

5. ЛЕКЦИЯ. Нейронные сети Кохонена

Задачи кластеризации

Сети (слои) Кохонена относятся к самоорганизующимся нейронным сетям. Самоорганизующаяся сеть позволяет выявлять кластеры (группы) входных векторов, обладающих некоторыми общими свойствами.

Кластеризация — это разделение исследуемого множества объектов на группы "похожих" объектов, называемых кластерами. Задача кластеризации принципиально отличается от задачи классификации. Решением задачи классификации является отнесение каждого из объектов к одному из заранее определенных классов. В задаче кластеризации происходит отнесение объекта к одному из заранее неопределенных классов. Разбиение объектов по кластерам осуществляется при одновременном формировании кластеров.

Формально задача кластеризации описывается следующим образом. Дано множество объектов $I = \{i_1, i_2, \dots, i_n\}$, каждый из которых характеризуется вектором $x_j, j=1, 2, \dots, n$ атрибутов (параметров): $x_j = \{x_{j1}, x_{j2}, \dots, x_{jm}\}$. Требуется построить множество кластеров C и отображение F множества I на множество C , то есть $F : I \rightarrow C$. Задача кластеризации состоит в построении множества

$$C = \{c_1, c_2, \dots, c_k, \dots, c_g\}$$

где c_c — кластер, содержащий «похожие» объекты из множества I

$$c_k = \{i_j, i_p \mid i_j \in I, i_p \in I \text{ и } d(i_j, i_p) < \sigma\}, \quad (1)$$

σ — величина, определяющая меру близости для включения объектов в один кластер, $d(i_j, i_p)$ — мера близости между объектами, называемая расстоянием.

Если расстояние $d(i_j, i_p)$ меньше некоторого значения σ , то объекты считаются близкими и помещаются в один кластер. В противном случае считается, что объекты отличны друг от друга и их помещают в разные кластеры. Условие (1) известно, как *гипотеза компактности*.

Кластеризация основана на использовании расстояния между векторами. Неотрицательное число $d(x, y)$ называется расстоянием (метрикой) между векторами x и y , если выполняются следующие условия:

1. $d(x, y) \geq 0$ для всех x и y .
2. $d(x, y) = 0$, тогда и только тогда, когда $x = y$.
3. $d(x, y) = d(y, x)$.
4. $d(x, y) \leq d(x, k) + d(k, y)$ — неравенство треугольника.

Кластеризация позволяет сгруппировать сходные данные, что облегчает решение ряда задач Data Mining (интеллектуальный анализ данных):

- Изучение данных, облегчение анализа. Содержательный анализ полученных кластеров позволяет обнаружить закономерности. Например, можно выявить группы клиентов сети сотовой связи, для которых можно предложить новый тарифный план. Другие примеры — выявление групп покупателей торговой сети, сегментация рынка. Анализ содержания кластера позволяет применить к объектам различных кластеров разные методы анализа.

- Прогнозирование. Относя новый объект к одному из кластеров, можно прогнозировать поведение объекта, поскольку его поведение будет схожим с поведением объектов кластера.

- Обнаружение аномалий. Содержательный анализ кластеров помогает выявить аномалии. Обычно, это кластеры, в которые попадает мало объектов.

Структура сети Кохонена

Сеть Кохонена – это искусственная нейронная сеть, содержащая входной слой и слой активных нейронов. Поскольку входной слой просто распределяет входные сигналы по нейронам активного слоя, такую сеть можно считать однослойной.

Обычно активный слой одномерный или двумерный. Это отражает представления нейробиологии, в соответствии с которыми сенсорная информация, представленная множеством сигналов, отображается в линейные или планарные структуры коры головного мозга.

Различают два варианта сетей Кохонена:

1. Слой Кохонена. В нем нейроны активного слоя не упорядочены. В процессе обучения подстраиваются веса только одного нейрона-победителя.

2. Карта Кохонена (Self-Organized Map – SOM). В данном случае нейроны активного слоя образуют регулярную структуру. Такая карта применяется обычно для кластеризации графических изображений и звуковых сигналов

В процессе обучения SOM после предъявления достаточного числа образов все выходные нейроны сети разбиваются на подмножества, каждое из которых «откликается» на образы соответствующего класса.

Важное свойство SOM после обучения заключается в том, что чем более похожи объекты из обучающей выборки, тем ближе должны находиться нейроны активного слоя, ассоциированные с кластерами данных объектов. Расстояние между нейронами определяется топологией сети. На рис. 1 приведены варианты нейрона с восемью или шестью соседями.

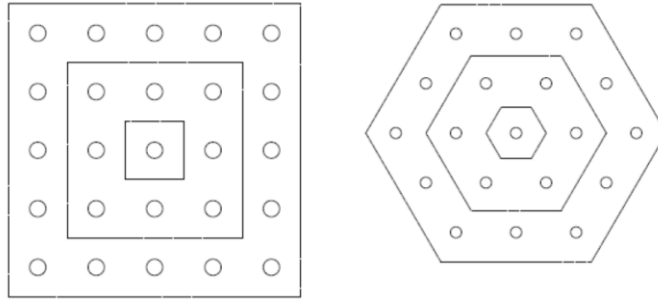


Рис. 1. Варианты топологии двумерной SOM

Таким образом, нейроны с номерами i и j характеризуются своими позициями I и J , с помощью которых можно описать функцию близости (neighbourhood function), которая может иметь вид гауссовой функции

$$g(i, j) = \exp\left(-\frac{\|I - J\|^2}{2\sigma^2}\right),$$

где σ – уменьшающаяся во времени константа

С помощью функции близости описывается область взаимодействия нейронов слоя Кохонена при обучении. На рис. 2 представлена одномерная сеть Кохонена из N нейронов.

