Сети встречного распространения

Введение в 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. Будет показано, что, помимо преодоления ограничений других сетей, встречное распространение обладает собственными интересными и полезными свойствами.

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. 6.1](https://www.intuit.ru/studies/courses/88/88/lecture/20537?page=1#image.6.1) показана упрощенная версия прямого действия сети встречного распространения. Здесь иллюстрируются функциональные свойства этой парадигмы. Полная двунаправленная сеть основана на тех же принципах, она обсуждается в этой лекции позднее.

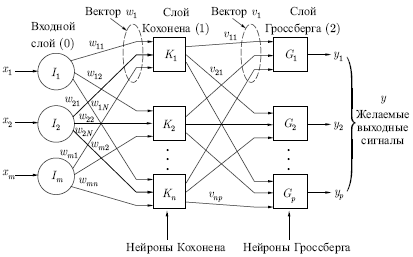


Рис. 6.1.

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_{mn}. 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_{np}. 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.

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

Слои Кохонена

В своей простейшей форме слой Кохонена функционирует в духе "победитель забирает все", т.е. для данного входного вектора один и только один нейрон Кохонена выдает на выходе логическую единицу, а все остальные выдают ноль. Нейроны Кохонена можно воспринимать как набор электрических лампочек, и для любого входного вектора "загорается" одна из них.

Ассоциированное с нейронами Кохонена множество весов связывает каждый нейрон с каждым входом. Например, на [рис. 6.1](https://www.intuit.ru/studies/courses/88/88/lecture/20537?page=1#image.6.1) нейрон Кохонена K_1 имеет веса w_{11}, w_{21}, \ldots, w_{m1}, составляющие весовой вектор W_1. Они соединяются через входной слой с входными сигналами x_1, x_2, \ldots, x_m, составляющими входной вектор \bf X. Подобно нейронам большинства сетей, выход NET каждого нейрона Кохонена является просто суммой взвешенных входов. Это может быть выражено следующим образом:

NET_j=\sum_i x_i w_{ij}

где NET_j — это выход NET нейрона Кохонена j, или, в векторной записи,

N=XW,

где N — вектор выходов NET слоя Кохонена.

Нейрон Кохонена с максимальным значением NET является "победителем". Его выход равен единице, у остальных он равен нулю.

Слой Гроссберга

Слой Гроссберга функционирует в сходной манере. Его выход NET является взвешенной суммой выходов k_1, k_2, \ldots, k_n слоя Кохонена, образующих вектор K. Вектор соединяющих весов, обозначенный через V, состоит из весов v_{11}, v_{12},
\ldots, v_{\text{np}}. Тогда выход NET каждого нейрона Гроссберга есть

NET_j=\sum_i k_i v_{ij},

где NET_j — выход j -го нейрона Гроссберга, или, в векторной форме,

Y = KV,

где Y — выходной вектор слоя Гроссберга, K — выходной вектор слоя Кохонена, V — матрица весов слоя Гроссберга.

Если слой Кохонена функционирует таким образом, что лишь у одного нейрона величина NET равна единице, а у остальных равна нулю, то всего один элемент вектора K отличен от нуля и 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}}. | ( 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. 6.2](https://www.intuit.ru/studies/courses/88/88/lecture/20537?page=3#image.6.2) такой двумерный вектор \bf 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 \bf V на пять дает вектор \bf V с компонентами 4/5 и 3/5, где \bf V' указывает в том же направлении, что и \bf V, но имеет единичную длину.

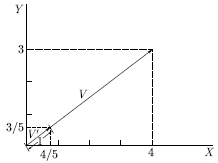


Рис. 6.2.

На [рис. 6.3](https://www.intuit.ru/studies/courses/88/88/lecture/20537?page=3#image.6.3) 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).

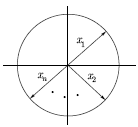


Рис. 6.3.

