E_j \le 0$, то $x_j^{(l+1)} = \hat{x}_j^{(l+1)}$.

Если $\Delta E_j > 0$ и $P_j \ge P_0$, то $x_j^{(l+1)} = \hat{x}_j^{(l+1)}$.

28. Если имеются скрытые нейроны, которые изменили состояние, то переход к 27.

29. Если $l > 0$, то переход к 32.

30. Квантование Гиббса для видимых нейронов нулевого слоя ($j \in \overline{1, N^{(0)}}$)

$$\Delta x_j^{(0)} = \hat{x}_j^{(0)} - x_j^{(0)}.$$

$$\Delta E_j = -\Delta x_j^{(0)} \left(\sum_{i=1}^{N^{(1)}} w_{ij}^{(1)} x_i^{(1)} \right).$$

$$P_j = \frac{1}{1 + \exp\left(-\dfrac{\Delta E_j}{T(k)} \right)}.$$

Если $\Delta E_j \le 0$, то $x_j^{(0)} = \hat{x}_j^{(0)}$.

Если $\Delta E_j > 0$ и $P_j \geq P_0$, то $x_j^{(0)} = \hat{x}_j^{(0)}$.

31. Если имеются видимые нейроны, которые изменили состояние, то переход к 30.

32. Уменьшение температурного коэффициента в соответствии с методом моделирования отжига

$T(k+1) = \beta T(k)$,

где β – коэффициент охлаждения, $0 < \beta < 1$.

33. Если $T(k) > T_{\min}$, то $k = k+1$, переход к 27.

34. $\mathbf{x}4^{(l+1)} = \mathbf{x}^{(l+1)}$.

Если $l = 0$, то $\mathbf{x}4^{(0)} = \mathbf{x}^{(0)}$.

35. Если $l > 0$, то $l = l-1$, переход к 20.

Результатом является образец $\mathbf{x}4^{(0)}$.

7.10.2. Дескриптивная стохастическая глубинная машина Больцмана

Обучение ИНС

1. Номер итерации обучения $n = 1$, инициализация посредством равномерного распределения на интервале (0,1) или [-0.5, 0.5] весов $w_{ij}^{(l)}(n)$, $w_{ii}^{(l)}(n) = 0$, $w_{ij}^{(l)}(n) = w_{ji}^{(l)}(n)$, $i \in \overline{1, N^{(l-1)}}$, $j \in \overline{1, N^{(l)}}$, $l \in \overline{1, L+1}$, где $N^{(l)}$ – количество нейронов в l-м скрытом слое, L – количество скрытых слоев. Задание параметра P_0, температурных параметров $T_{\max}, T_{\min}$, максимального количества итераций N.

2. Инициализация бинарных векторов состояний скрытых нейронов $\mathbf{x}^{(l)} = (x_1^{(l)}, ..., x_{N^{(l)}}^{(l)})$, $l \in \overline{1, L}$. Задается обучающее множество $\{(\mathbf{x}_\mu^{in}, \mathbf{x}_\mu^{out}) \mid \mathbf{x}_\mu^{in} \in \{0,1\}^{N^{(0)}}, \mathbf{x}_\mu^{out} \in \{0,1\}^{N^{(L-1)}}\}$, $\mu \in \overline{1, P}$, где $\mathbf{x}_\mu^{in}$ – μ-й обучающий вектор состояний видимых входных нейронов, $\mathbf{x}_\mu^{out}$ – μ-й обучающий вектор состояний видимых выходных нейронов, P – мощность обучающего множества.

Предобучение (шаги 3-7)

3. Настройка синаптических весов $w_{ij}^{(1)}$ на основе обучения генеративной ограниченной машины Больцмана (RBM), при этом видимый слой DBM рассматривается как видимый слой RBM, а

первый скрытый слой DBM рассматривается как скрытый слой RBM. Связи между первым и вторым скрытыми слоями DBM не учитываются.

4. $l = 2$.

5. Выполнение положительной фазы $l-1$-й обученной RBM и получение вектора состояний нейронов $l-1$-го скрытого слоя $\mathbf{x}_{\mu}^{(l-1)}$, $\mu \in \overline{1, P}$.

6. Настройка синаптических весов $w_{ij}^{(l)}$ на основе обучения RBM, при этом $l-1$-й скрытый слой DBM рассматривается как видимый слой RBM, а l-й скрытый слой DBM рассматривается как скрытый слой RBM. Если $l = L$, то используется дескриптивная RBM, иначе используется генеративная RBM. Связи между l-м и $l+1$-й скрытыми слоями DBM не учитываются

7. Если $l \le L$, то $l = l+1$, переход к 5.

Вычисление средних значений состояний скрытых нейронов
(шаги 8-11)

8. $\mu = 1$.

9. Инициализировать случайным образом вероятности $m_{\mu j}^{(l)} \in (0,1)$, $l \in \overline{1, L}$, $j \in \overline{1, N^{(l)}}$.

10. Вычислить вероятности $m_{\mu j}^{(l)}$, $l \in \overline{1, L}$, как средние значения состояний скрытых нейронов

$$m_{\mu j}^{(l)} = \begin{cases} \dfrac{1}{1+\exp\left(-\left(\sum\limits_{i=1}^{N^{(0)}} w_{ij}^{(1)}(n) x_{\mu i}^{in} + \sum\limits_{i=1}^{N^{(2)}} w_{ij}^{(2)}(n) m_{\mu i}^{(2)}\right)\right)}, & l=1 \\ \dfrac{1}{1+\exp\left(-\left(\sum\limits_{i=1}^{N^{(l-1)}} w_{ij}^{(l)}(n) m_{\mu i}^{(l-1)} + \sum\limits_{i=1}^{N^{(l+1)}} w_{ij}^{(l+1)}(n) m_{\mu i}^{(l+1)}\right)\right)}, & 1<l<L \\ \dfrac{1}{1+\exp\left(-\left(\sum\limits_{i=1}^{N^{(L-1)}} w_{ij}^{(L)}(n) m_{\mu i}^{(L-1)} + \sum\limits_{i=1}^{N^{(L+1)}} w_{ij}^{(L+1)}(n) x_{\mu i}^{out}\right)\right)}, & l=L \end{cases}$$

$j \in \overline{1, N^{(l)}}$.

11. Если $\mu < P$, то $\mu = \mu + 1$, переход к 9.

Проход снизу вверх (фаза распознавания) (шаги 8-33)

12. $\mu = 1$.

13. Инициализация бинарных векторов состояний скрытых нейронов $\mathbf{x}^{(l)} = (x_1^{(l)}, ..., x_{N^{(l)}}^{(l)})$, $l \in \overline{1, L}$.

14. $l = 1$, $\mathbf{x1}_{\mu}^{(0)} = \mathbf{x}_{\mu}^{in}$, $\mathbf{x1}_{\mu}^{(L+1)} = \mathbf{x}_{\mu}^{out}$.

1. Положительная фаза (шаги 15-20)

15. $k = 1$, $T(k) = T_{\max}$.

$\mathbf{x}^{(l-1)} = \mathbf{x1}_{\mu}^{(l-1)}$.

16. Квантование Гиббса для скрытых нейронов l-го слоя ($j \in \overline{1, N^{(l)}}$)

$\Delta x_j^{(l)} = \hat{x}_j^{(l)} - x_j^{(l)}$.

$$\Delta E_j = -\Delta x_j^{(l)} \left(\sum_{i=1}^{N^{(l-1)}} w_{ij}^{(l)}(n) x_i^{(l-1)} + \sum_{i=1}^{N^{(l)}} w_{ij}^{(l+1)}(n) x_i^{(l+1)} \right).$$

$$P_j = \frac{1}{1+\exp\left(-\dfrac{\Delta E_j}{T(k)}\right)}.$$

Если $\Delta E_j \le 0$, то $x_j^{(l)} = \hat{x}_j^{(l)}$.

Если $\Delta E_j > 0$ и $P_j \ge P_0$, то $x_j^{(l)} = \hat{x}_j^{(l)}$.

17. Если имеются скрытые нейроны, которые изменили состояние, то переход к 16.

18. Уменьшение температурного коэффициента в соответствии с методом моделирования отжига

$T(k+1) = \beta T(k)$,

где β – коэффициент охлаждения, $0 < \beta < 1$.

19. Если $T(k) > T_{\min}$, то $k = k+1$, переход к 16.

20. $\mathbf{x}1_\mu^{(l)} = \mathbf{x}^{(l)}$.

2. Отрицательная фаза (шаги 21-31)

21. $k = 1$, $T(k) = T_{\max}$.

$\mathbf{x}^{(l-1)} = \mathbf{x}1_\mu^{(l-1)}$, $\mathbf{x}^{(l)} = \mathbf{x}1_\mu^{(l)}$.

Если $l > 1$, то $\mathbf{x}^{(l-2)} = \mathbf{x}2_\mu^{(l-2)}$.