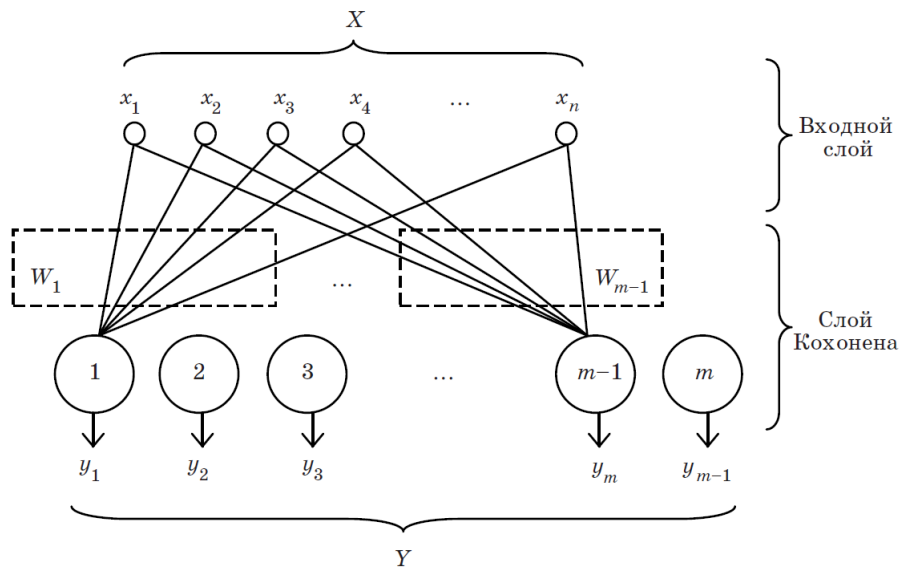


Рис. 2. Одномерная сеть Кохонена

С каждым нейроном Кохонена ассоциируется весовой вектор W , которому соответствует точка входного пространства.

Входной слой просто передает компоненты входного вектора X на каждый нейрон слоя Кохонена. Предполагается, что $N \gg m$.

Каждый i -й нейрон 2-го слоя имеет собственный вектор весов Wi , который сравнивается с входным вектором X . Сравнение предполагает вычисление

расстояния между X и W_i , так что в слое Кохонена появляется нейрон-победитель с номером j , веса которого имеют наименьшее расстояние до входного вектора:

$$j = \arg \min_i \|X - W_i\|.$$

В качестве метрики здесь может выступать евклидово расстояние

$$\|X - W\| = \sqrt{\sum_{i=1}^n (x_i - w_i)^2} \quad (1)$$

Если векторы X и W нормализованные, то в качестве меры близости можно использовать скалярное произведение, и тогда выход нейрона можно описать формулой

$$y_j = W_j X = \sum_{i=1}^n w_{ij} x_i, \quad (2)$$

и выход нейрона j оказывается максимальным при одинаковых X и W :

$$j = \arg \max_i \|W_i X\|.$$

Нормализация векторов выполняется по формулам

$$w'_i = \frac{w_i}{\sqrt{\sum_{j=1}^n w_j^2}}, \quad x'_i = \frac{x_i}{\sqrt{\sum_{j=1}^n x_j^2}}, \quad i = \overline{1, n}.$$

Таким образом, результатом работы слоя конкурирующих нейронов при подаче на входной слой некоторого вектора X является определение нейрона, который имеет наибольший выходной сигнал y_j (нейрон-победитель). Этот нейрон обладает весовым вектором W_j , который наиболее близок к входному вектору.

Нейроны слоя Кохонена работают не изолированно, между ними существуют соревновательные связи, с помощью которых близкие нейроны усиливают сигналы друг друга. Формула (2) дополняется вторым слагаемым:

$$y_j = \sum_{i=1}^n w_{ij} x_i + \sum_{l \neq j} g(j, l) y_l, \quad (3)$$

где $g(j, l)$ – сила связи между нейронами, которую часто называют функцией соседства.

Возможны разные варианты $g(j, l)$.

Простейший вариант функции соседства одномерного слоя

$$g(i, j) = \begin{cases} 1, & |i - j| \leq r, \\ 0, & |i - j| > r. \end{cases}$$

Гауссова функция соседства может быть описана в виде

$$g(i, j) = \exp\left(-\frac{R^2}{2\sigma^2}\right)$$

где σ – выбранная константа; R – расстояние между нейронами Кохонена, вычисляемое по формуле

$$R = \|r_i - r_j\|$$

Здесь r_i и r_j – координаты нейронов.

Помимо гауссовой функции можно использовать функцию, которая имеет вид «мексиканской шляпы» (вэйвлет — математическая функция, позволяющая анализировать различные частотные компоненты данных) (рис. 3).

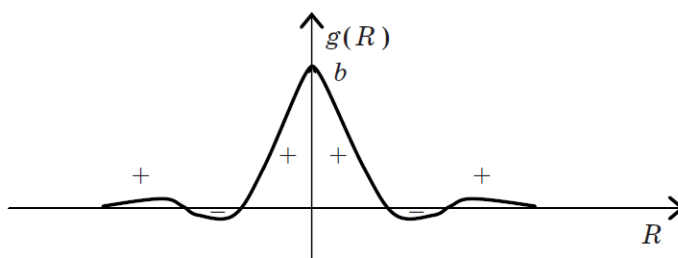


Рис. 3. Вариант описания взаимодействия нейронов

Как следует из рис. 3, вокруг точки возбуждения имеется небольшая окрестность, в которой нейроны переходят в активное состояние. За пределами данной окрестности происходит торможение активности нейронов. Наблюдается также слабая связь активности между нейронами с места возбуждения и нейронами с удаленных участков. На практике можно использовать упрощенный подход к описанию функции взаимодействия (рис. 4).

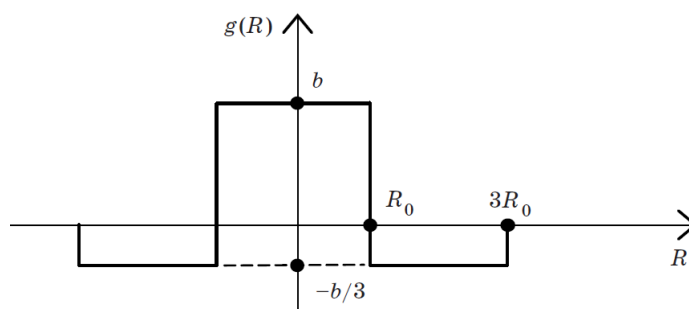


Рис. 4. Упрощенное представление силы связей слоя Кохонена

В соответствии с рис. 4 силу связи можно описать следующим образом:

$$g(R) = \begin{cases} b, & R \in [-R_0, R_0], \\ -\frac{b}{3}, & R \in [-R_0, -3R_0], \\ \frac{b}{3}, & R \in [R_0, 3R_0], \\ 0 & \text{в противном случае.} \end{cases}$$

При рассмотрении сети с двумерным слоем Кохонена функции взаимного влияния нейронов должны быть описаны для плоскости (рис. 5). Можно рассматривать также трехмерный слой Кохонена.