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. Уравнение, описывающее процесс обучения, имеет следующий вид:

w_{\text{н}}=w_c+\alpha(x-w_c),

где w_{\text{н}} — новое значение веса, соединяющего входную компоненту x с выигравшим нейроном; w_c — предыдущее значение этого веса; \alpha — 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. 6.4](https://www.intuit.ru/studies/courses/88/88/lecture/20537?page=3#image.6.4) этот процесс показан геометрически в двумерном виде. 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 \alpha, меньшую единицы, в результате чего получаем вектор изменения \delta. Окончательно новый весовой вектор W_{\text{н}} является отрезком, направленным из начала координат в конец вектора \delta. Отсюда можно видеть, что эффект обучения состоит во вращении весового вектора в направлении входного вектора без существенного изменения его длины.

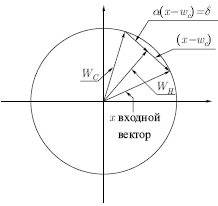


Рис. 6.4.

Переменная \alpha 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 \alpha 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 \bf X придается значение

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

где n — число входов. В начале \alpha очень мало, вследствие чего все входные векторы имеют длину, близкую к 1/\sqrt n, и почти совпадают с векторами весов. В процессе обучения сети \alpha 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.

Примеры обучения

Рассмотрим примеры обучения сети Кохонена обычным методом и методом выпуклой комбинации. В первом методе будем выбирать равномерно распределенные случайные векторы весов (ядер классов). На [рисунке 6.5](https://www.intuit.ru/studies/courses/88/88/lecture/20537?page=5#image.6.5) представлен пример обучения. Точками обозначены векторы x^p обучающего множества, кружками — векторы весовых коэффициентов.

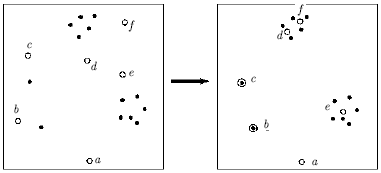


Рис. 6.5.

Вектор весов нейрона a не обучается, т.к. ни для одного из векторов обучающего множества этот нейрон не получает максимального выхода. Кроме того, в области из шести обучающих векторов (справа внизу) оказывается всего один вектор весов нейрона e, что не соответствует высокой плотности обучающих векторов в этой области. Эти недостатки присущи обычному методу обучения сети Кохонена.

Разберем работу метода выпуклой комбинации. Последовательное изменение картины векторов и весов показано на [рис. 6.6](https://www.intuit.ru/studies/courses/88/88/lecture/20537?page=5#image.6.6).

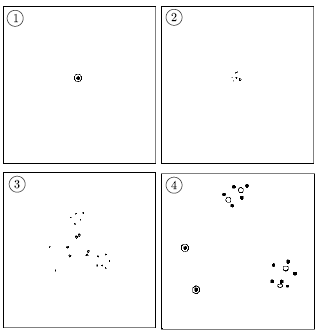


Рис. 6.6.

На первой схеме все векторы весов и обучающего множества имеют одно и то же значение. По мере обучения обучающие векторы расходятся к своим истинным значениям, а векторы весов следуют за ними. В итоге в сети не остается необученных нейронов и плотность векторов весов соответствует плотности векторов обучающего множества. Однако метод выпуклой комбинации хорошо работает, но замедляет процесс обучения, так как весовые векторы подстраиваются к изменяющейся цели. Другой подход состоит в добавлении шума к входным векторам. Тем самым они подвергаются случайным изменениям, схватывая в конце концов весовой вектор. Этот метод также работоспособен, но еще более медленнен, чем метод выпуклой комбинации.

Третий метод начинает работу со случайных весов, но на начальной стадии обучающего процесса подстраивает все веса, а не только связанные с выигравшим нейроном Кохонена. Тем самым весовые векторы перемещаются ближе к области входных векторов. В процессе обучения коррекция весов начинает производиться лишь для ближайших к победителю нейронов Кохонена. Этот радиус коррекции постепенно уменьшается, так что в конце корректируются только веса, связанные с выигравшим нейроном Кохонена.

