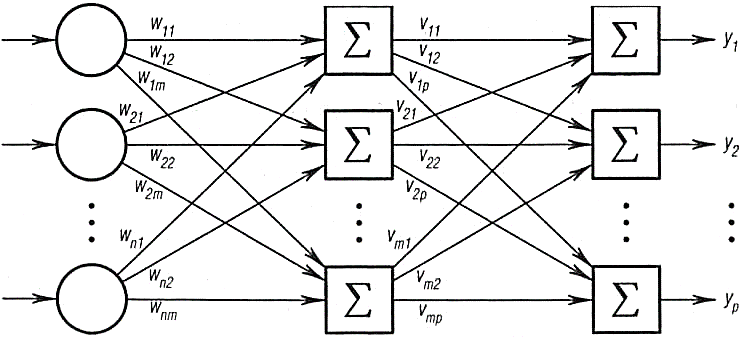
**Л 13. Нейронные cети обратного распространения**

Нейронная сеть обратного распространения. Алгоритм обучения сети обратного распространения.Сеть встречного распространения (сеть Кохонена). Алгоритм обучения сети Кохонена. Входные и выходные звезды и их обучение.Входные и выходные звезды Гроссберга и их обучение. Двухслойная сеть встречного распространения. Алгоритм обучения двухслойной сети встречного распространения.

Рассмотренный в предыдущем параграфе алгоритм обучения однослойного персептрона очень прост. Однако долгие годы не удавалось обобщить этот алгоритм на случай многослойных сетей, что спровоцировало в научных кругах значительный спад интереса к нейронным сетям. Только в 1986 году Румельхарт (D. Е. Rumelhart) разработал эффективный алгоритм корректировки весов, названный**алгоритмом обратного распространения ошибок**(back propagation).

Нейронные сети обратного распространения - это современный инструмент поиска закономерностей, прогнозирования, качественного анализа. Такое название -**сети обратного распространения** - они получили из-за используемого алгоритма обучения, в котором ошибка распространяется от выходного слоя к входному, т.е. в направлении, противоположном направлению распространения сигнала при нормальном функционировании сети.

Нейронная сеть обратного распространения состоит из нескольких слоев нейронов, причем каждый нейрон предыдущего слоя связан с каждым нейроном последующего слоя. В большинстве практических приложений оказывается достаточно рассмотрения двухслойной нейронной сети, имеющей входной (скрытый) слой нейронов и выходной слой (рис. 1).



*Скрытый слой Выходной слой*

Рис. 1. Нейронная сеть обратного распространения

Матрица весовых коэффициентов от входов к скрытому слою - , а матрица весов, соединяющих скрытый и выходной слой - . Обозначения для индексов: входы индекс , элементы скрытого слоя - , а выходы - . Число входов сети - , число нейронов в скрытом слое - , число нейронов в выходном слое - *.* Пусть сеть обучается на выборке

При **обучении нейронной сети ставится задача минимизации целевой функции ошибки**, которая находится по методу наименьших квадратов:

где — полученное реальное значение *-го* выхода нейросети при подаче на нее одного из входных образов обучающей выборки; – требуемое значение -го выхода для этого образца.

Обучение нейросети производится известным **оптимизационным методом градиентного спуска**, т. е. на каждой итерации производится следующееизменение веса

где - параметр, определяющий скорость обучения.

В качестве активационной функции в сети обратного распространения обычно используется сигмоидальная функция

,

где *-* взвешенная сумма входов нейрона. Эта функция удобна для вычислений в градиентном методе, так как имеет простую производную:

Функция ошибки в явном виде не содержит зависимости от весовых коэффициентов и *.*

После некоторых упрощений, вычислений и введения обозначений

Получимследующие выражения для производных:

**Алгоритм обучения сети обратного распространения**

Рассмотрим полный алгоритм обучения нейросети:

**Шаг 1.** Инициализация сети.

Весовым коэффициентам присваиваются малые случайные значения, например, из диапазона (-0,3; 0,3); задаются - параметр точности обучения, - параметр скорости обучения (как правило, и может еще уменьшаться в процессе обучения), - максимально допустимое число итераций.