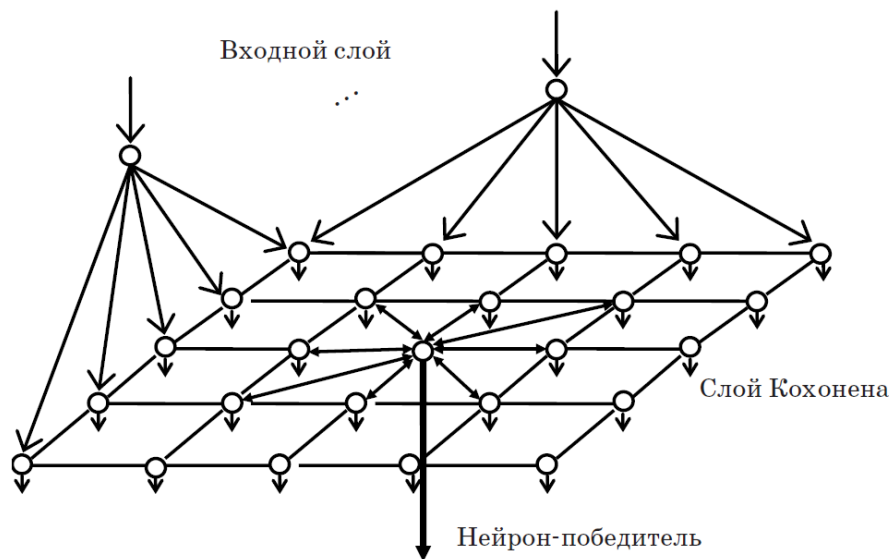


Рис. 5. ИНС с двумерным слоем Кохонена

Таким образом, сеть (слой) Кохонена – это однослойная сеть, построенная из нейронов типа WTA (Winner Takes All — победитель получает все).

Обучение сети Кохонена

Алгоритмы обучения сети Кохонена реализуют процесс обучения без учителя. При обучении n входных весов нейрона Кохонена рассматриваются как вектор в n -мерном пространстве. До обучения веса получают малые случайные значения. Затем каждый n -мерный вектор нормализуется в вектор единичной длины (с тем же направлением, что и исходный вектор).

При $n = 2$ все входные векторы можно изобразить на единичной окружности, при $n = 3$ – на единичной сфере, а для $n > 3$ следует рассматривать единичную гиперсферу.

Входные векторы обучающего набора также нормализуются, после чего сеть обучается по следующему алгоритму:

1. Подается входной вектор X .
2. Определяется нейрон-победитель с номером j по формуле (2).

3. Происходит подстройка весов нейронов.

а) если обучается слой конкурирующих нейронов, то обучается один нейрон:

$$W_i^T(t+1) = W_i^T(t) + \eta(X(t) - W_i^T(t));$$

б) если обучается SOM, то происходит подстройка весов нейронов Кохонена по формуле

$$W_i^T(t+1) = W_i^T(t) + \eta g(i,j)(X(t) - W_i^T(t)).$$

Здесь η – коэффициент скорости обучения.

4. Предъявляется новый входной вектор X , и шаги 1–4 повторяются.

Значение η должно постепенно уменьшаться.

В конце обучения влияние нейронов друг на друга может стать таким малым, чтобы обучался всего один нейрон, т. е. при подаче на вход сети некоторого входного вектора будет возбуждаться только один нейрон.

Каждый вес, связанный с выигравшим нейроном Кохонена, изменяется пропорционально разности между его величиной и величиной входа, к которому он присоединен. Для случая двух входов это можно представить геометрически (рис. 6).

Если бы с каждым нейроном Кохонена ассоциировался только один вектор, то слой Кохонена мог бы быть обучен путем простого присвоения весам значений этого входного вектора. Но поскольку обучение происходит по всем векторам, веса нейрона получают в конце концов усреднением по всем векторам, которые должны его активизировать.

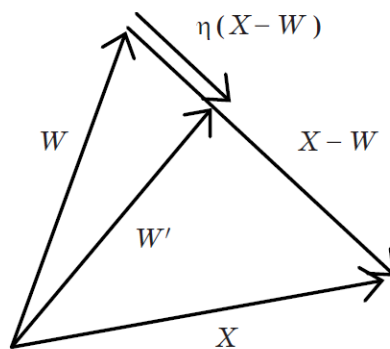


Рис. 6. Сближение вектора весов и входного вектора

Нормализованные входные и весовые векторы можно рассматривать как точки на поверхности единичной гиперсферы. Входные векторы группируются в классы (образы). Если переписать весовой вектор в область, соответствующую центру некоторого класса, то нейрон будет возбуждаться при появлении на входе любого вектора этого класса.

Входные векторы, которым обучают слой Кохонена, обычно распределены по поверхности гиперсферы не равномерно, а занимают локальные ее участки. Если до обучения выбирать веса случайно, можно получить ситуацию, при которой большинство весов будет настолько удалено от входных векторов, что многие нейроны Кохонена всегда будут иметь нулевой выход. Поэтому необходимо, чтобы в окрестности каждого входного вектора находились весовые векторы. Для того чтобы добиться этого, применяют *метод выпуклой комбинации*.

Все веса приравниваются к одной и той же величине $1/\sqrt{n}$ где n – размерность входного и весового векторов, т. е. все компоненты весовых векторов совпадают и сами они имеют единичную длину. Каждой компоненте вектора входа X придается значение

$$X_i = \alpha X_i + \left(\frac{1}{\sqrt{n}}\right)(1-\alpha)$$

Сначала α мало, и все входные компоненты близки к весовым компонентам. Потом α увеличивают, и входной вектор становится все более похож на самого себя. Весовые векторы отслеживают один вектор или небольшую их группу.

Кроме метода выпуклой комбинации существуют и другие методы. Например, к входным векторам можно добавлять шумы, чтобы каждый нейрон в конце концов захватил свой весовой вектор.

