Data Science

Кластеризация (Сеть Кохонена)

MVP

DEDICATED TEAM

SUPPORT

Sloboda STUDIO since 2010

Принцип самообучения

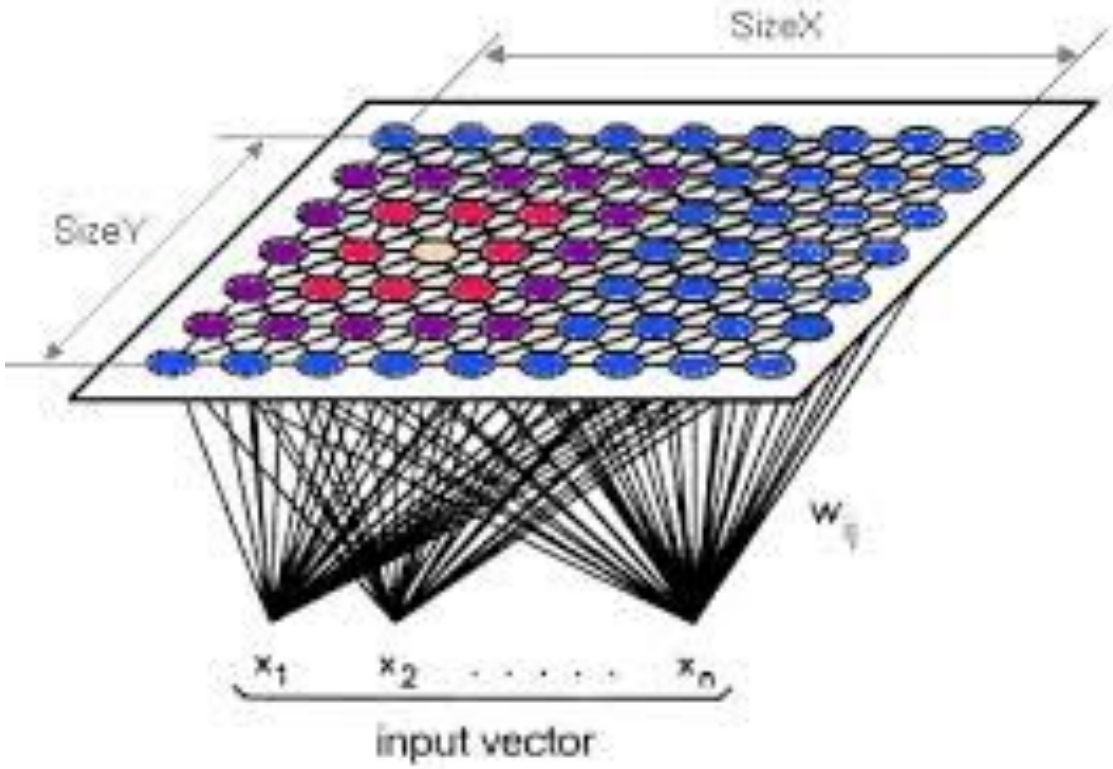
Самообучение сети заключается в подстройке весов синапсов на основании информации содержащейся в поданном входном векторе. Вид откликов на каждый класс входных событий заранее неизвестен и будет представлять собой произвольное сочетание состояний нейронов выходного слоя, обусловленное случайным распределением весов при инициализации сети.

Общая идея алгоритмов обучения заключается в том, что при самообучении путем коррекции весов усиливаются связи между возбужденными нейронами. Так происходит коррекция и закрепление образа, отвечающего конкретной части из всей группы рассматриваемых событий. В результате сеть способна обобщать схожие образы относя их к одному классу и этим выполнять сжатие информации.

Архитектура слоя Кохонена и алгоритм его настройки предполагает, что для каждого входного вектора будет активизирован лишь один нейрон-победитель. Для данного входного вектора только один нейрон Кохонена выдает логическую единицу, все остальные выдают ноль.

