Приложение № 2
к распоряжению РТУ МИРЭА
от__ _____№_____

Титульный лист материалов по дисциплине
(заполняется по каждому виду учебного материала)

ДИСЦИЛИНА Проектирование и обучение нейронных сетей
(полное наименование дисциплины без сокращений)

ИНСТИТУТ Информационые технологии

КАФЕДРА Вычислительная техника
полное наименование кафедры

ВИД УЧЕБНОГО МАТЕРИАЛА Лекция
(в соответствии с пп.1-11)

ПРЕПОДАВАТЕЛЬ Сорокин Алексей Борисович
(фамилия, имя, отчество)

СЕМЕСТР 7 семестр, 2023/2024
(указать семестр обучения, учебный год)

10. ЛЕКЦИЯ. Сети встречного распространения

Звезды Гроссберга

Рассмотрим конфигурации входных и выходных звезд Гроссберга. Нейроны типа инстар и оутстар – это взаимодополняющие элементы. Инстар адаптирует веса сигналов, поступающих на сумматор нейрона, к своим входным сигналам, а оутстар согласовывает веса выходящих из нейронов связей с узлами, в которых формируются значения выходных сигналов.

Нейрон в форме входной звезды (инстар) имеет N входов $x_1, x_2, \ldots, x_N$, которым соответствуют веса $w_{i1}, w_{i2}, \ldots, w_{iN}$, и один выход y_i, являющийся взвешенной суммой входов. Входная звезда обучается выдавать сигнал на выходе всякий раз, когда на входы поступает определенный вектор. Таким образом, входная звезда является детектором совокупного состояния своих входов (рис. 1).

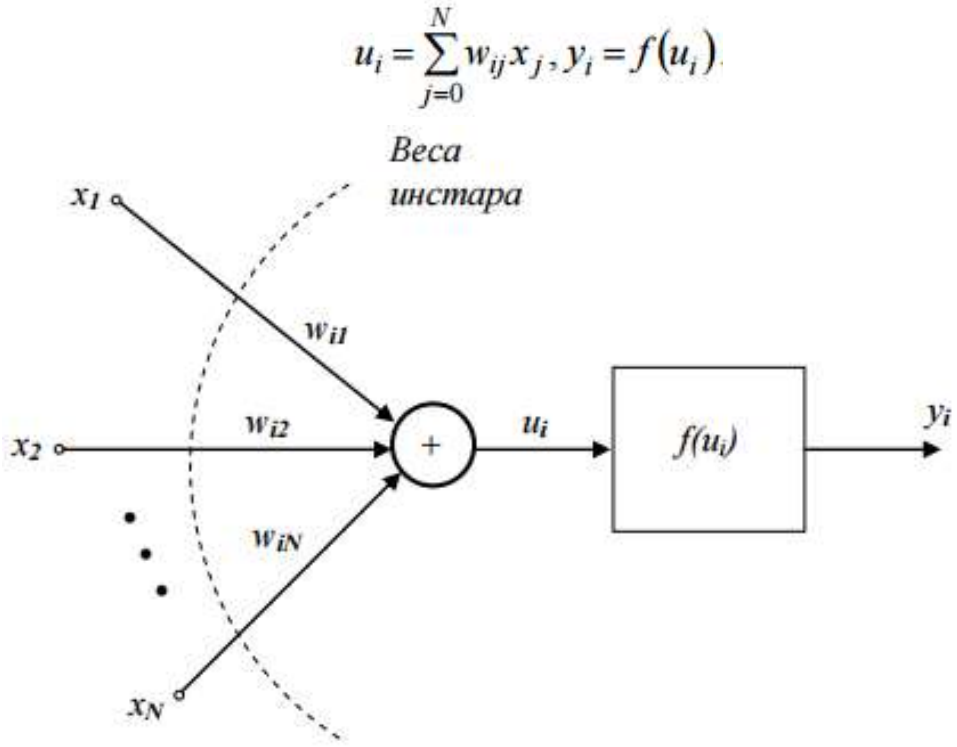

Рис. 1 Структурная схема входной звезды (инстара)

В качестве функции активации часто используется линейная функция, тогда $y_i = u_i$. Процесс обучения представляется в следующей итерационной форме:

$$w_{ij}(t+1) = w_{ij}(t) + \eta\, y_i \left[x_j - w_{ij}(t)\right],$$

где η – коэффициент обучения, значение которого, как правило, выбирается из интервала (0,1). Если принять $\eta = 1$, то $w_{ij} = x_j$ уже после первой итерации обучения. Ввод нового вектора x вызовет адаптацию весов к его

компонентам и «забывание» предыдущих значений. При η <1 инстар принимает усреднённое значение обучающих векторов. Входные данные должны быть нормированы, то есть $x = 1$. При этом

$$x_j \leftarrow \frac{x_j}{\sqrt{x_1^2 + x_2^2 + \ldots + x_N^2}}.$$

Инстар обучается как с учителем, так и без учителя. Если применяется обучение с учителем, то выходные значения y_i являются эталонными значениями d_i.

Выходная звезда Гроссберга (оутстар) выполняет противоположную функцию – функцию командного нейрона, выдавая на выходах определенный вектор, необходимый связанным с ним нейронам, при поступлении сигнала на вход. Структурная схема оутстара представлена на рис. 2.

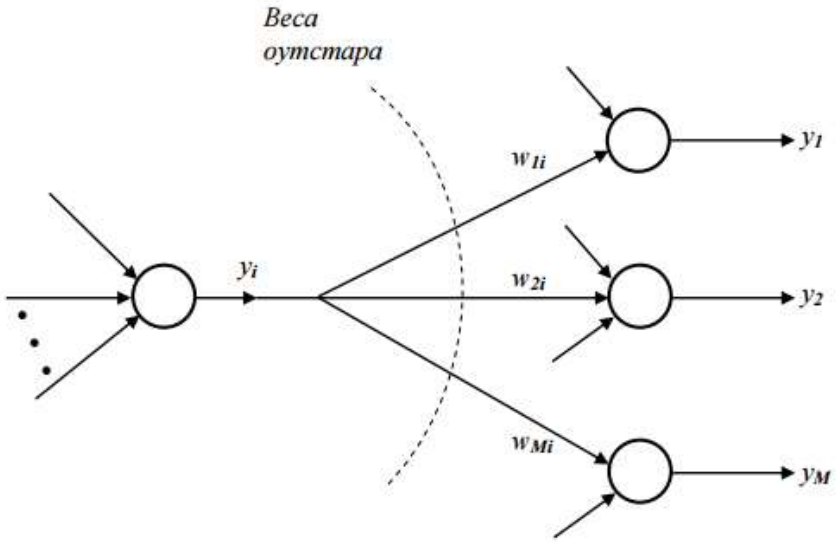

Рис. 2 Структурная схема выходной звезды (оутстара)

Нейрон этого типа имеет один вход и M выходов с весами $w_{1i}, w_{2i}, \ldots, w_{Mi}$. i-ый нейрон-источник высылает свой выходной сигнал y_i взаимодействующим с ним нейронам, выходные сигналы которых обозначены y_j (j = 1, 2, …, M). Оутстар, как правило, является линейным нейроном. Обучение состоит в таком подборе его весов w_{ij}, чтобы выходные сигналы оутстара были равны ожидаемым значениям y_j взаимодействующих с ним нейронов.

Обучение оутстара согласно правилу Гроссберга проводится в соответствии с выражением

$$w_{ji}(t+1) = w_{ji}(t) + \eta \cdot y_i \cdot (y_j - w_{ji}(t)),$$

в котором η – это коэффициент обучения, а y_i – выходной сигнал -го нейрона, выступающего в роли источника. В режиме распознавания в момент

активизации нейрона-источника оутстар будет генерировать сигналы, соответствующие ожидаемым значениям y_j.

Сети встречного распространения

По своим возможностям сети встречного распространения превосходят возможности однослойных сетей. Время же их обучения, по сравнению с обратным распространением, может уменьшаться в сто раз. Встречное распространение не настолько общее, как обратное распространение, но оно может давать решение в тех приложениях, где долгая обучающая процедура невозможна.

Во встречном распространении объединены два хорошо известных алгоритма: самоорганизующаяся карта Кохонена и звезда Гроссберга. При этом появляются свойства, которых нет ни у одного из них в отдельности.

Методы, которые, подобно встречному распространению, объединяют различные сетевые парадигмы как строительные блоки, могут привести к сетям, более близким по архитектуре к мозгу, чем любые другие однородные структуры. Похоже, что в естественном мозге именно каскадные соединения модулей различной специализации позволяют выполнять требуемые вычисления.

Сеть встречного распространения функционирует подобно столу справок, способному к обобщению. В процессе обучения входные векторы ассоциируются с соответствующими выходными векторами; они могут быть двоичными, состоящими из нулей и единиц, или непрерывными. Когда сеть обучена, приложение входного вектора приводит к требуемому выходному вектору. Обобщающая способность сети позволяет получать правильный выход даже при приложении входного вектора, который является неполным или слегка неверным. Таким образом, возможно, использовать данную сеть для распознавания образов, восстановления образов и усиления сигналов.

Структура сети

На рис.3 показана упрощенная версия прямого действия сети встречного распространения. Здесь иллюстрируются функциональные свойства этой парадигмы.

Нейроны слоя 0 (показанные кружками) служат лишь точками разветвления и не выполняют вычислений. Каждый нейрон слоя 0 соединен с каждым нейроном слоя 1 (называемого слоем Кохонена) отдельным весом w_{min}. Эти веса в целом рассматриваются как матрица весов W Аналогично, каждый нейрон в слое Кохонена (слое 1) соединен с каждым нейроном в слое Гроссберга (слое 2) весом $v_{\text{пр}}$Эти веса образуют матрицу весов V. Все это весьма напоминает

другие сети, встречавшиеся в предыдущих лекциях; различие, однако, в операциях, выполняемых нейронами Кохонена и Гроссберга.

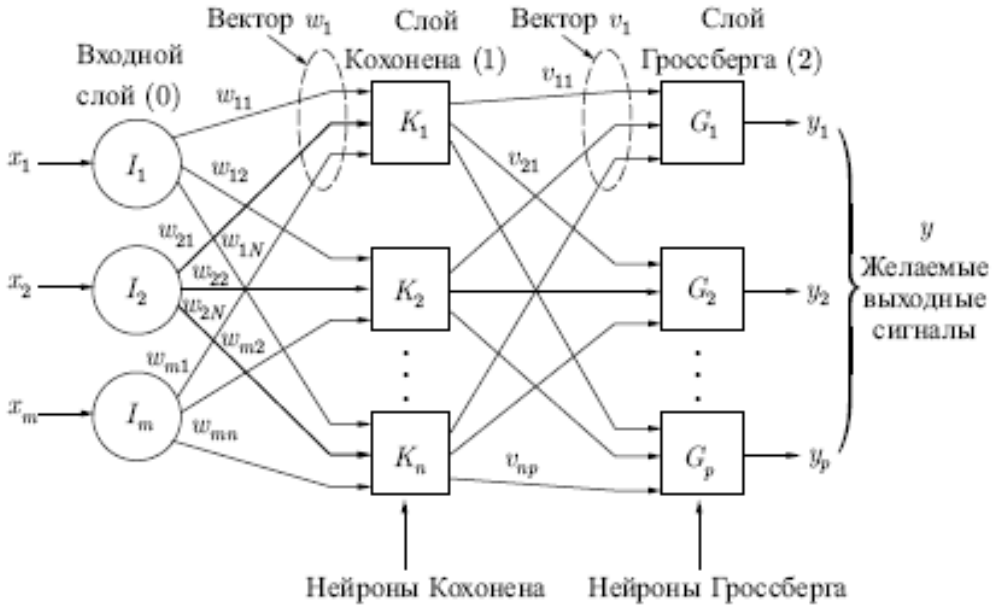

Рис. 3 Упрощенная версия прямого действия сети встречного распространения

Как и многие другие сети, встречное распространение функционирует в двух режимах: в нормальном режиме, при котором принимается входной вектор X и выдается выходной вектор Y, и в режиме обучения, при котором подается входной вектор и веса корректируются, чтобы дать требуемый выходной вектор.

Нормальное функционирование

Слой Кохонена

В своей простейшей форме слой Кохонена функционирует по принципу «победитель получает все», то есть для данного входного вектора только один нейрон Кохонена выдает на выходе логическую единицу, все остальные выдают ноль.

Каждый нейрон Кохонена соединён с каждой компонентой входного вектора. Подобно нейронам большинства сетей выход u каждого нейрона Кохонена является просто суммой взвешенных входов. Это может быть выражено следующей формулой:

$$u_i = w_{i1} \cdot x_1 + w_{i2} \cdot x_2 + \ldots + w_{iN} \cdot x_N ;$$

де u_i – это выход ui –го нейрона Кохонена,

$$u_i = \sum_{j=1}^{N} x_j \cdot w_{ij}$$

или в векторной записи

$$u = x \cdot w .$$

Нейрон Кохонена с максимальным значением u является нейроном-победителем.

Слой Гроссберга

Слой Гроссберга функционирует в сходной манере. Его выход y является взвешенной суммой выходов $u_1, u_2, , \ldots, u_K$, слоя Кохонена, образующих вектор u. Вектор соединяющих весов, обозначенный через v, состоит из весов $v_{11}, v_{21}, \ldots, v_{MK}$. Тогда выход y каждого нейрона Гроссберга есть

$$y_s = \sum_{i=1}^{M} u_{s_i} \cdot v_{si},$$

где y_s – выход -го нейрона Гроссберга, или в векторной форме

$$\mathbf{y} = \mathbf{u} \cdot \mathbf{v}$$

где y – выходной вектор слоя Гроссберга, u – выходной вектор слоя Кохонена, v – матрица весов слоя Гроссберга. Если слой Кохонена функционирует таким образом, что лишь один элемент вектора u отличен от нуля, то вычисления очень просты. Фактически каждый нейрон слоя Гроссберга лишь выдает величину веса, который связывает этот нейрон с единственным ненулевым нейроном Кохонена.

Обучение слоя Кохонена

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 задачей слоя Гроссберга является получение требуемых выходов. Обучение Кохонена является самообучением, протекающим без учителя. Поэтому трудно (и не нужно) предсказывать, какой именно нейрон Кохонена будет активироваться для заданного входного вектора. Необходимо лишь гарантированно добиться, чтобы в результате обучения разделялись несхожие входные векторы.

Предварительная обработка входных векторов

Весьма желательно (хотя и не обязательно) нормализовать входные векторы перед тем, как предъявлять их сети. Операция выполняется с помощью деления каждой компоненты входного вектора на длину вектора. Эта длина находится извлечением квадратного корня из суммы квадратов компонент вектора. В алгебраической записи

$$s_i' = \frac{x_i}{\sqrt{x_1^2 + x_2^2 + \ldots + x_n^2}}. \quad (1)$$

Таким образом, входной вектор превращается в единичный вектор с тем же самым направлением, т.е. в вектор единичной длины в n-мерном пространстве.

Уравнение (1) обобщает хорошо известный случай двух измерений, когда длина вектора равна гипотенузе прямоугольного треугольника, образованного его x и y компонентами, как это следует из известной теоремы Пифагора. На рис. 4. такой двумерный вектор V представлен в координатах $x - y$, причем координата x равна четырем, а координата y – трем. Квадратный корень из суммы квадратов этих компонент равен пяти. Деление каждой компоненты V на пять дает вектор V с компонентами 4/5 и 3/5, где V' указывает в том же направлении, что и V, но имеет единичную длину.

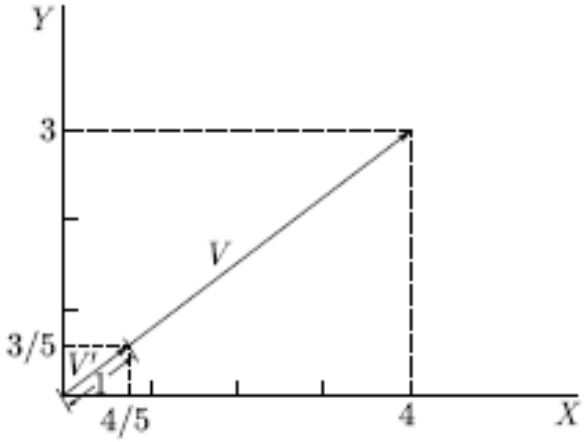

Рис. 4. Двумерный вектор

На рис. 5. показано несколько единичных векторов. Они оканчиваются в точках единичной окружности (окружности единичного радиуса), а это происходит, когда у сети лишь два входа. В случае трех входов векторы представлялись бы стрелками, оканчивающимися на поверхности единичной сферы. Такие представления могут быть перенесены на сети, имеющие произвольное число входов, где каждый входной вектор является стрелкой, оканчивающейся на поверхности единичной гиперсферы (полезной абстракцией, хотя и не допускающей непосредственной визуализации).

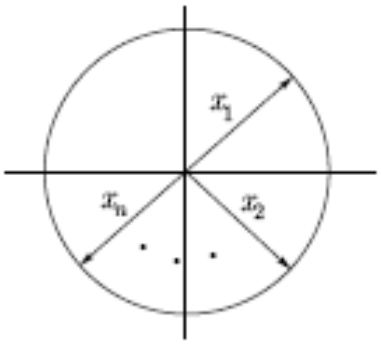

Рис. 5. Множество единичных векторов

При обучении слоя Кохонена на вход подается входной вектор и вычисляются его скалярные произведения с векторами весов, связанными со всеми нейронами Кохонена. Нейрон с максимальным значением скалярного произведения объявляется "победителем", и его веса подстраиваются. Так как скалярное произведение, используемое для вычисления величин NET, является мерой сходства между входным вектором и вектором весов, то процесс обучения состоит в выборе нейрона Кохонена с весовым вектором, наиболее близким к входному вектору, и дальнейшем приближении весового вектора к входному. Снова отметим, что процесс является самообучением, выполняемым без учителя. Сеть самоорганизуется таким образом, что данный нейрон Кохонена имеет максимальный выход для данного входного вектора. Напомню, уравнение, описывающее процесс обучения, имеет следующий вид:

$$w_н = w_c + \alpha(x - w_c),$$

где w_H – новое значение веса, соединяющего входную компоненту с выигравшим нейроном; w_C – предыдущее значение этого веса; α – коэффициент скорости обучения, который может варьироваться в процессе обучения.

Каждый вес, связанный с выигравшим *нейроном Кохонена*, изменяется пропорционально разности между его величиной и величиной входа, к которому он присоединен. Направление изменения минимизирует разность между весом и его входом.

На рис. 6. этот процесс показан геометрически в двумерном виде.

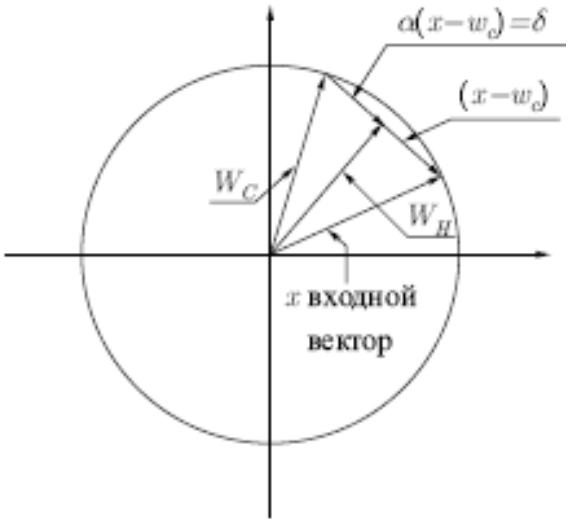

Рис. 6. Разность между весом и его входом

Сначала ищем вектор $X - W_C$, для этого проводится отрезок из конца W в конец X. Затем этот вектор укорачиваем умножением его на скалярную величину α, меньшую единицы, в результате чего получаем вектор изменения δ. Окончательно новый весовой вектор W_H является отрезком, направленным из

начала координат в конец вектора δ. Отсюда можно видеть, что эффект обучения состоит во вращении весового вектора в направлении входного вектора без существенного изменения его длины.

Переменная α является коэффициентом скорости обучения, который вначале обычно равен $\sim 0{,}7$ и может постепенно уменьшаться в процессе обучения. Это позволяет делать большие начальные шаги для быстрого грубого обучения и меньшие шаги при подходе к окончательной величине.

Если бы с каждым нейроном Кохонена ассоциировался один входной вектор, то слой Кохонена мог бы быть обучен с помощью одного вычисления на вес. Веса нейрона-победителя приравнивались бы к компонентам обучающего вектора ($\alpha = 1$). Как правило, обучающее множество включает много сходных между собой входных векторов, и сеть должна быть обучена активировать один и тот же нейрон Кохонена для каждого из них. В этом случае веса этого нейрона должны вычисляться усреднением входных векторов, которые его активируют. Постепенное уменьшение величины α уменьшает воздействие каждого обучающего шага, и окончательное значение будет средней величиной от входных векторов, на которых происходит обучение. Таким образом, веса, ассоциированные с нейроном, примут значение вблизи «центра» входных векторов, для которых данный нейрон является «победителем».

Выбор начальных значений весовых векторов

Всем весам сети перед началом обучения следует придать начальные значения. Общепринятой практикой при работе с нейронными сетями является присваивание весам небольших случайных значений. При обучении слоя Кохонена случайно выбранные весовые векторы следует нормализовать. Окончательные значения весовых векторов после обучения совпадают с нормализованными входными векторами. Поэтому нормализация перед началом обучения приближает весовые векторы к их окончательным значениям, сокращая, таким образом, продолжительность обучающего процесса.

Рандомизация весов слоя Кохонена может породить серьезные проблемы при обучении, так как в результате весовые векторы распределяются равномерно по поверхности гиперсферы. Из-за того, что входные векторы, как правило, распределены неравномерно и имеют тенденцию группироваться на относительно малой части поверхности гиперсферы, большинство весовых векторов будут так удалены от любого входного вектора, что они никогда не смогут дать наилучшее соответствие. Эти *нейроны Кохонена* будут всегда иметь нулевой выход и окажутся бесполезными. Более того, оставшихся весов, дающих наилучшие соответствия, может оказаться слишком мало, чтобы разделить

входные векторы на классы, которые расположены близко друг к другу на поверхности гиперсферы.

Допустим, что имеется несколько множеств входных векторов, все эти множества сходные, но необходимо разделить их на различные классы. Сеть должна быть обучена активировать отдельный нейрон Кохонена для каждого класса. Если начальная плотность весовых векторов в окрестности обучающих векторов слишком мала, то, возможно, не удастся разделить сходные классы из-за того, что весовых векторов в интересующей нас окрестности не хватит, чтобы приписать по одному из них каждому классу входных векторов.

Наоборот, если несколько входных векторов получены незначительными изменениями из одного и того же образца и должны быть объединены в один класс, то они должны включать один и тот же нейрон Кохонена. Если же плотность весовых векторов очень высока вблизи группы слегка различных входных векторов, то каждый входной вектор может активировать отдельный нейрон Кохонена. Это не является катастрофой, так как слой Гроссберга может отобразить различные нейроны Кохонена в один и тот же выход, но это расточительная трата нейронов Кохонена.

Наиболее желательное решение будет таким: распределить весовые векторы в соответствии с плотностью входных векторов, подлежащих разделению, и для этого поместить больше весовых векторов в *окрестности* большого числа входных векторов. Конечно, на практике это невыполнимо, но существует несколько методов приближенного достижения тех же целей.

Одно из решений, известное под названием ***метода выпуклой комбинации*** (*convex combination method*), состоит в том, что все веса приравниваются к одной и той же величине

$$w_i = \frac{1}{\sqrt{n}}$$

где n – число входов и, следовательно, число компонент каждого весового вектора. Благодаря этому все весовые векторы совпадают и имеют единичную длину. Каждой же компоненте входа X придается значение

$$x_i = ax_i + \frac{1-a}{\sqrt{n}},$$

где n – число входов. В начале α очень мало, вследствие чего все входные векторы имеют длину, близкую к $1/\sqrt{n}$, и почти совпадают с векторами весов. В процессе обучения сети α постепенно возрастает, приближаясь к единице. Это позволяет разделять входные векторы и окончательно приписывать им их

истинные значения. Весовые векторы отслеживают один или небольшую группу входных векторов и в конце обучения дают требуемую картину выходов.

Обучение слоя Гроссберга

Слой Гроссберга обучается относительно просто. Входной вектор, являющийся выходом слоя Кохонена, подается на слой нейронов Гроссберга, и выходы слоя Гроссберга вычисляются как при нормальном функционировании. Далее, каждый вес корректируется только в том случае, если он соединен с нейроном Кохонена, имеющим ненулевой выход. Величина коррекции веса пропорциональна разности между весом и требуемым выходом нейрона Гроссберга, с которым этот вес соединен. В символьной записи

$$v_{ijн} = v_{ijc} + \beta(y_j - v_{ijc})k_i,$$

где k_i — выход -го нейрона Кохонена (только для одного нейрона Кохонена он отличен от нуля);y_j — j-я компонента вектора желаемых выходов.

Первоначально βберется равным приблизительно 0,1 и затем постепенно уменьшается в процессе обучения. Отсюда видно, что веса слоя Гроссберга будут сходиться к средним величинам от желаемых выходов, тогда как веса слоя Кохонена обучаются на средних значениях входов. Обучение слоя Гроссберга — это обучение с учителем, алгоритм располагает желаемым выходом, по которому он обучается. Обучающийся без учителя, самоорганизующийся слой Кохонена дает выходы в недетерминированных позициях. Они отображаются в желаемые выходы слоем Гроссберга.

Сеть встречного распространения полностью

На рис.7 показана структура полной сети встречного распространения. В режиме нормального функционирования предъявляются входные векторы X_1 и X_2, и обученная сеть дает на выходе векторы Y_1 и Y_2, являющиеся аппроксимациями X_1 и X_2 соответственно.

Предполагается, что векторы X_1 и X_2 являются нормализованными единичными векторами, следовательно, выходные векторы также будут нормализованными. В процессе обучения векторы X_1 и X_2 подаются одновременно и как входные векторы сети, и как эталонные выходные сигналы. Вектор X_1 используется для обучения выходов Y_1, а вектор X_2 – для обучения выходов Y_2 слоя Гроссберга. Гибридная сеть обучается с использованием того же самого метода, который описывался для сети прямого действия. Нейроны Кохонена принимают входные сигналы, как от вектора X_1, так и от вектора X_2 , что аналогично ситуации, когда имеется один общий вектор, составленный из

векторов X_1 и X_2. В качестве результата получается единичное отображение, при котором предъявление пары входных векторов порождает их копии на выходе.

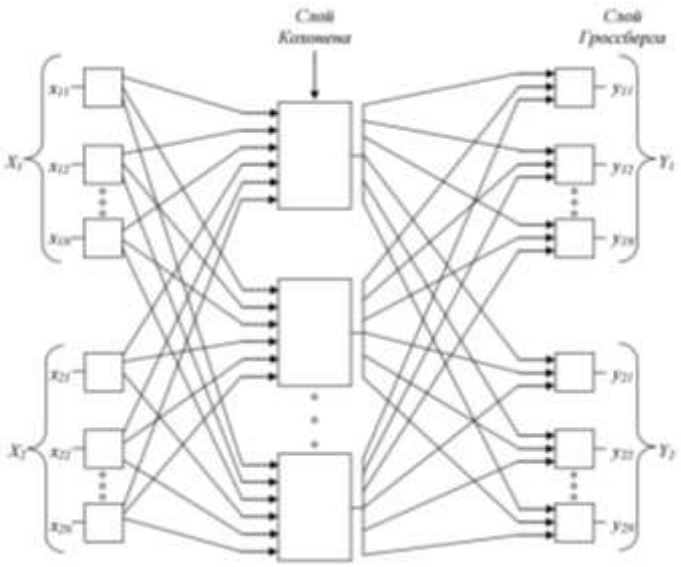

Рис.7. Структура полной сети встречного распространения.

Это не представляется особенно интересным, если не заметить, что предъявление только вектора X_1 порождает как выходы Y_1, так и выходы Y_2. Если F – функция, отображающая X_1 в Y_2 , то сеть аппроксимирует ее. Также, если F обратима, то предъявление только вектора X_2 порождает Y_1. Уникальная способность порождать функцию и обратную к ней делает сеть встречного распространения полезной в ряде приложений. Сеть встречного распространения быстро обучается и при правильном использовании может сэкономить значительное количество машинного времени. Она полезна также для быстрого моделирования систем, где большая точность встречного распространения вынуждает отдать ему предпочтение в окончательном варианте, но важна быстрая начальная аппроксимация. Возможность порождать функцию и обратную к ней также нашло применение в ряде систем.

Приложение: сжатие данных.

В дополнение к обычным функциям отображения векторов, встречное распространение оказывается полезным и в некоторых менее очевидных прикладных областях. Одним из наиболее интересных примеров является сжатие данных (рис.8).

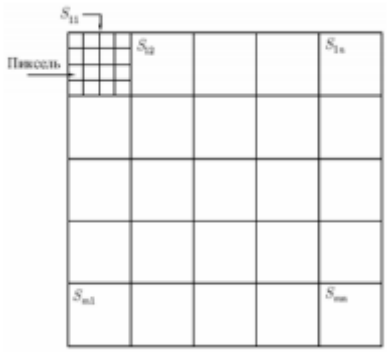

Рис.8. Сжатие данных

Сеть встречного распространения может быть использована для сжатия данных перед их передачей, уменьшая тем самым число битов, которые должны быть переданы.

Допустим, что требуется передать некоторое изображение. Оно может быть разбито на подизображения S, как показано на рис. Каждое подизображение разбито на пиксели (мельчайшие элементы изображения). Тогда каждое подизображение является вектором, элементами которого являются пиксели, из которых состоит подизображение. Допустим для простоты, что каждый пиксель – это единица (свет) или нуль (чернота). Если в подизображении имеется пикселей n, то для его передачи потребуется n бит. Если допустимы некоторые искажения, то для передачи типичного изображения требуется существенно меньшее число битов, что позволяет передавать изображение быстрее. Это возможно из-за статистического распределения векторов подизображений. Некоторые из них встречаются часто, тогда как другие встречаются так редко, что могут быть грубо аппроксимированы. Метод, называемый векторным квантованием, находит более короткие последовательности битов, наилучшим образом представляющие эти подизображения. Сеть встречного распространения может быть использована для выполнения векторного квантования. Множество векторов подизображений используется в качестве входа для обучения слоя Кохонена по методу аккредитации, когда выход единственного нейрона равен 1. Веса слоя Гроссберга обучаются выдавать бинарный код номера того нейрона Кохонена, выход которого равен 1. Например, если выходной сигнал нейрона 7 равен 1 (а все остальные равны 0), то слой Гроссберга будет обучаться выдавать 00...000111 (двоичный код числа 7). Это и будет являться более короткой битовой последовательностью передаваемых символов. На приемном конце идентичным образом обученная сеть встречного распространения принимает двоичный код и реализует обратную функцию, аппроксимирующую первоначальное

подизображение. Этот метод применялся на практике как к речи, так и к изображениям, с коэффициентом сжатия данных от 10:1 до 100:1. Качество было приемлемым, хотя некоторые искажения данных на приемном конце признаются неизбежным

Гибридная сеть Кохонена

Главной особенностью сетей с самоорганизацией на основе конкуренции является очень высокая скорость обучения, много больше, чем у сетей, обучающихся с учителем. Так как сеть с самоорганизацией выполняет обработку только входного вектора x, то основным недостатком такой сети является отсутствие аппроксимирующих свойств, которыми обладают многослойный персептрон и радиально-базисные сети. Очень хороший результат достигается при объединении самоорганизующегося слоя Кохонена и многослойного персептрона. Подобная структура, называемая гибридной сетью Кохонена, приведена на рисунке 9.

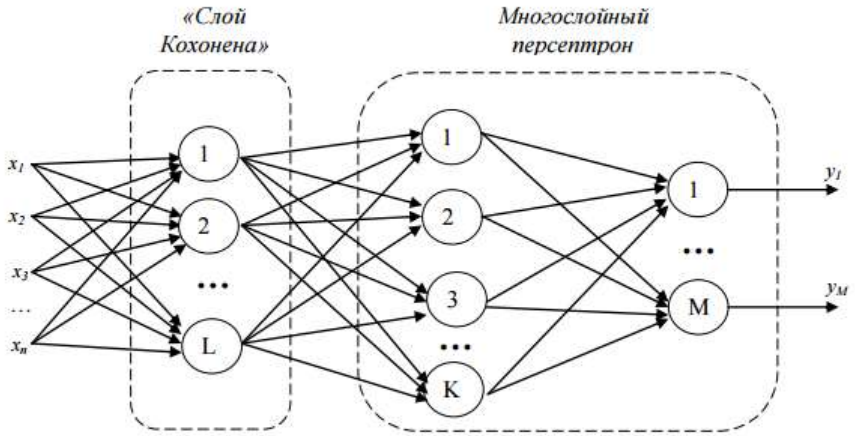

Рис.9. Гибридная сеть.

Слой Кохонена локализует значимые признаки процесса на основе анализа входного вектора x, после чего формируется входной вектор персептронного слоя. Вследствие хорошей локализации признаков первым слоем, обычно достаточно использовать персептронную сеть с одним скрытым слоем нейронов (зачастую линейных).

Обучение гибридной сети состоит из двух последовательных этапов. Сначала обучается слой Кохонена. В результате обучения векторы весов нейронов данного слоя с минимальной погрешностью отображают распределение данных обучающих векторов x. Обучение слоя Кохонена проводится по рассмотренным алгоритмам.

После завершения обучения, значения весов нейронов слоя Кохонена замораживаются, и производится анализ выходных сигналов. Выход нейрона-победителя переводится в состояние 1, а выходы остальных нейронов

переводятся в состояния из интервала (0-1). Переход от фактических выходных сигналов к нормализованным сигналам может производиться по различным формулам. Хорошие результаты можно получить, используя формулу

$$y_r = \exp\left(-\frac{|u_r - u_{\max}|^2}{\sigma^2}\right),$$

где u_{r} – реальный выход -го нейрона слоя Кохонена, y_{r} - нормализованный выход r-го нейрона слоя Кохонена, $u_{\max}$ – выход нейрона-победителя, σ - параметр, подбираемый для конкретной задачи.

Затем обучается персептронная сеть по одному из алгоритмов обучения с учителем. Обучающими сигналами является множество пар ($\mathrm{y_t}$, d_{t}), , где $\mathrm{y_t}$ - нормализованный выходной вектор слоя Кохонена, а d_{t} - это вектор эталонных значений выходных нейронов персептрона, соответствующих входному вектору x_{t} , t =1,2,…, p.

Чаще всего используются градиентные алгоритмы и метод обратного распространения ошибки. Процесс обучения будет протекать гораздо быстрее, чем для обычного многослойного персептрона, благодаря хорошей локализации данных в слое Кохонена. Кроме того, при этом, как правило, достигается глобальный минимум функции погрешности. Следует отметить, что данная сеть считается обобщением сети встречного распространения, в отличии от которой в сети Кохонена допускается дробная активность нейронов от значения 1 для победителя и от 0 до 1 для остальных нейронов. Учёт активности многих нейронов позволяет лучше локализовать входной вектор x в многомерном пространстве и получить лучшее отображение данных нейронной сетью в целом.

Рекуррентные сети

Общие положения

Отдельную группу нейронных сетей составляют сети с обратной связью между различными слоями нейронов. Это так называемые рекуррентные сети. Их общая черта состоит в передаче сигналов с выходного либо скрытого слоя во входной слой. Главная особенность таких сетей – динамическая зависимость на каждом этапе функционирования. Изменение состояния одного нейрона отражается на всей сети вследствие обратной связи типа «один ко многим». В сети возникает переходный процесс, который завершается формированием нового устойчивого состояния, отличающегося в общем случае от предыдущего. Другой особенностью рекуррентных сетей является тот факт, что для них не подходит ни обучение с учителем, ни обучение без учителя. В таких сетях весовые коэффициенты синапсов рассчитываются только однажды перед

началом функционирования сети на основе информации об обрабатываемых данных, и все обучение сети сводится именно к этому расчету. С одной стороны, предъявление априорной информации можно расценивать, как помощь учителя, но с другой – сеть фактически просто запоминает образцы до того, как на ее вход поступают реальные данные, и не может изменять свое поведение, поэтому говорить о звене обратной связи с учителем не приходится. Из сетей с подобной логикой работы наиболее известны сеть Хопфилда и сеть Хемминга, которые обычно используются для организации ассоциативной памяти. Ассоциативная память играет роль системы, определяющей взаимную зависимость векторов. В случае, когда на взаимозависимость исследуются компоненты одного и того же вектора, говорят об ассоциативной памяти. Если же взаимозависимыми оказываются два различных вектора, можно говорить о памяти гетероассоциативного типа. Типичным представителем первого класса является сеть Хопфилда, а второго – сеть Хемминга. Главная задача ассоциативной памяти сводится к запоминанию входных обучающих выборок таким образом, чтобы при представлении новой выборки система могла сгенерировать ответ, – какая из запомненных ранее выборок наиболее близка к вновь поступившему образу. Наиболее часто в качестве меры близости отдельных векторов применяется мера Хемминга.

При использовании двоичных значений (1,0) расстояние Хемминга между двумя векторами $y=[y_1, y_2,\ldots,y_N]^T$ и $d=[d_1, d_2,\ldots,d_N]^T$ определяется в виде:

$$d_H(y,d)=\sum_{i=1}^{N}[d_i(1-y_i)+(1-d_i)y_i]. \qquad (1)$$

При биполярных (± 1) значениях элементов обоих векторов расстояние Хемминга рассчитывается по формуле:

$$d_H(y,d)=\frac{1}{2}\left[N-\sum_{i=1}^{N}y_i d_i\right]. \qquad (2)$$

Сети Хопфилда

Американским исследователем Хопфилдом в 80-х годах предложен специальный тип нейронных сетей. В отличие от сетей с прямыми связями (Feedforward Networks или FF - Nets), сети Хопфилда являются рекуррентными или сетями с обратными связями (Feedback Networks).

Сети Хопфилда обладают следующими свойствами:

1. Симметрия дуг: сети содержат n нейронов, соединенных друг с

другом. Каждая дуга (соединение) характеризуется весом w_{ij}.

2. Симметрия весов: вес соединения нейрона n_i с нейроном n_j равен весу обратного соединения:

$$w_{ij} = w_{ji} ; w_{ii} = 0 .$$

3. Бинарные входы: сеть Хопфилда обрабатывает бинарные входы {0,1} или {-1,1}. В литературе встречаются модели сетей как со значениями 0 и 1, так и –1, 1. Для структуры сети это безразлично. Однако формулы для распознавания образов (изображений) при использовании значений –1 и 1 для входов и выходов нейронов сети Хопфилда получаются нагляднее.

Определение. Бинарная сеть Хопфилда определяется симметричной матрицей с нулевыми диагональными элементами, вектором T порогов нейронов и знаковой функцией активации или выхода нейронов.

Каждый вектор X с компонентами –1 или 1, удовлетворяющий уравнению $X = f(WX - T)$, называется образом для сети Хопфилда.

Обобщенная структура сети Хопфилда представляется, как правило, в виде системы с непосредственной обратной связью выхода с входом (рис.1).

Характерная особенность такой системы состоит в том, что выходные сигналы нейронов являются одновременно входными сигналами сети:

$$x_i(t) = y_i(t-1)$$

В классической системе Хопфилда отсутствует связь нейрона с собственным выходом, что соответствует $w_{ij} = 0$, а матрица весов является симметричной $W = W^T$.

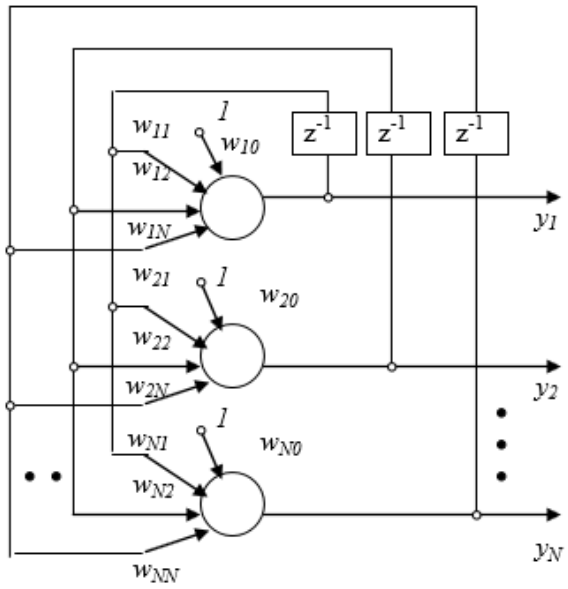

Рис.1. Обобщенная структура сети Хопфилда

Сеть Хопфилда состоит из единственного слоя нейронов, число которых является одновременно числом входов и выходов сети. Поэтому считается, что сеть не имеет входных элементов, а входной вектор задает первоначальную активность нейронов, которая затем изменяется в ходе итерационного процесса, обусловленного наличием обратных связей.

Каждый нейрон связан синапсами со всеми остальными нейронами, а также имеет один входной синапс, через который осуществляется ввод сигнала. В качестве функции активации нейронов сети Хопфилда используется знаковая функция,

$$s_j = \begin{cases} +1, & \text{если } net_j > 0, \\ -1, & \text{если } net_j < 0. \end{cases} \quad (3)$$

хотя для сетей Хопфилда также можно использовать пороговую функцию, линейную функцию с насыщением или сигмоидальные функции активации.

Зависимость 3 называется сигнум-функцией и в более краткой форме она записывается в виде

$$s_j = \mathrm{sgn}(net_j).$$

Это означает, что выходной сигнал i-го нейрона определяется функцией:

$$y_i = \mathrm{sgn}\left(\sum_{j=1}^{N} w_{ij} x_j + b_i\right), \quad (4)$$

где N обозначает число нейронов.

Будем считать, что пороговые элементы являются компонентами вектора x. Без учета единичных задержек сети, представляющих собой способ синхронизации процесса передачи сигналов, основные зависимости, определяющие сеть Хопфилда, можно представить в виде:

$$y_i(t) = \mathrm{sgn}\left(\sum_{j=0, i \neq j}^{N} w_{ij} y_j(t-1)\right), \quad (5)$$

с начальным условием $y_i(0) = x_j$. В процессе функционирования сети Хопфилда можно выделить два режима: обучения и классификации. В режиме обучения на основе известных обучающих выборок x подбираются весовые коэффициенты w_{ij} . В режиме классификации при зафиксированных значениях весов и вводе конкретного начального состояния нейронов $y(0) = x$ возникает переходной процесс, протекающий в соответствии с выражением (5) и завершающийся в одном из локальных минимумов, для которого $y(t) = y(t -$

1).

При вводе только одной обучающей выборки x процесс изменений продолжается до тех пор, пока зависимость (5) не начнет соблюдаться для всех N нейронов. Это условие автоматически выполняется в случае выбора значений весов, соответствующих отношению

$$w_{ij} = \frac{1}{N} x_i x_j, \tag{6}$$

поскольку только тогда $\frac{1}{N}\left(\sum_{j=1}^{N} x_i x_j x_j\right) = x_i$ (вследствие биполярных значений элементов вектора x всегда выполняется соотношение $x_j^2 = (\pm 1)^2 = 1$) Следует отметить, что зависимость (6) представляет собой правило обучения Хебба. При вводе большого числа обучающих выборок $x(t)$ для $t = 1,2,\ldots,p$ веса w_{ij} подбираются согласно обобщенному правилу Хебба, в соответствии с которым

$$w_{ij} = \frac{1}{N} \sum_{t=1}^{p} x_i^{(t)} x_j^{(t)} \tag{7}$$

Благодаря такому режиму обучения веса принимают значения, определяемые усреднением множества обучающих выборок.

В случае множества обучающих выборок становится актуальным фактор стабильности ассоциативной памяти. Для стабильного функционирования сети необходимо, чтобы реакция i-го нейрона y_i^l на l-ю обучающую выборку x^l совпадала с ее i-й составляющей x_i^l. Это означает, что с учетом выражения (7) получим

$$y_i^{(l)} = \operatorname{sgn}\left(\sum_{j=0}^{N} w_{ij} x_j^{(l)}\right) = \operatorname{sgn}\left(\frac{1}{N} \sum_{j=0}^{N} \sum_{t=1}^{p} x_i^{(t)} x_j^{(t)} x_j^{(l)}\right) = x_i^{(l)} \tag{8}$$

Если взвешенную сумму входных сигналов i-го нейрона обозначить u_i^l, то можно выделить в ней ожидаемое значение x_i^l и остаток, называемый диафонией:

$$u_i^{(l)} = x_i^{(l)} + \frac{1}{N} \sum_{j=0}^{N} \sum_{t \neq l} x_i^{(t)} x_j^{(t)} x_j^{(l)} \tag{9}$$

Вследствие применения знаковой функции активации, выполнение условия (8) возможно при малых значениях диафонии, не способных изменить знак x_i^l. Это означает, что, несмотря на определенное несовпадение битов,

переходный процесс завершается в нужной точке притяжения. При предоставлении тестовой выборки, отличающейся некоторым количеством битов, нейронная сеть может откорректировать эти биты и завершить процесс классификации в нужной точке притяжения. Тем не менее, правило Хебба обладает невысокой продуктивностью. Максимальная емкость ассоциативной памяти (число запомненных образцов) при обучении по правилу Хебба с допустимой погрешностью 1%, составляет примерно 14% от числа нейронов сети. Кроме того, при наличии шума, применение правила Хебба приводит к различным неточностям в виде локальных минимумов, далеких от исходного решения. Поэтому в качестве альтернативы используют методы обучения, основанные на псевдоинверсии. Идея этого метода состоит в том, что при правильно подобранных весах, каждая поданная на вход выборка x генерирует на выходе саму себя, мгновенно приводя к исходному состоянию (зависимость (5)). В матричной форме это можно представить в виде:

$$WX = X, \tag{10}$$

где W - матрица весов сети размерностью $N \times N$, а X - прямоугольная матрица размерностью N_p, составленная из p последовательных обучающих векторов $x^{(t)}$, то есть $X = \left[x^{(1)}, x^{(2)}, \ldots, x^{(p)}\right]$. Решение такой линейной системы уравнений имеет вид:

$$W = X\,X^{+} \tag{11}$$

где знак + обозначает псевдоинверсию. Если обучающие векторы линейно независимы, последнее выражение можно представить в форме:

$$W = X\left(X^{T}\,X\right)^{-1}X^{T} \tag{12}$$

Псевдоинверсия матрицы размерностью $N \times p$ в этом выражении заменена обычной инверсией квадратной матрицы $X^{\mathrm{T}}X$ размерностью $p \times p$. Дополнительное достоинство выражения (12) – возможность записать его в итерационной форме, не требующей расчета обратной матрицы. В этом случае выражение (12) принимает вид функциональной зависимости от последовательности обучающих векторов $x^{(t)}$ для $t = 1, 2, \ldots, p$.

$$W^{(t)} = W^{(t-1)} + \frac{1}{\left[x^{(t)}\right]^{T} x^{(t)} - \left[x^{(t)}\right]^{T} W^{(t-1)}\,x^{(t)}} \times \left[W^{(t-1)}\,x^{(t)} - x^{(t)}\right] \times \left[W^{(t-1)}\,x^{(t)} - x^{(t)}\right]^{T} \tag{13}$$

при начальных условиях $W^{(0)} = 0$. Такая форма предполагает однократное предъявление всех обучающих выборок, в результате чего матрица весов принимает значение $W = W^{p}$. Зависимости (12) и (13) называются методом проекций. Применение метода псевдоинверсии

увеличивает максимальную емкость сети Хопфилда, которая становится равной $N-1$.

Модифицированный вариант метода проекций – метод Δ-проекций – это градиентная форма алгоритма минимизации целевой функции. В соответствии с этим способом веса подбираются рекуррентно с помощью циклической процедуры, повторяемой для всех обучающих выборок:

$$W \leftarrow W + \frac{\eta}{N}\left[x^{(t)} - W x^{(t)}\right]\left[x^{(t)}\right]^T \tag{14}$$

Коэффициент η - это коэффициент обучения, выбираемый обычно из интервала [0.7 – 0.9]. Процесс обучения завершается, когда изменение вектора весов становится меньше априорно принятого значения ε. По завершении подбора весов сети их значения «замораживаются», и сеть можно использовать в режиме распознавания. В этой фазе на вход сети подается тестовый вектор x и рассчитывается ее отклик в виде:

$$y(t) = \mathrm{sgn}(W y(t-1)) \tag{15}$$

(в начальный момент $y(0) = x$, причем итерационный процесс повторяется для последовательных значений $y(t)$ вплоть до стабилизации отклика. В процессе распознавания образа по зашумленным сигналам, образующим начальное состояние нейронов, возникают проблемы с определением конечного состояния, соответствующего одному из запомненных образов. Возможны ошибочные решения. Одной из причин нахождения ошибочных решений является возможность перемешивания различных компонентов запомненных образов и формирования стабильного состояния, воспринимаемого как локальный минимум.

Для пояснения алгоритма формирования матрицы синаптических весов приведем пример. Допустим, что предметная область содержит два образа, закодированных с помощью двух векторов: [−1,1,−1] и [1,−1,1]. Перемножая их самих на себя и складывая, получим квадратную матрицу

$$\begin{bmatrix}-1\\1\\-1\end{bmatrix}\begin{bmatrix}-1&1&-1\end{bmatrix}+\begin{bmatrix}1\\-1\\1\end{bmatrix}\begin{bmatrix}1&-1&1\end{bmatrix}=\begin{bmatrix}1&-1&1\\-1&1&-1\\1&-1&1\end{bmatrix}+\begin{bmatrix}1&-1&1\\-1&1&-1\\1&-1&1\end{bmatrix}=\begin{bmatrix}2&-2&2\\-2&2&-2\\2&-2&2\end{bmatrix}.$$

Выполнив обнуление главной диагонали, окончательно получим:

$$w_{ij}=\begin{bmatrix}0&-2&2\\-2&0&-2\\2&-2&0\end{bmatrix}$$

Теперь предположим, что мы закодировали и ввели в сеть Хопфилда

матрицу синаптических весов, соответствующую трем образам, изображенным на рис. 2.

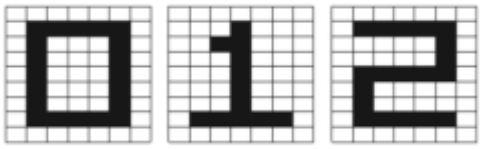

Рис. 2. Три образа, запомненные сетью Хопфилда

После этого мы предъявляем сети входной вектор (т.е. задаем первоначальную активность нейронов), соответствующий некоторому искаженному образу, изображенному на рис. 3 в левом верхнем углу.

Как видно из последующих кадров рис. 3, итерационный процесс привел к тому, что на выходе сети Хопфилда сформировался вектор, в точности соответствующий одному из ранее введенных образов. В этом случае говорят, что входной образ ассоциировался с одним из введенных ранее образов и что рекуррентные сети рассмотренного типа выступают в роли ассоциативных запоминающих устройств.

Хопфилду удалось математически строго показать, что при любом входном векторе итерационный процесс всегда приведет к распознаванию одного из введенных ранее образов, наиболее близкому к предъявленному. Однако максимальное количество запоминаемых сетью образов pmax ограничено формулой:

$p_{max} = \frac{N}{2lnN}$ где N — число нейронов сети Хопфилда.

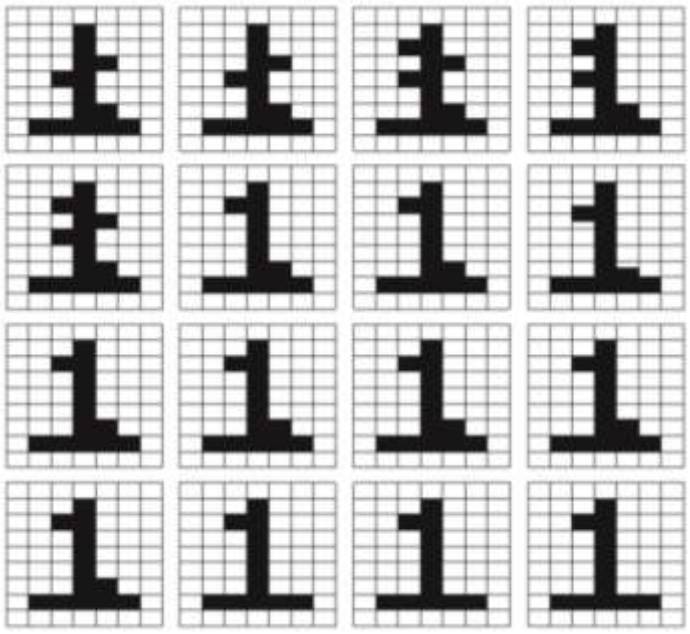

Рис. 3. Предъявленный нейронной сети Хопфилда искаженный образ и его деформация за последующие 15 итераций