22. Квантование Гиббса для видимых входных либо скрытых нейронов $l-1$-го слоя ($j \in \overline{1, N^{(l-1)}}$)

$\Delta x_j^{(l-1)} = \hat{x}_j^{(l-1)} - x_j^{(l-1)}$.

$$\Delta E_j = \begin{cases} -\Delta x_j^{(l-1)} \left(\sum_{i=1}^{N^{(l)}} w_{ij}^{(l)}(n) x_i^{(l)} \right), & l = 1 \\ -\Delta x_j^{(l-1)} \left(\sum_{i=1}^{N^{(l)}} w_{ij}^{(l)}(n) x_i^{(l)} + \sum_{i=1}^{N^{(l-2)}} w_{ij}^{(l-2)}(n) x_i^{(l-2)} \right), & 1 < l \le L \end{cases}.$$

$$P_j = \frac{1}{1 + \exp\left(-\frac{\Delta E_j}{T(k)} \right)}.$$

Если $\Delta E_j \le 0$, то $x_j^{(l-1)} = \hat{x}_j^{(l-1)}$.

Если $\Delta E_j > 0$ и $P_j \ge P_0$, то $x_j^{(l-1)} = \hat{x}_j^{(l-1)}$.

23. Если имеются видимые входные либо скрытые нейроны, которые изменили состояние, то переход к 22.

24. Если $l < L$, то переход к 29.

25. Квантование Гиббса для видимых выходных нейронов $L+1$-го слоя ($j \in \overline{1, N^{(L+1)}}$)

$$\Delta x_j^{(l-1)} = \hat{x}_j^{(l-1)} - x_j^{(l-1)}.$$

$$\Delta E_j = -\Delta x_j^{(L+1)} \left(\sum_{i=1}^{N^{(L)}} w_{ij}^{(L+1)}(n) x_i^{(L)} \right).$$

$$P_j = \frac{1}{1 + \exp\left(-\frac{\Delta E_j}{T(k)} \right)}.$$

Если $\Delta E_j \le 0$, то $x_j^{(l-1)} = \hat{x}_j^{(l-1)}$.

Если $\Delta E_j > 0$ и $P_j \ge P_0$, то $x_j^{(l-1)} = \hat{x}_j^{(l-1)}$.

26. Если имеются видимые выходные нейроны, которые изменили состояние, то переход к 25.

27. Квантование Гиббса для скрытых нейронов L-го слоя ($j \in \overline{1, N^{(L)}}$)

$$\Delta x_j^{(L)} = \hat{x}_j^{(L)} - x_j^{(L)}.$$

$$\Delta E_j = -\Delta x_j^{(L)} \left(\sum_{i=1}^{N^{(L-1)}} w_{ij}^{(L-1)}(n) x_i^{(L-1)} \right).$$

$$P_j = \frac{1}{1 + \exp\left(-\frac{\Delta E_j}{T(k)} \right)}.$$

Если $\Delta E_j \le 0$, то $x_j^{(L)} = \hat{x}_j^{(L)}$.

Если $\Delta E_j > 0$ и $P_j \ge P_0$, то $x_j^{(L)} = \hat{x}_j^{(L)}$.

28. Если имеются скрытые нейроны, которые изменили состояние, то переход к 27.

29. Уменьшение температурного коэффициента в соответствии с методом моделирования отжига

$$T(k+1) = \beta T(k),$$

где β – коэффициент охлаждения, $0 < \beta < 1$.

30. Если $T(k) > T_{\min}$, то $k = k+1$, переход к 22.

31. $\mathbf{x}2_{\mu}^{(l-1)} = \mathbf{x}^{(l-1)}$.

Если $l = L$, то $\mathbf{x}2_{\mu}^{(L)} = \mathbf{x}^{(L)}$, $\mathbf{x}2_{\mu}^{(L+1)} = \mathbf{x}^{(L+1)}$.

32. Если $l < L$, то $l = l + 1$, переход к 15.

33. Если $\mu < P$, то $\mu = \mu + 1$, переход к 13.

34. Настройка синаптических весов на основе стохастического правила Больцмана

$$w_{ij}^{(l)}(n) = \begin{cases} w_{ij}^{(l)}(n) + \eta\left(\frac{1}{P}\sum_{\mu=1}^{P} x_{\mu i}^{in} m_{\mu j}^{(l)} - \frac{1}{P}\sum_{\mu=1}^{P}\varphi(x2_{\mu i}^{(l-1)}, x2_{\mu j}^{(l)})\right), & l = 1 \\ w_{ij}^{(l)}(n) + \eta\left(\frac{1}{P}\sum_{\mu=1}^{P} m_{\mu i}^{(l-1)} m_{\mu j}^{(l)} - \frac{1}{P}\sum_{\mu=1}^{P}\varphi(x2_{\mu i}^{(l-1)}, x2_{\mu j}^{(l)})\right), & 1 < l \le L \\ w_{ij}^{(l)}(n) + \eta\left(\frac{1}{P}\sum_{\mu=1}^{P} m_{\mu i}^{(l-1)} x_{\mu j}^{out} - \frac{1}{P}\sum_{\mu=1}^{P}\varphi(x2_{\mu i}^{(l-1)}, x2_{\mu j}^{(l)})\right), & l = L + 1 \end{cases}$$

$i \in \overline{1, N^{(l-1)}}$, $j \in \overline{1, N^{(l)}}$, $l \in \overline{1, L+1}$,

$$\varphi(x_i, x_j) = \begin{cases} 1, & x_i = x_j \\ 0, & x_i \ne x_j \end{cases}.$$

35. Если $n < N$, то $n = n + 1$, переход к 8.

Функционирование ИНС

Проход снизу вверх (фаза распознавания) (шаги 1-20)

1. Инициализация бинарных векторов состояний скрытых нейронов $\mathbf{x}^{(l)} = (x_1^{(l)}, ..., x_{N^{(l)}}^{(l)})$, $l \in \overline{1, L}$.

Положительная фаза (шаги 2-8)

2. $l = 1$, $\mathbf{x}1^{(0)} = \mathbf{x}^{(0)}$.

3. $k = 1$, $T(k) = T_{\max}$.

$\mathbf{x}^{(l-1)} = \mathbf{x}1^{(l-1)}$.

4. Квантование Гиббса для скрытых нейронов l-го слоя ($j \in \overline{1, N^{(l)}}$)

$$\Delta x_j^{(l)} = \hat{x}_j^{(l)} - x_j^{(l)}.$$

$$\Delta E_j = -\Delta x_j^{(l)}\left(\sum_{i=1}^{N^{(l-1)}} w_{ij}^{(l)} x_i^{(l-1)} + \sum_{i=1}^{N^{(l)}} w_{ij}^{(l+1)} x_i^{(l+1)}\right).$$

$$P_j = \frac{1}{1 + \exp\left(-\frac{\Delta E_j}{T(k)}\right)}.$$

Если $\Delta E_j \le 0$, то $x_j^{(l)} = \hat{x}_j^{(l)}$.

Если $\Delta E_j > 0$ и $P_j \ge P_0$, то $x_j^{(l)} = \hat{x}_j^{(l)}$.

5. Если имеются скрытые нейроны, которые изменили состояние, то переход к 4.

6. Уменьшение температурного коэффициента в соответствии с методом моделирования отжига

$$T(k+1) = \beta T(k),$$

где β – коэффициент охлаждения, $0 < \beta < 1$.

7. Если $T(k) > T_{\min}$, то $k = k+1$, переход к 4.

8. $\mathbf{x}1^{(l)} = \mathbf{x}^{(l)}$.

Отрицательная фаза (шаги 9-20)

9. $k = 1$, $T(k) = T_{\max}$.

$\mathbf{x}^{(l-1)} = \mathbf{x}1^{(l-1)}$, $\mathbf{x}^{(l)} = \mathbf{x}1^{(l)}$.

Если $l > 1$, то $\mathbf{x}^{(l-2)} = \mathbf{x}2^{(l-2)}$.

10. Квантование Гиббса для видимых входных либо скрытых нейронов $l-1$-го слоя ($j \in \overline{1, N^{(l-1)}}$)

$$\Delta x_j^{(l-1)} = \hat{x}_j^{(l-1)} - x_j^{(l-1)}.$$

$$\Delta E_j = \begin{cases} -\Delta x_j^{(l-1)} \left(\sum_{i=1}^{N^{(l)}} w_{ij}^{(l)} x_i^{(l)} \right), & l = 1 \\ -\Delta x_j^{(l-1)} \left(\sum_{i=1}^{N^{(l)}} w_{ij}^{(l)} x_i^{(l)} + \sum_{i=1}^{N^{(l-2)}} w_{ij}^{(l-2)} x_i^{(l-2)} \right), & 1 < l \le L \end{cases}.$$

$$P_j = \frac{1}{1 + \exp\left(-\frac{\Delta E_j}{T(k)}\right)}.$$

Если $\Delta E_j \le 0$, то $x_j^{(l-1)} = \hat{x}_j^{(l-1)}$.

Если $\Delta E_j > 0$ и $P_j \ge P_0$, то $x_j^{(l-1)} = \hat{x}_j^{(l-1)}$.

11. Если имеются видимые входные либо скрытые нейроны, которые изменили состояние, то переход к 10.

12. Если $l < L$, то переход к 20.

13. Квантование Гиббса для видимых выходных нейронов $L+1$-го слоя ($j \in \overline{1, N^{(L+1)}}$)

$\Delta x_j^{(l-1)} = \widehat{x}_j^{(l-1)} - x_j^{(l-1)}$.

$$\Delta E_j = -\Delta x_j^{(L+1)} \left(\sum_{i=1}^{N^{(L)}} w_{ij}^{(L+1)} x_i^{(L)} \right).$$

$$P_j = \frac{1}{1 + \exp\left(-\dfrac{\Delta E_j}{T(k)} \right)}.$$

Если $\Delta E_j \le 0$, то $x_j^{(l-1)} = \widehat{x}_j^{(l-1)}$.