Еще один метод наделяет каждый нейрон Кохонена "чувством справедливости". Если он становится победителем чаще своей "законной доли" (примерно 1/k, где k — число нейронов Кохонена ), он временно увеличивает свой порог, что уменьшает его шансы на выигрыш, давая тем самым возможность обучаться и другим нейронам.

Во многих приложениях точность результата существенно зависит от распределения весов. К сожалению, эффективность различных решений исчерпывающим образом не оценена и остается проблемой, ожидающей своего решения.

Модификации алгоритма обучения

Чувство справедливости: чтобы не допустить отсутствие обучения по любому из нейронов, вводится "чувство справедливости". Если нейрон чаще других выигрывает "состязание", т.е. получает максимальный выход чаще, чем в 1 из M случаев, то его значение выхода искусственно уменьшается, чтобы дать возможность выиграть другим нейронам. Это включает все нейроны сети в процесс обучения.

Коррекция весов пропорционально выходу: в этой модификации корректируются веса не только выигравшего нейрона, но и всех остальных, пропорционально их нормированному выходу. Нормировка выполняется по максимальному значению выхода слоя или по его среднему значению. Этот метод также исключает "мертвые" нейроны и улучшает распределение плотности весов.

Режим интерполяции

До сих пор мы обсуждали алгоритм обучения, в котором для каждого входного вектора активировался только один нейрон Кохонена. Это называется методом аккредитации. Его точность ограничена, так как выход полностью является функцией лишь одного нейрона Кохонена.

В методе интерполяции целая группа нейронов Кохонена, имеющих максимальные выходы, может передавать свои выходные сигналы в слой Гроссберга. Число нейронов в такой группе должно выбираться в зависимости от задачи, и убедительных данных относительно оптимального размера группы не имеется. Как только группа определена, ее множество выходов NET рассматривается как вектор, длина которого нормализуется на единицу делением каждого значения NET на корень квадратный из суммы квадратов значений NET в группе. Все нейроны вне группы имеют нулевые выходы.

Метод интерполяции способен устанавливать более сложные соответствия и может давать более точные результаты. По-прежнему, однако, нет убедительных данных, позволяющих сравнить достоинства и недостатки режимов интерполяции и аккредитации.

Статистические свойства обученной сети

Метод обучения Кохонена обладает полезной и интересной способностью извлекать статистические свойства из множества входных данных. Как показано Кохоненом, для полностью обученной сети вероятность того, что случайно выбранный входной вектор (в соответствии с функцией плотности вероятности входного множества) будет ближайшим к любому заданному весовому вектору, равна 1/k, где k — число нейронов Кохонена. Это является оптимальным распределением весов на гиперсфере. (Предполагается, что используются все весовые векторы, а это возможно лишь в том случае, если используется один из вышеупомянутых методов распределения весов.)

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_{ij\text{н}}=v_{ijc}+\beta(y_j-v_{ijc})k_i,

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

Первоначально \beta берется 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. 6.7](https://www.intuit.ru/studies/courses/88/88/lecture/20537?page=6#image.6.7) показана сеть встречного распространения целиком. В режиме нормального функционирования предъявляются входные векторы X и Y, и обученная сеть дает на выходе векторы X' и Y', являющиеся аппроксимациями соответственно для X и Y. Векторы X и Y предполагаются здесь нормализованными единичными векторами, следовательно, порождаемые на выходе векторы также будут иметь тенденцию быть нормализованными.

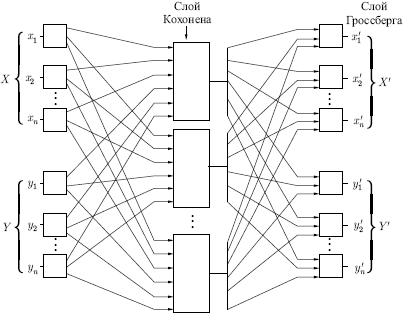


Рис. 6.7.