**Шаг 2.** Вычисление текущего выходного сигнала.

На вход сети подается один из образов обучающей выборки и определяются значения выходов всех нейронов нейросети.

**Шаг 3.** Настройка синаптических весов.

Рассчитать изменение весов для выходного слоя нейронной сети по формулам

Рассчитать изменение весов для скрытого слоя по формулам

**Шаг 4.** Шаги 2-3 повторяются для всех обучающих векторов. Обучение завершается по достижении для каждого из обучающих образов значения функции ошибки, не превосходящего или после максимально допустимого числа итераций - .

**Замечание 1**. На шаге 2 векторы из обучающей последовательности лучше предъявлять на вход в случайном порядке.

**Замечание 2.** Желательно наделять каждый нейрон обучаемым смещением. Это позволяет сдвигать начало отсчета логистической функции, с эффектом, аналогичном подстройке порога персептронного нейрона, и приводит к **ускорению процесса обучения**. Эта возможность может быть легко введена в обучающий алгоритм с помощью добавления к каждому нейрону дополнительного входа. Его вес обучается так же, как и все остальные веса, за исключением того, что подаваемый на него сигнал всегда равен +1, а не выходу нейрона предыдущего слоя.

**Замечание 3.** **Количество входов и выходов сети, как правило, диктуется условиями задачи, а размер скрытого слоя находят экспериментально. Обычно число нейронов в нем составляет 30—50% от числа входов. Слишком большое количество нейронов скрытого слоя приводит к тому, что сеть теряет способность к обобщению (она просто досконально запоминает элементы обучающей выборки и не реагирует на схожие образцы, что неприемлемо для задач распознавания). Если же число нейронов в скрытом слое слишком мало, сеть оказывается просто не в состоянии обучиться.**

**Замечание 4.** Выходы каждого нейрона сети лежат в диапазоне (0, 1) - области значений логистической функции - это надо учитывать при формировании обучающей выборки. Если необходимо получить от сети бинарный выход, то, как правило, вместо 0 используют 0,1, а вместо 1 - 0,9, так как границы интервала недостижимы.

Модификации алгоритма обратного распространения связаны с использованием различных функций ошибки, других активационных функций, различных процедур определения направления и величины шага.

Несмотря на многочисленные успешные применения обратного распространения, оно не является панацеей. Больше всего неприятностей приносит неопределенно долгий процесс обучения. В сложных задачах для обучения сети могут потребоваться часы или даже дни, она может и вообще не обучиться. Неудачи в обучении часто возникают по причине попадания сети в локальный минимум, что является характерной особенностью методов градиентного спуска. Исправить ситуацию в таком случае иногда помогают небольшие случайные изменения весовых значений сети.

**Сеть встречного распространения (сеть Кохонена)**

Рассмотрим алгоритм обучения «победитель забирает все» для задачи классификации Кохонена (Т. Kohonen). При предъявлении входного вектора возбуждается единственный нейрон, наиболее точно соответствующий этому образу.

**Сеть Кохонена. Классификация образов**

Задача классификации заключается в разбиении объектов на классы, причем основой разбиения служит вектор параметров объекта. Чаще бывает так, что сами классы заранее неизвестны, и их приходится формировать динамически. Назовем**прототипом** класса объект, наиболее типичный для своего класса. Один из самых простых подходов к классификации состоит в том, чтобы предположить существование определенного числа классов и произвольным образом выбрать координаты прототипов. Затем каждый вектор из набора данных связывается с ближайшим к нему прототипом, и новыми прототипами становятся центроиды всех векторов, связанных с исходным прототипом. В качестве меры близости двух векторов обычно выбирается евклидово расстояние:

На этих принципах основано функционирование сети Кохонена, обычно используемой для решения задач классификации. Данная сеть обучается**без учителя** на основе самоорганизации. По мере обучения векторы весов нейронов становятся прототипами классов - групп векторов обучающей выборки. На этапе решения информационных задач сеть относит новый предъявленный образ к одному из сформированных классов.

Сеть Кохонена состоит из одного слоя нейронов. Число входов каждого нейрона равно размерности вектора параметров объекта. Количество нейронов *t* совпадает с требуемым числом классов, на которые нужно разбить объекты (меняя число нейронов, можно динамически менять число классов).