Если $\Delta E_j > 0$ и $P_j \ge P_0$, то $x_j^{(l-1)} = \widehat{x}_j^{(l-1)}$.

14. Если имеются видимые выходные нейроны, которые изменили состояние, то переход к 13.

15. Квантование Гиббса для скрытых нейронов L-го слоя ($j \in \overline{1, N^{(L)}}$)

$\Delta x_j^{(L)} = \widehat{x}_j^{(L)} - x_j^{(L)}$.

$$\Delta E_j = -\Delta x_j^{(L)} \left(\sum_{i=1}^{N^{(L-1)}} w_{ij}^{(L-1)} x_i^{(L-1)} \right).$$

$$P_j = \frac{1}{1 + \exp\left(-\dfrac{\Delta E_j}{T(k)} \right)}.$$

Если $\Delta E_j \le 0$, то $x_j^{(L)} = \widehat{x}_j^{(L)}$.

Если $\Delta E_j > 0$ и $P_j \ge P_0$, то $x_j^{(L)} = \widehat{x}_j^{(L)}$.

16. Если имеются скрытые нейроны, которые изменили состояние, то переход к 15.

17. Уменьшение температурного коэффициента в соответствии с методом моделирования отжига

$T(k+1) = \beta T(k)$,

где β – коэффициент охлаждения, $0 < \beta < 1$.

18. Если $T(k) > T_{\min}$, то $k = k + 1$, переход к 10.

19. $\mathbf{x}2^{(l-1)} = \mathbf{x}^{(l-1)}$.

Если $l = L$, то $\mathbf{x}2_1^{(L)} = \mathbf{x}^{(L)}$, $\mathbf{x}2^{(L+1)} = \mathbf{x}^{(L+1)}$.

20. Если $l < L$, то $l = l + 1$, переход к 3.

Результатом является образец $\mathbf{x}^{(L+1)}$.

Достоинства

1. Используется для восстановления или классификации образцов.
2. Обеспечивает хорошее качество обобщения.
3. Количество классов может быть больше двух.
4. В отличие от DHNN гарантирует нахождение глобального минимума.
5. Распределение вероятности, представляемое нейронами, близко к распределению вероятности среды, в которой работает ИНС.
6. В отличие от DBN дает более точный результат.

Недостатки

1. Отсутствует автоматическое определение числа скрытых слоев и числа нейронов в этих слоях.
2. Обучение DBM происходит медленнее, чем в случае MLP.
3. В отличие от ART, не решает проблему пластичности-стабильности.
4. Стохастическая DBM работает только с биполярными или бинарными данными.

7.11. Сигмоидальная сеть доверия

На рис. 7.15-7.16 приведена сигмоидальная сеть доверия (SBN) [76], которая является нерекуррентной на этапе обучения однослойной ИНС и рекуррентной на этапе функционирования.

Генеративная SBN реализует автоассоциативную память (ее элемент представлен парой образцов $(\mathbf{m}_x, \mathbf{m}_y)$, $\mathbf{m}_y = \mathbf{m}_x$) и восстанавливает запомненный образец $\mathbf{m}_y$ по ключевому образцу $\mathbf{m}_x$, соответствующему входному вектору $\mathbf{x}$.

Важнейшим свойством SBN является то, что одна и та же ИНС с одними и теми же весами связей может хранить и воспроизводить несколько запомненных образцов.

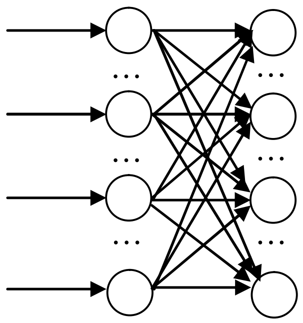


Рис. 7.15. Сигмоидальная сеть доверия (SBN) на этапе обучения

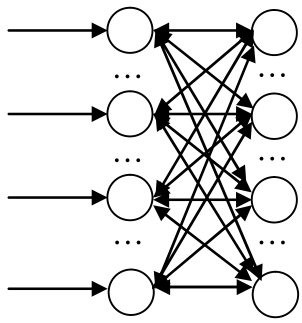


Рис. 7.16. Сигмоидальная сеть доверия (SBN) на этапе функционирования

Для генеративной стохастической SBN используется стохастическое обучение (обучение без учителя). В отличие от генеративной RBM, SBN работает только в положительной фазе.

Для генеративной стохастической SBN в положительной фазе фиксируются видимые нейроны, а ИНС функционирует до тех пор, пока не установятся скрытые нейроны.

Компонентами стохастической SBN являются стохастические нейроны, состояние которых описывается на основе распределения Бернулли в виде

$$x_j = \begin{cases} 1, & \text{с вероятностью } P_j \\ 0, & \text{с вероятностью } 1 - P_j \end{cases}.$$

Величина изменения состояния j-го стохастического нейрона определена в виде

$$\Delta x_j = \hat{x}_j - x_j,$$

где $\hat{x}_j$ – состояние, противоположное состоянию x_j, т.е. $\hat{x}_j = 1 - x_j$.

Вероятность перехода j-го стохастического нейрона из текущего состояния x_j в новое состояние $\hat{x}_j$ определяется на основе имитации отжига в виде

$$P_j = \frac{1}{1 + \exp\left(-\frac{\Delta E_j}{T}\right)},$$

где ΔE_j – приращение энергии сети при изменении состояния j-го нейрона, T – температура.

Обучение ИНС

1. Номер итерации обучения $n=1$, инициализация посредством равномерного распределения на интервале (0,1) или [-0.5, 0.5] весов $w_{ij}(n)$, $i \in \overline{1,N^v}$, $j \in \overline{N^v+1,N^v+N^h}$. Задание параметра P_0, температурных параметров $T_{\max}, T_{\min}$.

2. Инициализация бинарного вектора состояний скрытых нейронов $\mathbf{x}^h=(x_1^h,...,x_{N^h}^h)$. Задается обучающее множество $\{\mathbf{x}_\mu^v \mid \mathbf{x}_\mu^v \in \{0,1\}^{N^v}\}$, $\mu \in \overline{1,P}$, где $\mathbf{x}_\mu^v$ – μ-й обучающий вектор состояний видимых нейронов, P – мощность обучающего множества.

Положительная фаза (шаги 3-10)

3. μ=1.

4. $k=1$, $T(k)=T_{\max}$.

$\mathbf{x}=(x_{\mu 1}^v,...,x_{\mu N^v}^v,x_1^h,...,x_{N^h}^h)$

5. Квантование Гиббса для скрытых нейронов ($j \in \overline{N^v+1,N^v+N^h}$)

$\Delta x_j = \hat{x}_j - x_j$.

$$\Delta E_j = -\Delta x_j \sum_{i=1}^{N^v} w_{ij}(n) x_i .$$

$$P_j = \frac{1}{1+\exp\left(-\frac{\Delta E_j}{T(k)}\right)}.$$

Если $\Delta E_j \le 0$, то $x_j = \hat{x}_j$.

Если $\Delta E_j > 0$ и $P_j \ge P_0$, то $x_j = \hat{x}_j$.

6. Если есть скрытые нейроны, которые не изменили состояние, переход к 5.

7. Уменьшение температурного коэффициента в соответствии с методом моделирования отжига

$T(k+1)=\beta T(k)$,

где β – коэффициент охлаждения, $0<\beta<1$.

8. Если $T(k)>T_{\min}$, то $k=k+1$, переход к 5.

10. $\mathbf{x}1_\mu = \mathbf{x}$. Если $\mu < P$, то $\mu=\mu+1$, переход к 4.

11. Настройка синаптических весов на основе правила обучения сигмоидальной сети доверия

$$w_{ij}(n) = w_{ij}(n) + \eta\rho_{ij}, i \in \overline{1, N^v}, \ j \in \overline{N^v + 1, N^v + N^h},$$

$$\rho_{ij} = \frac{1}{P}\sum_{\mu=1}^{P} \varphi(x1_{\mu j}\, x1_{\mu i}) \frac{1}{1 + \exp\left(-\frac{x1_{\mu j}}{T} \sum_{r=1}^{N^v} w_{rj}(n) x1_{\mu r} \right)},$$

$$\varphi(x_i, x_j) = \begin{cases} 1, & x_i = x_j \\ 0, & x_i \neq x_j \end{cases}.$$

12. Если $|\rho_{ij}| > \varepsilon$, то $n = n + 1$, переход к 2.

Функционирование ИНС

1. Инициализация бинарного вектора состояний скрытых нейронов $\mathbf{x}^h = (x_1^h, ..., x_{N^h}^h)$. Добавляются обратные связи, т.е. $w_{ij} = w_{ji}$, $i \in \overline{1, N^v}$, $j \in \overline{N^v + 1, N^v + N^h}$.

Положительная фаза (шаги 2-7)

2. $k = 1$, $T(k) = T_{\max}$.

$\mathbf{x} = (x_1^v, ..., x_{N^v}^v, x_1^h, ..., x_{N^h}^h)$.