В процессе обучения векторы X и Y подаются одновременно и как входные векторы сети, и как желаемые выходные сигналы. Вектор X используется для обучения выходов X', а вектор Y — для обучения выходов Y' слоя Гроссберга. Сеть встречного распространения целиком обучается с использованием того же самого метода, который описывался для сети прямого действия. Нейроны Кохонена принимают входные сигналы как от векторов X, так и от векторов Y. Но эта ситуация неотличима от той, когда имеется один большой вектор, составленный из векторов X и Y, и тем самым не влияет на алгоритм обучения.

В качестве результирующего получается единичное отображение, при котором предъявление пары входных векторов порождает их копии на выходе. Этот вывод не представляется особенно интересным, если не заметить, что предъявление только вектора X (с вектором Y, равным нулю) порождает как выходы X', так и выходы Y'. Если F — функция, отображающая X в Y', то сеть аппроксимирует ее. Также, если F обратима, то предъявление только вектора Y (приравнивая X нулю) порождает X'. Уникальная способность сети встречного распространения — порождать функцию и обратную к ней — делает эту сеть полезной в ряде приложений.

[Рис. 6.7](https://www.intuit.ru/studies/courses/88/88/lecture/20537?page=6#image.6.7), в отличие от первоначальной конфигурации, не демонстрирует противоток в сети, по которому она получила свое название. Такая форма выбрана потому, что она также иллюстрирует сеть без обратных связей и позволяет обобщить понятия, развитые в предыдущих лекциях.

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.

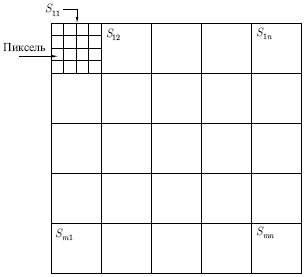


Рис. 6.8.

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. 6.8](https://www.intuit.ru/studies/courses/88/88/lecture/20537?page=7#image.6.8). 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 n пикселей, 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 неизбежными.

* Просмотр
* [Редактировать](https://moodle.bgpu.ru/mod/lesson/edit.php?id=71569)
* [Отчеты](https://moodle.bgpu.ru/mod/lesson/report.php?id=71569)
* [Оценить эссе](https://moodle.bgpu.ru/mod/lesson/essay.php?id=71569)

Сети встречного распространения

Введение в 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. Будет показано, что, помимо преодоления ограничений других сетей, встречное распространение обладает собственными интересными и полезными свойствами.

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. 6.1](https://www.intuit.ru/studies/courses/88/88/lecture/20537?page=1#image.6.1) показана упрощенная версия прямого действия сети встречного распространения. Здесь иллюстрируются функциональные свойства этой парадигмы. Полная двунаправленная сеть основана на тех же принципах, она обсуждается в этой лекции позднее.

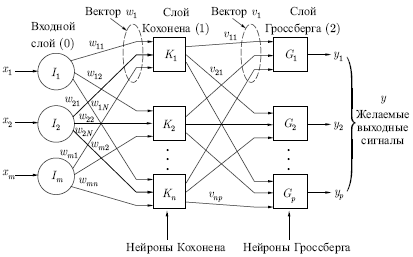


Рис. 6.1.

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_{mn}. 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_{np}. 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.

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

Слои Кохонена

В своей простейшей форме слой Кохонена функционирует в духе "победитель забирает все", т.е. для данного входного вектора один и только один нейрон Кохонена выдает на выходе логическую единицу, а все остальные выдают ноль. Нейроны Кохонена можно воспринимать как набор электрических лампочек, и для любого входного вектора "загорается" одна из них.

Ассоциированное с нейронами Кохонена множество весов связывает каждый нейрон с каждым входом. Например, на [рис. 6.1](https://www.intuit.ru/studies/courses/88/88/lecture/20537?page=1#image.6.1) нейрон Кохонена K_1 имеет веса w_{11}, w_{21}, \ldots, w_{m1}, составляющие весовой вектор W_1. Они соединяются через входной слой с входными сигналами x_1, x_2, \ldots, x_m, составляющими входной вектор \bf X. Подобно нейронам большинства сетей, выход NET каждого нейрона Кохонена является просто суммой взвешенных входов. Это может быть выражено следующим образом:

NET_j=\sum_i x_i w_{ij}

где NET_j — это выход NET нейрона Кохонена j, или, в векторной записи,

N=XW,

где N — вектор выходов NET слоя Кохонена.