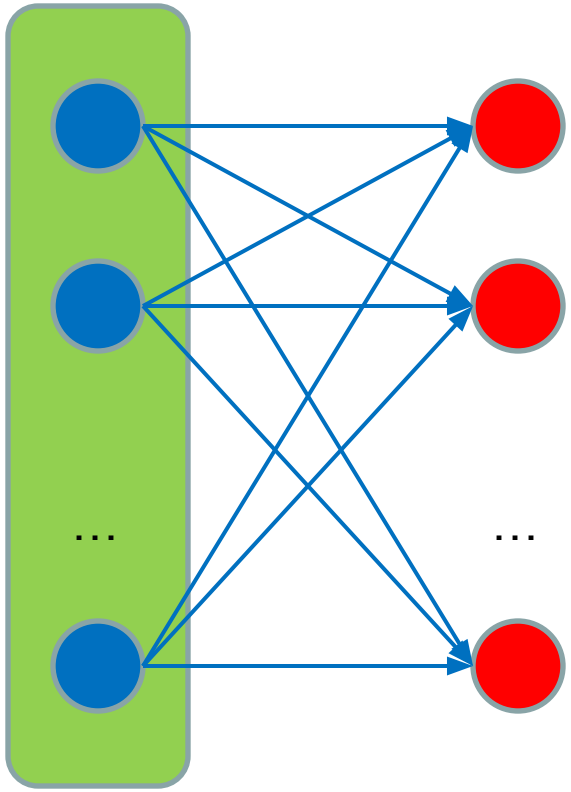
Слой Кохонена классифицирует входные векторы в группы схожих векторов. Это достигается с помощью такой подстройки весов, что близкие входные векторы активизируют один и тот же нейрон. В результате обучения слой получает способность разделять несхожие входные векторы.

Сеть Кохонена



Слой Кохонена

- классифицирует входные векторы в группы



Слой рецепторов (входной вектор)

Слой Кохонена (классификатор)

В простейшей форме реализуется модель «WTA» («Победитель забирает всё»):

Алгоритм Кохонена формирования карт признаков:

Шаг 1. Инициализация сети:
Весовым коэффициентам сети присваиваются малые случайные значения. Общее число синаптических весов - *M*N* Начальная зона соседства показана на рис.

Шаг 2. Предъявление сети нового входного сигнала.

Шаг 3. Вычисление расстояния до всех нейронов сети:

Расстояния d_j от входного сигнала до каждого нейрона *j* определяется по формуле:

$$d_j = \sum_{i=0}^{n-1} (x_i(t) - w_{ij}(t))^2$$

Где x_i - *i*-ый элемент входного сигнала в момент времени *t*,
$w_{ij}(t)$ - вес связи от *i*-го элемента входного сигнала к нейрону *j* в момент времени *t*.

Шаг 4. Выбор нейрона с наименьшим расстоянием:
Выбирается нейрон *j, для которого расстояние d_j наименьшее.**

Шаг 5. Настройка весов нейрона *j и его соседей:**

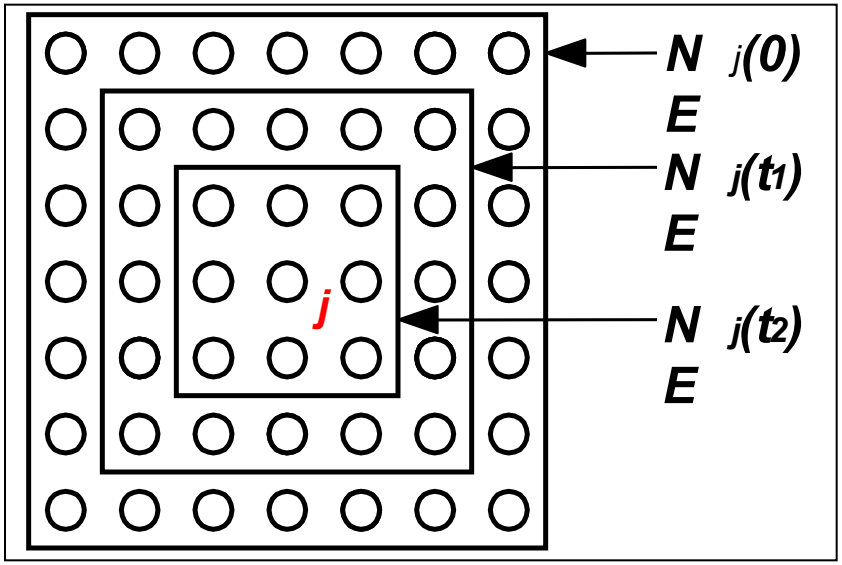
Производится подстройка весов для нейрона j^* и всех нейронов из его зоны соседства NE. Новые значения весов:

$$w_{ij}(t+1) = w_{ij}(t) + r(t)\,(x_i(t) - w_{ij}(t)),$$

где $r(t)$ - шаг обучения, уменьшающийся с течением времени (положительное число, меньше единицы).

Шаг 6. Возвращение к шагу 2 .

Зоны топологического соседства нейронов



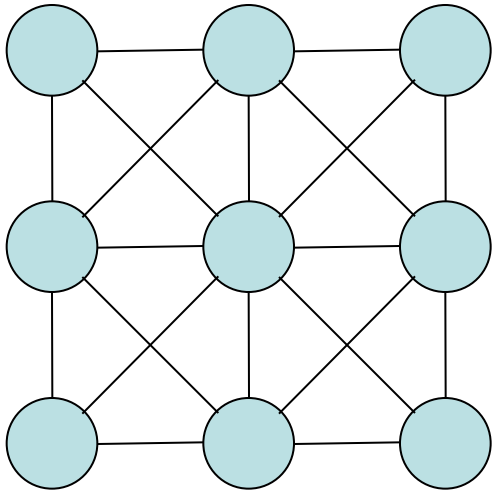
$NE_j(t)$ − множество нейронов, которые являются соседями нейрона *j* в момент времени *t*.

Окрестность - это несколько нейронов, окружающих выигравший нейрон.

SOM (Карты Кохенена)
Модель сетки

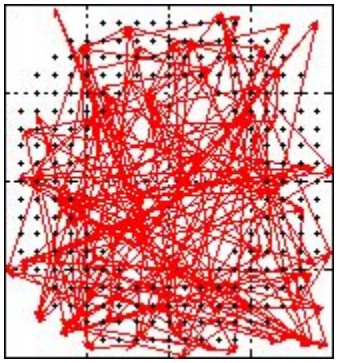
- Матрица узлов $M^{KL} = (m_{kl})$
- Соседство - 4 или 8 связность
- Каждому узлу соответствует точка в исходном пространстве

$$m_{kl} \in R^d$$

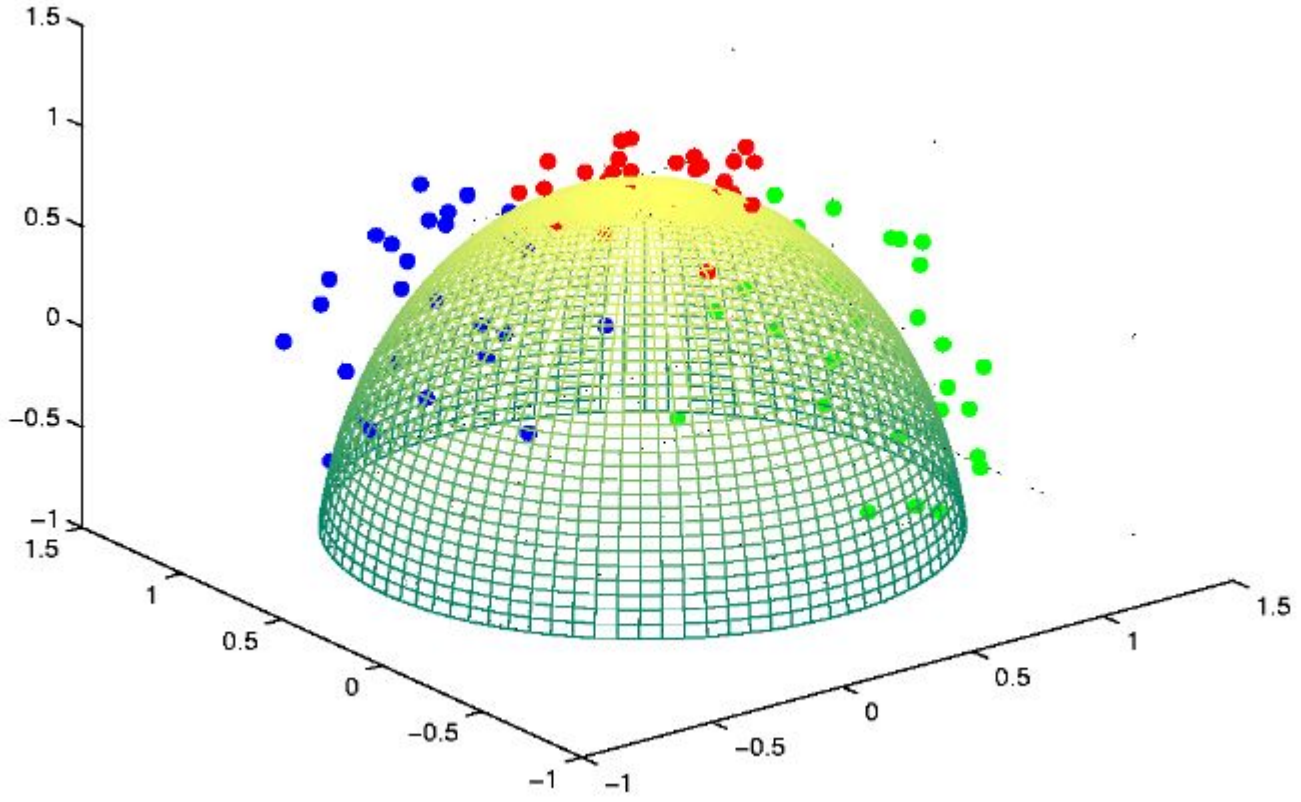


SOM (Карты Кохенена) Алгоритм построения

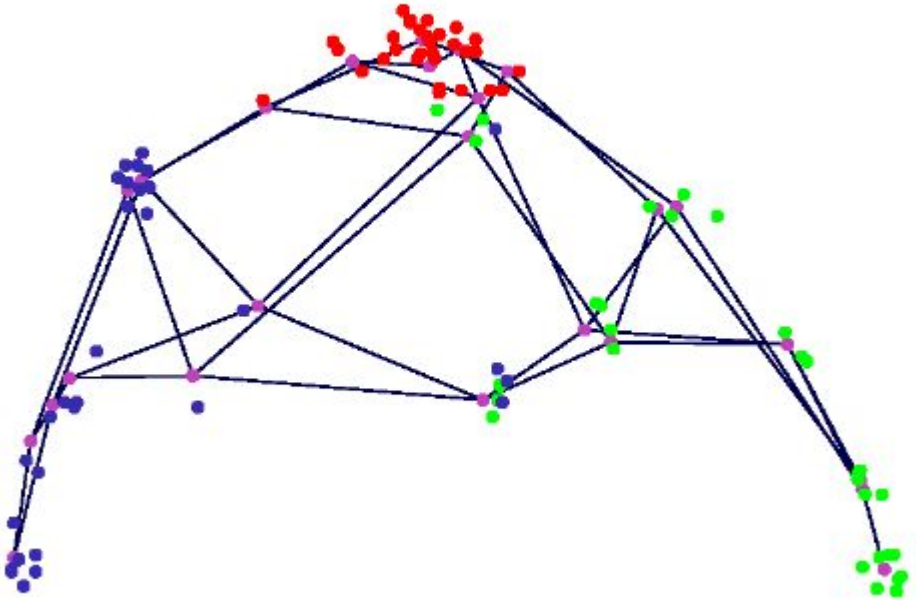
- Проинициализируем (m_{ij}) случайными значениями
- Далее, в случайном порядке будем предъявлять наблюдения и для каждого:
  - Вычисляем ближайший узел
  - Выберем множество соседей узла, такое что расстояние на сетке между ними меньше r
  - Для некоторого множества соседей узла, включая сам узел, изменяем их положения согласно: $m_{kl} \leftarrow m_{kl} + \alpha(x_i - m_{kl})$
  - Повторяем процедуру уменьшая r и α пока сеть не стабилизируется



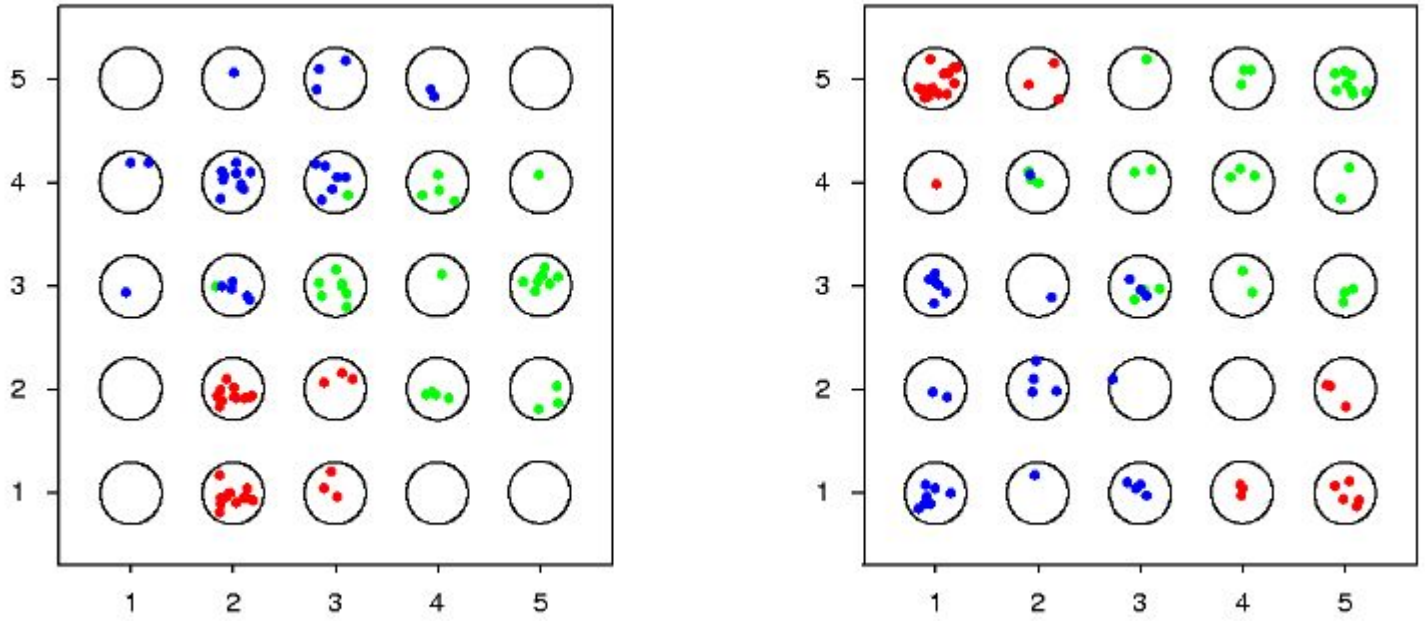
SOM (Карты Кохенена)
Иллюстрация: исходные данные



SOM (Карты Кохенена)
Иллюстрация: сетка



SOM (Карты Кохенена)
Иллюстрация: проекции на матрицу



Интерпретация работы слоя Кохонена

Вид откликов на каждый класс входных образов не известен заранее и будет представлять собой произвольное сочетание состояний нейронов выходного слоя, обусловленное случайным распределением весов на стадии инициализации. Вместе с тем, сеть способна обобщать схожие образы, относя их к одному классу.

Для приведения откликов обученной сети к удобному представлению сеть дополняется одним слоем, который по алгоритму обучения однослойного перцептрона необходимо заставить отображать выходные реакции сети в требуемые образы.

SOM (Карты Кохенена) Практическое использование

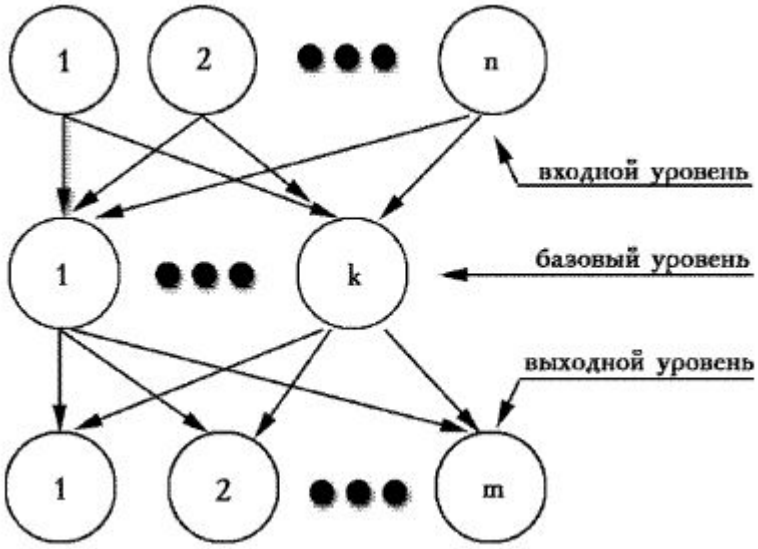
- Данные представляют некоторую поверхность, требуется сократить размерность
- Хорошо подходят для последующей кластеризации
- Могут работать «online»
- В случае слишком сложных данных не информативны

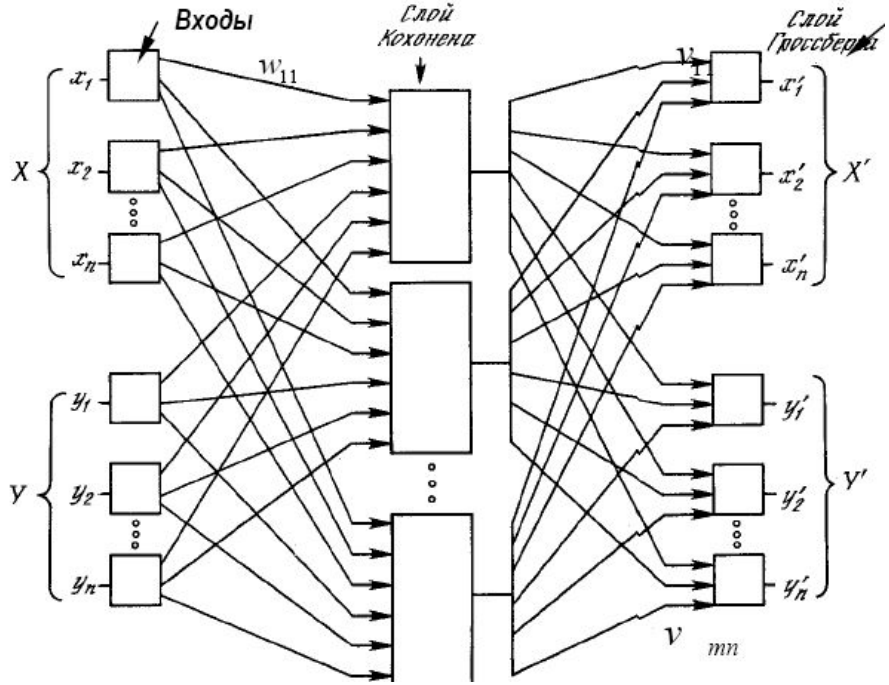
Монохромные (бинарные) изображения

2 цвета (черный и белый)

Изображение представляется в виде растра (таблицы) со значениями яркости 0 (черный) и 1 (белый)

Нейронная сеть Цао-Ена (Гроссберга-Кохонена)





Отличия сети Цао-Ена от многослойного персептрона

Функция состояния

Персептрон

$$s = \sum_{i=0}^{n} x_i \cdot w_i$$

Сеть Цао-Ена

$$s = \sum_{i=1}^{n} w_{ij} x_i$$

Функция активации

Персептрон

$$f(s) = \begin{cases} 0, s < 0; \\ 1, s \geq 0 \end{cases} \qquad f(s) = \frac{1}{1 + e^{-s}}$$

Сеть Цао-Ена

$$f(s) = s$$

На первый взгляд разница между сетями несущественна, но при рассмотрении процесса обучения будут видны принципиальные преимущества

Классификация изображений. 1 этап сжатия

Задача классификации заключается в разбиении объектов на классы (формирующиеся динамически), причем основой разбиения служит вектор параметров объекта.

Назовём *прототипом* класса объект, наиболее типичный для своего класса. Один из самых простых подходов к классификации состоит в том, чтобы предложить существование определённого числа классов и произвольным образом выбрать координаты прототипов.

В качестве меры близости двух векторов обычно выбирается евклидово расстояние:

$$d(x, y) = \sum (x_i - y_i)^2 .$$

Алгоритм обучения сети Цао-Ена. Шаги инициализации

Шаг 1. Нормализация входов

Единичная нормировка всех векторов (X,Y) обучающего множества

Шаг 2. Инициализация весов и нормализация столбцов матрицы весов

Весовым коэффициентам сети Wij,Vji, i=1,n, j=1,m присвоить случайные значения и произвести единичную нормировку матриц W.V по столбцам

Инициализация констант α и β

Алгоритм обучения сети Цао-Ена. Обучение слоя Кохонена

При обучении сети самоорганизация происходит следующим образом. Для каждого j-го нейрона определяется расстояние от него до вектора X:

$$d_j = \sum_{i=1}^{n} (x_i - w_{ij})^2$$

Далее выбирается нейрон с номером k для которого это расстояние минимально (то есть сеть отнесла входной вектор к классу с номером k).

На текущем шаге обучения N будут модифицироваться только веса нейронов и окрестности нейрона k:

$$w_{ij}^{N+1} = w_{ij}^{N} + \alpha_N (x_i - w_{ij}^{N}).$$

Замечание по обучению

Проведем предварительную единичную нормировку входных векторов:

$$x_i' = \frac{x_i}{\sqrt{x_1^2 + x_2^2 + \ldots + x_n^2}}$$

В таком случае, вычисление расстояния будет выглядеть как вычисление скалярного произведения:

$$d_j = \sum_{i=1}^{n}(x_i - w_{ij}^N)^2 = \sum_{i=1}^{n} x_i^2 - 2\sum_{i=1}^{n} x_i w_{ij}^N + \sum_{i=1}^{n} w_{ij}^2 = 2 - 2\sum_{i=1}^{n} x_i w_{ij}^N .$$

Таким образом, наименьшим будет расстояние до того нейрона, скалярное произведение с весами которого у входного вектора максимально.