3. Квантование Гиббса для скрытых нейронов ($j \in \overline{N^v + 1, N^v + N^h}$)

$$\Delta x_j = \hat{x}_j - x_j.$$

$$\Delta E_j = -\Delta x_j \sum_{i=1}^{N^v} w_{ij} x_i .$$

$$P_j = \frac{1}{1 + \exp\left(-\frac{\Delta E_j}{T(k)} \right)}.$$

Если $\Delta E_j \le 0$, то $x_j = \hat{x}_j$.

Если $\Delta E_j > 0$ и $P_j \ge P_0$, то $x_j = \hat{x}_j$.

4. Если имеются скрытые нейроны, которые изменили состояние, то переход к 3.

5. Уменьшение температурного коэффициента в соответствии с методом моделирования отжига

$T(k+1) = \beta T(k)$,

где β – коэффициент охлаждения, $0 < \beta < 1$.

6. Если $T(k) > T_{\min}$, то $k = k+1$, переход к 3.

7. $\mathbf{x}1 = \mathbf{x}$.

Отрицательная фаза (шаги 8-13)

8. $k = 1$, $T(k) = T_{\max}$.

$\mathbf{x} = \mathbf{x}1$.

9. Квантование Гиббса для видимых нейронов ($j \in \overline{1, N^v}$)

$\Delta x_j = \widehat{x}_j - x_j$.

$$\Delta E_j = -\Delta x_j \sum_{i=N^v+1}^{N^v+N^h} w_{ij} x_i .$$

$$P_j = \frac{1}{1+\exp\left(-\frac{\Delta E_j}{T(k)}\right)}.$$

Если $\Delta E_j \le 0$, то $x_j = \widehat{x}_j$.

Если $\Delta E_j > 0$ и $P_j \ge P_0$, то $x_j = \widehat{x}_j$.

10. Если имеются видимые нейроны, которые изменили состояние, то переход к 9.

11. Уменьшение температурного коэффициента в соответствии с методом моделирования отжига

$T(k+1) = \beta T(k)$,

где β – коэффициент охлаждения, $0 < \beta < 1$.

12. Если $T(k) > T_{\min}$, то $k = k+1$, переход к 9.

13. $\mathbf{x}^v = (x_1, ..., x_{N^v})$.

Результатом является образец $\mathbf{x}^v$.

Замечание. Если состояние нейронов перестает изменяться, то это означает, что система достигла температурного равновесия, т.е. энергия сети $E(\mathbf{x}) = -\sum_{i=1}^{N^v} \sum_{j=N^v+1}^{N^v+N^h} w_{ij} x_i x_j \to \min$.

Замечание. В случае биполярных данных, т.е. $x_j \in \{-1,1\}$, $E(\mathbf{x}) = -\sum_{i=1}^{N^v} \sum_{j=N^v+1}^{N^v+N^h} w_{ij}(n) x_i x_j$ заменяется на $E(\mathbf{x}) = -\frac{1}{2} \sum_{i=1}^{N^v} \sum_{j=N^v+1}^{N^v+N^h} w_{ij} x_i x_j$,

$\Delta E_j = -\Delta x_j \sum_{i=1}^{N^v} w_{ij}(n) x_i$ заменяется на $\Delta E_j = -\frac{1}{2} \Delta x_j \sum_{i=1}^{N^v} w_{ij}(n) x_i$,

$\hat{x}_j = 1 - x_j$ заменяется на $\hat{x}_j = -x_j$.

Замечание. На основе SBN создаются глубинная сеть доверия (DBN) и машина Гельмгольца (HM), которые представляют собой композицию SBN,. Они имеют один видимый слой и более одного скрытого слоя.

Достоинства

1. Используется для восстановления образцов.
2. Обеспечивает хорошее качество обобщения.
3. Количество классов может быть больше двух.
4. В отличие от DHNN гарантирует нахождение глобального минимума.
5. Распределение вероятности, представляемое нейронами, близко к распределению вероятности среды, в которой работает ИНС.
6. Автоматически определяется количество скрытых слоев (равно одному).
7. Обучение SBN происходит быстрее, чем RBM (только положительная фаза).

Недостатки

1. Отсутствует автоматическое определение числа нейронов в скрытом слое.
2. Обучение стохастической SBN происходит медленнее, чем в случае MLP.
3. В отличие от ART, не решает проблему пластичности-стабильности.
4. Стохастическая SBN работает только с биполярными или бинарными данными.

7.12. Нейросеть ART-1

Нейросеть ART-1 [77] основана на теории адаптивного резонанса (ART) и состоит из пяти функциональных модулей (рис.7.16): нейронного слоя сравнения, нейронного слоя распознавания, нейрона сброса, нейрона управления $G1$, нейрона управления $G2$.

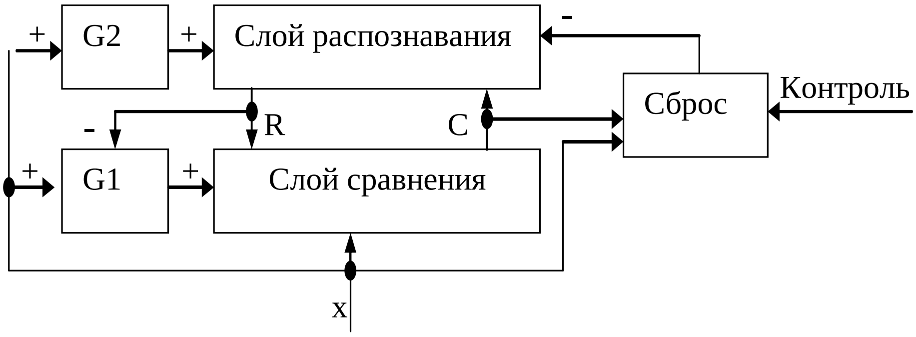


Рис.7.16. Нейросеть ART-1

Если входной вектор близок (входит в резонанс) к вектору весовых коэффициентов определенного нейрона в слое распознавания, то этот вектор весовых коэффициентов модифицируется, иначе в слой распознавания включается новый нейрон с вектором весовых коэффициентов, который соответствует входному вектору.

Нейрон $G1$ выполняет функцию детектора новизны (если входной вектор близок к вектору весовых коэффициентов определенного нейрона в слое распознавания, то $G1=0$). Нейрон $G2$ гасит активность в слое распознавания, если в ИНС не поступило никакой информации. Нейрон сброса определяет сходство между входным вектором $\mathbf{x}$ и вектором весовых коэффициентов определенного нейрона в слое распознавания.

Нейросеть ART-1 реализует автоассоциативную память (ее элемент представлен парой образцов $(\mathbf{m}_x, \mathbf{m}_y)$, $\mathbf{m}_y = \mathbf{m}_x$) и восстанавливает запомненный образец $\mathbf{m}_y$ по ключевому образцу $\mathbf{m}_x$, соответствующему входному вектору $\mathbf{x}$.

Важнейшим свойством нейросети ART-1 является то, что одна и та же ИНС с одними и теми же весами связей может хранить и воспроизводить несколько различных запомненных образцов.

Для нейросети ART-1 используется конкурентное обучение (обучение без учителя),

Обучение ИНС

1. Инициализация весов $v_{ji} = 1$, $w_{ij} = \lambda / (\lambda - 1 + N^{(0)})$, $\lambda \in [1,2]$, $i \in \overline{1, N^{(1)}}$, $j \in \overline{1, N^{(2)}}$, где w_{ij} – бинарный вес связи от i-го элемента слоя сравнения к j-му нейрону слоя распознавания, v_{ji} – вещественный вес связи от j-го элемента слоя распознавания к i-му нейрону слоя сравнения, $N^{(1)}$ – количество нейронов слоя сравнения, $N^{(2)}$ – количество нейронов слоя распознавания.

Задается параметр сходства ρ, $0 \le \rho \le 1$, который показывает, насколько должен входной сигнал совпадать с одним из векторов весовых коэффициентов определенного нейрона в слое сравнения, чтобы они считались похожими. Если $\rho = 1$, то входной сигнал и вектор весовых коэффициентов должны совпадать.

Задается обучающее множество $\{\mathbf{x}_\mu \mid \mathbf{x}_\mu \in \{0,1\}^{N^{(1)}}\}$, $\mu \in \overline{1, P}$, где $\mathbf{x}_\mu$ – μ-й обучающий входной вектор, P – мощность обучающего множества. Номер вектора из обучающего множества $\mu = 1$.

Множество заторможенных нейронов-победителей $J = \varnothing$.

2. Установка значений сигналов

Значение нейрона управления $G1 = 1$.

Взвешенный выходной сигнал слоя распознавания $\mathbf{u}$ равен нулевому вектору, т.е. $u_i = 0$, $i \in \overline{1, N^{(1)}}$.

3. Вычисление выходного сигнала слоя сравнения

$G2 = x_{\mu 1} \vee ... \vee x_{\mu N^{(1)}}$.

Если $G2 = 0$, то $u_i = 0$, $i \in \overline{1, N^{(1)}}$.

$c_i = G1 x_{\mu i} \vee G1 u_i \vee x_{\mu i} u_i$, $i \in \overline{1, N^{(1)}}$.