Обучение начинается с задания небольших случайных значений элементам весовой матрицы . В дальнейшем происходит процесс самоорганизации, состоящий в модификации весов при предъявлении на вход векторов обучающей выборки. Каждый столбец весовой матрицы представляет собой параметры соответствующего нейрона-классификатора. Для каждого *j* -го нейрона определяется расстояние от него до входного вектора :

Далее выбирается нейрон с номером , для которого это расстояние минимально (т. е. сеть отнесла входной вектор к классу с номером *).* На текущем шаге обучения будут модифицироваться только веса нейронов из окрестности нейрона

Первоначально в окрестности любого из нейронов находятся все нейроны сети, но с каждым шагом эта окрестность сужается. В конце этапа обучения подстраиваются только веса нейрона с номером *.* Темп обучения с течением времени также уменьшается (часто полагают Образы обучающей выборки предъявляются последовательно и каждый раз происходит подстройка весов.

**Алгоритм обучения сети Кохонена**

**Шаг 1.** Инициализация сети. Весовым коэффициентам сети присваиваются малые случайные значения. Задаются значения - начальный темп обучения и*—* максимальное расстояние между весовыми векторами (столбцами матрицы ).

**Шаг 2.** Предъявление сети нового входного сигнала .

**Шаг 3.** Вычисление расстояния от входа X до всех нейронов сети:

**Шаг 4.** Выбор нейрона с наименьшим расстоянием *.*

**Шаг 5.** Настройка весов нейрона и всех нейронов, находящихся от него на расстоянии, не превосходящем

**Шаг 6.** Уменьшение значений

**Шаг 7.** Шаги 2—6 повторяются до тех пор, пока веса не перестанут меняться (или пока суммарное изменение всех весов станет очень мало).

После обучения классификация выполняется посредством подачи на вход сети испытуемого вектора, вычисления расстояния от него до каждого нейрона с последующим выбором нейрона с наименьшим расстоянием как индикатора правильной классификации.

**Замечание**. Если предварительно провести единичную нормировку всех входных векторов, т. е. подавать на вход сети образы , компоненты которого связаны с компонентами вектора по формулам

а также если после каждой итерации процесса обучения осуществлять нормировку весов каждого нейрона (столбцов матрицы ), то в качестве меры близости входных векторов и весовых векторов нейронов сети можно рассматривать скалярное произведение между ними. Действительно, в этом случае

Таким образом, наименьшим будет расстояние до того нейрона, скалярное произведение с весами которого у входного вектора максимально. В этом случае можно считать, что каждый нейрон Кохонена реализует тождественную активационную функцию *,* где

Нейрон с максимальным значением активационной функции объявляется «победителем» и его веса (а также веса нейронов из его окружения) пересчитываются.

Сеть Кохонена нашла широкое применение в задачах финансового анализа. С ее помощью успешно решаются задачи предсказания рисков, нахождения рейтинга и др.

**Входные и выходные звезды Гроссберга**

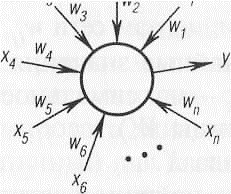
Входная звезда Гроссберга (S. Grossberg), как показано на рис. 2, состоит из нейрона, на который подается группа входов, умноженных на синапсические веса.

Рис. 2. Входная звезда Гроссберга

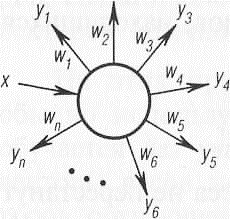
Выходная звезда, показанная на рис. 3, является нейроном, управляющим группой весов. Входные и выходные звезды могут быть взаимно соединены в сети любой сложности.

Рис. 3. Выходная звезда Гроссберга

**Обучение входной звезды**

Входная звезда выполняет распознавание образов, т. е. она обучается реагировать на определенный входной вектор и ни на какой другой. Это обучение реализуется путем настройки весов таким образом, чтобы они соответствовали входному вектору. Входная звезда имеет тождественную активационную функцию *,* т. е. выход входной звезды определяется как взвешенная сумма ее входов:

С другой точки зрения, выход можно рассматривать как скалярное произведение входного вектора с весовым вектором. Если эти векторы имеют единичную норму, то скалярное произведение будет максимальным для того входного образа, которому нейрон был обучен.