Обучение слоя Гроссберга. 2 этап сжатия

Шаг 4

Корректировка весов нейрона

$$v^{N+1}_{kj} = v^{N}_{kj} + \beta_N (z_i - v^{N}_{kj})$$

(k-номер выигравшего нейрона, весовой вектор которого даёт максимальное скалярное произведение с входным вектором)

Алгоритм обучения сети Цао-Ена. Итерации

Шаг 5

Уменьшение значений α_N, β_N

Шаг 6

Повтор шагов 3-5

Общее количество итераций должно быть достаточно большим. Все образы обучающей выборки желательно предъявить сети несколько десятков или даже сотен раз.

Сравнение с JPEG

Исходное изображение

Сжатие JPEG

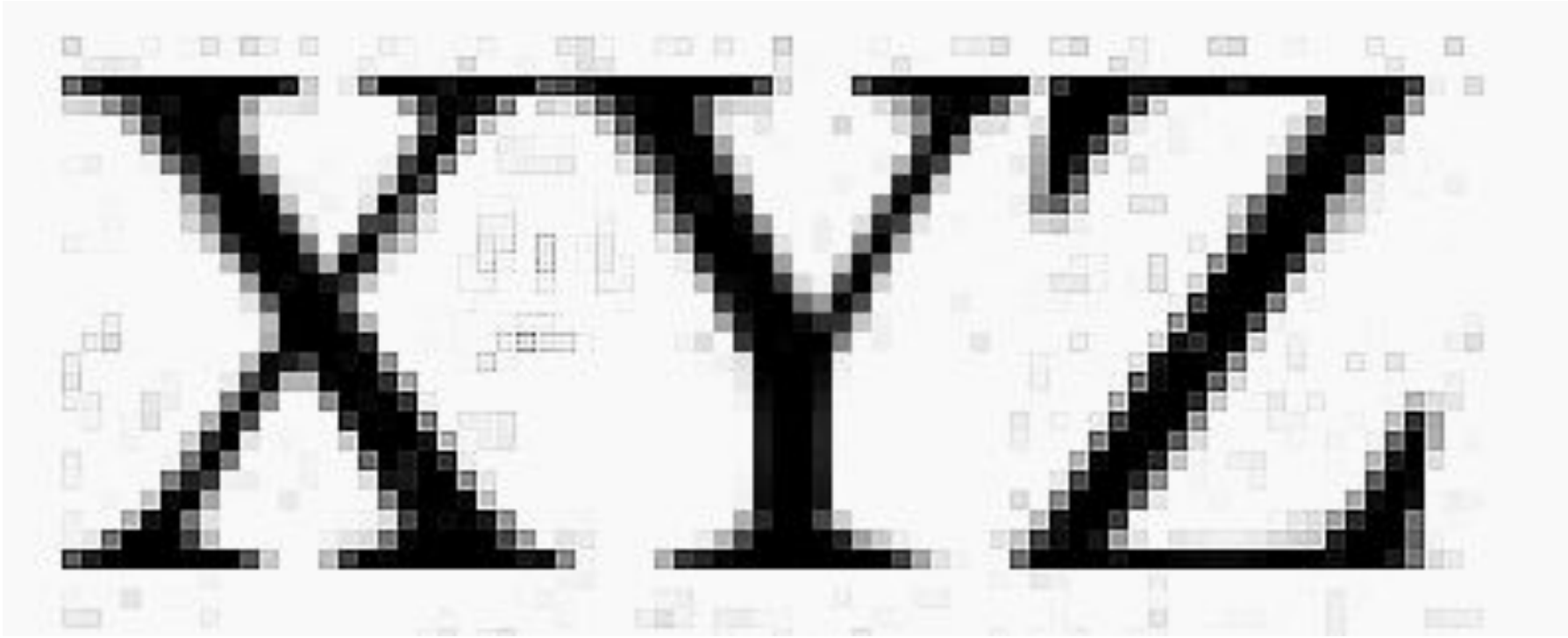
Сжатие Цао-Ена

Крупный масштаб

Исходное изображение

Крупный масштаб

Сжатие JPEG



Крупный масштаб

Сжатие Цао-Ена

СПАСИБО)

ВОПРОСЫ