4. Взвешивание выходного сигнала слоя сравнения

$$t_j = \sum_{i=1}^{N^{(1)}} w_{ij} c_i, \quad j \in \overline{1, N^{(2)}},$$

5. Определение нейрона-победителя в слое распознавания

$$j^* = \arg\max_j t_j, \quad j \in \overline{1, N^{(2)}}, \quad j \notin J,$$

6. Вычисление выходного сигнала слоя распознавания

$$r_j = \begin{cases} 1, & j = j^* \\ 0, & j \neq j^* \end{cases}, \quad j \in \overline{1, N^{(2)}}.$$

Если $r_{j^*} = 1$, то сигнал управления $G1 = 0$.

7. Взвешивание выходного сигнала слоя распознавания

$$u_i = \sum_{j=1}^{N^{(2)}} v_{ji} r_j, \quad i \in \overline{1, N^{(1)}}.$$

8. Вычисление выходного сигнала слоя сравнения

$$c_i = G1 x_{\mu i} \vee G1 u_i \vee x_{\mu i} u_i, \quad i \in \overline{1, N^{(1)}}.$$

9. Сравнение векторов $\mathbf{x}_\mu$ и $\mathbf{c}$

$$S = \frac{\| \mathbf{c} \|}{\| \mathbf{x} \|} = \frac{\sum_{i=1}^{N^{(1)}} c_i}{\sum_{i=1}^{N^{(0)}} x_{\mu i}}$$

Если $S > \rho$, то осуществляется переход к шагу 10.

Если $S \leq \rho$ и $| J | = N^{(2)} - 1$, то $j^* = N^{(2)} + 1$, в слой распознавания добавляется новый нейрон с вектором весовых коэффициентов, который соответствует входному вектору $\mathbf{x}_\mu$, $N^{(2)} = N^{(2)} + 1$, и осуществляется переход к шагу 10.

Если $S \leq \rho$ и $| J | < N^{(2)} - 1$, то вырабатывается сигнал сброса, который тормозит нейрон-победитель j^* в слое распознавания (j^* включается во множество J) и осуществляется переход к шагу 5.

10.Модификация весов нейрона-победителя на основе правила входной звезды (считаем, что $\eta = 1$):

$$v_{j^* i} = c_i, \quad i \in \overline{1, N^{(1)}},$$

$$w_{ij^*} = \frac{\lambda c_i}{\lambda - 1 + \sum_{i=1}^{N^{(1)}} c_i}, \quad i \in \overline{1, N^{(1)}}.$$

11. Проверка условия завершения

Если $\mu < P$, то $J = \varnothing$, $\mu = \mu + 1$, переход к 2.

Функционирование ИНС

1. $J = \varnothing$

2. Установка значений сигналов

$G1 = 1$,

$u_i = 0, \ i \in \overline{1, N^{(1)}}$.

3. Вычисление выходного сигнала слоя сравнения

$G2 = x_1 \vee ... \vee x_{N^{(0)}}$.

Если $G2 = 0$, то $u_i = 0, \ i \in \overline{1, N^{(1)}}$.

$c_i = G1 x_i \vee G1 u_i \vee x_i u_i, \ i \in \overline{1, N^{(1)}}$.

4. Взвешивание выходного сигнала слоя сравнения

$$t_j = \sum_{i=1}^{N^{(1)}} w_{ij} c_i, \ j \in \overline{1, N^{(2)}},$$

5. Определение нейрона-победителя в слое распознавания

$$j^* = \arg\max_j t_j, \ j \in \overline{1, N^{(2)}}, \ j \notin J,$$

Результатом является образец $(v_{j^*1}, ..., v_{j^*N^{(1)}})$.

Достоинства

1. Используется для кластеризации.

2. В отличие от других ИНС, решает проблему пластичности-стабильности.

3. Автоматически определяется количество скрытых слоев (равно нулю).

4. Автоматически определяется число нейронов скрытого слоя (равно нулю).

5. Количество кластеров может быть больше двух (в слое распознавания может быть больше одного нейрона).

7. В отличие от SOM, которая требует задавать количество кластеров (статическая самоорганизация), в ART-1 количество кластеров определяется динамически (динамическая самоорганизация).

Недостатки:

1. В отличие от SOM, необходимо задавать параметр сходства.

2. Если параметр сходства большой, то слишком большое количество нейронов в слое распознавания. Это приводит к более медленному функционированию сети.

3. Если параметр сходства большой, то плохое качество обобщения.

4. Если параметр сходства большой, то плохое качество аппроксимации.

5. В отличие от SOM, обучается медленнее.

6. Работает только с биполярными или бинарными данными.

7.13. Нейросеть ART-2

Нейросеть ART-2 [78,79] основана на теории адаптивного резонанса (ART) и состоит из трех функциональных модулей (рис.7.17): нейронного слоя сравнения, нейронного слоя распознавания, нейрона сброса.

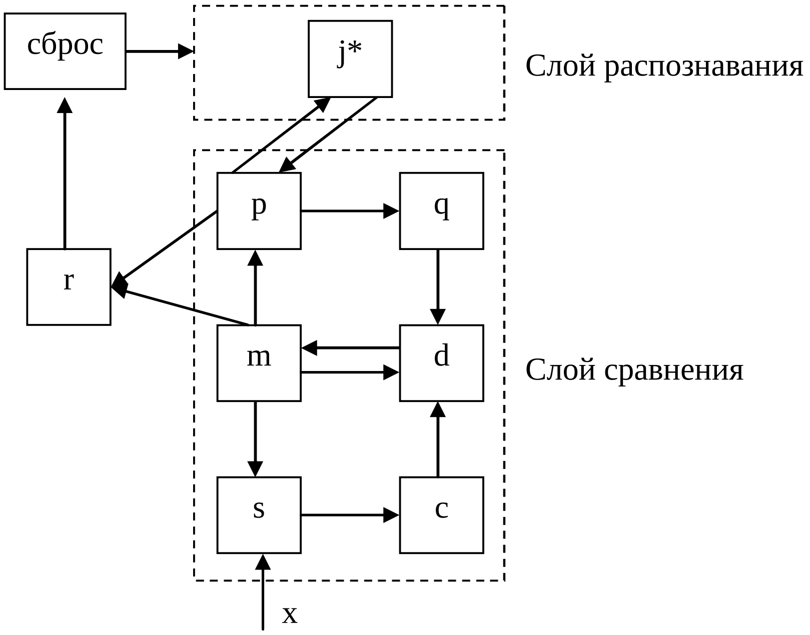


Рис.7.17. Нейросеть ART-2

Если входной вектор близок (входит в резонанс) к вектору весовых коэффициентов определенного нейрона в слое распознавания, то этот вектор весовых коэффициентов модифицируется, иначе в слой распознавания включается новый нейрон с вектором весовых коэффициентов, который соответствует входному вектору.

Нейрон сброса определяет сходство между входным вектором $\mathbf{x}$ и вектором весовых коэффициентов определенного нейрона в слое распознавания.

Нейросеть ART-2 реализует автоассоциативную память (ее элемент представлен парой образцов $(\mathbf{m}_x, \mathbf{m}_y)$, $\mathbf{m}_y = \mathbf{m}_x)$ и восстанавливает запомненный образец $\mathbf{m}_y$ по ключевому образцу $\mathbf{m}_x$, соответствующему входному вектору $\mathbf{x}$.

Важнейшим свойством нейросети ART-2 является то, что одна и та же ИНС с одними и теми же весами связей может хранить и воспроизводить несколько различных запомненных образцов.

Для нейросети ART-2 используется конкурентное обучение (обучение без учителя),

Обучение ИНС

1. Инициализация весов $v_{ji} = 0$, $w_{ij} \le 1/((1-k)\sqrt{N^{(0)}})$, $k \in (0,1)$, $i \in \overline{1, N^{(1)}}$, $j \in \overline{1, N^{(2)}}$, где w_{ij} – вещественный вес связи от *i*-го элемента слоя сравнения к *j*-му нейрону слоя распознавания, v_{ji} – вещественный вес связи от *j*-го элемента слоя распознавания к *i*-му нейрону слоя сравнения, $N^{(1)}$ – количество нейронов слоя сравнения, $N^{(2)}$ – количество нейронов слоя распознавания.

Задается параметр сходства ρ, $0 \le \rho \le 1$, который показывает, насколько должен входной сигнал совпадать с одним из векторов весовых коэффициентов определенного нейрона в слое сравнения, чтобы они считались похожими. Если $\rho = 1$, то входной сигнал и вектор весовых коэффициентов должны совпадать.

Задается обучающее множество $\{\mathbf{x}_\mu \mid \mathbf{x}_\mu \in R^{N^{(1)}}\}$, $\mu \in \overline{1, P}$, где $\mathbf{x}_\mu$ – μ-й обучающий входной вектор, P – мощность обучающего множества. Номер вектора из обучающего множества $\mu = 1$.

Множество заторможенных нейронов-победителей $J = \varnothing$.

2. Установка значений сигналов

Выходные сигналы слоя сравнения равны нулевому вектору, т.е. $m_i = 0$, $p_i = 0$, $q_i = 0$, $d_i = 0$, $i \in \overline{1, N^{(1)}}$. Выходной сигнал слоя сравнения равен нулевому вектору, т.е. $h_j = 0$, $i \in \overline{1, N^{(2)}}$.

3. Вычисление выходных сигналов в слое сравнения

$m_i = \dfrac{d_i}{e + \|\mathbf{d}\|}$, $e > 0$, $i \in \overline{1, N^{(1)}}$, где $\|\cdot\|$ – норма Евклида.