Еще один метод заставляет каждый нейрон Кохонена действовать «по справедливости», так, чтобы нейроны, которые возбуждаются чаще других, были подавлены, и могли обучиться все нейроны.

Пример

Рассмотрим простой пример обучения слоя конкурирующих нейронов. Пусть даны четыре входных вектора:

$$X_1 = \begin{bmatrix} 1 \\ 1 \\ 0 \\ 0 \end{bmatrix}, \quad X_2 = \begin{bmatrix} 0 \\ 0 \\ 0 \\ 1 \end{bmatrix}, \quad X_3 = \begin{bmatrix} 1 \\ 0 \\ 0 \\ 0 \end{bmatrix}, \quad X_4 = \begin{bmatrix} 0 \\ 0 \\ 1 \\ 1 \end{bmatrix}.$$

Они должны быть кластеризованы сетью Кохонена, имеющей два нейрона. При обучении изменяются веса только нейрона-победителя. Пусть веса нейронов получили некоторые случайные начальные значения (ненормализованные):

$$W_1^T = \begin{bmatrix} 0,2 \\ 0,6 \\ 0,5 \\ 0,9 \end{bmatrix}, \quad W_2^T = \begin{bmatrix} 0,8 \\ 0,4 \\ 0,7 \\ 0,3 \end{bmatrix}.$$

На первом шаге обучения подается входной вектор X_1 и вычисляется расстояние от него до весовых векторов (по формуле (1)):

$$d_1 = \|X_1 - W_1^T\| = \sqrt{\sum_{i=1}^4 (x_i - w_i)^2} \approx 1,36,$$

$$d_1 = \sqrt{(1 - 0,2)^2 + (1 - 0,6)^2 + (0 - 0,5)^2 + (0 - 0,9)^2} \approx 1,36$$

$$d_2 = \|X_1 - W_2^T\| = \sqrt{\sum_{i=1}^4 (x_i - w_i)^2} \approx 0,99.$$

$$d_2 = \sqrt{(1 - 0,8)^2 + (1 - 0,4)^2 + (0 - 0,7)^2 + (0 - 0,3)^2} \approx 0,99$$

Таким образом, победившим оказывается второй нейрон, и его веса подстраиваются ($\eta = 0,6$):

$$W_2^T = \begin{bmatrix} 0,8 \\ 0,4 \\ 0,7 \\ 0,3 \end{bmatrix} + 0,6 \cdot \begin{bmatrix} 1 \\ 1 \\ 0 \\ 0 \end{bmatrix} - \begin{bmatrix} 0,8 \\ 0,4 \\ 0,7 \\ 0,3 \end{bmatrix} = \begin{bmatrix} 0,92 \\ 0,76 \\ 0,28 \\ 0,12 \end{bmatrix}.$$

На втором шаге обучения подается входной вектор X_2 :

$$d_1 = \|X_2 - W_1^T\| = \sqrt{\sum_{i=1}^4 (x_i - w_i)^2} \approx 0,81,$$

$$d_2 = \|X_2 - W_2^T\| = \sqrt{\sum_{i=1}^4 (x_i - w_i)^2} \approx 1,5.$$

Победившим оказывается первый нейрон, и его веса подстраиваются:

$$W_1^T = \begin{bmatrix} 0,2 \\ 0,6 \\ 0,5 \\ 0,9 \end{bmatrix} + 0,6 \cdot \begin{bmatrix} 0 \\ 0 \\ 0 \\ 1 \end{bmatrix} - \begin{bmatrix} 0,2 \\ 0,6 \\ 0,5 \\ 0,9 \end{bmatrix} = \begin{bmatrix} 0,08 \\ 0,24 \\ 0,20 \\ 0,96 \end{bmatrix}.$$

Аналогично подаются векторы X_3 и X_4 , корректируются веса, и первая эпоха обучения заканчивается.

На второй и последующих эпохах значение η постепенно уменьшается, происходит приближение весов к предельным значениям:

$$W_1^T = \begin{bmatrix} 0 \\ 0 \\ 0,5 \\ 1 \end{bmatrix}, \quad W_2^T = \begin{bmatrix} 1 \\ 0,5 \\ 0 \\ 0 \end{bmatrix}.$$

Таким образом, центр первого кластера (W_1) оказывается усреднением компонент векторов X_2 и X_4 , а центр второго кластера (W_2) – усреднением компонент векторов X_1 и X_3 .

Контекстные карты

Существуют два фундаментально отличных метода визуализации самоорганизующихся карт признаков. Первый из этих методов состоит в построении гибкой сети, в которой векторы синоптических весов являются указателями, направленными от соответствующих нейронов во входное пространство. Этот метод визуализации особенно полезен для правильного отображения свойства топологического упорядочивания алгоритма SOM.

Во втором методе визуализации нейронам в двумерной решетке (представляющим выходной слой сети) назначаются метки классов, в зависимости от того, как каждый из примеров (не встречавшихся ранее) возбудил конкретный нейрон в самоорганизующейся сети. В результате этой второй ступени моделирования нейроны в двумерной решетке разбиваются на некоторое количество когерентных областей (coherent region). Здесь под когерентностью понимается то, что каждая из групп нейронов представляет обособленное множество непрерывных символов или меток. Это подразумевает, что при восстановлении хорошо упорядоченной карты использовались правильные условия.

Для примера рассмотрим множество данных, представленных в табл. 1. В этой таблице приведены характеристики отдельных животных. На основании данных каждого из столбцов таблицы можно составить описание некоторого животного. При этом значение 1 подразумевает наличие, а 0 - отсутствие одного из 13 свойств, перечисленных в таблице слева. Отдельные атрибуты (например, "2 ноги" и "4 ноги") Коррелируют, в то время как остальные нет. Для каждого из представленных животных можно составить характеристический код (attribute code) X_a , состоящий из 13 элементов. Само животное определяется символьным кодом (symbolic code) X_s , по которому никак нельзя собрать информацию о сходствах и различиях между отдельными животными. Например, в данном примере X_s представляет собой вектор, k -й элемент (соответствующий номеру

животного в списке) которого имеет некоторое значение a ; при этом все остальные элементы равны нулю. Параметр a определяет относительное влияние характеристического и символьного кодов друг на друга (рис.7).

<i>Животное</i>		<i>Г</i>	<i>К</i>	<i>У</i>	<i>Г</i>	<i>С</i>	<i>Я</i>	<i>О</i>	<i>Л</i>	<i>С</i>	<i>В</i>	<i>К</i>	<i>Т</i>	<i>Л</i>	<i>Л</i>	<i>З</i>	<i>К</i>
		<i>о</i>	<i>у</i>	<i>т</i>	<i>у</i>	<i>о</i>	<i>с</i>	<i>р</i>	<i>и</i>	<i>о</i>	<i>о</i>	<i>о</i>	<i>и</i>	<i>е</i>	<i>о</i>	<i>е</i>	<i>о</i>
		<i>л</i>	<i>р</i>	<i>к</i>	<i>с</i>	<i>в</i>	<i>т</i>	<i>е</i>	<i>с</i>	<i>б</i>	<i>л</i>	<i>ш</i>	<i>г</i>	<i>в</i>	<i>ш</i>	<i>б</i>	<i>р</i>
		<i>у</i>	<i>и</i>	<i>а</i>	<i>ь</i>	<i>а</i>	<i>р</i>	<i>л</i>	<i>а</i>	<i>а</i>	<i>к</i>	<i>к</i>	<i>р</i>		<i>а</i>	<i>р</i>	<i>о</i>
		<i>б</i>	<i>ц</i>				<i>е</i>			<i>к</i>	<i>а</i>				<i>д</i>	<i>а</i>	<i>в</i>
		<i>ь</i>	<i>а</i>				<i>б</i>			<i>а</i>					<i>ь</i>		<i>а</i>
<i>Размер</i>	<i>Маленький</i>	1	1	1	1	1	1	0	0	0	0	1	0	0	0	0	0
	<i>Средний</i>	0	0	0	0	0	0	1	1	1	1	0	0	0	0	0	0
	<i>Крупный</i>	0	0	0	0	0	0	0	0	0	0	0	1	1	1	1	1
<i>Имеет</i>	<i>2 ноги</i>	1	1	1	1	1	1	1	0	0	0	0	0	0	0	0	0
	<i>4 ноги</i>	0	0	0	0	0	0	0	1	1	1	1	1	1	1	1	1
	<i>Шерсть</i>	0	0	0	0	0	0	0	1	1	1	1	1	1	1	1	1
	<i>Копыта</i>	0	0	0	0	0	0	0	0	0	0	0	0	0	1	1	1
	<i>Гриву</i>	0	0	0	0	0	0	0	0	0	1	0	0	1	1	1	0
	<i>Перья</i>	1	1	1	1	1	1	1	0	0	0	0	0	0	0	0	0
<i>Любит</i>	<i>Охоту</i>	0	0	0	0	1	1	1	1	0	1	1	1	1	0	0	0
	<i>Бегать</i>	0	0	0	0	0	0	0	0	1	1	0	1	1	1	1	0
	<i>Летать</i>	1	0	0	1	1	1	1	0	0	0	0	0	0	0	0	0
	<i>Плавать</i>	0	0	1	1	0	0	0	0	0	0	0	0	0	0	0	0

Рис. 7. Пример контекстной карты

Для того чтобы гарантировать, что данный характеристический код является доминантным, а выбирается равным 0,2. Входной вектор для каждого из животных состоит из 29 элементов и представляет собой объединение характеристического X_a и символьного X_s кодов:

$$\mathbf{x} = \begin{bmatrix} \mathbf{x}_s \\ \mathbf{x}_a \end{bmatrix} = \begin{bmatrix} \mathbf{x}_s \\ \mathbf{0} \end{bmatrix} + \begin{bmatrix} \mathbf{0} \\ \mathbf{x}_a \end{bmatrix}$$

Наконец, каждый из векторов данных нормализуется к единичной длине. Примеры из таким образом созданного множества данных подаются на вход двумерной решетки нейронов размером 10×10 , при этом синаптические веса нейронов настраиваются в соответствии с алгоритмом SOM. Обучение продолжалось в течение 2000 итераций, после чего карта признаков должна была достичь стабильного состояния. После этого самоорганизующейся сети подавались на вход векторы $\mathbf{x} = [x_s, 0]^T$, содержащие только символьные коды одного из животных, и определялся нейрон с самым сильным откликом. Эта процедура повторялась для всех 16 животных.

В процессе выполнения описанного алгоритма была получена карта, показанная на рис. 8.

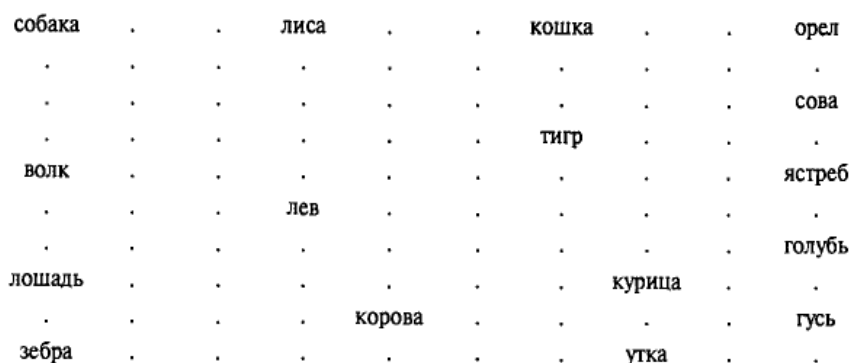


Рис. 8. Контекстная карта

В ней маркированные нейроны дали самый сильный отклик на соответствующие примеры. Точками на карте обозначены нейроны с более слабыми откликами.

На следующем рис. показан результат моделирования для той же самоорганизующейся сети (рис.9).

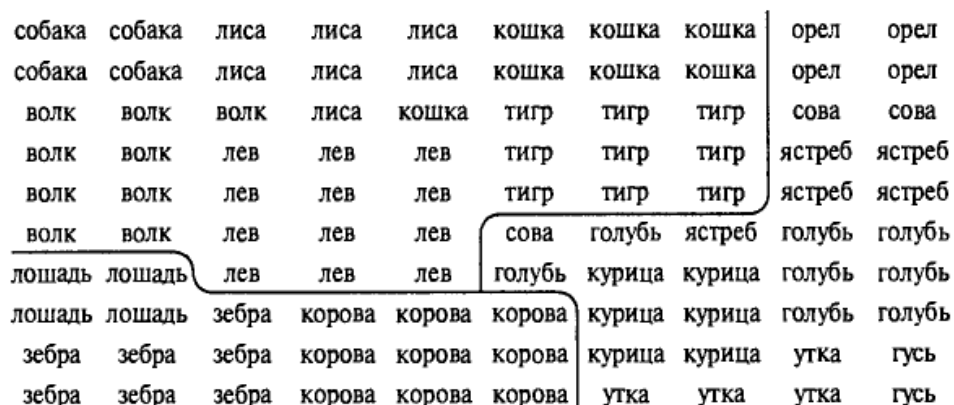


Рис. 9. Результат моделирования

Однако на этот раз каждый из нейронов сети быть маркирован названием того животного, для которого его отклик был самым сильным. На этом рисунке видно, что карта признаков четко отслеживает "родственные взаимосвязи" между 16 различными животными. На нем можно выделить три обособленных кластера, один из которых представляет птиц, второй мирных животных, а третий хищников. Карта признаков, показанная на последнем рис., называется контекстной (contextual) или семантической (semantic).

Контекстные карты, получившиеся в результате использования алгоритма SOM, нашли свое применение в решении задач создания фонетических классов для текста, исследования Земли или дистанционного зондирования, а также в исследовании данных и извлечении скрытых закономерностей (data exploration & data mining).

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, \dots, x_N , которым соответствуют веса $w_{i1}, w_{i2}, \dots, w_{iN}$, 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).

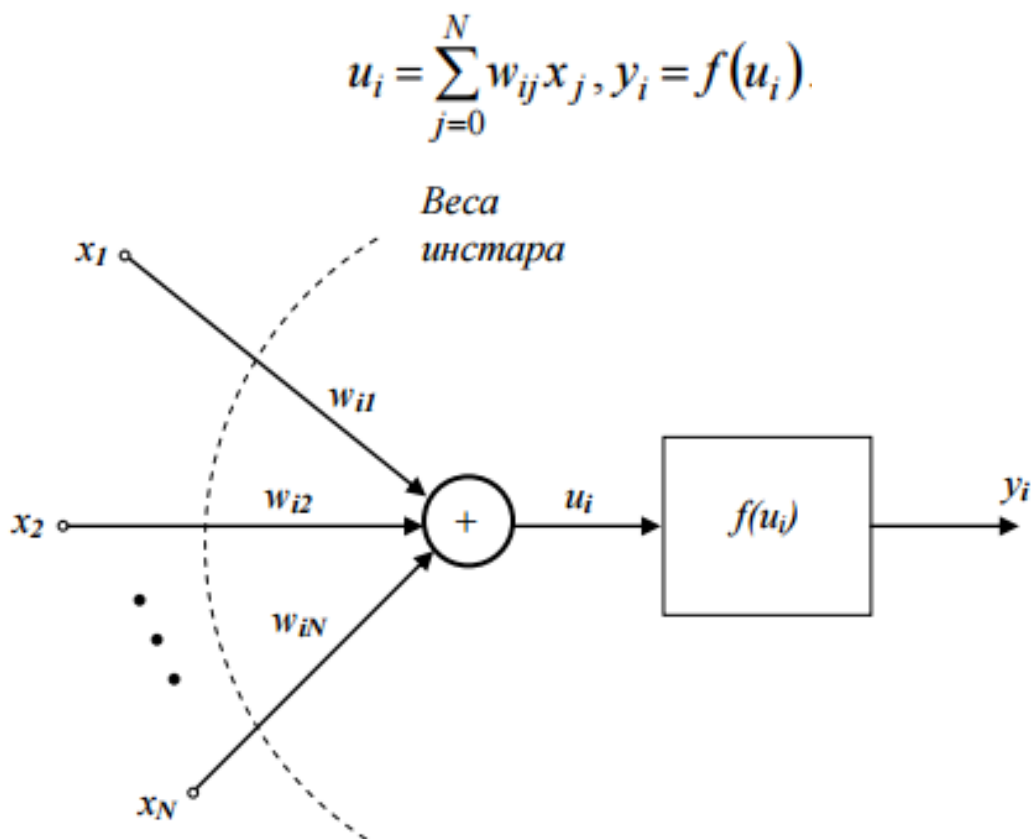


Рис. 1 Структурная схема входной звезды (инстара)

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

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

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 $\eta < 1$ инстар принимает усреднённое значение обучающих векторов. Входные данные должны быть нормированы, то есть $x = 1$. При этом

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

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.

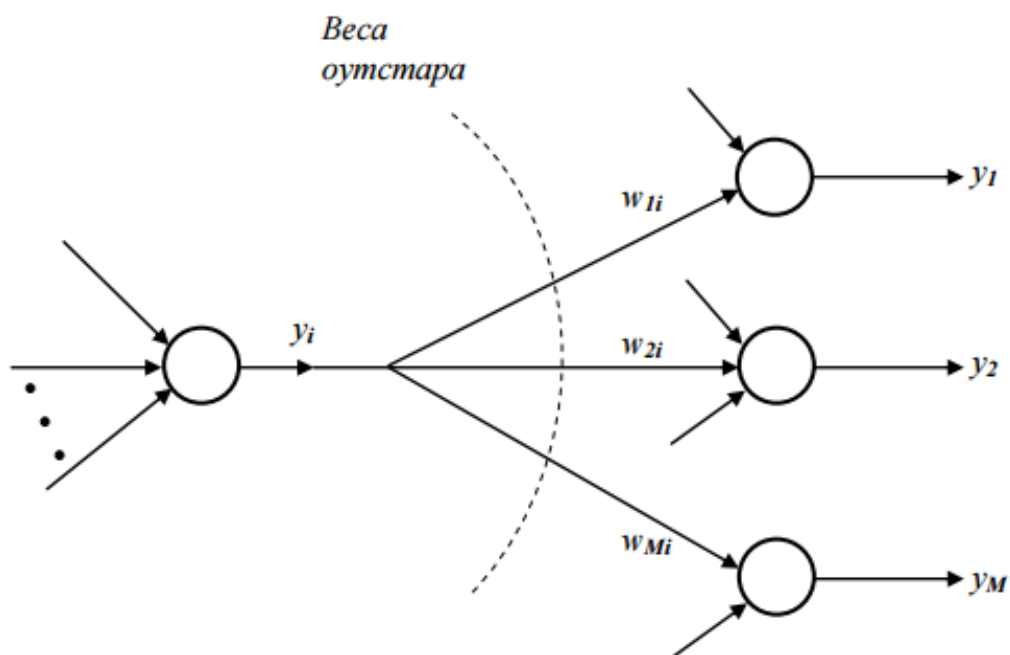


Рис. 2 Структурная схема выходной звезды (оутстара)

Нейрон этого типа имеет один вход и M выходов с весами $w_{1i}, w_{2i}, \dots, w_{Mi}$. i -ый нейрон-источник высылает свой выходной сигнал y_i взаимодействующим с ним нейронам, выходные сигналы которых обозначены y_j ($j = 1, 2, \dots, M$). 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 i -го нейрона, выступающего в роли источника. В режиме распознавания в момент активизации нейрона-источника оутстар будет генерировать сигналы, соответствующие ожидаемым значениям y_j .

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.

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.

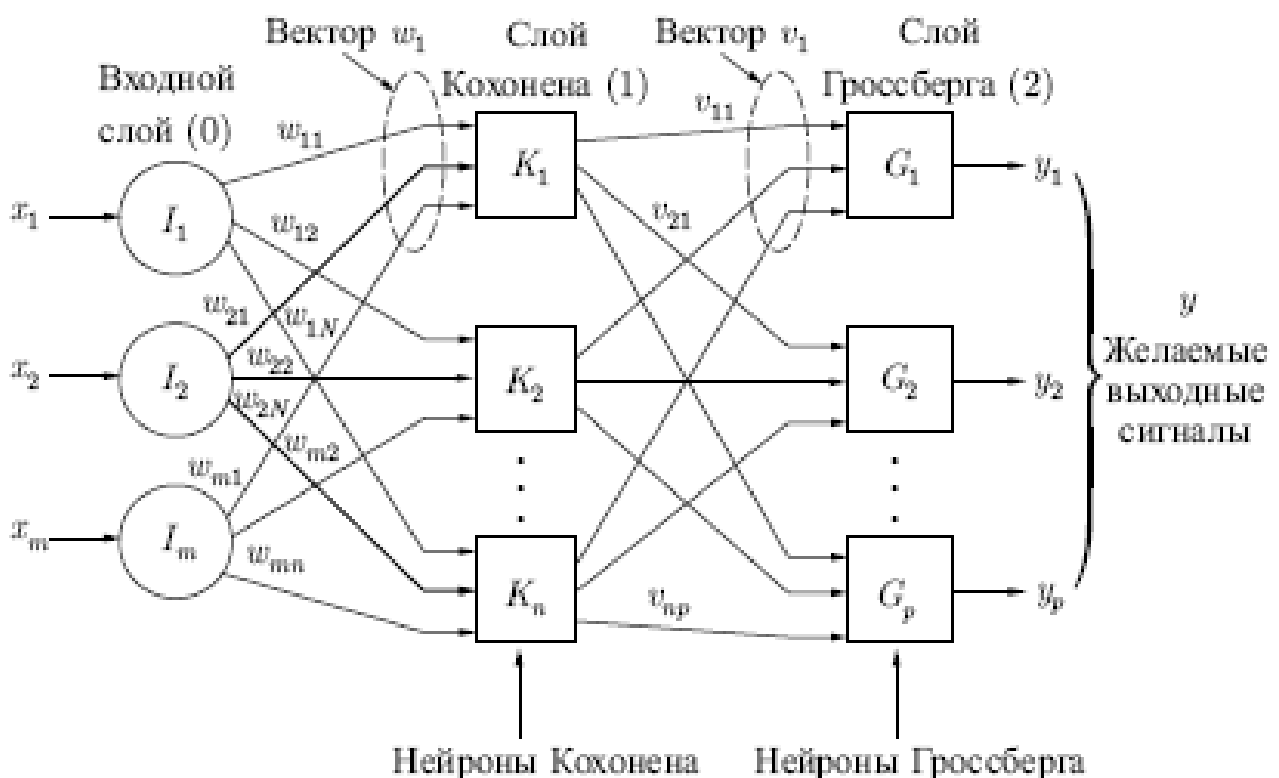


Рис. 3 Упрощенная версия прямого действия сети встречного распространения

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 $w_{пр}$. 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.

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_i каждого нейрона Кохонена является просто суммой взвешенных входов. Это может быть выражено следующей формулой:

$$u_i = w_{i1} \cdot x_1 + w_{i2} \cdot x_2 + \dots + w_{iN} \cdot x_N,$$

де u_i – это выход i -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, \dots, u_K , слоя Кохонена, образующих вектор u . Вектор соединяющих весов, обозначенный через v , состоит из весов $v_{11}, v_{21}, \dots, v_{MK}$. Тогда выход y каждого нейрона Гроссберга есть

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

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

$$y = u \cdot v$$

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.

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:

$$w_H = w_c + \alpha(x - w_c),$$

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.

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

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

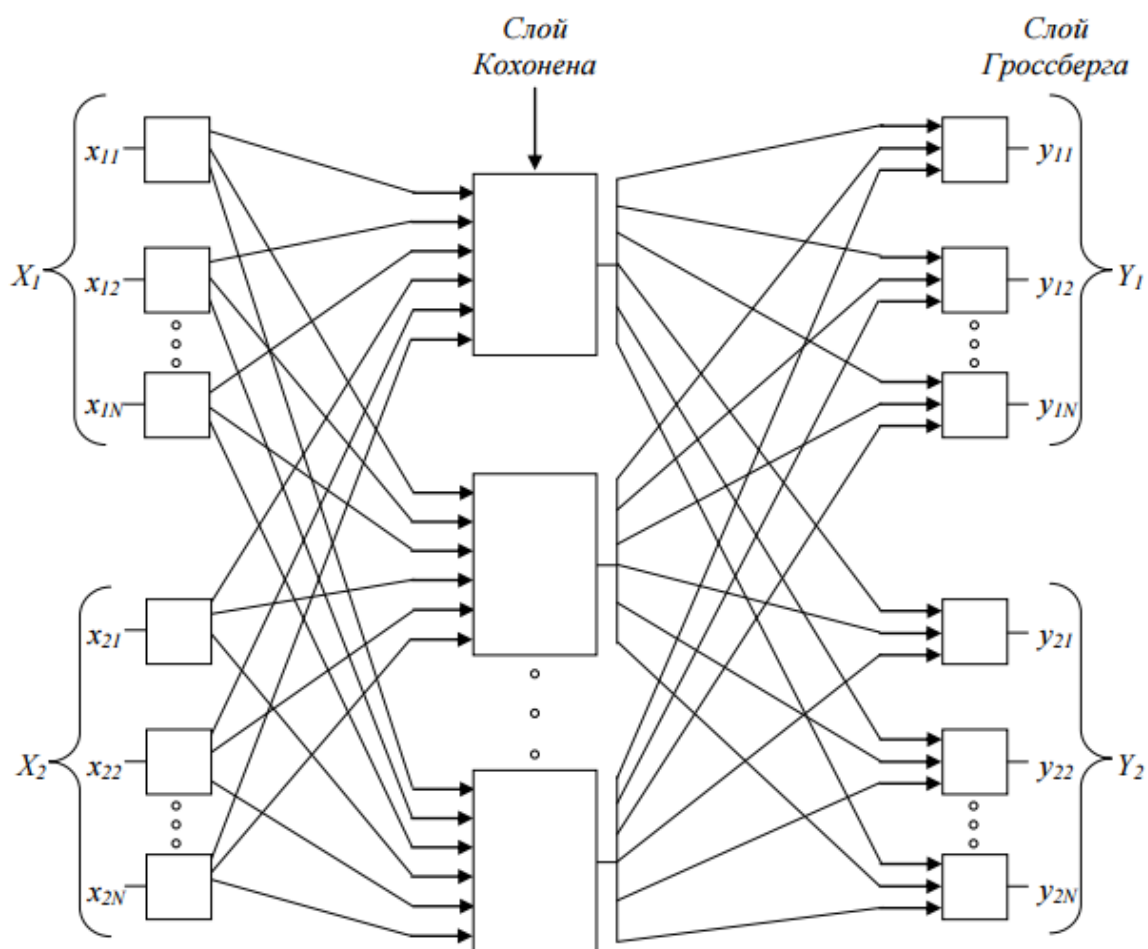
где k_i — выход i -го нейрона Кохонена (только для одного нейрона Кохонена он отличен от нуля); y_j — j -я компонента вектора желаемых выходов.

Первоначально β 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. 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.

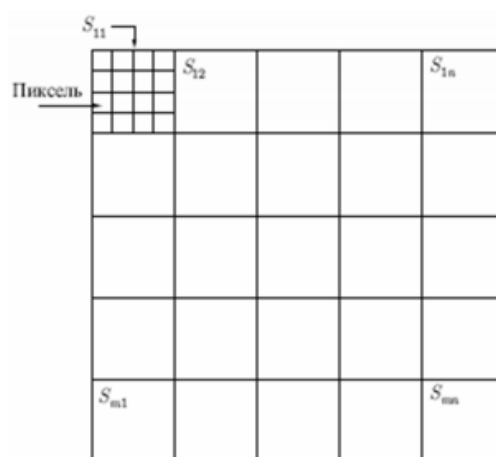


Предполагается, что векторы X_1 и X_2 являются нормализованными единичными векторами, следовательно, выходные векторы также будут нормализованными. В процессе обучения векторы X_1 и X_2 подаются одновременно и как входные векторы сети, и как эталонные выходные сигналы. Вектор X_1 используется для обучения 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. 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.



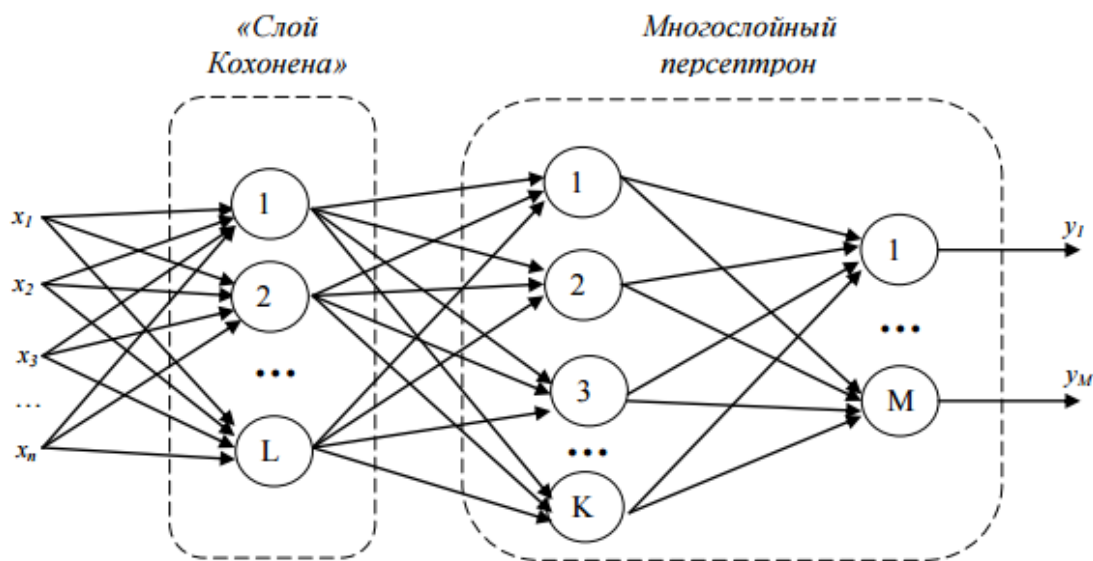
Сеть встречного распространения может быть использована для сжатия данных перед их передачей, уменьшая тем самым число битов, которые должны быть переданы.

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.



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 r -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 (y_t, d_t) , где y_t - нормализованный выходной вектор слоя Кохонена, а d_t - это вектор эталонных значений выходных нейронов персептрона, соответствующих входному вектору x_t , $t = 1, 2, \dots, p$.

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 отличие 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.