Нейрон Кохонена с максимальным значением NET является "победителем". Его выход равен единице, у остальных он равен нулю.

Слой Гроссберга

Слой Гроссберга функционирует в сходной манере. Его выход NET является взвешенной суммой выходов k_1, k_2, \ldots, k_n слоя Кохонена, образующих вектор K. Вектор соединяющих весов, обозначенный через V, состоит из весов v_{11}, v_{12},
\ldots, v_{\text{np}}. Тогда выход NET каждого нейрона Гроссберга есть

NET_j=\sum_i k_i v_{ij},

где NET_j — выход j -го нейрона Гроссберга, или, в векторной форме,

Y = KV,

где Y — выходной вектор слоя Гроссберга, K — выходной вектор слоя Кохонена, V — матрица весов слоя Гроссберга.

Если слой Кохонена функционирует таким образом, что лишь у одного нейрона величина NET равна единице, а у остальных равна нулю, то всего один элемент вектора K отличен от нуля и 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}}. | ( 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. 6.2](https://www.intuit.ru/studies/courses/88/88/lecture/20537?page=3#image.6.2) такой двумерный вектор \bf 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 \bf V на пять дает вектор \bf V с компонентами 4/5 и 3/5, где \bf V' указывает в том же направлении, что и \bf V, но имеет единичную длину.

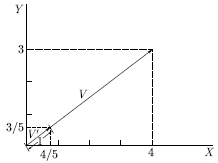


Рис. 6.2.

На [рис. 6.3](https://www.intuit.ru/studies/courses/88/88/lecture/20537?page=3#image.6.3) 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).

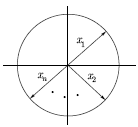


Рис. 6.3.

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. Уравнение, описывающее процесс обучения, имеет следующий вид:

w_{\text{н}}=w_c+\alpha(x-w_c),

где w_{\text{н}} — новое значение веса, соединяющего входную компоненту x с выигравшим нейроном; w_c — предыдущее значение этого веса; \alpha — 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. 6.4](https://www.intuit.ru/studies/courses/88/88/lecture/20537?page=3#image.6.4) этот процесс показан геометрически в двумерном виде. 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 \alpha, меньшую единицы, в результате чего получаем вектор изменения \delta. Окончательно новый весовой вектор W_{\text{н}} является отрезком, направленным из начала координат в конец вектора \delta. Отсюда можно видеть, что эффект обучения состоит во вращении весового вектора в направлении входного вектора без существенного изменения его длины.

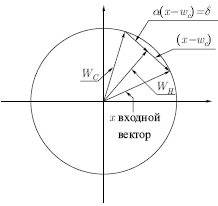


Рис. 6.4.

Переменная \alpha 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 \alpha 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 \bf X придается значение

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

где n — число входов. В начале \alpha очень мало, вследствие чего все входные векторы имеют длину, близкую к 1/\sqrt n, и почти совпадают с векторами весов. В процессе обучения сети \alpha 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.

Примеры обучения

Рассмотрим примеры обучения сети Кохонена обычным методом и методом выпуклой комбинации. В первом методе будем выбирать равномерно распределенные случайные векторы весов (ядер классов). На [рисунке 6.5](https://www.intuit.ru/studies/courses/88/88/lecture/20537?page=5#image.6.5) представлен пример обучения. Точками обозначены векторы x^p обучающего множества, кружками — векторы весовых коэффициентов.

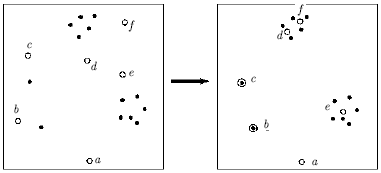


Рис. 6.5.