$s_i = x_i + a m_i$, $a \in (0,1)$, $i \in \overline{1, N^{(1)}}$

$$p_i = m_i + \sum_{j=1}^{N^{(2)}} h_j v_{ji}, \; i \in \overline{1, N^{(1)}}$$

$$c_i = \frac{s_i}{e + \| \mathbf{s} \|}, \; e > 0, \; i \in \overline{1, N^{(1)}}$$

$$q_i = \frac{p_i}{e + \| \mathbf{p} \|}, \; e > 0, \; i \in \overline{1, N^{(1)}}$$

$$d_i = f(c_i) + bf(q_i), \; b > 0, \; i \in \overline{1, N^{(1)}}$$

Повторить шаг 3 еще раз и перейти к шагу 4.

4. Взвешивание выходного сигнала слоя сравнения

$$T_j = \sum_{i=1}^{N^{(1)}} p_i w_{ij}, \; j \in \overline{1, N^{(2)}}$$

5. Определение нейрона-победителя в слое распознавания

$$j^* = \arg\max_j T_j, \; j \in \overline{1, N^{(2)}}, \; j \notin J$$

6. Вычисление выходного сигнала слоя распознавания

$$h_j = \begin{cases} k, & j = j^* \\ 0, & j \neq j^* \end{cases}, \; k \in (0,1), \; j \in \overline{1, N^{(2)}}$$

7. Вычисление выходных сигналов в слое сравнения

$$m_i = \frac{d_i}{e + \| \mathbf{d} \|}, \; e > 0, \; i \in \overline{1, N^{(1)}}$$

$$p_i = m_i + \sum_{j=1}^{N^{(2)}} h_j v_{ji}, \; i \in \overline{1, N^{(1)}}$$

8. Сравнение входного вектора и эталона в памяти

$$r_i = \frac{m_i + l p_i}{e + \| \mathbf{m} \| + \| l \mathbf{p} \|}, \; l \in (0,1), \; e > 0, \; i \in \overline{1, N^{(1)}}$$

$$Q = \frac{\rho}{e + \| \mathbf{r} \|}, \; \rho \in (0,1), \; e > 0$$

Если $Q \le 1$, то осуществляется переход к шагу 9.

Если $Q > 1$ и $|J| = N^{(2)} - 1$, то $j^* = N^{(2)} + 1$, в слой распознавания добавляется новый нейрон с вектором весовых коэффициентов, который соответствует входному вектору $\mathbf{x}_\mu$, $N^{(2)} = N^{(2)} + 1$, и осуществляется переход к шагу 9.

Если $Q > 1$ и $|J| < N^{(2)} - 1$, то вырабатывается сигнал сброса, который тормозит нейрон-победитель j^* в слое распознавания (j^* включается во множество J) и осуществляется переход к шагу 5.

9.Модификация весов на основе правила входной звезды:

$$v_{ji} = v_{ji} + \eta h_j (p_i - v_{ji}),\ i \in \overline{1, N^{(1)}},\ j \in \overline{1, N^{(2)}},$$

$$w_{ij} = w_{ij} + \eta h_j (p_i - w_{ij}),\ i \in \overline{1, N^{(1)}},\ j \in \overline{1, N^{(2)}},$$

где η – параметр, определяющий скорость обучения (при большом η обучение происходит быстрее, но увеличивается опасность получить неверное решение), $0 < \eta < 1$.

10. Проверка условия завершения

Если $\mu < P$, то $J = \varnothing$, $\mu = \mu + 1$, переход к шагу 2.

$$f(x) = \begin{cases} \dfrac{20x^2}{x^2 + \theta^2}, & x \in [0, \theta] \\ x, & x > \theta \end{cases},\ \theta = 1/\sqrt{N^{(0)}}$$

или

$$f(x) = \begin{cases} 0, & x \in [0, \theta] \\ x, & x > \theta \end{cases},\ \theta = 1/\sqrt{N^{(0)}}$$

Функционирование ИНС

1. $J = \varnothing$

2. Установка значений сигналов

$$m_i = 0,\ p_i = 0,\ q_i = 0,\ d_i = 0,\ i \in \overline{1, N^{(1)}}.$$

$$h_j = 0,\ i \in \overline{1, N^{(2)}}.$$

3. Вычисление выходных сигналов в слое сравнения

$$m_i = \frac{d_i}{e + \|\mathbf{d}\|},\ e > 0,\ i \in \overline{1, N^{(1)}}$$

$$s_i = x_i + am_i,\ a \in (0,1),\ i \in \overline{1, N^{(1)}}$$

$$p_i = m_i + \sum_{j=1}^{N^{(2)}} h_j v_{ji},\ i \in \overline{1, N^{(1)}}$$

$$c_i = \frac{s_i}{e + \|\mathbf{s}\|},\ e > 0,\ i \in \overline{1, N^{(1)}}$$

$$q_i = \frac{p_i}{e + \| \mathbf{p} \|}, \; e > 0, \; i \in \overline{1, N^{(1)}}$$

$$d_i = f(c_i) + bf(q_i), \; b > 0, \; i \in \overline{1, N^{(1)}}$$

4. Взвешивание выходного сигнала слоя сравнения

$$T_j = \sum_{i=1}^{N^{(1)}} p_i w_{ij}, \; j \in \overline{1, N^{(2)}}$$

5. Определение нейрона-победителя в слое распознавания

$$j^* = \arg\max_j T_j, \; j \in \overline{1, N^{(2)}}, \; j \notin J$$

Результатом является образец $(v_{j^*1}, ..., v_{j^*N^{(1)}})$.

Замечание. Существует также ART-3, которая основана на ART-2, имеет иерархическую структуру и обучается без учителя, и ARTMAP, которая состоит из ART-1 и ART-2 и обучается с учителем.

Достоинства

1. Используется для кластеризации.
2. В отличие от других ИНС, решает проблему пластичности-стабильности.
3. Автоматически определяется количество скрытых слоев (равно нулю).
4. Автоматически определяется число нейронов скрытого слоя (равно нулю).
5. Количество кластеров может быть больше двух (в слое распознавания может быть больше одного нейрона).
7. В отличие от SOM, которая требует задавать количество кластеров (статическая самоорганизация), в ART-2 количество кластеров определяется динамически (динамическая самоорганизация).
8. В отличие от ART-1 работает с вещественными данными.

Недостатки:

1. В отличие от SOM, необходимо задавать параметр сходства.
2. Если параметр сходства большой, то слишком большое количество нейронов в слое распознавания. Это приводит к более медленному функционированию сети.
3. Если параметр сходства большой, то плохое качество обобщения.

4. Если параметр сходства большой, то плохое качество аппроксимации.

5. В отличие от SOM, обучается медленнее.

7.14. Рекуррентна нейросеть анализа главных компонент

На рис.7.18 приведена рекуррентная нейросеть анализа главных компонент (PCARNN) [35], которая является рекуррентной однослойной ИНС. Выход ИНС является линейным. PCARNN выделяет для классифицирующих ИНС информативные признаки, сокращая размерность данных. Таким образом, пространство данных (входное пространство) преобразуется в пространство информативных признаков. Для PCARNN используется обучение Хебба (обучение без учителя), при этом применяются алгоритмы алгоритмы декорреляции (например, APEX).

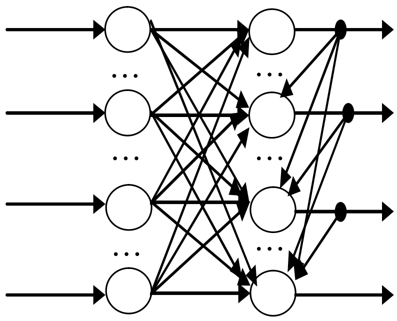


Рис.7.18. Рекуррентная нейросеть анализа главных компонент (PCARNN)

PCARNN реализует гетероассоциативную память (ее элемент представлен парой образцов $(\mathbf{m}_x, \mathbf{m}_y)$, $\mathbf{m}_y \neq \mathbf{m}_x$) и восстанавливает запомненный образец $\mathbf{m}_y$ по ключевому образцу $\mathbf{m}_x$, соответствующему входному вектору $\mathbf{x}$.

Важнейшим свойством PCARNN является то, что одна и та же ИНС с одними и теми же весами связей может хранить и воспроизводить несколько различных запомненных образцов.

Процедура обучения (алгоритм APEX)

1. Номер итерации обучения $n = 1$, инициализация посредством равномерного распределения на интервале (0,1) или [-0.5, 0.5] весов $w_{ij}(n)$ (для прямых связей), $a_{ij}(n)$ (для обратных связей),

$i \in \overline{1, N^{(0)}}$, $j \in \overline{1, N^{(1)}}$, где $N^{(0)}$ – количество нейронов входного слоя, $N^{(1)}$ – количество нейронов выходного слоя, $N^{(1)} < N^{(0)}$.

2. Задается обучающее множество $\{\mathbf{x}_\mu \mid \mathbf{x}_\mu \in R^{N^{(0)}}\}$, $\mu \in \overline{1, P}$, где $\mathbf{x}_\mu$ – μ-й обучающий входной вектор, P – мощность обучающего множества. Номер вектора из обучающего множества $\mu = 1$.