В процессе обучения веса корректируются следующим образом:

где . — весовой коэффициент входа ; - нормирующий коэффициент обучения, который имеет начальное значение 0,1 и постепенно уменьшается в процессе обучения.

После завершения обучения предъявление входного вектора будет активизировать обученный входной нейрон. Хорошо обученная входная звезда будет реагировать не только на определенный запомненный вектор, но также и на незначительные изменения этого вектора. Это достигается постепенной настройкой нейронных весов при предъявлении в процессе обучения векторов, представляющих нормированные вариации входного вектора. Веса настраиваются таким образом, чтобы усреднить величины обучающих векторов, и нейроны получают способность реагировать на любой вектор этого класса.

**Обучение выходной звезды**

Если входная звезда учится реагировать на определенный вход, то выходная звезда обучается выдавать требуемый целевой выход.

Чтобы обучить нейрон выходной звезды, его веса настраиваются в соответствии с требуемым целевым вектором .

Формула коррекции весов имеет следующий вид:

где представляет собой нормирующий коэффициент обучения, который вначале приблизительно равен единице и постепенно уменьшается до нуля в процессе обучения.

Как и в случае входной звезды, веса выходной звезды постепенно настраиваются на множество векторов, представляющих собой возможные вариации запоминаемого выходного вектора.

**Двухслойная сеть встречного распространения**

Сеть встречного распространения состоит из двух слоев: слоя нейронов Кохонена и слоя нейронов Гроссберга. Автор сети Р. Хехт-Нильсен (R. Hecht-Nielsen) объединил эти две архитектуры, в результате чего сеть приобрела свойства, которых не было у каждой из них в отдельности.

Слой Кохонена классифицирует входные векторы в группы схожих. Это достигается с помощью такой подстройки весов слоя Кохонена, что близкие входные векторы активируют один и тот же нейрон данного слоя. Затем слой Гроссберга дает требуемые выходы.

На рис. 4 показана сеть встречного распространения.

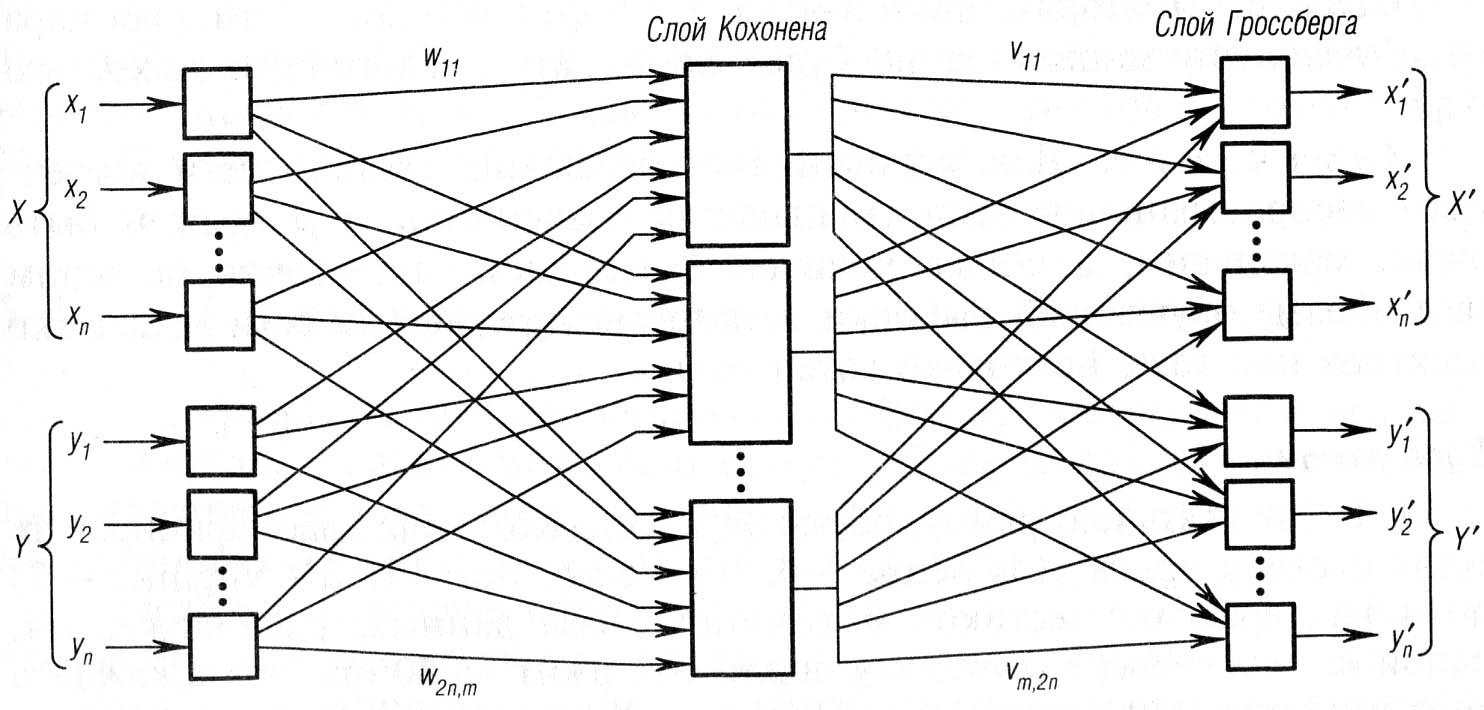


Рис.4. Двухслойная сеть встречного распространения

В режиме нормального функционирования предъявляются входные векторы и , и обученная сеть дает на выходе векторы и , являющиеся аппроксимациями соответственно для и *.* Векторы и предполагаются здесь нормированными векторами единичной длины, следовательно, порождаемые на выходе векторы также должны быть нормированными.

В процессе обучения векторы и подаются одновременно и как входные векторы сети, и как желаемые выходные сигналы. В результате получается отображение, при котором предъявление пары входных векторов порождает их копии на выходе.

Это не было бы интересно, если не учитывать способность этой сети к обобщению. Благодаря обобщению предъявление только вектора (с вектором) порождает как выходы так и выходы *.* Если *-* функция, отображающая в *,* то сеть аппроксимирует ее. Кроме того, если функция обратима (если функция *y = f(x)* такова, что для любого её значения *y0* уравнение *f(x) = y0* имеет относительно *x* единственный корень, то говорят, что функция *f* обратима), то предъявление только вектора (при *)* порождает выходы *.*

Уникальная способность порождать функцию и обратную к ней делает сеть встречного распространения полезной в ряде приложений. Например, в задаче аппроксимации многомерной векторной функции сеть обучается на известных значениях этой функции.

**Алгоритм обучения двухслойной сети встречного распространения**

Шаг 1. Произвести единичную нормировку всех векторов *,*  обучающего множества.

Шаг 2. Весовым коэффициентам сети присвоить малые случайные значения и произвести единичную нормировку матриц,по столбцам. Положить

Шаг 3. Подать на вход сети обучающий набор *(, )* и определить единственный нейрон - «победитель» в слое Кохонена (весовой вектор которого дает максимальное скалярное произведение с входным вектором). Выход этого нейрона установить равным 1, выходы всех остальных нейронов слоя Кохонена положить равными 0. Скорректировать веса выигравшего нейрона:

где

Шаг 4. Подать выходной вектор слоя Кохонена на вход слоя Гроссберга. Скорректировать веса слоя Гроссберга, связанные с выигравшим нейроном слоя Кохонена:

(здесь - номер выигравшего нейрона).

Шаг 5. Уменьшить значения

Шаг 6. Повторять шаги 3—5 до тех пор, пока каждая входная пара из обучающего множества не будет порождать аналогичную выходную пару.

Замечание. Для улучшения обобщающих свойств сети встречного распространения темп уменьшения значений и должен быть очень маленьким, а общее количество итераций достаточно большим (все образы обучающей выборки желательно предъявить сети несколько десятков или даже несколько сотен раз).