Вектор весов нейрона a не обучается, т.к. ни для одного из векторов обучающего множества этот нейрон не получает максимального выхода. Кроме того, в области из шести обучающих векторов (справа внизу) оказывается всего один вектор весов нейрона e, что не соответствует высокой плотности обучающих векторов в этой области. Эти недостатки присущи обычному методу обучения сети Кохонена.

Разберем работу метода выпуклой комбинации. Последовательное изменение картины векторов и весов показано на [рис. 6.6](https://www.intuit.ru/studies/courses/88/88/lecture/20537?page=5#image.6.6).

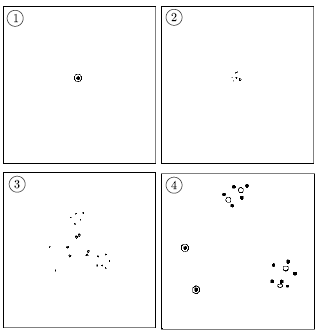


Рис. 6.6.

На первой схеме все векторы весов и обучающего множества имеют одно и то же значение. По мере обучения обучающие векторы расходятся к своим истинным значениям, а векторы весов следуют за ними. В итоге в сети не остается необученных нейронов и плотность векторов весов соответствует плотности векторов обучающего множества. Однако метод выпуклой комбинации хорошо работает, но замедляет процесс обучения, так как весовые векторы подстраиваются к изменяющейся цели. Другой подход состоит в добавлении шума к входным векторам. Тем самым они подвергаются случайным изменениям, схватывая в конце концов весовой вектор. Этот метод также работоспособен, но еще более медленнен, чем метод выпуклой комбинации.

Третий метод начинает работу со случайных весов, но на начальной стадии обучающего процесса подстраивает все веса, а не только связанные с выигравшим нейроном Кохонена. Тем самым весовые векторы перемещаются ближе к области входных векторов. В процессе обучения коррекция весов начинает производиться лишь для ближайших к победителю нейронов Кохонена. Этот радиус коррекции постепенно уменьшается, так что в конце корректируются только веса, связанные с выигравшим нейроном Кохонена.

Еще один метод наделяет каждый нейрон Кохонена "чувством справедливости". Если он становится победителем чаще своей "законной доли" (примерно 1/k, где k — число нейронов Кохонена ), он временно увеличивает свой порог, что уменьшает его шансы на выигрыш, давая тем самым возможность обучаться и другим нейронам.

Во многих приложениях точность результата существенно зависит от распределения весов. К сожалению, эффективность различных решений исчерпывающим образом не оценена и остается проблемой, ожидающей своего решения.

Модификации алгоритма обучения

Чувство справедливости: чтобы не допустить отсутствие обучения по любому из нейронов, вводится "чувство справедливости". Если нейрон чаще других выигрывает "состязание", т.е. получает максимальный выход чаще, чем в 1 из M случаев, то его значение выхода искусственно уменьшается, чтобы дать возможность выиграть другим нейронам. Это включает все нейроны сети в процесс обучения.

Коррекция весов пропорционально выходу: в этой модификации корректируются веса не только выигравшего нейрона, но и всех остальных, пропорционально их нормированному выходу. Нормировка выполняется по максимальному значению выхода слоя или по его среднему значению. Этот метод также исключает "мертвые" нейроны и улучшает распределение плотности весов.

Режим интерполяции

До сих пор мы обсуждали алгоритм обучения, в котором для каждого входного вектора активировался только один нейрон Кохонена. Это называется методом аккредитации. Его точность ограничена, так как выход полностью является функцией лишь одного нейрона Кохонена.

В методе интерполяции целая группа нейронов Кохонена, имеющих максимальные выходы, может передавать свои выходные сигналы в слой Гроссберга. Число нейронов в такой группе должно выбираться в зависимости от задачи, и убедительных данных относительно оптимального размера группы не имеется. Как только группа определена, ее множество выходов NET рассматривается как вектор, длина которого нормализуется на единицу делением каждого значения NET на корень квадратный из суммы квадратов значений NET в группе. Все нейроны вне группы имеют нулевые выходы.

Метод интерполяции способен устанавливать более сложные соответствия и может давать более точные результаты. По-прежнему, однако, нет убедительных данных, позволяющих сравнить достоинства и недостатки режимов интерполяции и аккредитации.

Статистические свойства обученной сети

Метод обучения Кохонена обладает полезной и интересной способностью извлекать статистические свойства из множества входных данных. Как показано Кохоненом, для полностью обученной сети вероятность того, что случайно выбранный входной вектор (в соответствии с функцией плотности вероятности входного множества) будет ближайшим к любому заданному весовому вектору, равна 1/k, где k — число нейронов Кохонена. Это является оптимальным распределением весов на гиперсфере. (Предполагается, что используются все весовые векторы, а это возможно лишь в том случае, если используется один из вышеупомянутых методов распределения весов.)

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_{ij\text{н}}=v_{ijc}+\beta(y_j-v_{ijc})k_i,

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

Первоначально \beta берется 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. 6.7](https://www.intuit.ru/studies/courses/88/88/lecture/20537?page=6#image.6.7) показана сеть встречного распространения целиком. В режиме нормального функционирования предъявляются входные векторы X и Y, и обученная сеть дает на выходе векторы X' и Y', являющиеся аппроксимациями соответственно для X и Y. Векторы X и Y предполагаются здесь нормализованными единичными векторами, следовательно, порождаемые на выходе векторы также будут иметь тенденцию быть нормализованными.

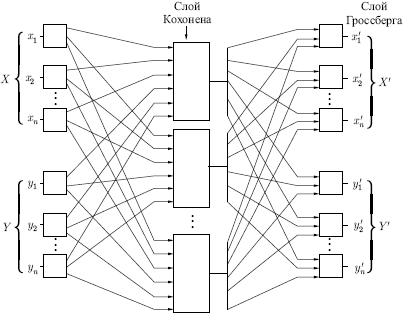


Рис. 6.7.

В процессе обучения векторы X и Y подаются одновременно и как входные векторы сети, и как желаемые выходные сигналы. Вектор X используется для обучения выходов X', а вектор Y — для обучения выходов Y' слоя Гроссберга. Сеть встречного распространения целиком обучается с использованием того же самого метода, который описывался для сети прямого действия. Нейроны Кохонена принимают входные сигналы как от векторов X, так и от векторов Y. Но эта ситуация неотличима от той, когда имеется один большой вектор, составленный из векторов X и Y, и тем самым не влияет на алгоритм обучения.

В качестве результирующего получается единичное отображение, при котором предъявление пары входных векторов порождает их копии на выходе. Этот вывод не представляется особенно интересным, если не заметить, что предъявление только вектора X (с вектором Y, равным нулю) порождает как выходы X', так и выходы Y'. Если F — функция, отображающая X в Y', то сеть аппроксимирует ее. Также, если F обратима, то предъявление только вектора Y (приравнивая X нулю) порождает X'. Уникальная способность сети встречного распространения — порождать функцию и обратную к ней — делает эту сеть полезной в ряде приложений.

[Рис. 6.7](https://www.intuit.ru/studies/courses/88/88/lecture/20537?page=6#image.6.7), в отличие от первоначальной конфигурации, не демонстрирует противоток в сети, по которому она получила свое название. Такая форма выбрана потому, что она также иллюстрирует сеть без обратных связей и позволяет обобщить понятия, развитые в предыдущих лекциях.

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.

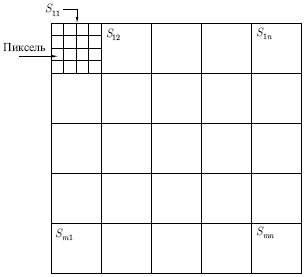


Рис. 6.8.

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. 6.8](https://www.intuit.ru/studies/courses/88/88/lecture/20537?page=7#image.6.8). 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 n пикселей, 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 неизбежными.