3. Вычисление выходного сигнала первого нейрона

$$y_1(n) = \sum_{i=1}^{N^{(0)}} w_{i1}(n) x_{\mu i}\,,\ j = 2\,.$$

4. Вычисление выходного сигнала нейронов от 1 до j

$$\tilde{y}_k(n) = \sum_{i=1}^{N^{(0)}} w_{ik}(n) x_{\mu i} + \sum_{i=1}^{j-1} a_{ik}(n) y_i(n),\ k \in \overline{1, j}$$

5. Настройка синаптических весов на основе правила Ойя

$$w_{ij}(n+1) = w_{ij}(n) + \eta y_j(n)(x_{\mu i}(n) - y_j(n) w_{ij}(n)),\ i \in \overline{1, N^{(0)}},$$

$$a_{ij}(n+1) = a_{ij}(n) - \eta y_j(n)(y_i(n) - y_j(n) a_{ij}(n)),\ i \in \overline{1, j-1}.$$

6. $\mathbf{y}(n+1) = \tilde{\mathbf{y}}(n)$

Если $j < N^{(1)}$, то $j = j+1$, переход к 4.

7. Проверка условия завершения

Если $\sum_{i=1}^{N^{(0)}} \sum_{j=1}^{N^{(1)}} | w_{ij}(n+1) - w_{ij}(n) | > \varepsilon$ и $\sum_{i=1}^{N^{(0)}} \sum_{j=1}^{N^{(1)}} | a_{ij}(n+1) | > \varepsilon$, то $n = n+1$, переход к 3.

Если $\mu < P$, то $n = 1, \mu = \mu + 1$, перейти к 3, иначе завершиться.

Функционирование ИНС

1. $y_1 = \sum_{i=1}^{N^{(0)}} w_{i1} x_i$.

2. $y_j = \sum_{i=1}^{N^{(0)}} w_{ij} x_i + \sum_{i=1}^{j-1} a_{ij} y_i$, $j \in \overline{2, N^{(1)}}$.

Результатом является образец (вектор главных компонент) $\mathbf{y}$.

Достоинства

Сокращает размерность данных.

Недостатки

Отсутствует автоматическое определение количества нейронов выходного слоя (информативных признаков).

ЛИТЕРАТУРА

1. Гаврилова Т.А. Базы знаний интеллектуальных систем / Т.А. Гаврилова, В.Ф. Хорошевский. – СПб.: Питер, 2001. – 384 с.

2. Хайкин С. Нейронные сети: полный курс / С. Хайкин. – М.: Издательский дом «Вильямс», 2006. – 1104 с.

3. Комарцова Л.Г. Нейрокомпьютеры / Л.Г. Комарцова, А.В. Максимов. – М.: Изд-во МГТУ им. Н.Э. Батмана, 2002. – 320 с.

4. Загоруйко Н.Г. Прикладные методы анализа данных и знаний / Н.Г. Загоруйко. – Новосибирск: Изд-во Ин-та математики, 1999. – 270 с.

5. Rumelhart D.E. Parallel distributed processing: explorations in the microstructure of cognition / D.E. Rumelhart, J.L.McClelland. – Cambridge, MA MIT Press, 1986. – 352 p.

6. Sivanandam S.N. Introduction to Neural Networks using Matlab 6.0 / S.N. Sivanandam, S. Sumathi, S.N. Deepa – New Delhi: The McGraw-Hill Comp., Inc., 2006. – 660 p.

7. Галушкин А.И. Теория нейронных сетей / А.И. Галушкин. – М.: ИПРЖР, 2000. – 416 с.

8. Moody J. Fast learning in networks of locally-tuned processing units / J. Moody, C.J. Darken. // Neural Computation. – 1989. – Vol.1. – P. 281-294.

9. Hartman E.J. Layered neural networks with Gaussian hidden units as universal approximation / E.J. Hartman, J.D. Keeler, J. Kowalski // Neural Computation. – 1990. – Vol.2. – P. 210-215.

10. Park J. Universal approximation using radial-bases function networks / J. Park, I.W. Sandberg // Neural Computation. – 1991. – Vol.3. – P. 246-257.

11. Leonard J.A. Using radial basis functions to approximate a function and its error bound / J.A. Leonard, M.A. Kramer, L.H. Ungar. // IEEE Trans. on Neural Networks. – 1994. – Vol.5. – P. 614-627.

12. Specht D.F. A general regression neural network / D.F. Specht // IEEE Trans. on Neural Networks. – 1991. – Vol.2. – P. 568-576.

13. Specht D.F. Probabilistic neural networks / D.F. Specht // Neural Networks. – 1990. – Vol.3. – P. 109-118.

14. Каллан Р. Основные концепции нейронных сетей / Р. Каллан. – М.: Издательский дом «Вильямс», 2001. – 288 с.

15. Cortes C. Support vector networks / C. Cortes, V. Vapnik // Machine Learning. – 1995. – Vol.20. – P. 273-297.

16. Kung S.Y. DecisionBased Neural Networks with Signal/Image Classification Applications / S.Y. Kung, J.S . Taur // IEEE Transactions on Neural Networks. – 1995. –Vol. 6, № 1. – P. 170-181.

17. Yiu K.K. Gaussian mixture models and probabilistic decision-based neural networks for pattern classification: A comparative study / K.K. Yiu, M.W. Mak, C.K. Li // Neural computing and applications. – Vol. 8. – 1999. – P. 235-245.

18. Lin S. Face recognition/detection by a probabilistic decision-based neural network / S. Lin, S. Kung, L. Lin. // IEEE Transactions on Neural Networks. – Vol. 8, № 1. – 1997. – P. 114-132.

19. Jordan M.I. Modular and Hierarchical Learning Systems / M.I.Jordan, R.A. Jacobs // The Handbook of brain Theory and Neural Networks / Ed. M.A. Arbib. – Cambridge, MA: MIT Press, 1995. – P.579-583.

20. Jordan M.I. Hierarchical mixtures of experts and the EM algorithm / M.I. Jordan, R.A. Jacobs // Neural Computation. – 1994. – Vol.6. – P. 181-214.

21. Sadek S. Image retrieval using cubic splines neural networks / S. Sadek, A. Al-Hamadi, B. Michaelis, U. Sayed // Intern. J. of Video & Image Processing and Network Security. – 2008. – Vol. 9, № 10. – P.17-22.

22. Роджерс Д. Математические основы машинной графики / Д. Роджерс, Дж. Адамс. – М.: Мир, 1980. – 201 с.

23. Vecci L. Approximation capabilities of adaptive spline neural networks / L.Vecci, P. Campolucci; F. Piazza, A. Uncini // International Conference on Neural Networks. – 1997. – Vol. 1. – P. 260 -265.

24. Brown M. Neurofuzzy Adaptive Modelling and Control / M. Brown, C. Harris. – Englewood Cliffs, NJ: Prentice Hall International, 1994. – 508 p.

25. Zhang O. Wavelet networks / Q. Zhang, A. Benveniste // IEEE Trans. on Neural Networks. – 1992. – Vol. 3, № 6. – P. 889-898.

26. Fahlman S. The recurrent cascade-correlation architecture, Technical Report CMU-CS-91-100 / S. Fahlman. – Pittsburgh, PA: Carnegie-Mellon University, 1991. – 12 pp.

27. Marwala T. Causality, Correlation and Artificial Intelligence for Rational Decision Making / T. Marwala. – Singapore: World Scientific Publishing Co. Pte. Ltd., 2015. – 200 p.

28. Bourlard H. Auto-Association by Multilayer Perceptrons and Singular Value Decomposition // H. Bourlard, Y. Kamp // Biological Cybernetycs. – 1988. – Vol. 59. – P. 291-294.

29. Kohonen T. Self-organizing Maps / T. Kohonen. – Berlin: Springer-Verlag, 1995. – 365 p.

30. Ritter H. On stationary state of the Kohonen self-organizing sensory mapping / H. Ritter, K. Schulten // Biolog. Cybern. – 1986. – Vol.53. – P. 405-411.

31. Martinez M. «Neural-gas» network for vector quantization and its application to time series prediction / M. Martinez, S. Berkovich, K. Schulten // IEEE Trans. on Neural Networks. – 1993. – Vol.4. – P. 558-569.

32. Руденко О.Г. Штучні нейронні мережі: Навчальний посібник / О.Г. Руденко, Є.В. Бодянський. – Харків: ТОВ «Компанія СМІТ», 2006. – 404 с.

33. Heht-Nielsen R. Counterpropagation networks / R. Heht-Nielsen // Proc. Int. Conf. on Neural Networks. – N.Y. – 1987. – Vol.2. – P. 19-32.

34. Heht-Nielsen R. Application counterpropagation networks / R. Heht-Nielsen // Neural Networks. – 1988. – Vol.1. – P. 131-139.

35. Sanger T.D. Optimal unsupervised learning in a single-layer linear feedforward neural network / T.D. Sanger // Neural Networks. – 1989. – Vol.2. – P. 459-473.

36. Bartlett M.S. Face Recognition by Independent Component Analysis / M.S. Bartlett, J.R. Movellan, T.J. Sejnowski // IEEE Trans. on Neural Networks. – 2002. – Vol. 13, № 6. – P. 1450-1464.

37. Draper B. Recognizing Faces with PCA and ICA / B. Draper, K. Baek, M.S. Bartlett, J.R. Beveridge Computer Vision and Image Understanding (Special Issue on Face Recognition). – 2003. – Vol. 91, Issues 1-2, P. 115-137.

38. Albus J.S. A new approach to manipulator control: the cerebellar model articulation controller (CMAC) // Journal of dynamic systems, measurement, and control trans. ASME. – 1975. – Vol. 97, № 6. – P. 228-233.

39. Connor J.T. Recurrent Neural Networks and Robust Time Series Prediction / J.T. Connor, R.D. Martin, L.E. Atlas // IEEE Transactions on Neural Networks. – 1994. –Vol. 5, № 2. – P. 240-254.

40. Осовский С. Нейронные сети для обработки информации / С. Осовский. – М.: Финансы и статистика, 2002. – 344 с.

41. Lang K.J. The development of the time-delay neural network architecture for speech recognition, Technical Report CMU-CS-88-152 / K.J. Lang, Hinton G.E. – Pittsburgh, PA: Carnegie-Mellon University, 1988. – 100 pp.

42. Waibel A. Phoneme recognition using time-delay neural networks / A. Waibel, T. Hanazawa, G. Hinton, K. Shikano, K.J. Lang // IEEE transactions on Acoustics, Speech and Signal Processing. – 1988. – Vol.37. – P. 328-329.

43. Брилюк Д.В. Распознавание человека по изображению лица нейросетевыми методами / Д.В. Брилюк, В.В. Старовойтов. – Минск: ИТК, 2002. – 54 с.

44. Cun L. Convolutional networks for images, speech, and time series / L. Cun, Y. Bengio // The Handbook of Brain Theory and Neural Networks / M.A. Arbib. – Cambridge: MITPress, 1995. – P. 255–258.

45. Lawrence S. Face Recognition: A Convolutional Neural Network Approach / S. Lawrence, C.L. Giles, A.C. Tsoi, A.D. Back // IEEE Transactions on Neural Networks, Special Issue on Neural Networks and Pattern Recognition. – 1997. – Vol.1 – P. 1-24.

46. Fukushima K. Cognitron: A self-organizing multiplayered neural network / K. Fukushima // Biological cybernetics. – 1975. – № 20. – P.121–126.

47. Нейронные сети: история развития теории: Учеб. пособие для вузов. / под ред. А.И. Галушкина, Я.З. Цыпкина. – М.: ИПРЖР, 2001. – 840 с.

48. Fukushima K. Neocognitron for handwritten digit recognition / K. Fukushima // Neurocomputing. – 2003. – № 51. – P. 161–180.

49. Bohte S.M. Error-Backpropagation in Temporally Encoded Networks of Spiking Neurons / S.M. Bohte, H.A. La Poutr´e, J.N. Kok // Neurocomputing, – 2002. – Vol. 48. – P. 17-37.

50. Song S. Competitive Hebbian learning through spike-timing-dependent synaptic plasticity / S. Song, K.D. Miller, L.F. Abbott // Nature neuroscience. – 2000. – Vol. 3, № 9. – P. 919-926.

51. Jordan M.I. Attractor dynamics and parallelism in a connectionist sequential machine / M.I. Jordan // Proc. of the Ninth Annual conference of the Cognitive Science Society. – Hillsdale, NJ. – 1986. – P. 531–546.

52. Jordan M. Forward models: supervised learning with a distal teacher / M. Jordan, D. Rumelhart // Cognitive Science. – 1992. – Vol. 16, P. 307–354.

53. Zhang Z. A novel learning method for Elman neural network using local search / Z. Zhang, Z. Tang, C. Vairappan // Neural Information Processing – Letters and Reviews. – 2007. – Vol. 11, № 8. – P. 181–188.

54. Wiles J. Learning to count without a counter: a case study of dynamics and activation landscapes in recurrent networks / J. Wiles, J. Elman // Proc. of the Seventeenth Annual Conference of the Cognitive Science Society. – Cambridge, MA. –1995. – P. 1200-1205.

55. Hochreiter S. Long short-term memory / S. Hochreiter, J. Schmidhuber // Technical Report FKI-207-95, Fakultat fur Informatik, Technische Universitat Munchen.

56. Gers F. Long Short-Term Memory in Recurrent Neural Networks / F. Gers // PhD thesis, Ecole Polytechnique Federale de Lausanne.

57. Jaeger H. A tutorial on training recurrent neural networks, covering BPPT, RTRL, EKF and the "echo state network" approach / H.Jaeger // GMD Report 169, Fraunhofer Institute AIS, 2002.

58. Schuster M. Bidirectional Recurrent Neural Networks / M. Schuster, K.K. Paliwal // IEEE Transactions on Signal Processing. – 1997. – Vol.45, № 11. – P. 2673-2681.

59. Pollack J.B. Recursive distributed representations / J.B. Pollack // Recursive distributed representations. – 1990. – Vol.46, Issue 1-2. – P. 77 - 105

60. Hopfield J.J. Neural networks and physical systems with emergent collective computation abilities / J.J. Hopfield // Proc. Nac. Academy of Sciences USA. – 1982. – Vol.79. – P. 2554-2558.

61. Hopfield J.J. Neural computation of decisions in optimization problems / J.J. Hopfield, D.W. Tank // Biolog. Cybern. – 1985. – Vol.52. – P. 141-152.

62. Akiyama Y. Combinational optimization with Gaussian machines / Y. Akiyama, A. Yamashita, M. Kajiura, H. Aiso // Neural Networks. – 1989. – Vol. 1. – P. 533-540.

63. Neelakanta P.S. Neural network modelling: statistical mechanics and cybernetic perspectives / P.S. Neelakanta, D. DeGroff. – Boca Raton, Florida: CRC Press, 1994. – 236 p.

64. Kosko B. Bidirectional associative memories / Kosko B. // IEEE Trans. on Syst., Man and Cybern. – 1988. – Vol.18. – P. 49-60.

65. Anderson J.A. Distinctive features, categorical perception and probability learning: Some applications of a neural models / J.A. Anderson, J.W. Silverstein, S.A. Ritz, R.S. Jones // Psychological Review. – 1977. – Vol.84. – P. 413-451.

66. Anderson J.A. Cognitive and psychological computation with neural models / J.A. Anderson // IEEE Trans. on Syst., Man and Cybern. – 1983. – Vol.13. – P. 799-815.

67. Lippmann R.P. An introduction to computing with neural nets / R.P. Lippmann // IEEE Acoustics, Speech and Signal Processing Magazine. – 1987. – April. – P. 4-22.

68. Куравский Л.С. Нейросетевая технология диагностики патологических состояний по аномалиям электроэнцефалограмм / Л.С. Куравский, С.Н. Баранов, О.Е. Буланова, Т.Е. Кравчук // Нейрокомпьютеры: разработка и применение. 2007. № 4. С. 4–14.

69. Ackley D.H. A learning algorithm for Boltzmann machines / D.H. Ackley, G.E. Hinton, T.J. Sejnowski // Cognitive Science. – 1985. – Vol.9. – P. 147-169.

70. Kirkpatrick S. Optimization by simulated annealing / S. Kirkpatrick, C.D.Jr. Gelatt, M.P. Vecchi // Science. – 1983. – Vol.220. – P. 671-680.

71. Hinton G.E. A Practical Guide to Training Restricted Boltzmann Machines / G.E. Hinton // UTML TR 2010–003, University of Toronto, 2010.

72. Fischer A. Training Restricted Boltzmann Machines: An Introduction / A. Fischer, C. Igel. // Pattern Recognition. – 2014. – Vol. 47. – P. 25-39.

73. Taylor G.W. Modeling Human Motion Using Binary Latent Variables / G.W. Taylor, G.E. Hinton, S.T. Roweis // NIPS 2006. – Vancouver. – 2006. – P. 1345-1352

74. Salakhutdinov R.R. Deep Boltzmann machines / R.R. Salakhutdinov, G.E. Hinton // Journal of Machine Learning Research. – 2009. – Vol. 5. – P. 448-455.

75. Salakhutdinov R.R. Efficient Learning of Deep Boltzmann Machines / R.R. Salakhutdinov, H. Larochelle // Journal of Machine Learning Research. – 2010. – Vol. 9. – P. 693-700.

76. Neal R.M. Connectionist learning of belief networks / R.M. Neal // Artificial Intelligence. – 1992. – Vol. 56. – P 71-113.

77. Grossberg S. Nonlinear neural networks: principles, mechanisms and architectures / S. Grossberg // Neural Networks. – 1988. – Vol.1. – P. 17-61.

78. Carpenter G.A. ART-2: self-organization of stable category recognition codes for analog input patterns / G.A. Carpenter, S. Grossberg // Applied Optics. – 1987. – Vol.26. – P. 4919-4930.

79. Carpenter G.A. ART-3: self-organization of stable category recognition codes for analog input patterns / G.A. Carpenter, S. Grossberg // Neural Networks. – 1990. – Vol.3. – P. 129-152.

Наукове видання

ФЕДОРОВ Євген Євгенович

ШТУЧНІ НЕЙРОННІ МЕРЕЖІ

Монографія

(російською мовою)

В авторській редакції

Формат $60\times84^{1}/_{16.}$ Ум. друк. арк. 19,6.
Тираж 500 прим. Замовлення № 371

Видавець: Державний вищий навчальний заклад «Донецький національний технічний університет», пл. Шибанкова, 2, м. Красноармійськ, 85300, Україна

Свідоцтво про державну реєстрацію суб'єкта видавничої справи: ДК 4911 від 09.06.2015.

Надруковано: