Лекция 1. Машинное обучение. Нейронные сети. Сверточные нейронные сети (CNN)

Ю.В. Визильтер

начальник подразделения 3000
ФГУП «Государственный научно-исследовательский институт авиационных систем»,
д.ф-м.н., профессор РАН

Москва, ГосНИИАС, 04.09.2018

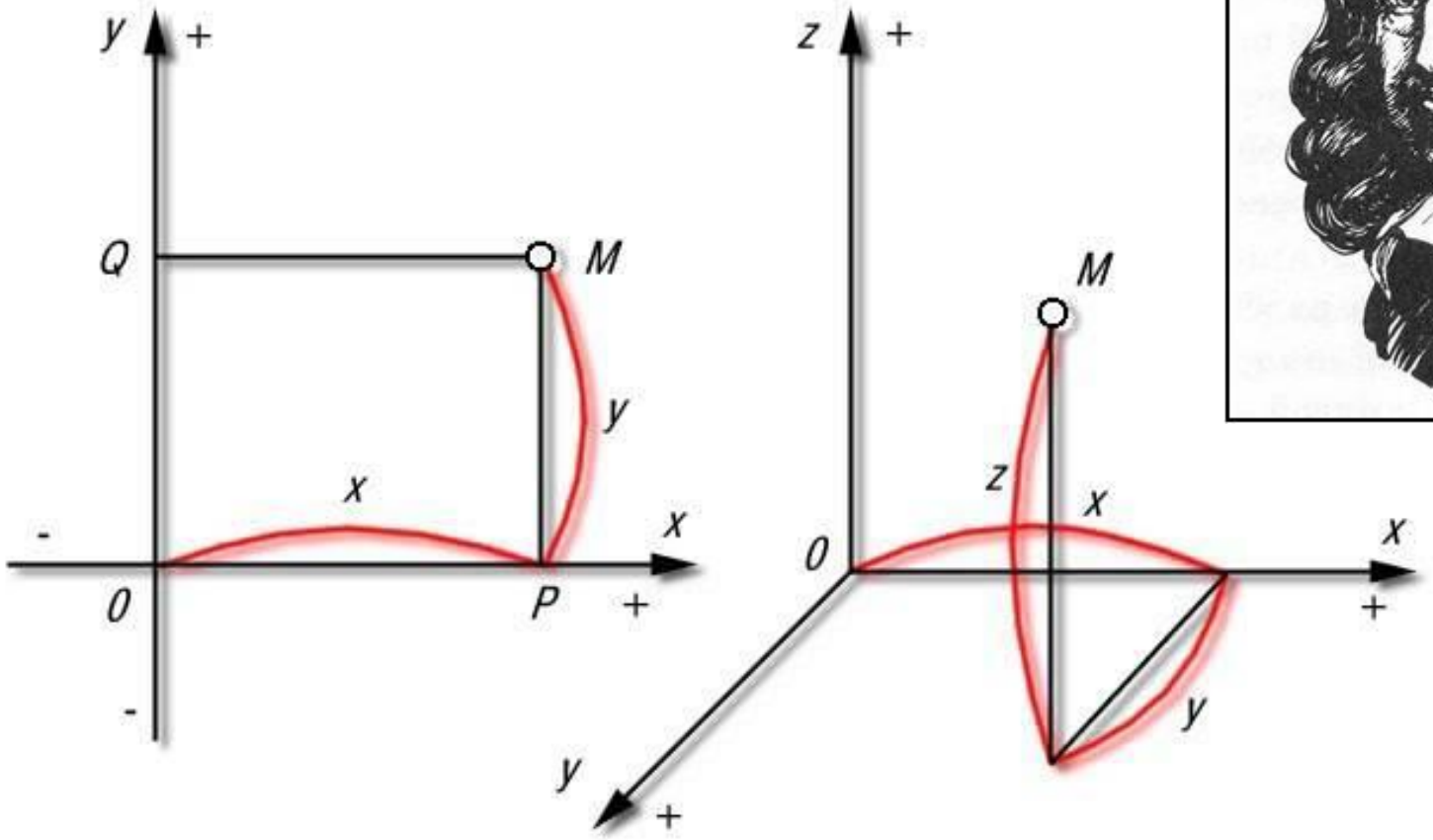
Содержание

- **Машинное обучение**
- **Линейные классификаторы**
- **Обучение методом градиентного спуска**
- **Нейронные сети**
- **Многослойные персептроны**
- **Идеи из компьютерного зрения: линейные фильтры, выделение и сбор признаков**
- **Сверточные нейронные сети (CNN) и глубокое обучение**

Машинное обучение

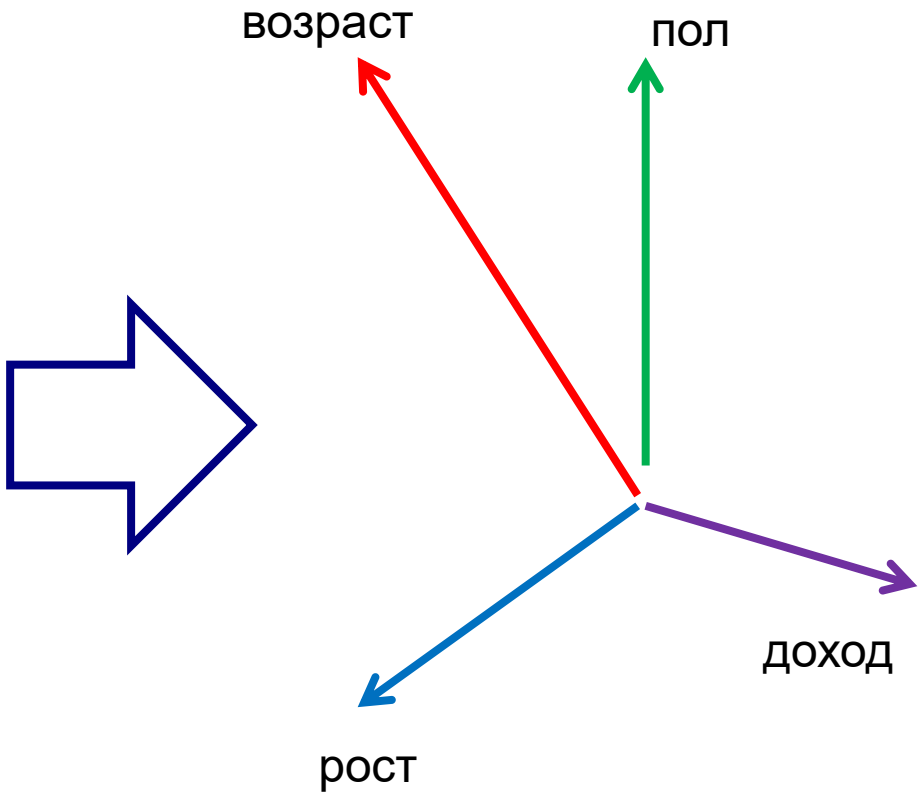
Краткое введение/напоминание

Главная идея: координатизация мира!



Главная идея: координатизация мира!

Признак – это одно число, характеризующее объект



Вектор признаков – это упорядоченный набор чисел, характеризующий объект $\mathbf{x} = (x_1,\ldots,x_n)$.

Пространство признаков –это множество векторов признаков, для любой пары которых можно определить, что они близки (похожи) или далеки (непохожи).

Обучение с учителем

Компоненты задачи:

пространство объектов $\mathcal{A}$, *множество классов* $C=\{c_1, \ldots, c_l\}$,

разбиение объектов по классам: $c_{\mathcal{A}}(a)$: $a \in \mathcal{A} \mid\!\rightarrow c \in C$,

пространство описаний (*признаков*) $\mathcal{X}$,

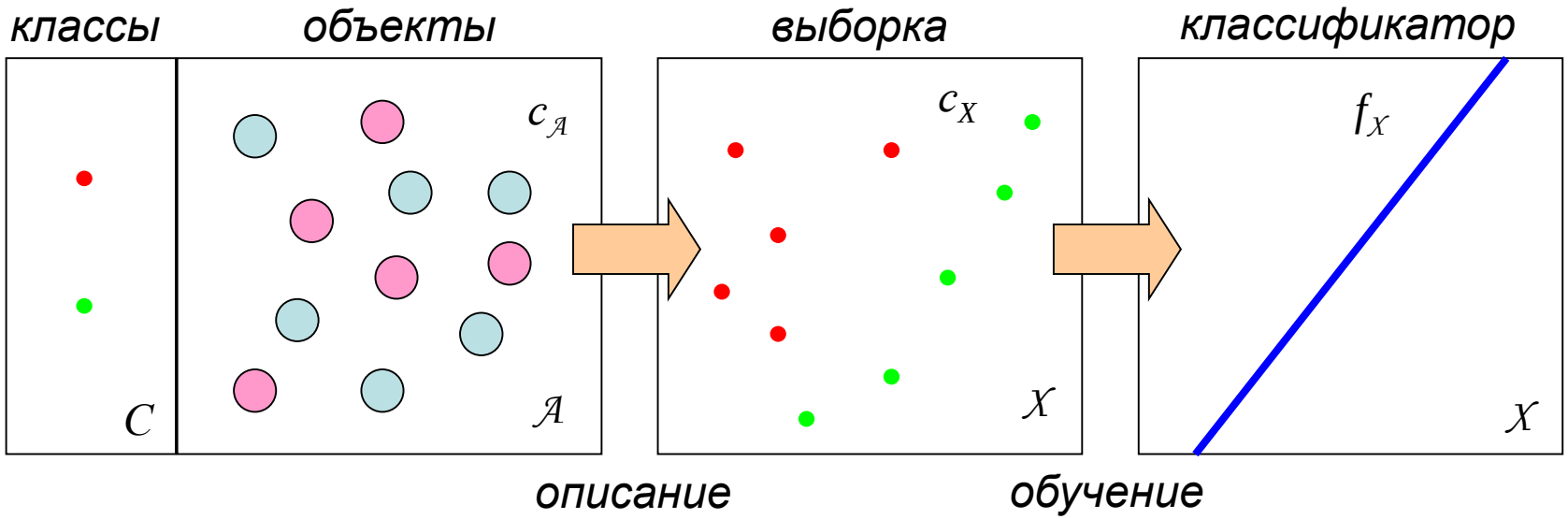
описание объектов признаками: $x_{\mathcal{A}}(a)$: $a \in \mathcal{A} \mid\!\rightarrow x \in \mathcal{X}$.

выборка объектов $A \subseteq \mathcal{A}$, $\|A\|<+\infty$, *выборка описаний* $X \subseteq \mathcal{X}$, $\|X\|<+\infty$.

обучающая выборка: $c_X(x)$: $x_{\mathcal{A}}(a) \in X \mid\!\rightarrow c_{\mathcal{A}}(a) \in C$.

распознающий алгоритм или *классификатор* $f_{\mathcal{X}}(x)$: $x \in \mathcal{X} \mid\!\rightarrow c \in C$.

Требуется: по информации о c_X построить такой $f_{\mathcal{X}}(x)$, который обеспечивает *в некотором смысле наилучшее разбиение* $\mathcal{X}$ на классы из C.



Задача синтеза классификатора. Как описать «наилучшее разбиение»?

Задача обучения классификатора - глядя на *обучающую* выборку минимизировать число ошибок на неизвестной заранее *тестовой* выборке.

Дополнительные компоненты задачи:

Тестовая выборка

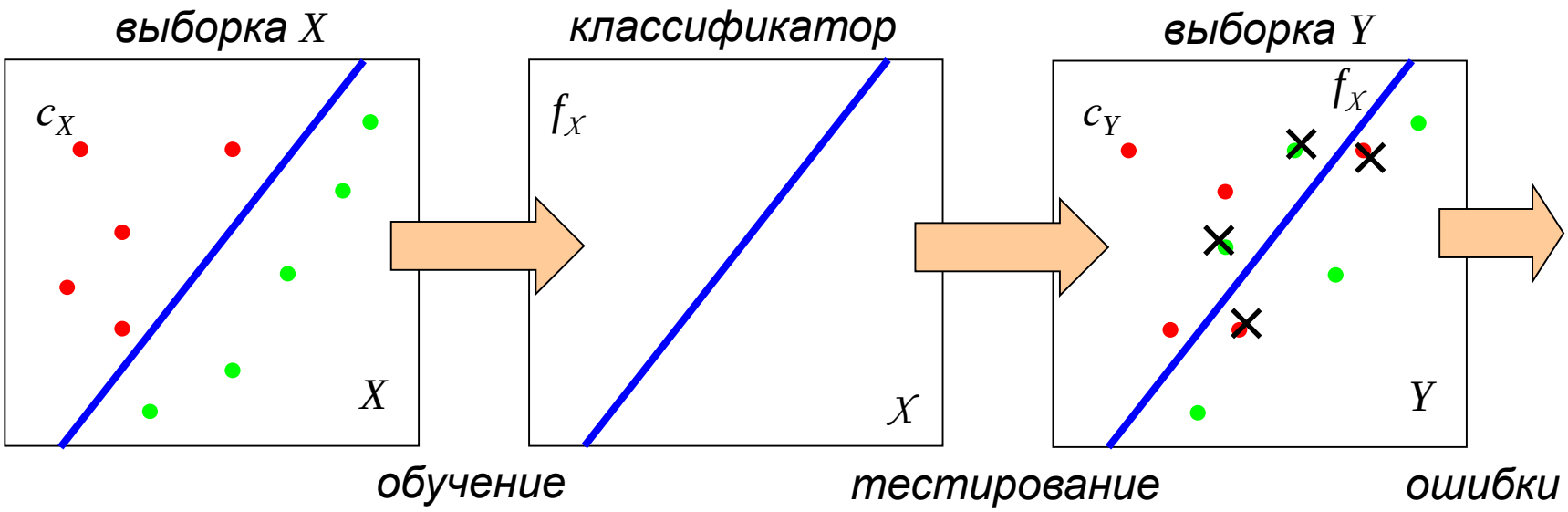
$$c'_Y(x)\colon x \in Y \mid\!\to c \in C,\ Y \subseteq X,\ Y \cap X = \varnothing,\ \|Y\| < +\infty,$$

Критерий эмпирического риска на выборке Y:

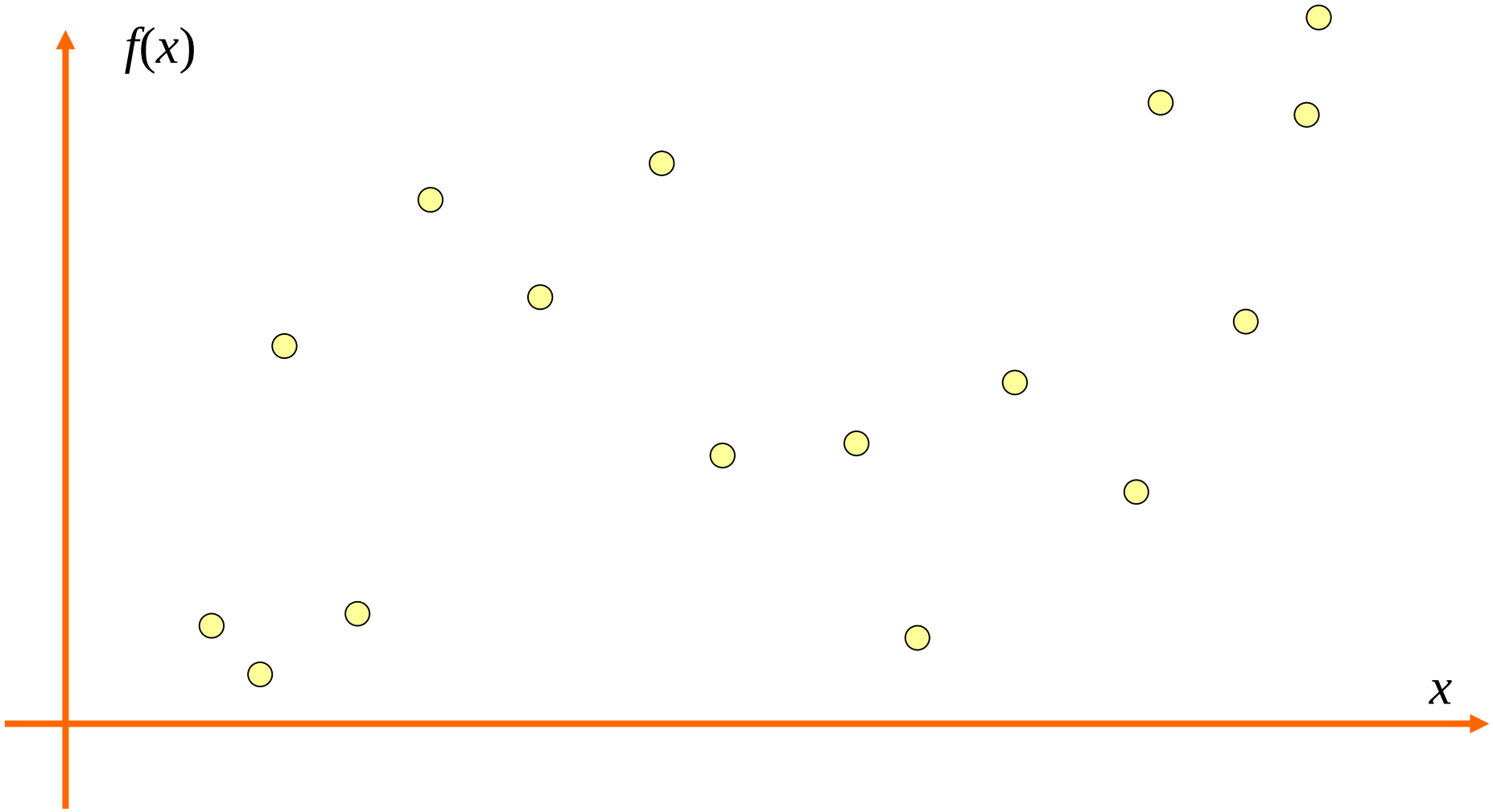
$$J_Y(f_X) = d_H(f_Y, c'_Y) / \| Y \|,$$

$d_H(f_Y, c'_Y) = \sum_{x \in Y} 1(f(x) \neq c'(x))$ – *число ошибок классификации* на выборке Y.

$\| Y \| = \sum_{x \in Y} 1$ – объем выборки Y.

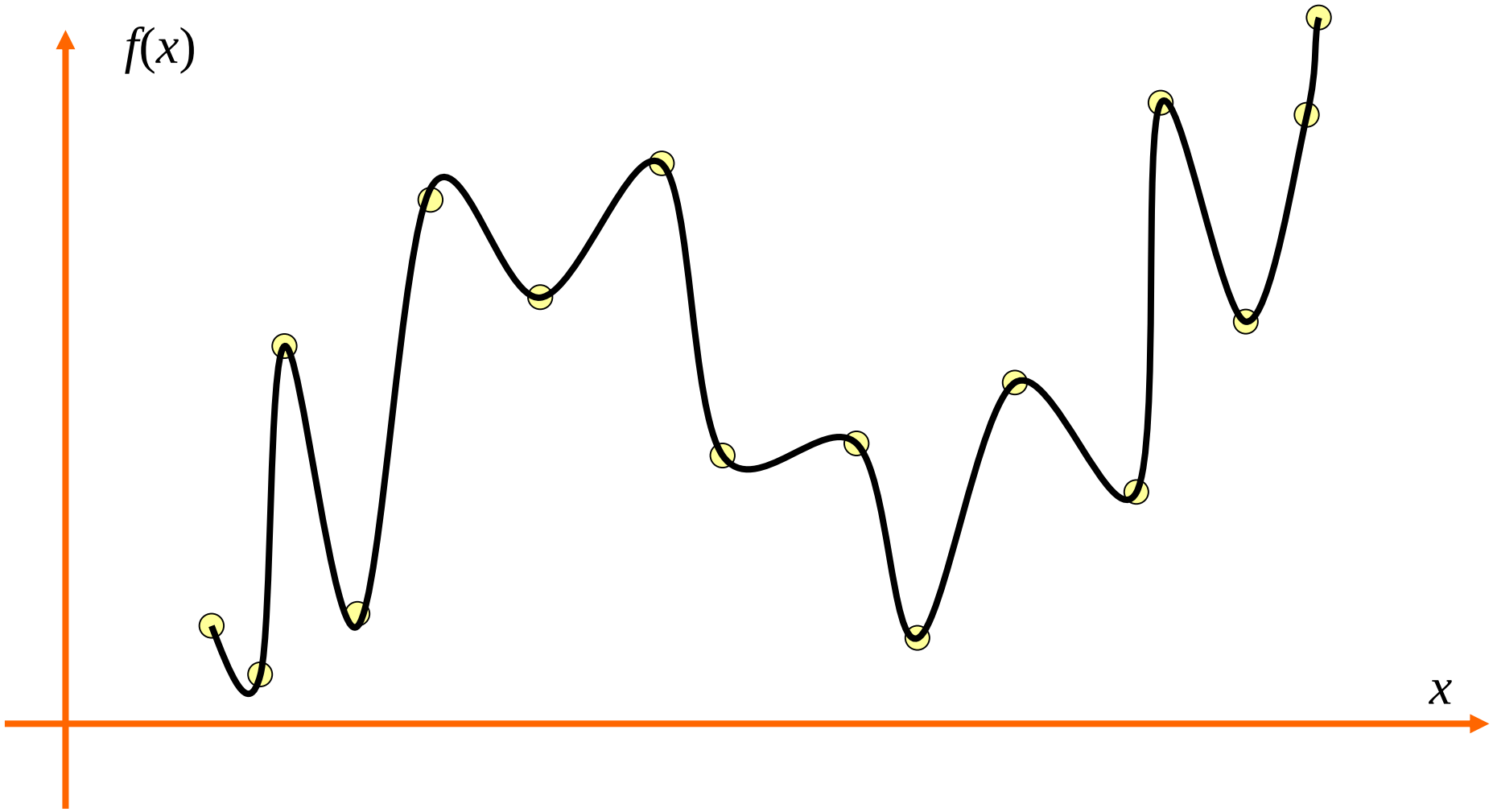


Проблема переобучения



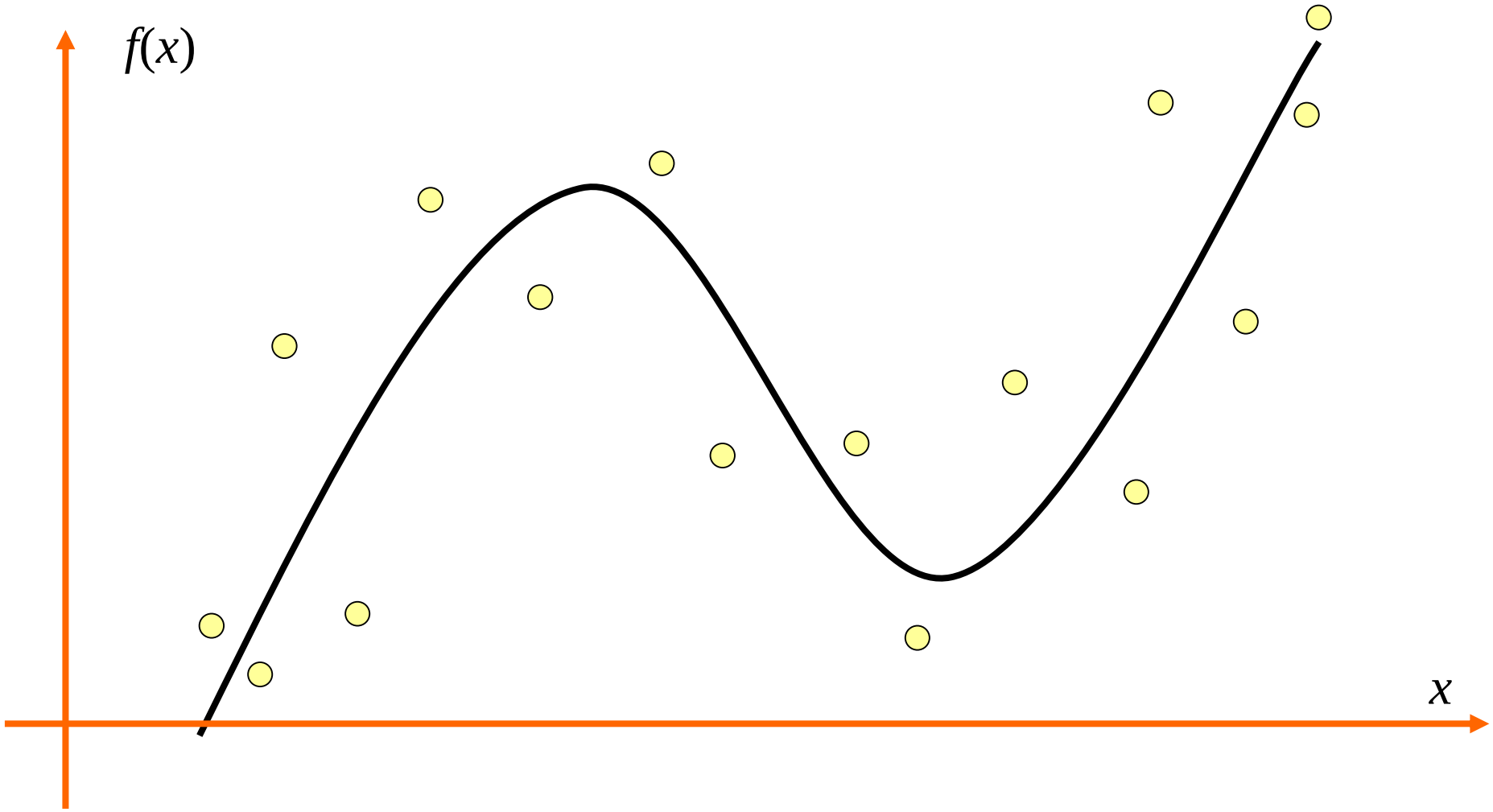
$$\Phi(\mathrm{A},\mathrm{L}) = \mathrm{J}(\mathrm{A},\mathrm{L}) \rightarrow min(\mathrm{L} \in F(x))$$

Проблема переобучения



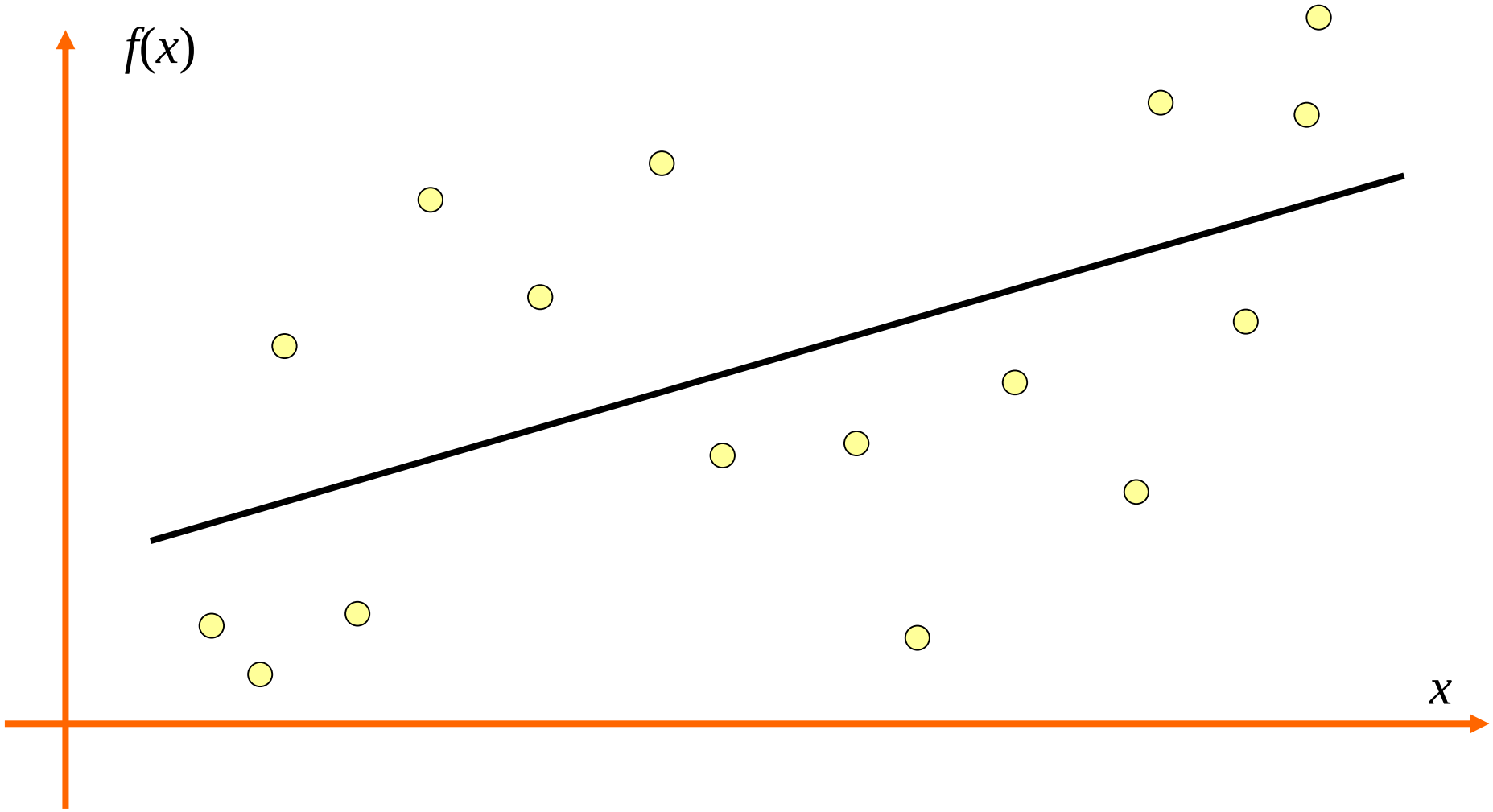
$$\Phi(\mathrm{A},\mathrm{L}) = \mathrm{J}(\mathrm{A},\mathrm{L}) \rightarrow min(\mathrm{L} \in F(x))$$

Регуляризация по сложности



$$\Phi(A,L)=J(A,L)+\alpha\times Q(L)\rightarrow min(L\in F(x))$$

Регуляризация по сложности

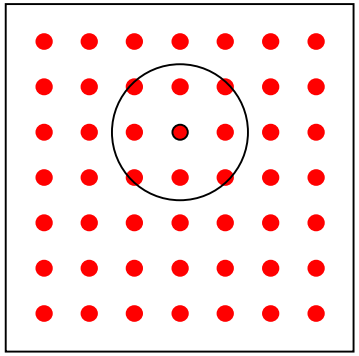


$$\Phi(A,L)=J(A,L)+\alpha\times Q(L)\rightarrow min(L\in F(x))$$

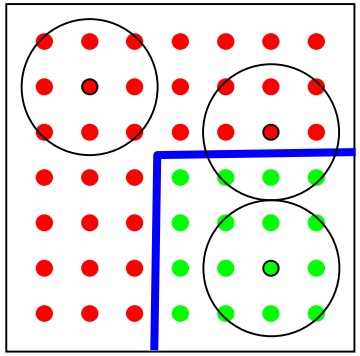
Почему мы можем пользоваться простыми классификаторами

Принцип компактности: более близкие объекты должны с большей вероятностью принадлежать к одному классу.

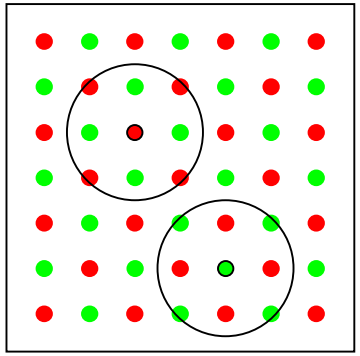
- *Айзерман М. А., Браверман Э. М., Розоноэр Л. И. Метод потенциальных функций в теории обучения машин. М.: Наука, 1970. 320 pp.*
- *Хачай М. Ю. Топологический подход к выводу условий равномерной по классу событий сходимости частот к вероятностям. // Интеллектуализация обработки информации: 8-я международная конференция (ИОИ-8), Кипр, г.Пафос, 2010 г.: Сборник докладов. – М.: МАКС Пресс, 2010, с.91-94.*



компактный класс



локально компактные классы



некомпактные классы

Свойством компактности обладает *хорошее для данной задачи распознавания* пространство признаков. Это не свойство конечного классификатора, а свойство пространства признаков. Если признаки выбраны (сформированы) удачно, то классы будут легко разделяться простыми классификаторами. Отсюда возникает задача *формирования пространства признаков*. Раньше это делал человек, теперь - сама CNN.

Линейные классификаторы

Краткое введение/напоминание

Двухклассовый классификатор

Случай двух классов

Степень принадлежности 1-му классу

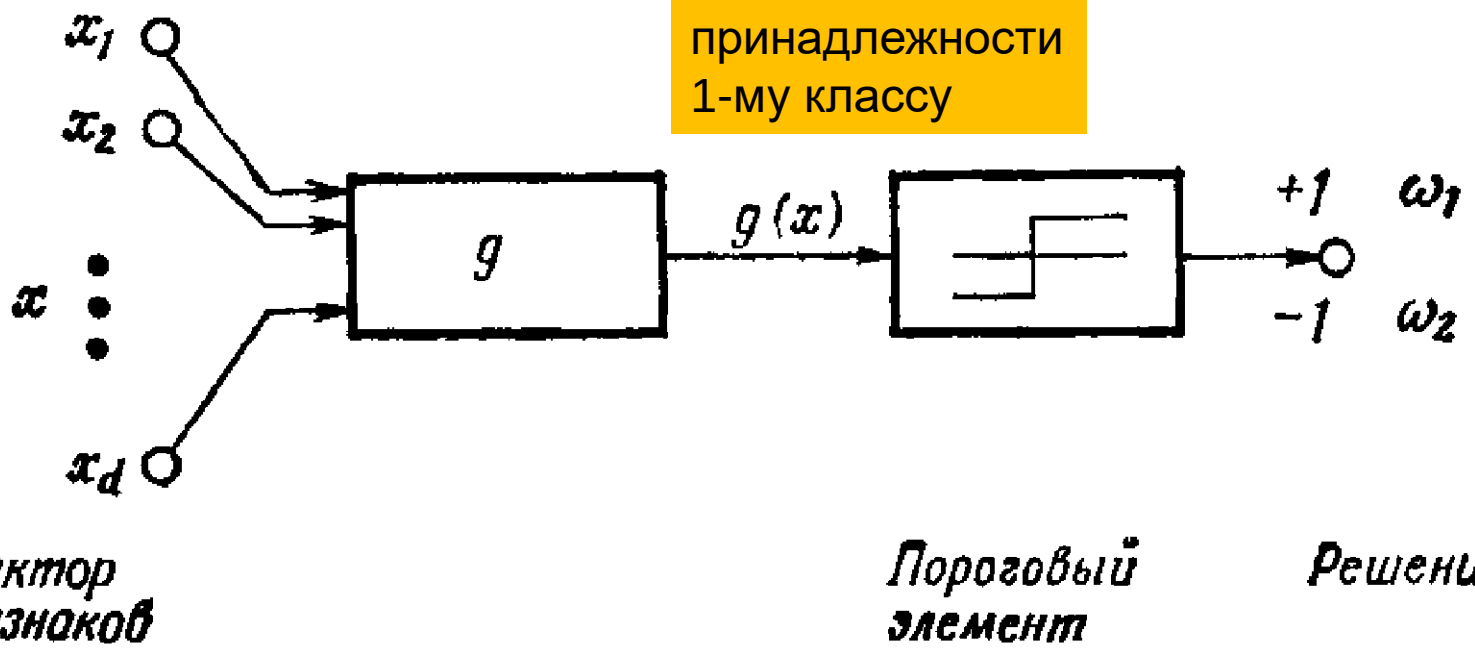


Рис. 2.5. Схема классификатора образов для двух классов.

Степень принадлежности– некое число, которое характеризует принадлежность наблюдаемого объекта (вектора признаков) классу. Чем оно больше, тем выше уверенность в этом.

Одномерный линейный классификатор

Пример. Один признак

Степень принадлежности классу = один из признаков (яркость)

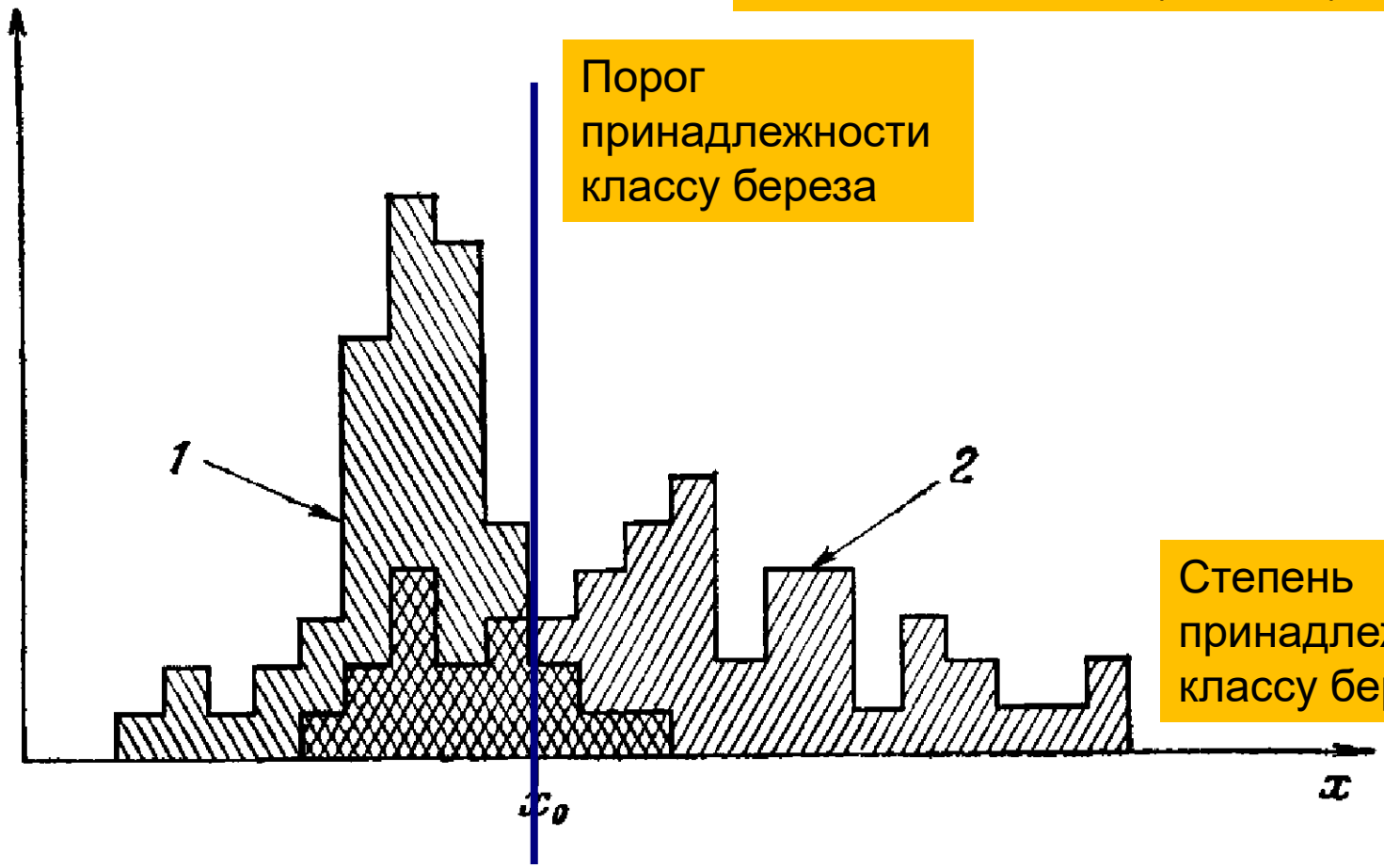


Рис. 1.2. Гистограмма яркости (по оси ординат — число кусков древесины; *1* — ясень, *2* — береза).

Двумерный линейный классификатор

Пример. Два признака

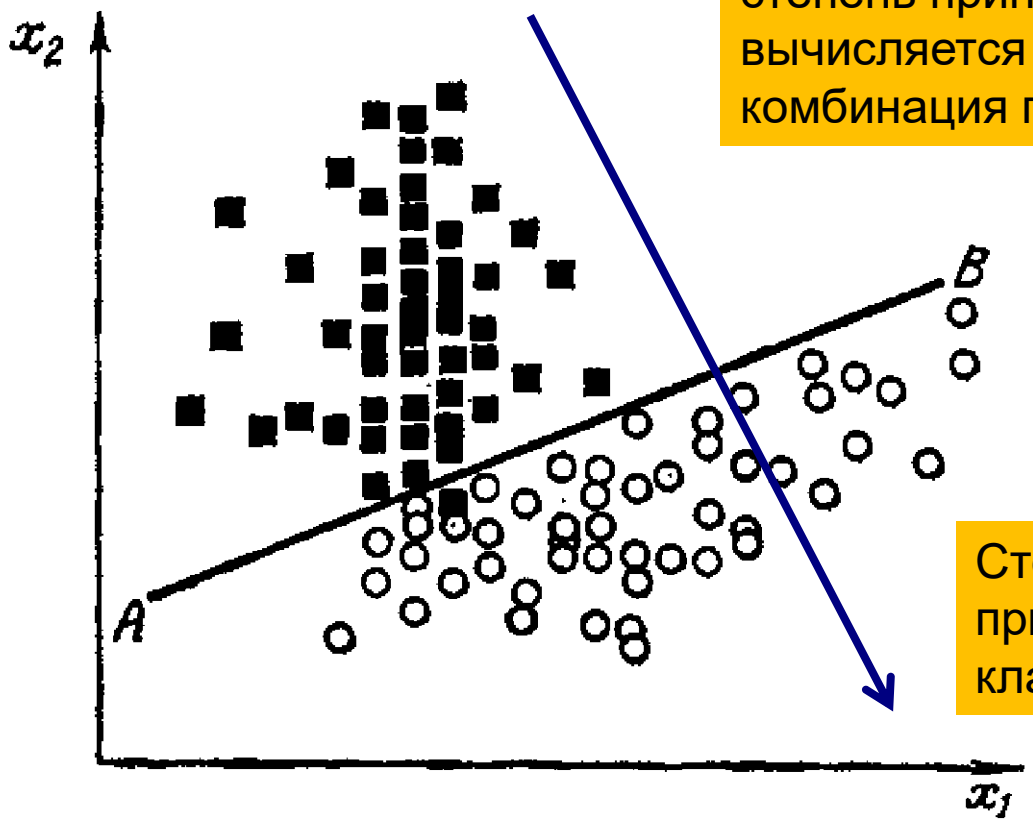


Рис. 1.3. Диаграмма разброса векторов признаков (■ — ясень, ○ — береза).

В линейных классификаторах степень принадлежности классу вычисляется как линейная комбинация признаков

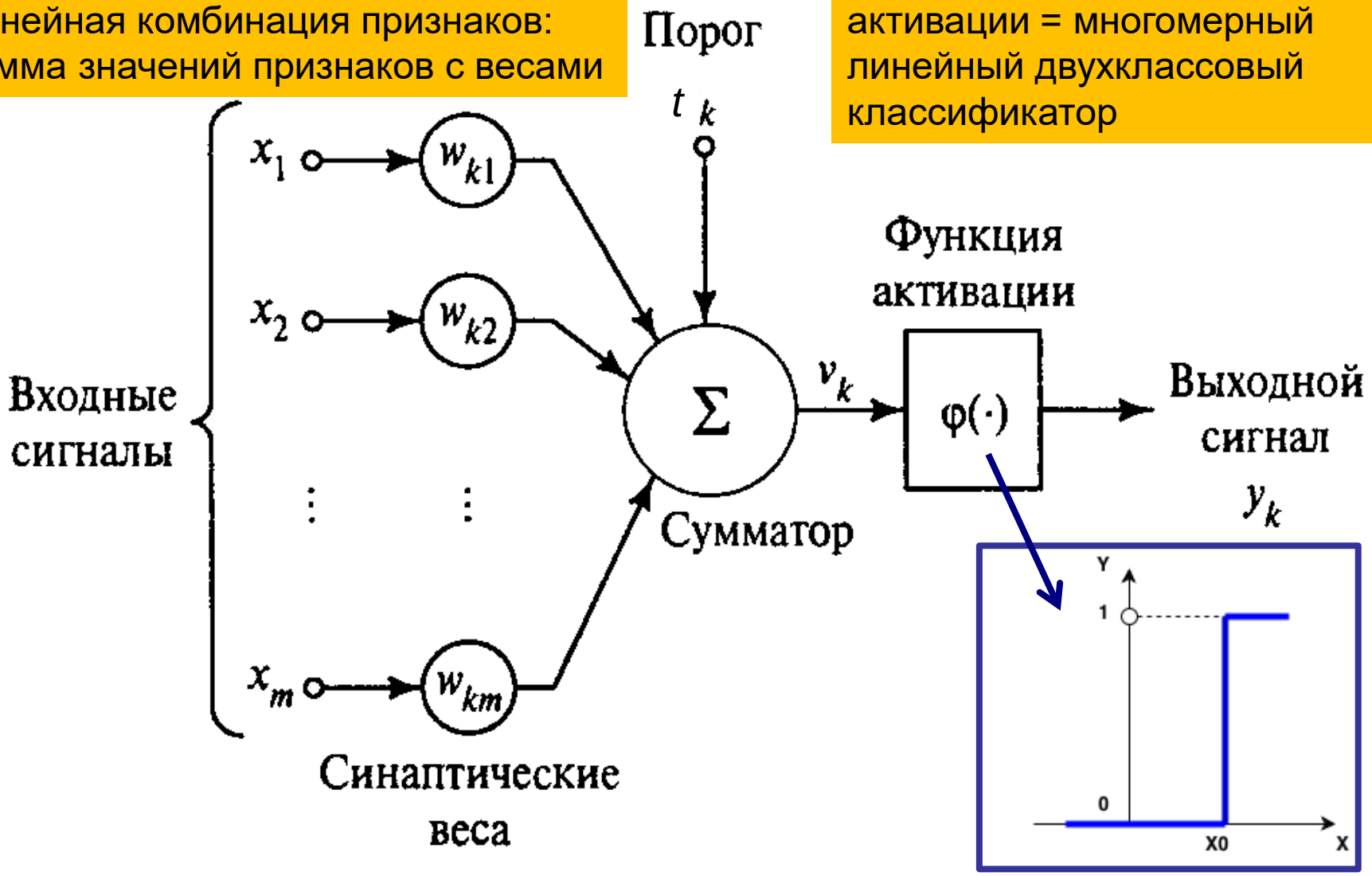
Порог принадлежности классу береза

Степень принадлежности классу береза

Многомерный линейный классификатор

Степень принадлежности классу – линейная комбинация признаков: сумма значений признаков с весами

Нейрон с пороговой функцией активации = многомерный линейный двухклассовый классификатор



$x_1*w_1+x_2*w_2+\ldots+x_m*w_m > t$ – правило принятия решения

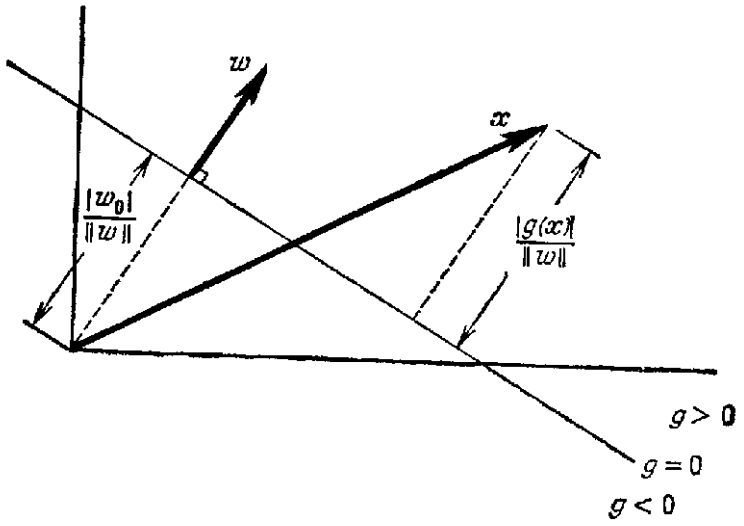
Линейные решающие правила

СЛУЧАЙ ДВУХ КЛАССОВ

Разделяющая функция, представляемая линейной комбинацией компонент вектора **x**, может быть записана в следующем виде:

$$g(\mathbf{x}) = \mathbf{w}^t\mathbf{x} + w_0, \qquad (1)$$

где **w** называется *весовым вектором*, а w_0 — *величиной порога*. В основу линейного классификатора для двух классов положено следующее решающее правило: принять решение ω_1, если $g(\mathbf{x})>0$, и ω_2, если $g(\mathbf{x})<0$. Таким образом, **x** приписывается к ω_1, если скалярное произведение $\mathbf{w}^t\mathbf{x}$ превышает порог $-w_0$. Если $g(\mathbf{x})=0$,



В двумерном случае речь идет просто об уравнении прямой или линейном неравенстве

Линейные решающие правила

Случай двух классов. Минимизация квадратичной ошибки

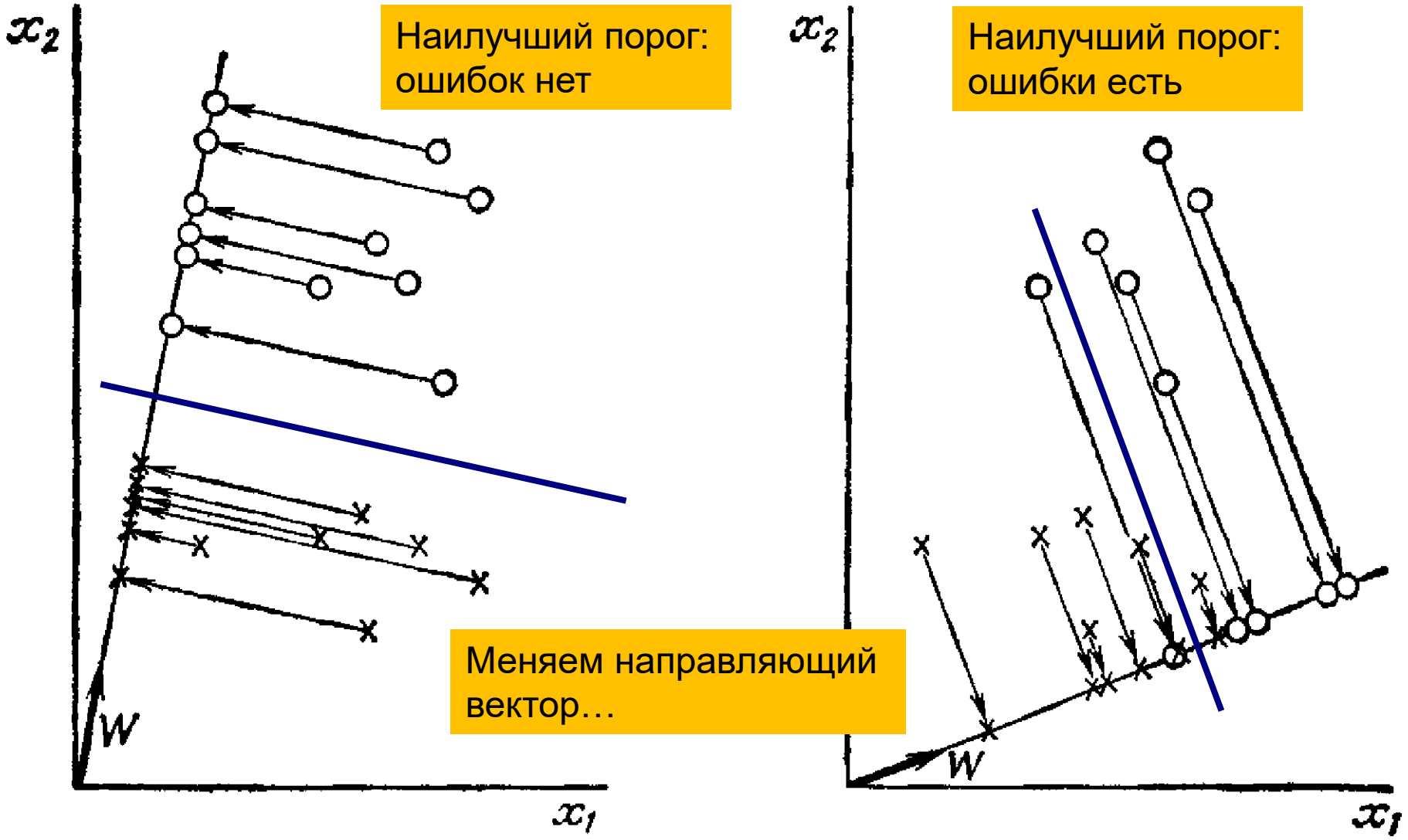


Рис. 4.6. Проекция выборок на прямую.

Обучение методом градиентного спуска

Краткое введение/напоминание

Поиск наилучшего классификатора путем минимизации функции потерь в пространстве параметров

Loss – количество ошибок классификации

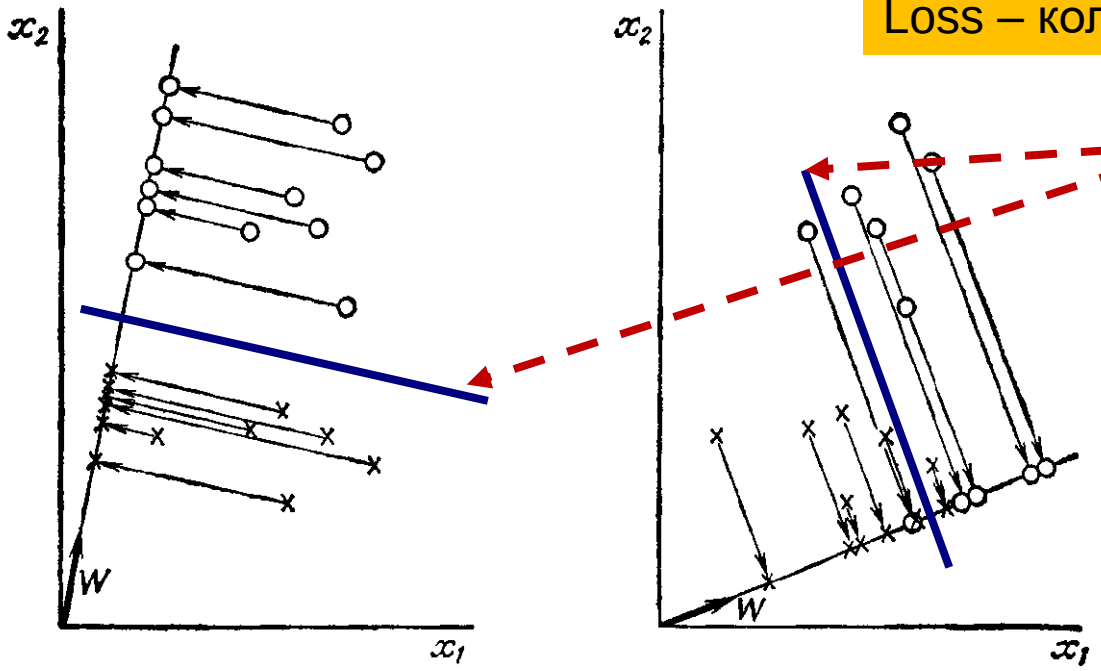
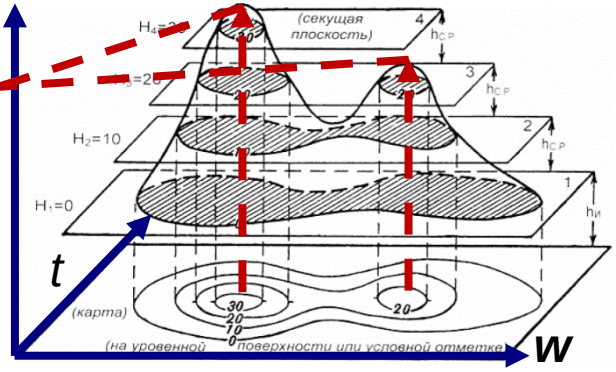
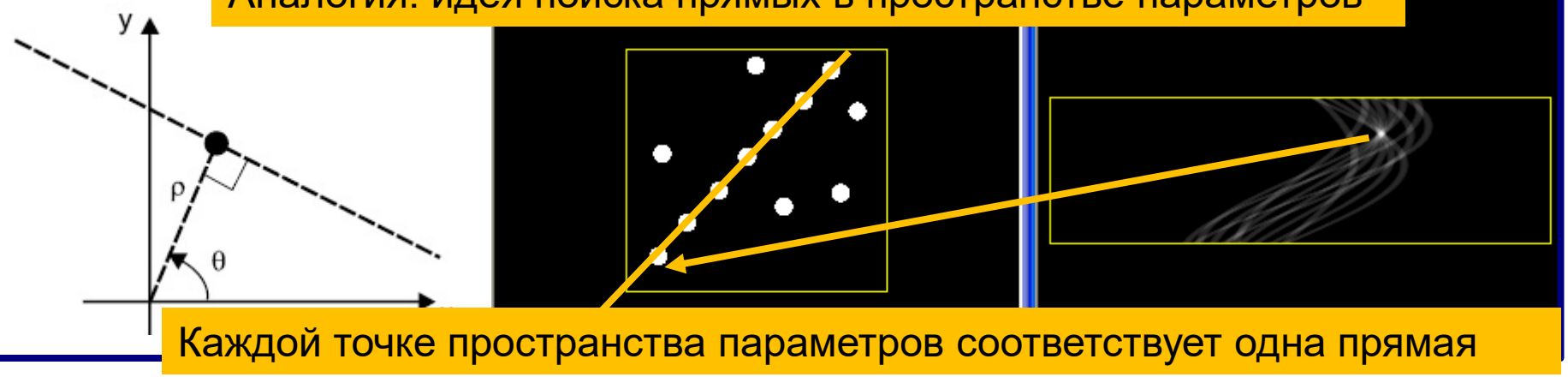


Рис. 4.6. Проекция выборок на прямую.



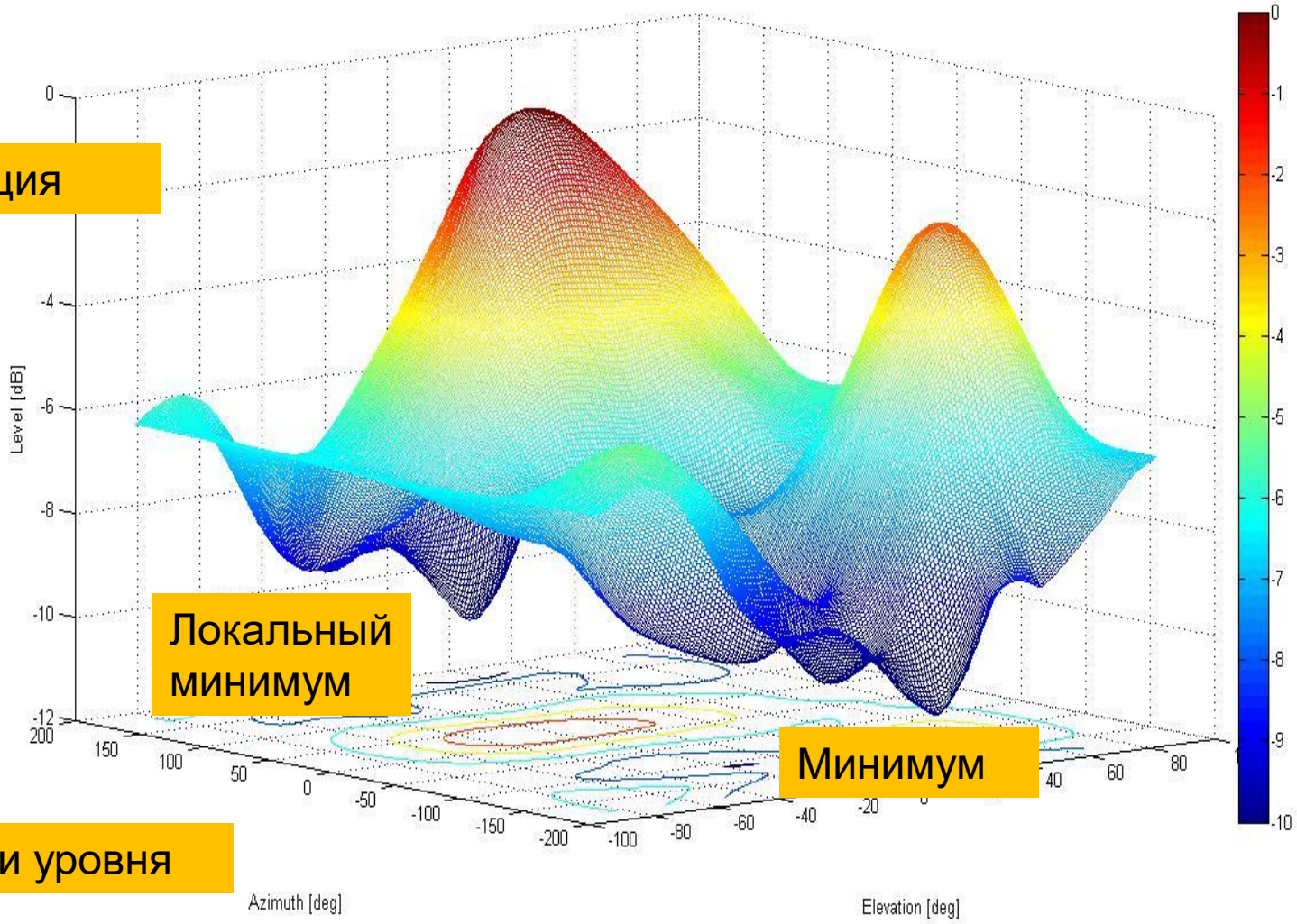
Каждой точке пространства параметров (***w,t***) соответствует одно линейное решающее правило, а ему соответствует значение Loss(***w,t***)

Аналогия: идея поиска прямых в пространстве параметров



Каждой точке пространства параметров соответствует одна прямая

Задача оптимизации (поиска минимума или максимума функции)



Многомерная оптимизация

Полный перебор по сетке – очень долго, особенно если переменных много (пространство признаков высокой размерности)

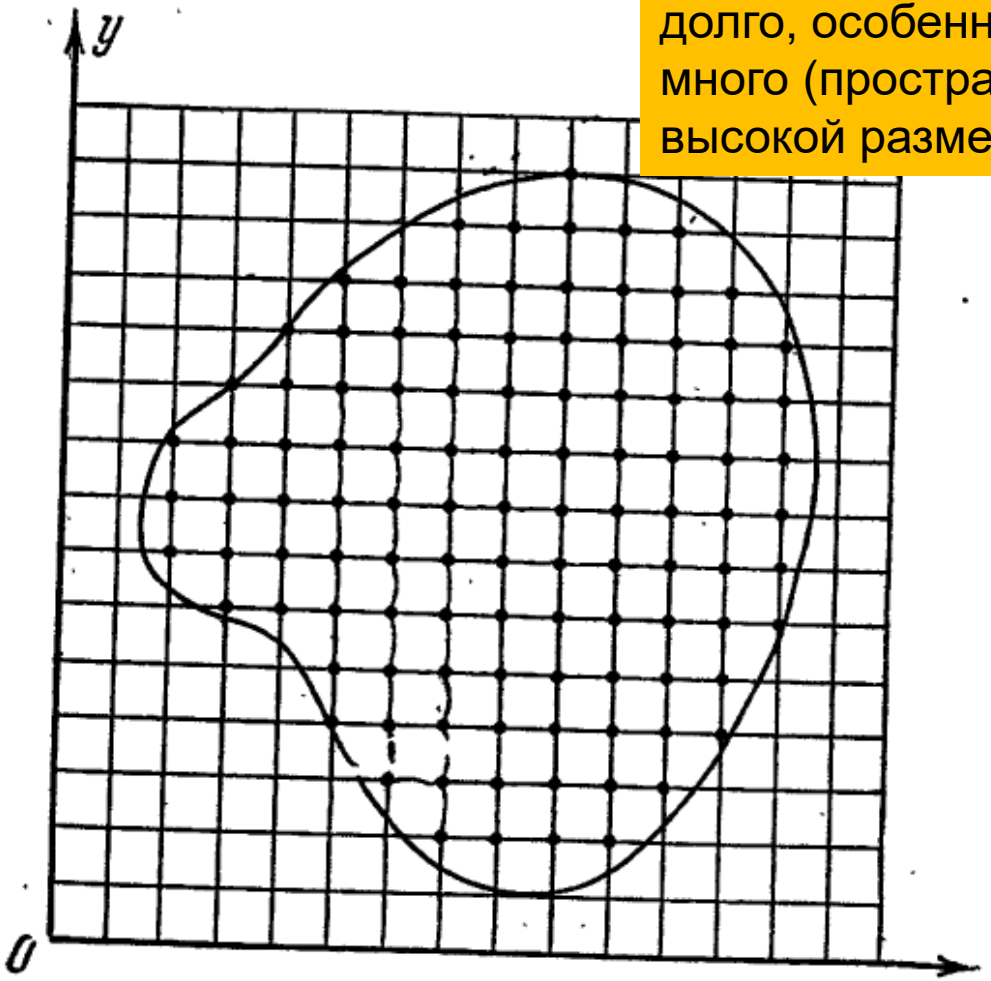
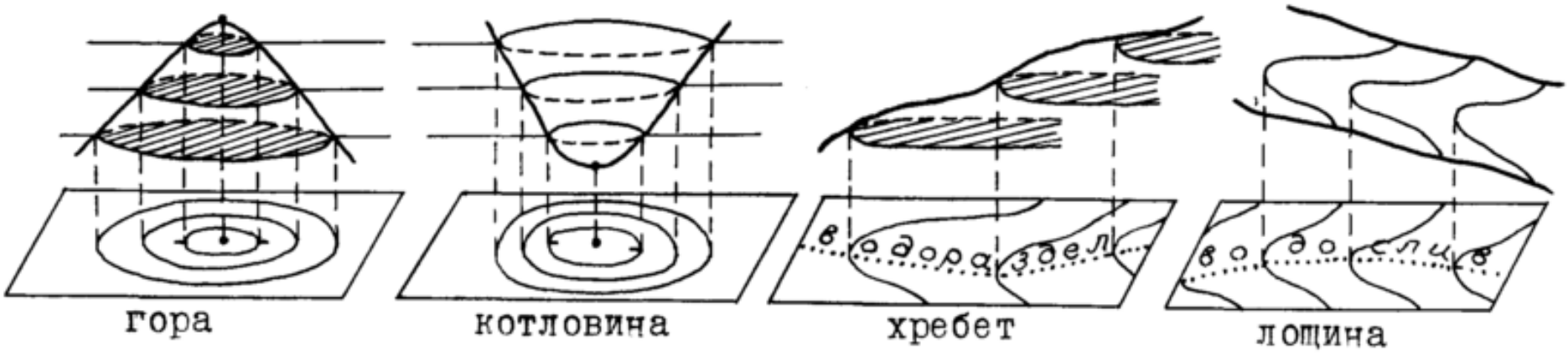


Рис. 33. Построение сетки с шагом h и выбор «пробных» точек в узлах сетки для приближенного определения наименьшего значения функции двух переменных.

Локальная оптимизация

Градиентный спуск — метод нахождения локального минимума (максимума) функции с помощью движения вдоль градиента. Для минимизации функции в направлении градиента используются методы одномерной оптимизации, например, метод золотого сечения. Также можно искать не наилучшую точку в направлении градиента, а какую-либо лучше текущей.
Наиболее простой в реализации из всех методов локальной оптимизации. Имеет довольно слабые условия сходимости, но при этом скорость сходимости достаточно мала (линейна). Шаг градиентного метода часто используется как часть других методов оптимизации.



Локальная оптимизация

Пусть целевая функция имеет вид:

$$F(\vec{x}) : \mathbb{X} \to \mathbb{R}.$$

Градиент функции в точке направлен в сторону ее наискорейшего роста

И задача оптимизации задана следующим образом:

$$F(\vec{x}) \to \min_{\vec{x} \in \mathbb{X}}$$

Основная идея метода заключается в том, чтобы идти в направлении наискорейшего спуска, а это направление задаётся антиградиентом $-\nabla F$:

$$\vec{x}^{[j+1]} = \vec{x}^{[j]} - \lambda^{[j]} \nabla F(\vec{x}^{[j]})$$

При градиентном спуске мы на каждом шаге двигаемся против направления градиента

где $\lambda^{[j]}$ выбирается

- постоянной, в этом случае метод может расходиться;
- дробным шагом, т.е. длина шага в процессе спуска делится на некое число;
- наискорейшим спуском:

$$\lambda^{[j]} = \operatorname{argmin}_{\lambda} F(\vec{x}^{[j]} - \lambda^{[j]} \nabla F(\vec{x}^{[j]}))$$

Длина шага спуска пропорциональна величине градиента с коэффициентом λ

Локальная оптимизация

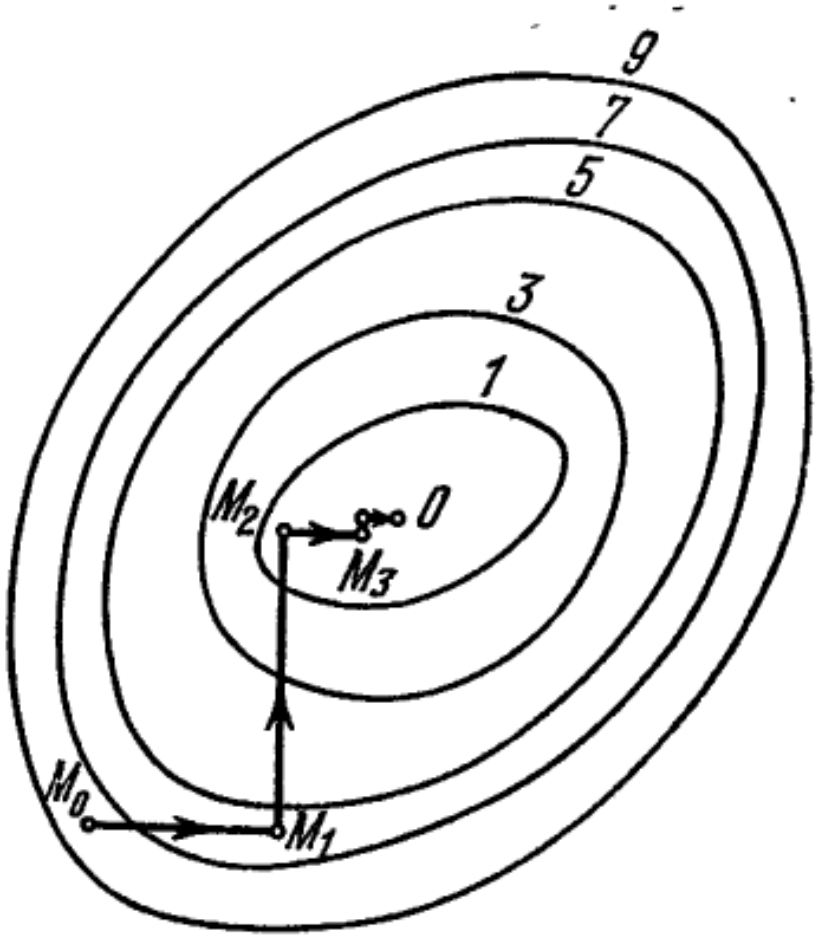


Рис. 34. Поиск наименьшего значения функции методом покоординатного спуска.

Локальная оптимизация

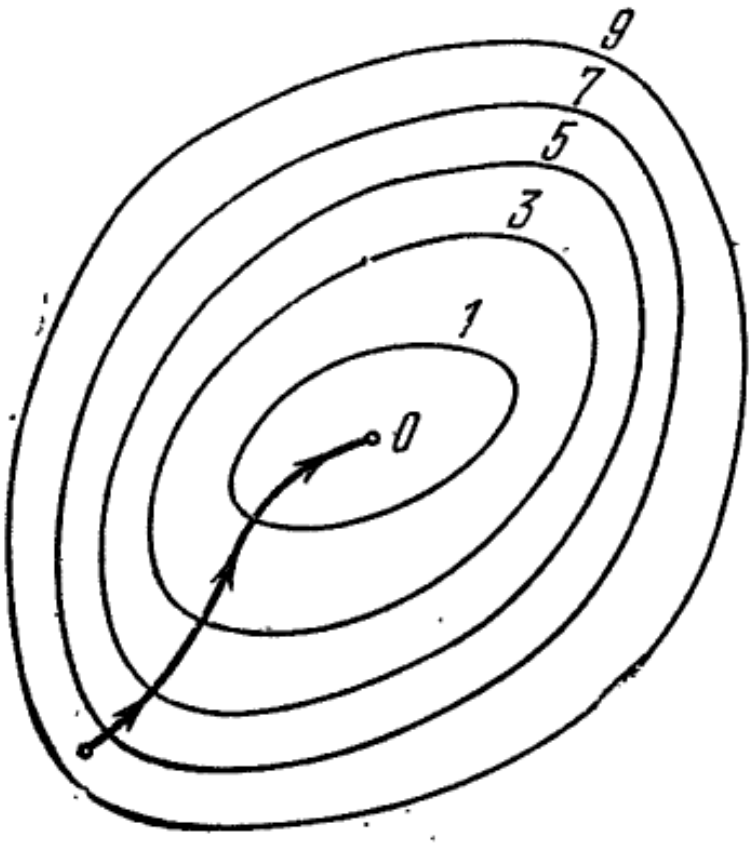


Рис. 35. Поиск наименьшего значения функции методом градиентного спуска.

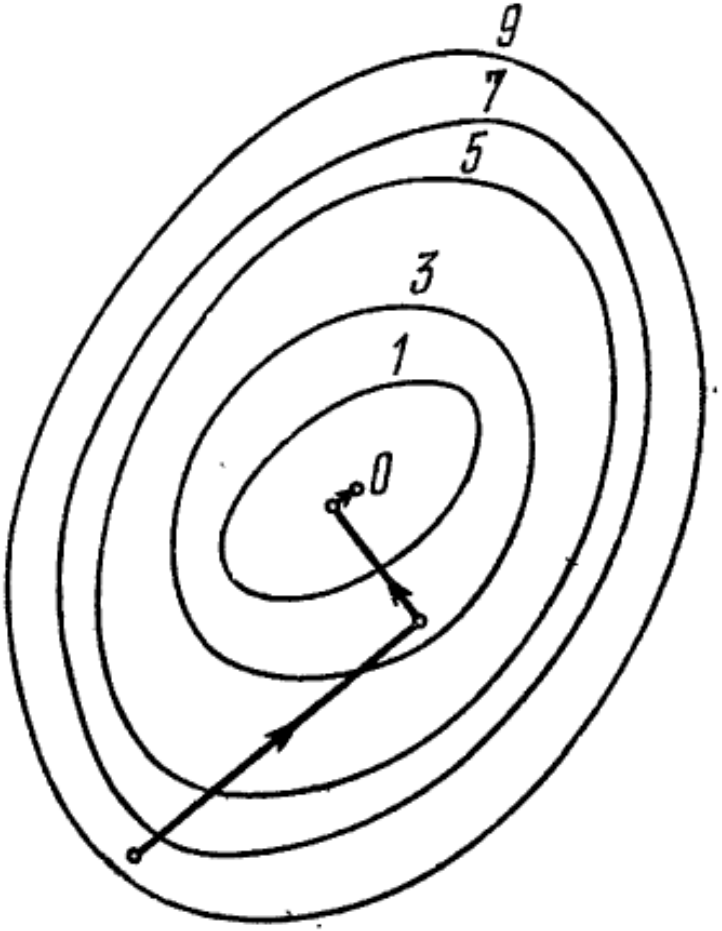
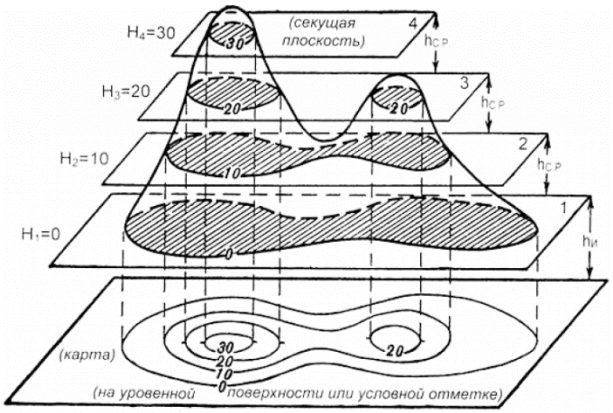


Рис. 36. Поиск наименьшего значения функции методом наибыстрейшего спуска.

Локальная оптимизация



Два примера случайной инициализации параметров (***w***,*t*)

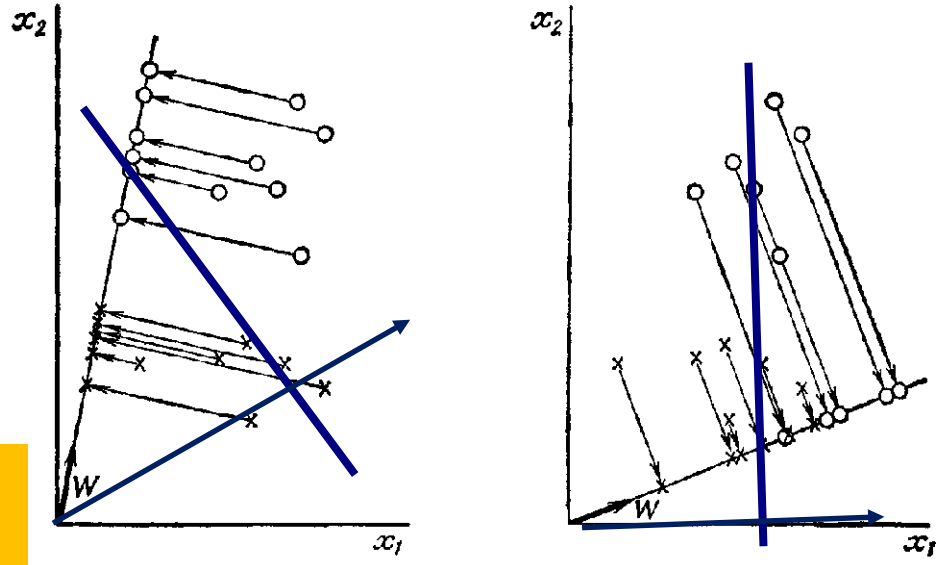


Рис. 4.6. Проекция выборок на прямую.

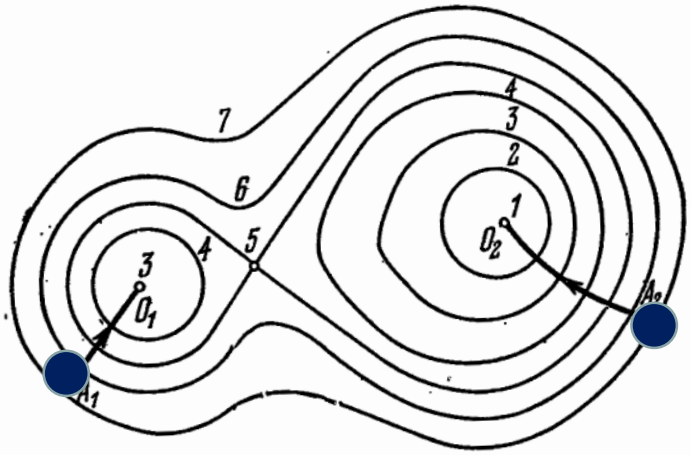
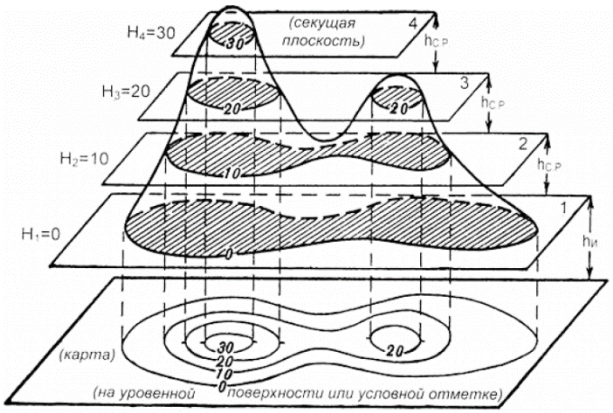


Рис. 38. Пример функции с двумя «локальными» минимумами в точках O_1 и O_2.

Пошаговая градиентная оптимизация функции потерь при различных начальных значениях параметров (инициализация) приводит к различным локальным оптимумам решающего правила

Локальная оптимизация



Изменение параметров (***w***,*t*) после нескольких шагов градиентного спуска.

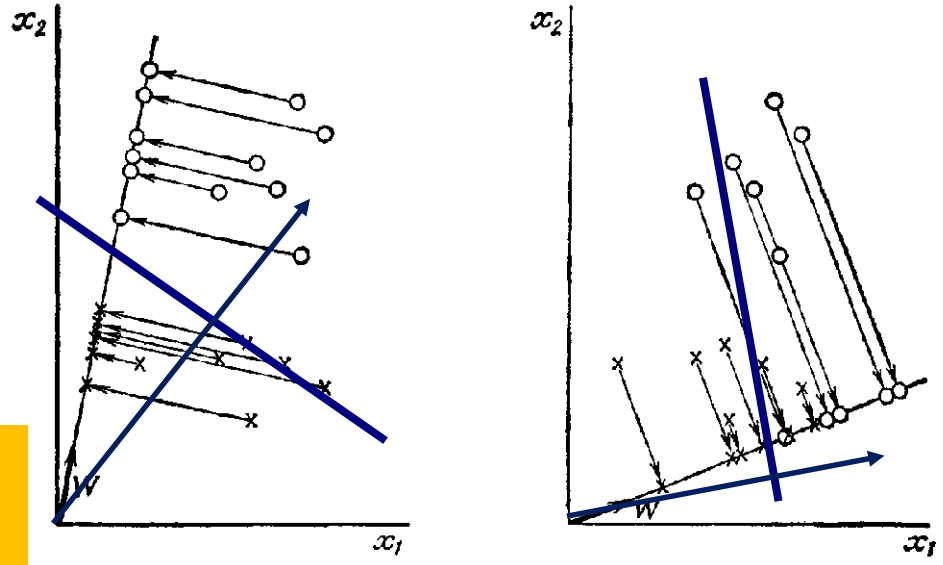


Рис. 4.6. Проекция выборок на прямую.

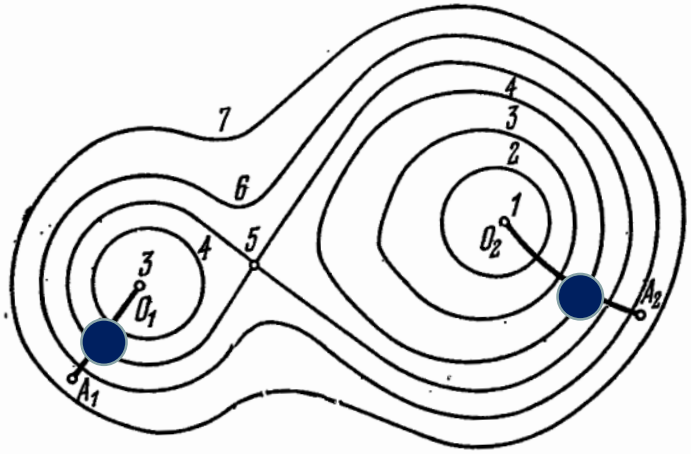
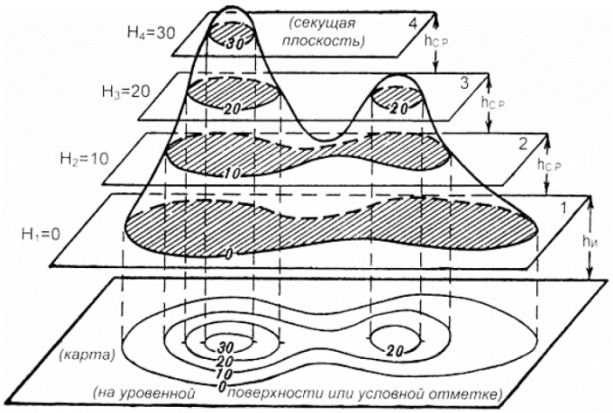


Рис. 38. Пример функции с двумя «локальными» минимумами в точках O_1 и O_2.

Пошаговая градиентная оптимизация функции потерь при различных начальных значениях параметров (инициализация) приводит к различным локальным оптимумам решающего правила

Локальная оптимизация



Окончательные значения параметров (***w***,*t*) после завершения градиентного спуска.

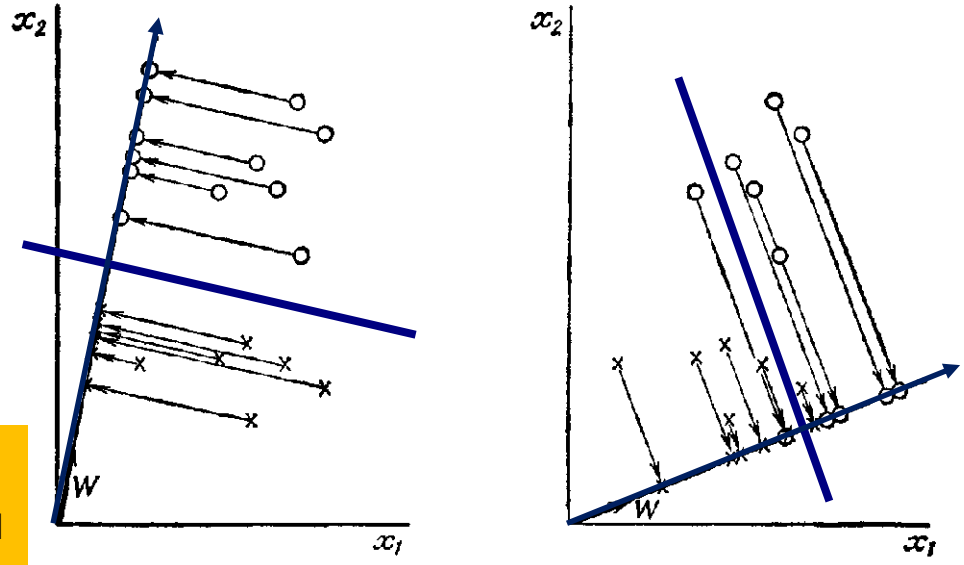


Рис. 4.6. Проекция выборок на прямую.

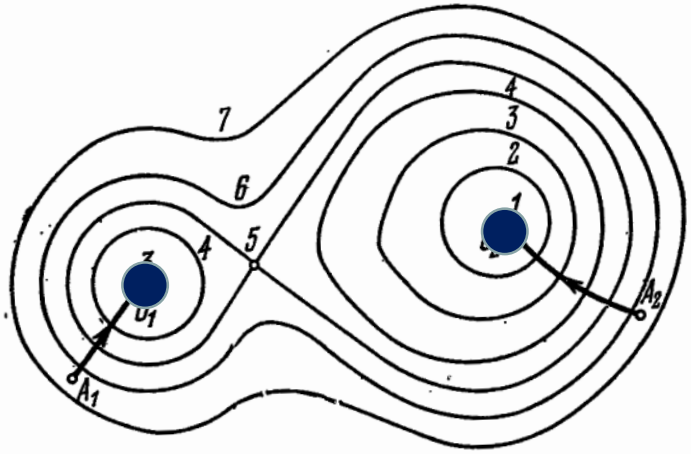


Рис. 38. Пример функции с двумя «локальными» минимумами в точках O_1 и O_2.

Пошаговая градиентная оптимизация функции потерь при различных начальных значениях параметров (инициализация) приводит к различным локальным оптимумам решающего правила

Локальная оптимизация

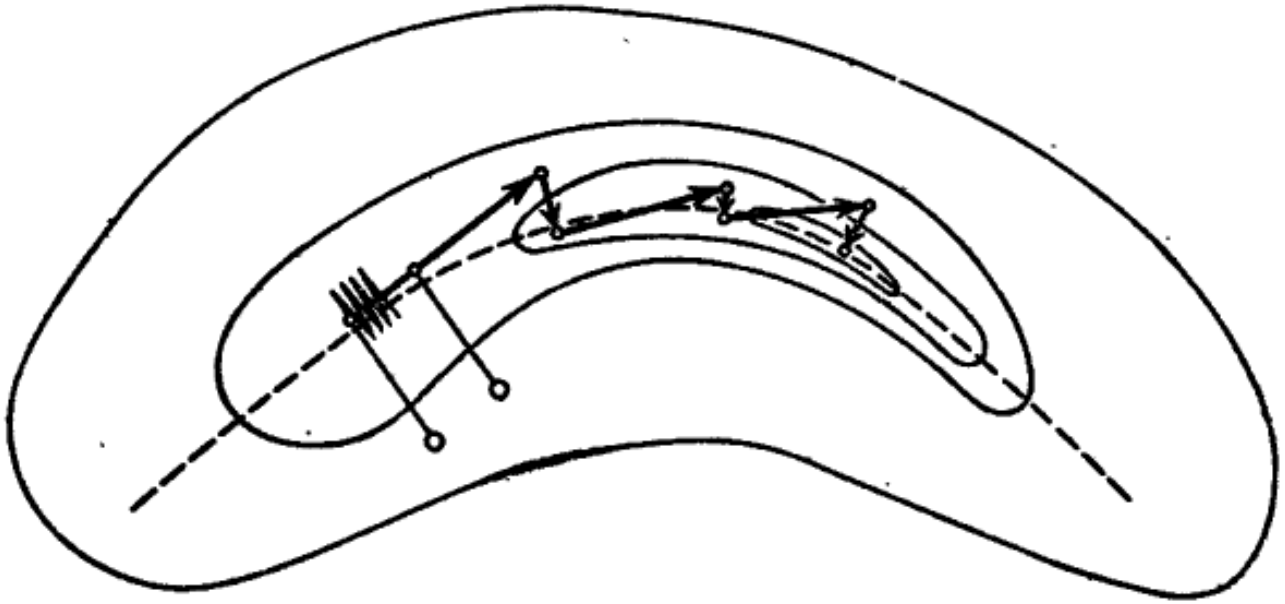


Рис. 37. Поиск наименьшего значения функции в случае «оврага».

В подобных случаях нелегко подобрать правильную скорость (шаг) градиентного спуска

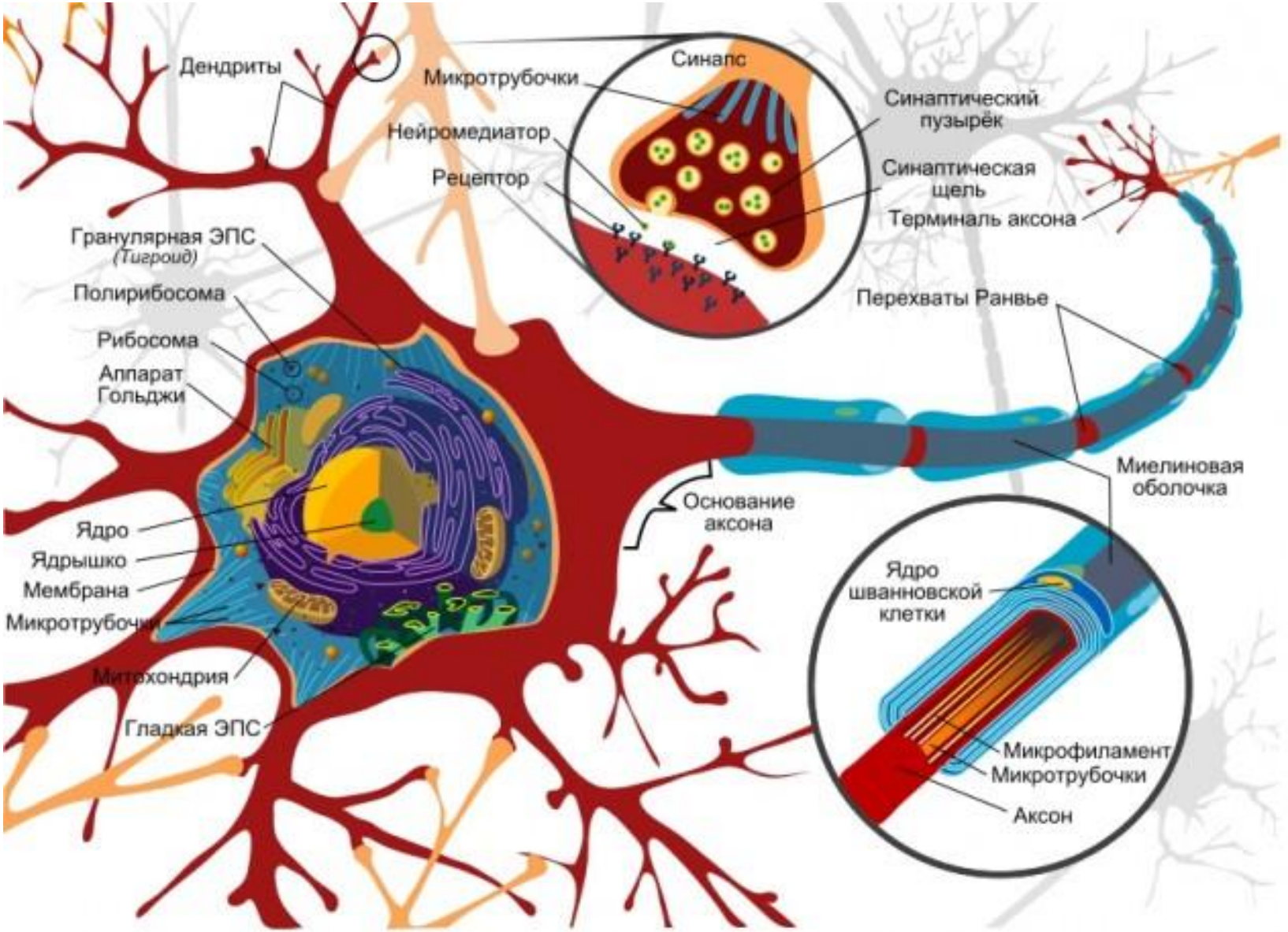
Когда мы мучимся с подбором параметров обучения, это мы на самом деле боремся с характером функции потерь, попадая во всякие овраги и т.п.

Нейронные сети

Краткое введение/напоминание

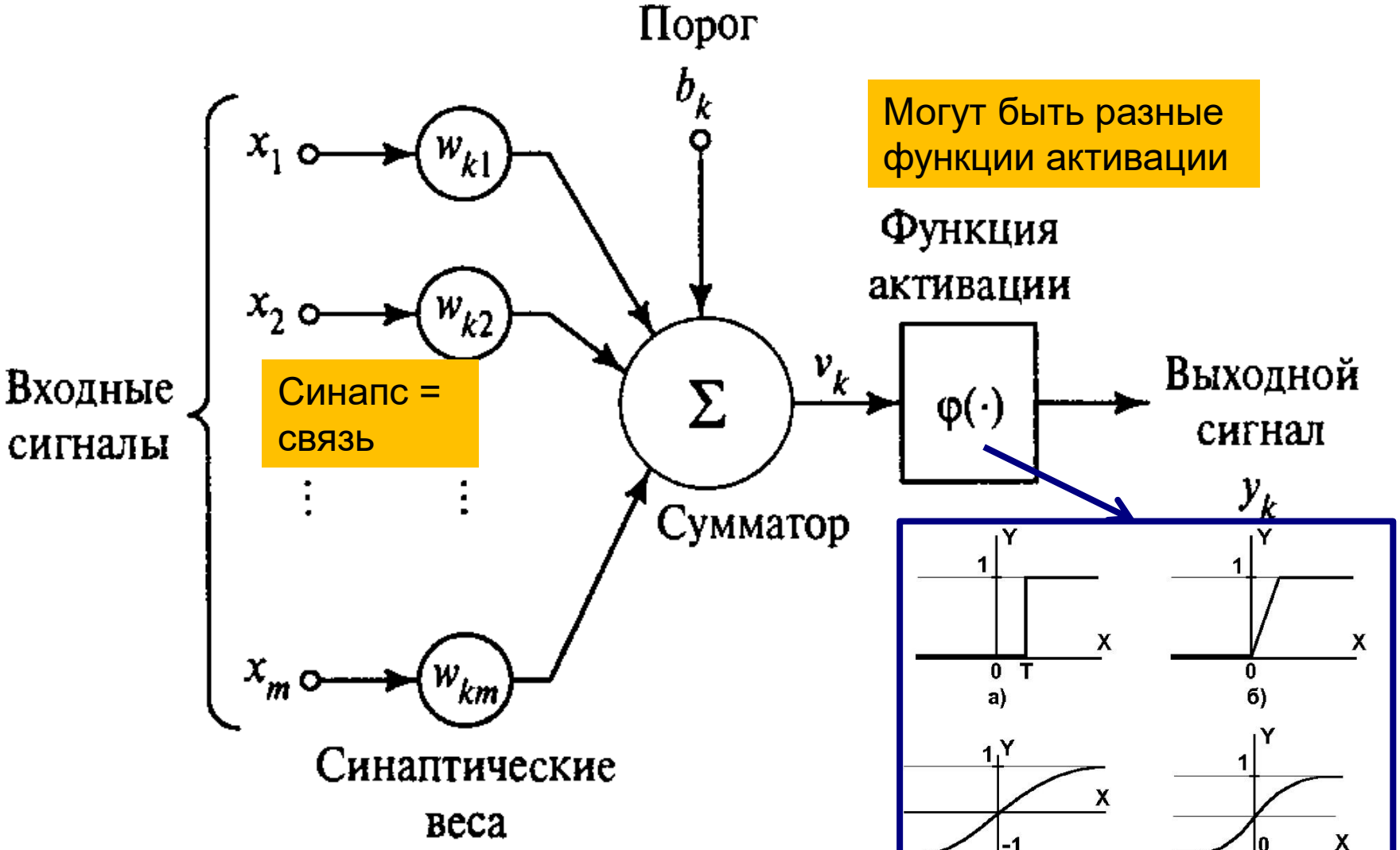
Нейроны: идея из биологии

Настоящие нейроны и нейронные сети в нервной системе



Нейроны и нейронные сети

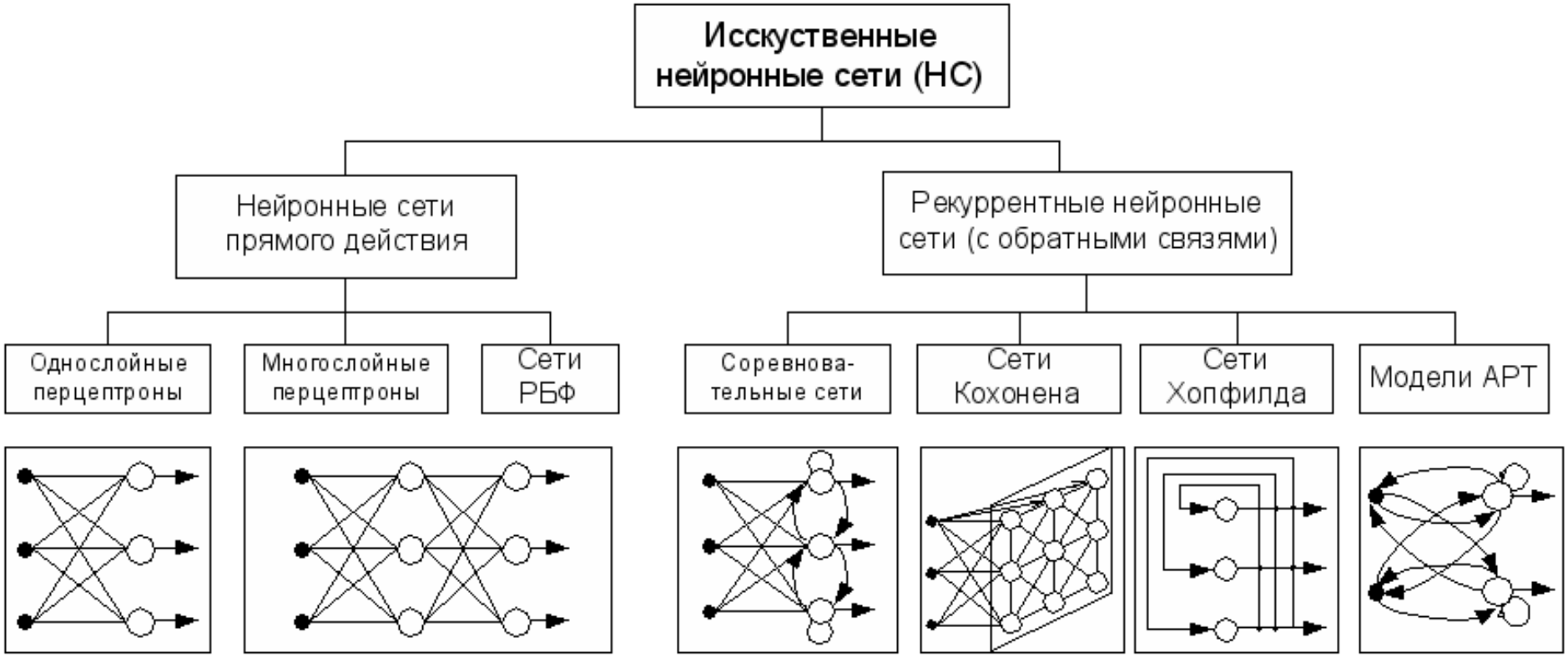
Модель искусственного нейрона – сумматор с нелинейностью



Корреляция выхода и входа × значимость!

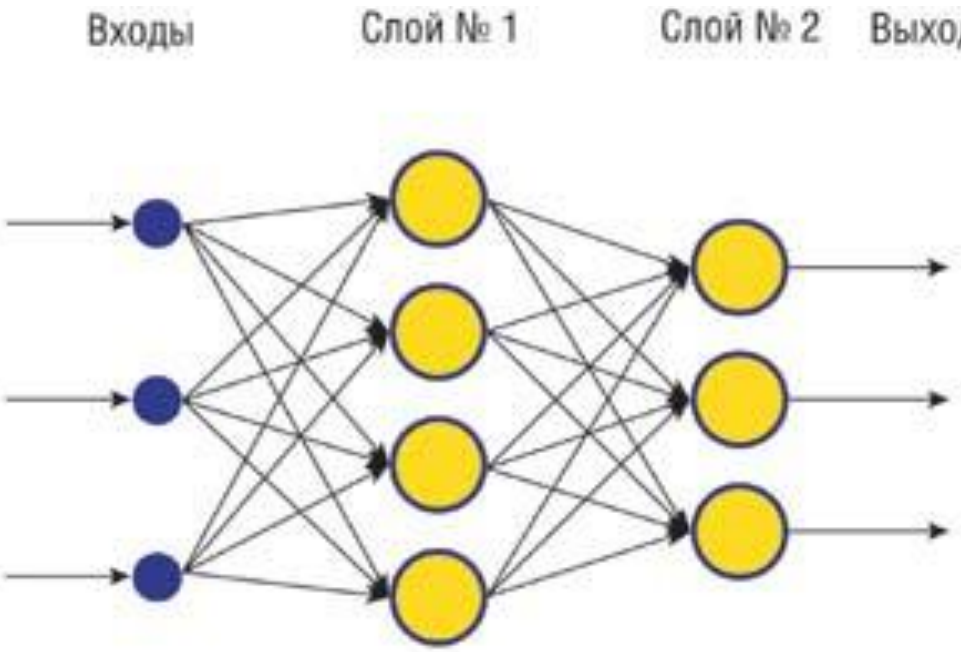
Нейроны и нейронные сети

Нейронные сети могут иметь различную архитектуру



Нейроны и нейронные сети

Нейронная сеть – группа связанных слоев нейронов



Входы нейронов слоя $i+1$ являются выходами нейронов слоя i

Математическая модель нейрона

$$x_{(k)}^{(i+1)} = f\left(\sum_{j=1}^{N} w_j^{(k)} x_j^{(i)}\right)$$

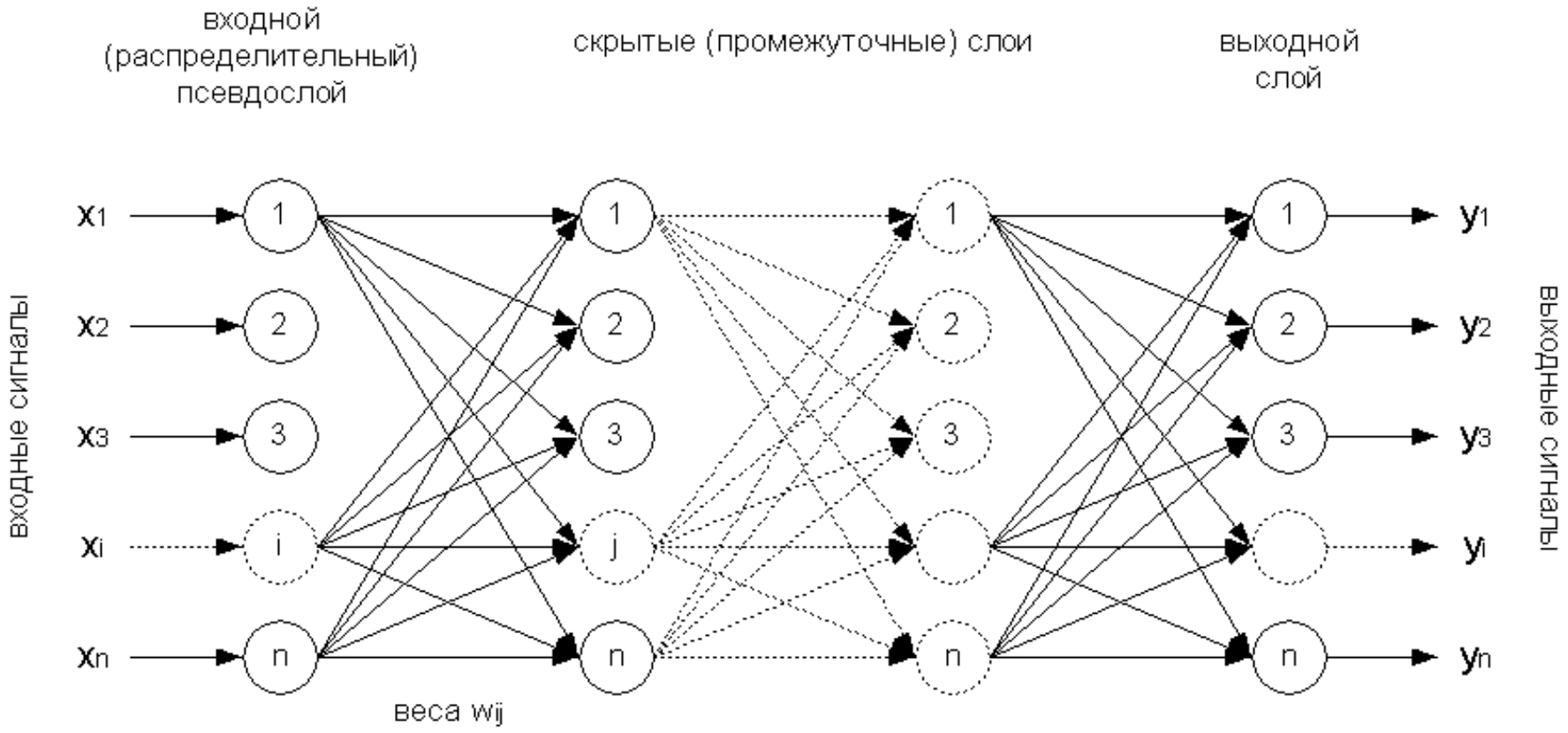
Выход k-го нейрона слоя $i+1$ рассчитывается как взвешенная сумма всех его входов со слоя i, к которой применена функция активации, нормализующая выходной сигнал

Персептрон – это нейронная сеть такой архитектуры, в которой все нейроны разделены на слои (уровни, layers) таким образом, что на вход нейронов каждого следующего слоя подаются выходы нейронов предыдущего слоя.

Нельзя возвращать информацию на нижние слои с верхних и нельзя перепрыгивать (связывать нейроны синапсами) через слои.

Нейроны и нейронные сети

Многослойный персептрон

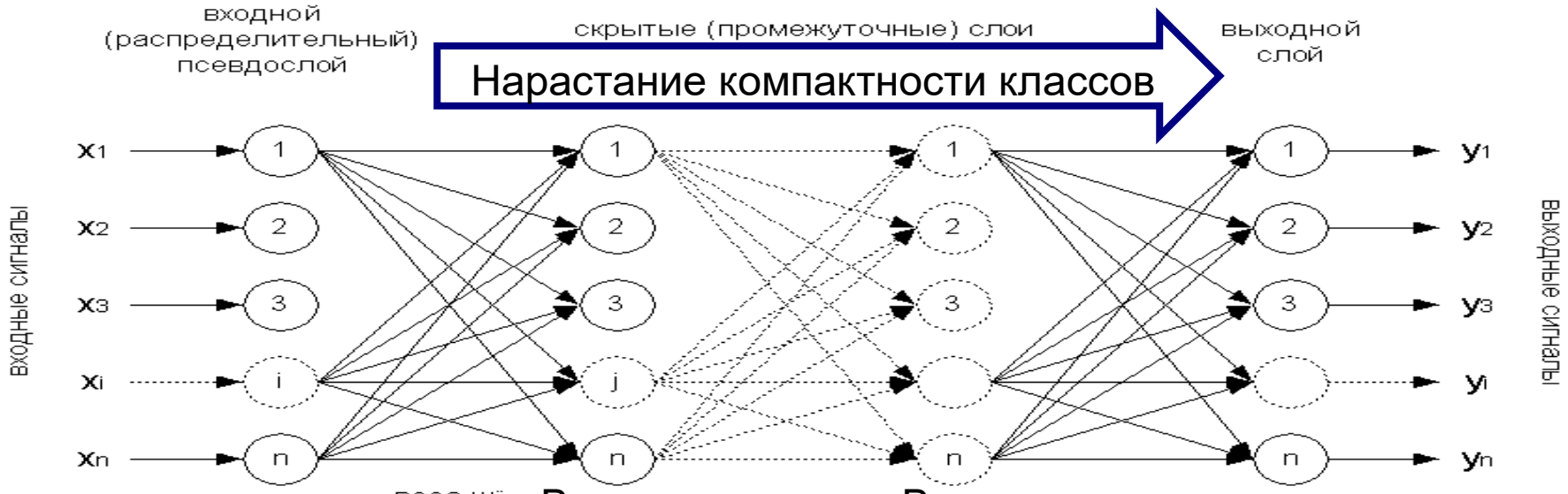


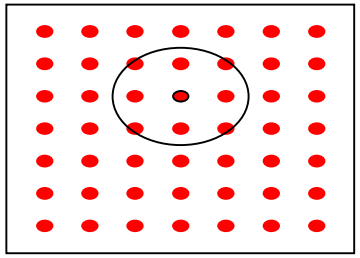
Зачем нам много слоев нейронов?

Зачем нелинейности между слоями? Затем, чтобы многослойный персептрон нельзя было заменить эксвивалентным однослойным, т.е. чтобы с ростом слоев росла сложность классификатора. **А зачем увеличивать сложность?**....

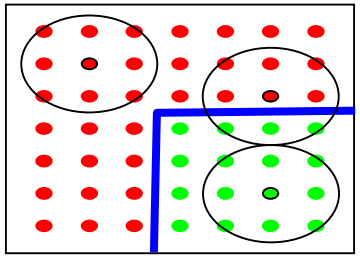
Нейроны и нейронные сети

Многослойный персептрон

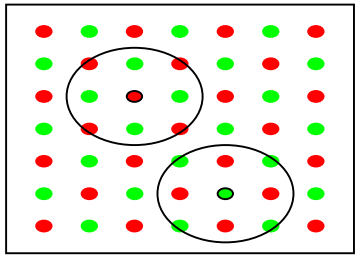




компактный класс



локально компактные классы

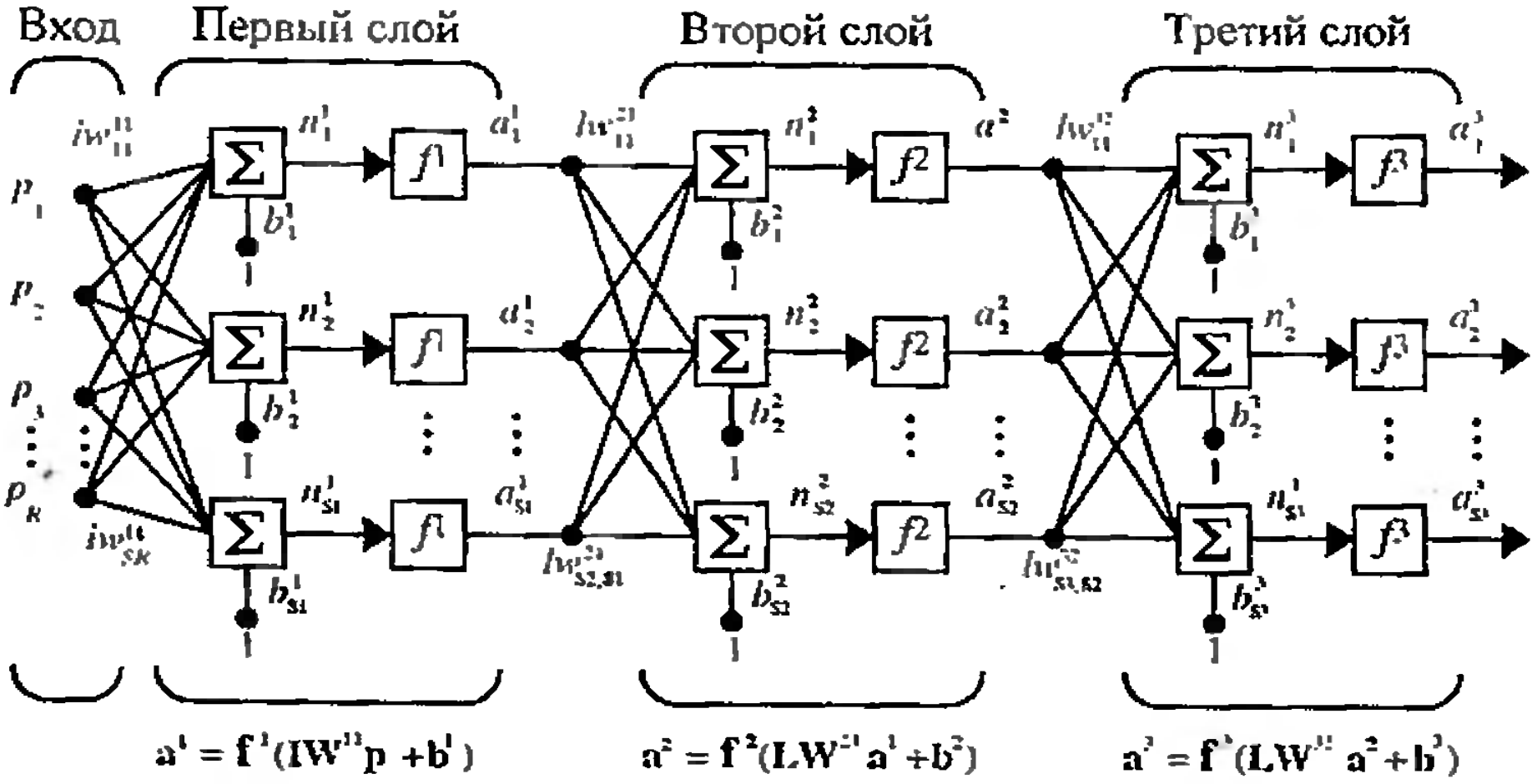


некомпактные классы

От слоя к слою происходит постепенное построение *хорошего для данной задачи распознавания* пространства признаков. Персептрон сам формирует такие выходные признаки, по которым классы легко разделяются линейными классификаторами.

Нейроны и нейронные сети

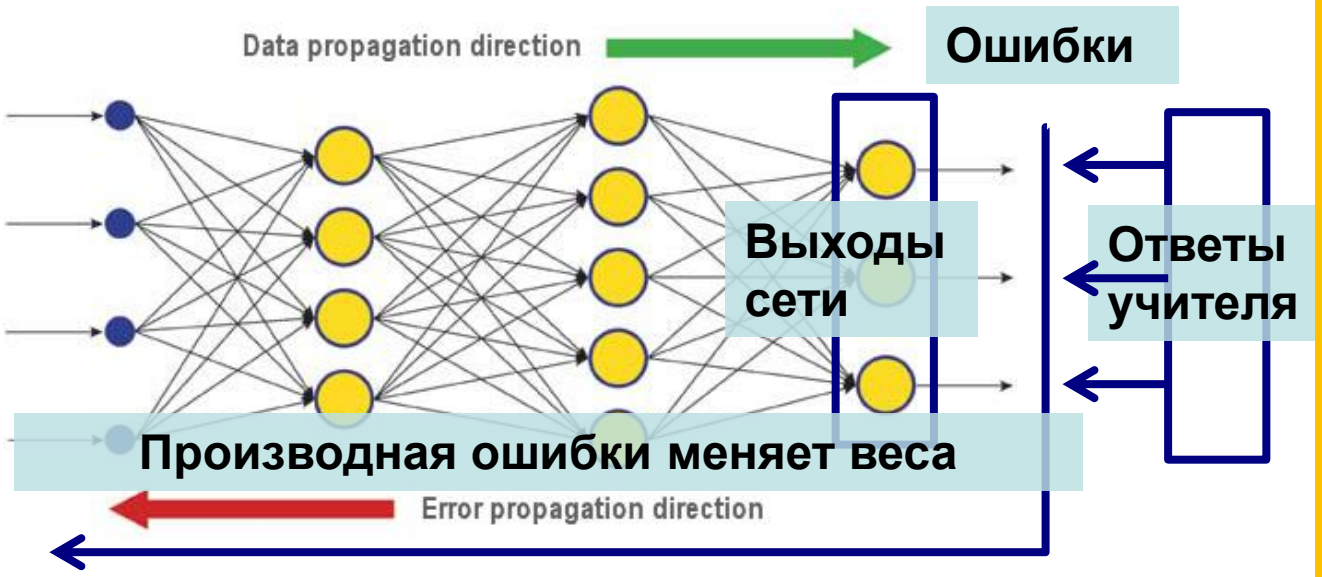
Многослойный персептрон как сложная функция



От слоя к слою происходит постепенное построение *хорошего для данной задачи распознавания* пространства признаков. Персептрон сам формирует такие выходные признаки, по которым классы легко разделяются линейными классификаторами.

Нейроны и нейронные сети

Многослойный персептрон: обучение коэффициентов обратным распространением ошибки



За счет back propagation для каждого нейрона формируется свой Loss как часть общего Loss сети. Для всех нейронов одновременно выполняется шаг градиентного спуска (см. пример выше). Так учится вся сеть

Сложная функция: $y = g(f(x))$.

Правило нахождения производной сложной функции

$$g'(f(x)) = g'(f) \cdot f'(x)$$

(производная сложной функции равна производной основной функции на производную внутренней функции)

Цифровые изображения. Линейная фильтрация изображений. Выделение признаков. Обнаружение и распознавание объектов на основе выделенных признаков

Краткий экскурс в классическое компьютерное зрение

Цифровые изображения

Изображение как двумерный массив данных

Видимое поле представляет собой лишь некоторую *функцию распределения яркости* или *цвета* на двумерной плоскости: *f(x,y)*, где *x* и *y* – декартовы координаты, описывающие плоскость изображения.

Цифровое изображение представляет собой двумерную матрицу Im[x,y] размера (DimX × DimY), где x – целое число от 0 до DimX – 1, описывающее номер элемента в строке матрицы, y – целое число от 0 до DimY – 1, описывающее номер строки матрицы, в которой расположен данный элемент.

Пиксель (pixel, picture element) – элемент цифрового изображения (ячейка прямоугольной матрицы). В простейшем случае каждый пиксель Im[x,y] имеет скалярное целочисленное значение, пропорциональное значению функции распределения яркости *f(x,y)* в данной точке плоскости.

174	190	60	31	41	17	26	18	20	28	15	21	17	16	24	38	49	22	14	13	20	17	76
189	59	41	26	35	18	24	17	16	17	19	15	18	19	42	119	132	76	28	12	15	14	36
77	48	52	34	17	11	23	34	34	16	29	11	14	61	60	158	194	162	141	38	14	45	29
61	53	32	27	23	15	27	63	58	31	67	17	16	90	27	137	191	183	180	121	35	56	74
81	72	49	48	29	9	31	62	94	38	58	56	67	40	39	191	196	202	196	167	75	35	91
93	92	83	78	66	15	16	56	85	86	35	27	28	50	146	198	198	198	185	156	98	55	90
89	98	96	104	85	45	36	64	84	93	94	70	91	151	184	190	195	194	186	121	112	117	70
93	102	116	108	110	93	71	67	37	86	107	123	129	146	160	174	182	167	141	126	109	84	79

Цифровое изображение как двумерная матрица интенсивностей

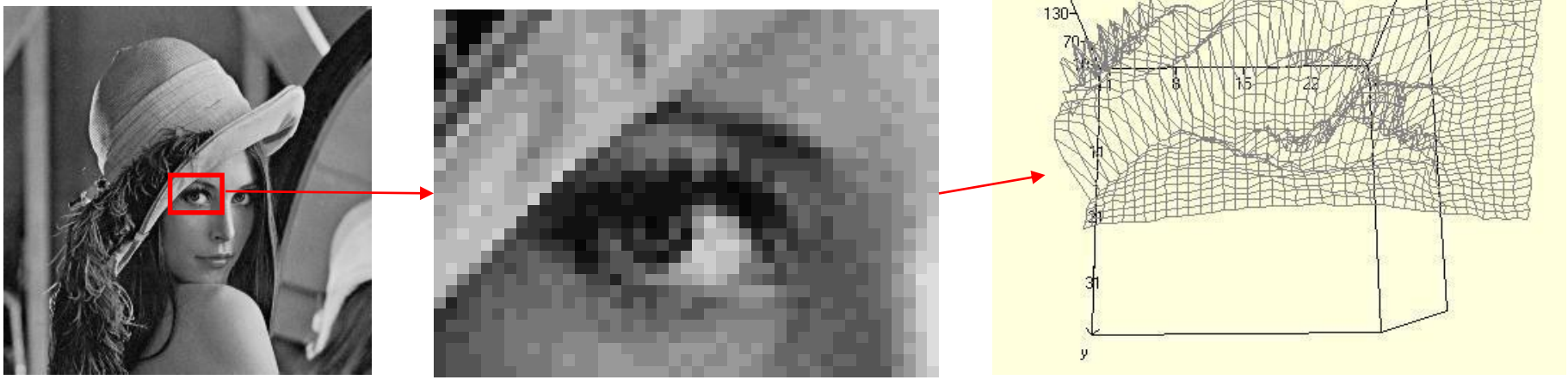
Цифровые изображения

Изображение как двумерный массив данных

Видимое поле представляет собой лишь некоторую *функцию распределения яркости* или *цвета* на двумерной плоскости: *f(x,y)*, где *x* и *y* – декартовы координаты, описывающие плоскость изображения.

Цифровое изображение представляет собой двумерную матрицу Im[x,y] размера (DimX × DimY), где x – целое число от 0 до DimX – 1, описывающее номер элемента в строке матрицы, y – целое число от 0 до DimY – 1, описывающее номер строки матрицы, в которой расположен данный элемент.

Пиксель (pixel, picture element) – элемент цифрового изображения (ячейка прямоугольной матрицы). В простейшем случае каждый пиксель Im[x,y] имеет скалярное целочисленное значение, пропорциональное значению функции распределения яркости *f(x,y)* в данной точке плоскости.

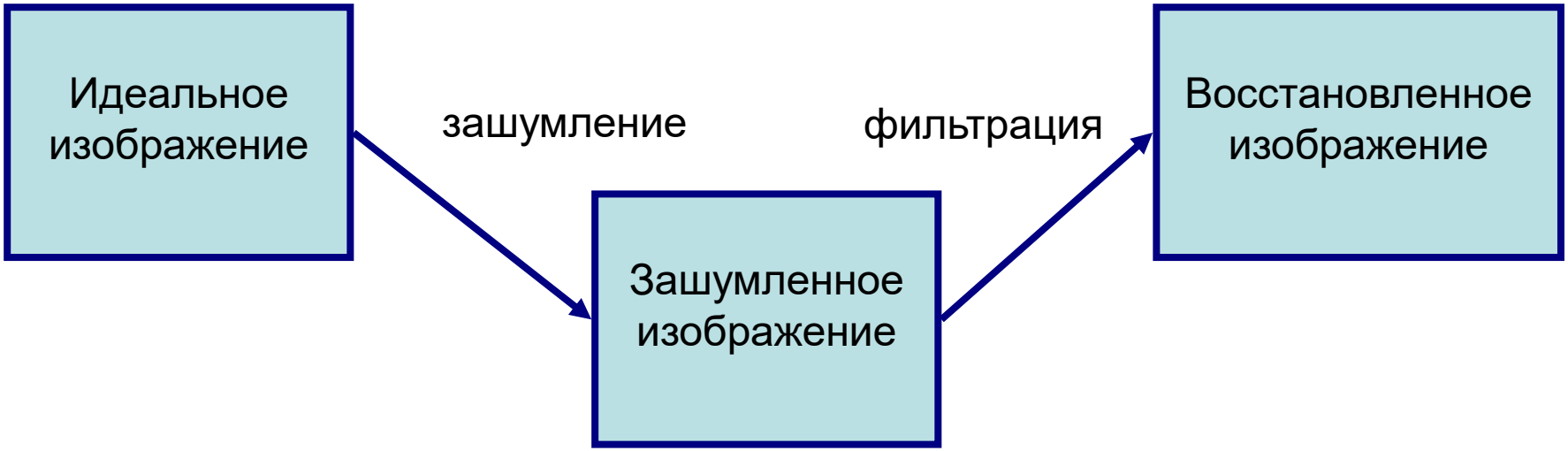


Цифровое изображение как псевдо-трехмерный рельеф

Задача фильтрации изображений

Фильтрация изображений в широком смысле – любые процедуры обработки изображений, при которых **на вход процедуры подается (одно) растровое изображение, и на выходе также формируется растровое изображение**. Такие процедуры (один растровый вход, один растровый выход) называют *фильтрами*.

Фильтрация в узком смысле – *помеховая фильтрация* или **фильтрация изображений от «шума»**.



Идеальный фильтр в точности восстанавливает исходное изображение. Проблема в том, что все фильтры неидеальны.

Оконная фильтрация изображений в пространственной области

Основная идея помеховой фильтрации изображений заключается в том, что для оценки исходного значения каждого пикселя изображения используется не только значение самого данного пикселя, но и значения еще нескольких близких к нему пикселей, попадающих в так называемое «*окно*» или *апертуру* фильтра. При этом «близость» пикселей к оцениваемому понимается в буквальном геометрическом смысле.

Прямоугольные окна (апертуры) фильтрации определяются условием «*все пиксели данного окна отстоят от тестируемого центрального пикселя на более чем на WinX/2 по горизонатали и WinY/2 по вертикали*», где WinX и WinY – горизонтальный и вертикальный размер окна фильтрации соответственно.

Оконная фильтрация изображений в пространственной области

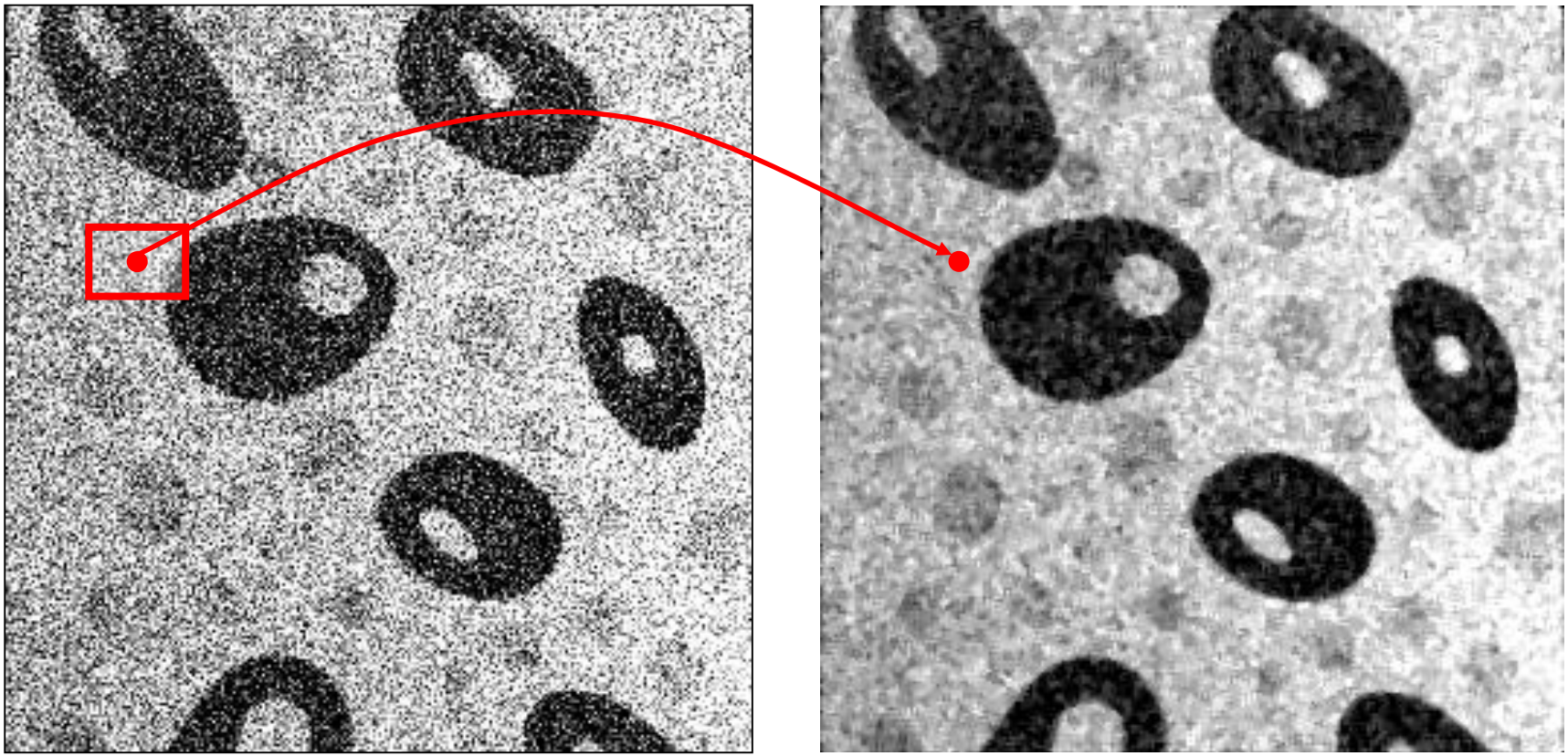
Окно фильтрации последовательно движется по *входному изображению* (например, сверху вниз по строкам, слева направо в каждой строке), при этом в каждом положении окна происходит анализ всех пикселей, принадлежащих в данный момент окну, и на основе такого анализа центральному пикселю окна на *выходном изображении* присваивается то или иное финальное значение.

Сформированное таким образом выходное изображение также называется *результатом фильтрации*.

Процедуры оконной фильтрации могут различаться:

- типом собираемых в окне локальных статистик;
- способом принятия решения на основе собранных статистик.
- размером и формой окна (апертуры);

Оконная фильтрация изображений в пространственной области



Входное изображение

Выходное изображение

Линейная оконная фильтрация изображений в пространственной области

Линейная оконная фильтрация изображений в пространственной области заключается в вычислении *линейной комбинации* значений яркости пикселей в окне фильтрации с коэффициентами матрицы весов фильтра, называемой также *маской* или *ядром* линейного фильтра:

$$Im'[x,y] = \Sigma_i \Sigma_j Im[x+i,y+j] \times Mask[i,j],$$

$i = -hWinX \ldots hWinX,$

$j = -hWinY \ldots hWinY,$

где $hWinX = [WinX/2]$, $hWinY = [WinY/2]$ – полуширина и полувысота окна фильтрации соответственно.

Результат применения данной операции ко всем пикселям изображения Im(x,y) называется *сверткой* изображения Im с маской Mask.

Линейная фильтрация изображений в пространственной области

В случае маски размера 3×3:

$$Im'[x,y] = \Sigma_i \Sigma_j\, Im[x+i,y+j] \times Mask[i,j],$$

i = –1 ... 1, j = –1 … 1.

При этом маска фильтра представляется матрицей вида:

Mask[–1,–1]	Mask[0,–1]	Mask[1,–1]
Mask[–1,0]	Mask[0,0]	Mask[1,0]
Mask[–1,1]	Mask[0,1]	Mask[1,1],

а фрагмент изображения с центральным пикселем Im(x,y) имеет вид:

Im[x–1,y–1]	Im[x,y–1]	Im[x+1,y–1]
Im[x–1,y]	Im[x,y]	Im[x+1,y]
Im[x–1,y+1]	Im[x,y+1]	Im[x+1,y+1].

Среднее в окне

Простейшим видом линейной оконной фильтрации в пространственной области является *среднее* в окне. Результатом такой фильтрации является значение математического ожидания, вычисленное по всем пикселям окна. Математически это эквивалентно свертке с маской, все элементы которой равны *1/n*, где *n* – число элементов маски. Например, маска среднего в окне размера 3×3 имеет вид:

$$\frac{1}{9} \times \begin{vmatrix} 1 & 1 & 1 \\ 1 & 1 & 1 \\ 1 & 1 & 1 \end{vmatrix}$$

Маска сглаживающего фильтра «среднее в окне»

Внимание! *Сглаживающие* или *фильтрующие* маски линейных фильтров должны иметь сумму всех элементов равную 1. Данное *условие нормировки* гарантирует адекватный отклик фильтра на постоянный сигнал (изображение Im[x,y] = const).

Гауссовская фильтрация

Повысить устойчивость результатов фильтрации на краях областей можно, если придать более близким точкам окрестности большее влияние на окончательный результат, чем дальним:

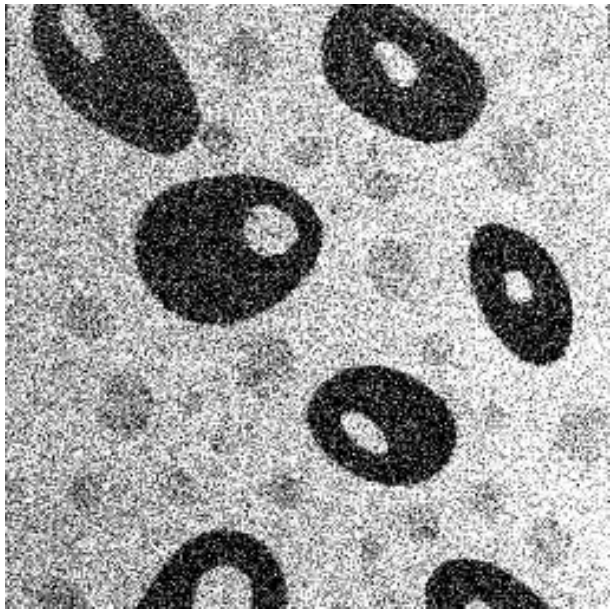
$$\frac{1}{16} \times \begin{vmatrix} 1 & 2 & 1 \\ 2 & 4 & 2 \\ 1 & 2 & 1 \end{vmatrix}$$

Маска Гауссового сглаживающего фильтра

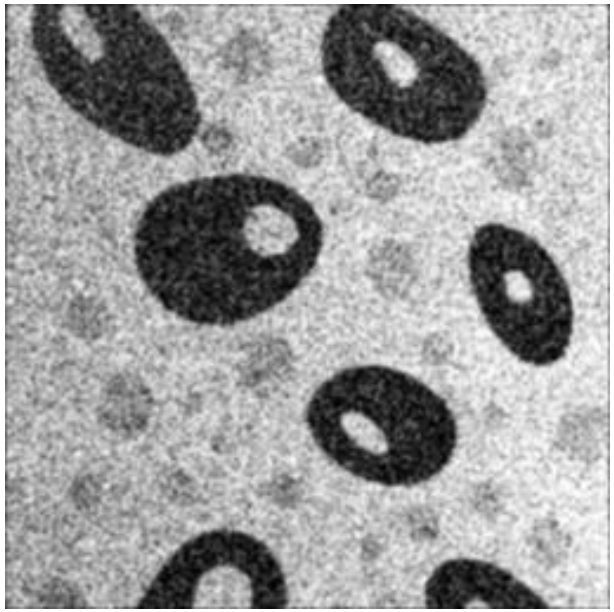
Такая маска называется Гауссовой, соответственно и использующий ее линейный фильтр также называется *гауссовым*. Можно определить гауссовы маски любого размера, используя дискретные приближения двумерного гауссова распределения.

Внимание! *Сглаживающие* или *фильтрующие* маски линейных фильтров должны иметь сумму всех элементов равную 1. Данное *условие нормировки* гарантирует адекватный отклик фильтра на постоянный сигнал (изображение Im[x,y] = const).

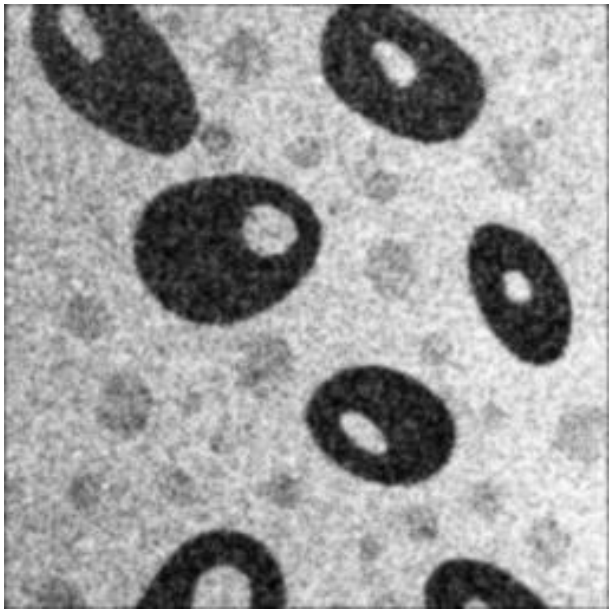
Гауссовская фильтрация



Зашумленное изображение



Результат гауссовой линейной фильтрации gauss 3×3

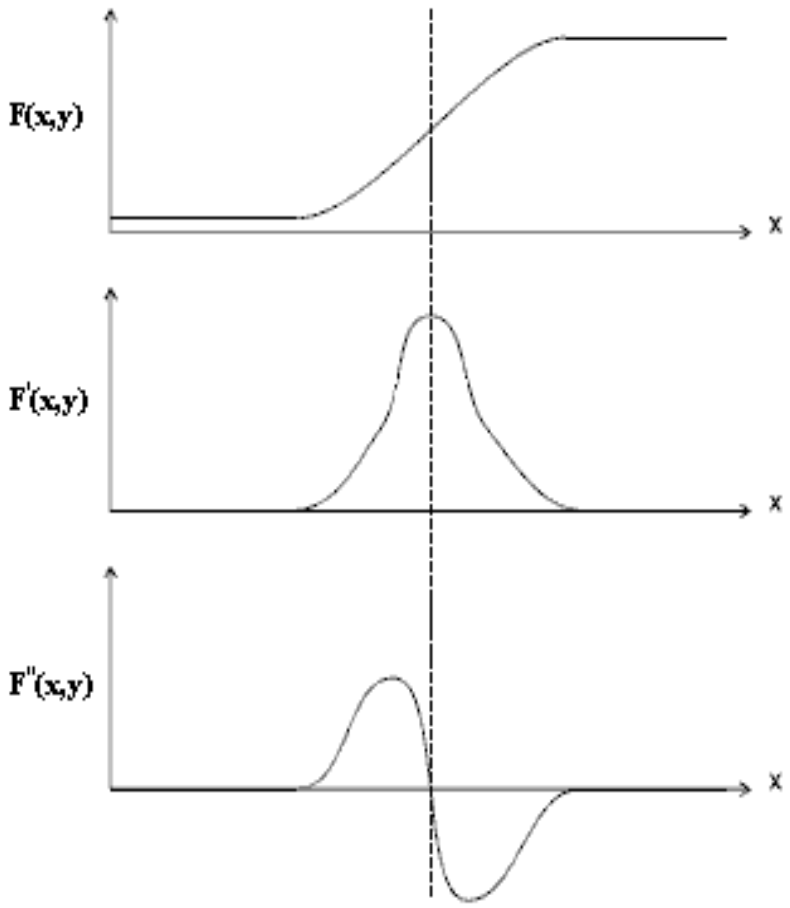


Результат гауссовой линейной фильтрации gauss 5×5

Выделение контуров на полутоновых изображениях

Контурные операторы (фильтры) не сглаживают изображение, а формируют в каждой его точке *признак* наличия контура

Задача выделения контуров



Традиционно рассматриваются две модели "края": "ступенька" и "излом". "Ступенька" предполагает скачкообразное изменение яркости вдоль некоторого контура на изображении. Край типа "излом" - это совокупность точек разрыва первой производной функции *f(x,y)*. «Ступенчатым" краевым точкам соответствуют точки смены знака второй производной (максимума первой производной), а "изломным" краевым точкам - точки смены знака первой производной (локальные максимумы яркостной функции).

Идея определения краевых перепадов интенсивности типа «ступенька»: а) функция интенсивности на границе перепада; б) первая производная функции; в) вторая производная функции.

Операторы вычисления векторов градиентов

Операторы Робертса, Превитта и Собела непосредственно вычисляют значения компонент вектора-градиента для каждой точки изображения путем *свертки* локальной окрестности точки с малоразмерными масками:

$$M_1 = \begin{vmatrix} 1 & 0 \\ 0 & -1 \end{vmatrix} \qquad M_2 = \begin{vmatrix} 0 & -1 \\ 1 & 0 \end{vmatrix}$$

для оператора Робертса, и

$$M_x = \begin{vmatrix} 1 & 0 & -1 \\ 2 & 0 & -2 \\ 1 & 0 & -1 \end{vmatrix} \qquad M_y = \begin{vmatrix} 1 & 2 & 1 \\ 0 & 0 & 0 \\ -1 & -2 & -1 \end{vmatrix}$$

для оператора Собела.

Маски фильтров для вычисления признаков-производных по направлению

Внимание! Дифференцирующие или *признаковые* фильтры должны иметь сумму всех элементов равную 0. Данное *условие нормировки* гарантирует адекватный *нулевой* отклик фильтра на неинформативный сигнал (изображение Im[x,y] = const).

Операторы вычисления векторов градиентов

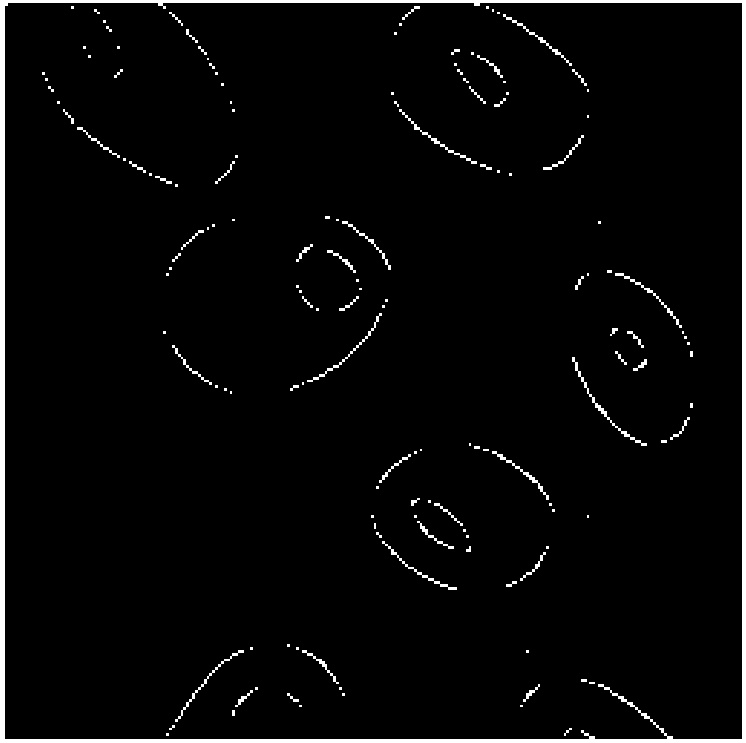
Оператор Превитта выглядит аналогичным образом:

$$M_x = \begin{matrix} 1 & 1 & 1 \\ 0 & 0 & 0 \\ -1 & -1 & -1 \end{matrix} \qquad M_y = \begin{matrix} -1 & 0 & 1 \\ -1 & 0 & 1 \\ -1 & 0 & 1 \end{matrix}$$

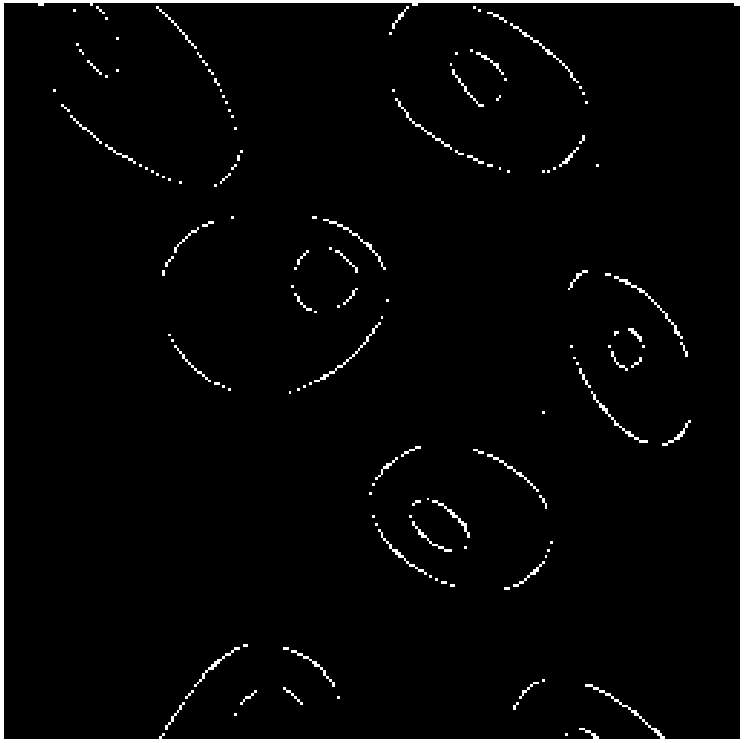
Маски фильтров для вычисления признаков-производных по направлению

Практические исследования показывают, что оператор Робертса не является в достаточной мере помехозащищенным, оператор же Собела обеспечивает вполне удовлетворительные результаты при обработке реальных изображений.

Операторы вычисления векторов градиентов



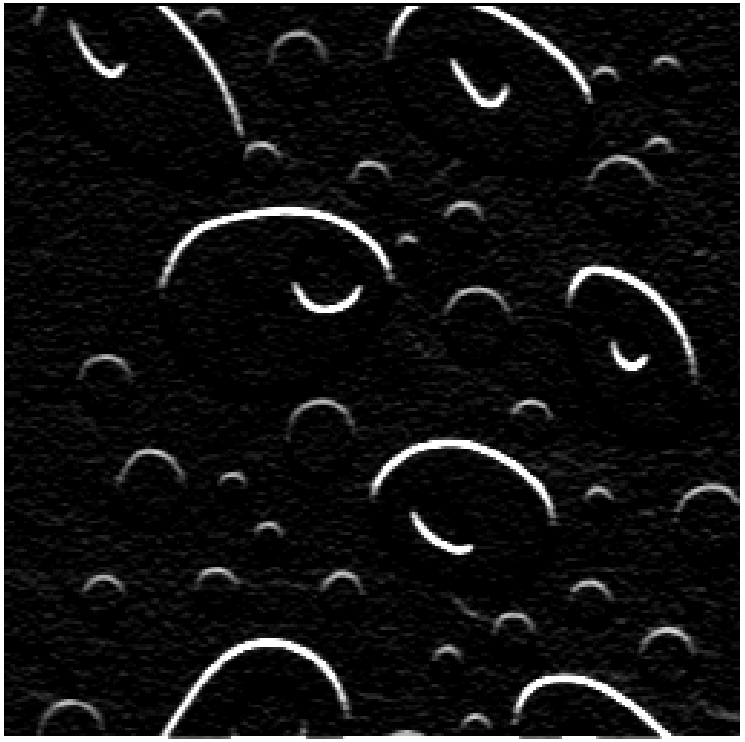
(а) Оператор Робертса: результат свертки с маской M1



(б) Оператор Робертса: результат свертки с маской M2

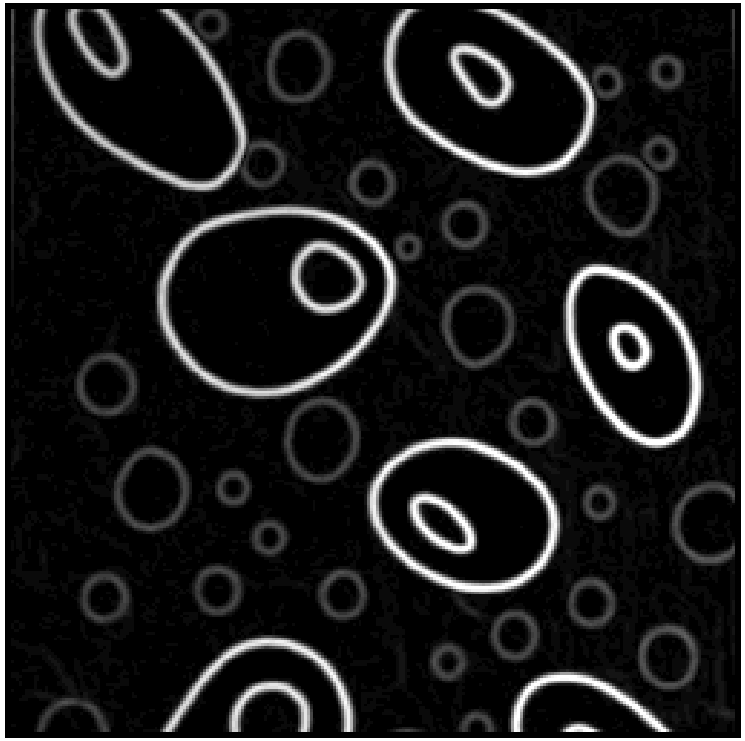
Операторы вычисления векторов градиентов

(в) Оператор Собела:
вертикальные контура (маска Mx)



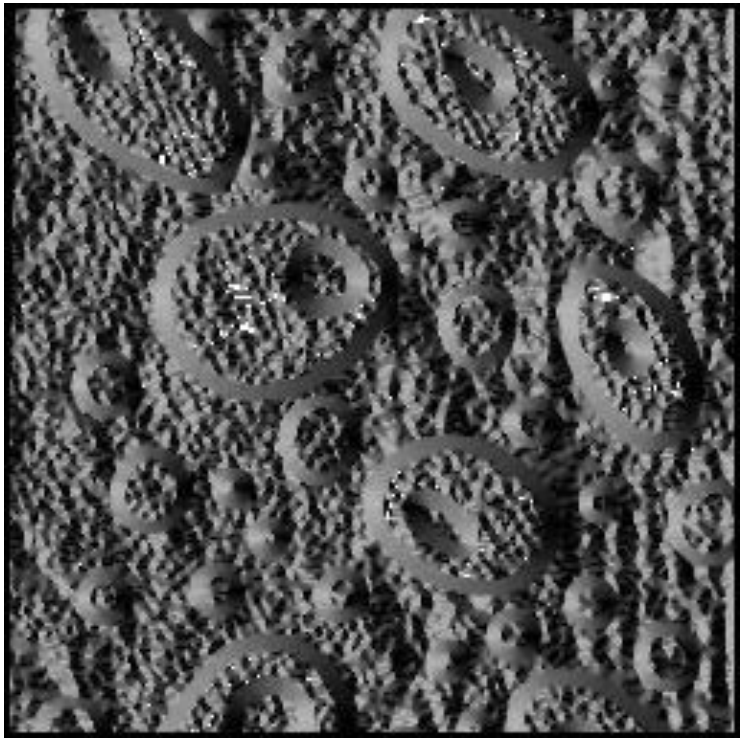
(г) Оператор Собела:
горизонтальные контура (маска My)

Операторы вычисления векторов градиентов



(д) Оператор Собела: амплитуда градиента

$$A = \sqrt{\left(g_x^2 + g_y^2\right)}$$



(е) Оператор Собела: поле направлений градиента

$\varphi = arctg(g_y / g_x)$

Оператор Лапласа

Рассмотрим операторы выделения краев, основанные на вычислении симметричных круговых производных.
Простейшим оператором такого рода является *оператор Лапласа*.
Оператор Лапласа (*Лапласиан*) 3x3 имеет маску следующего вида:

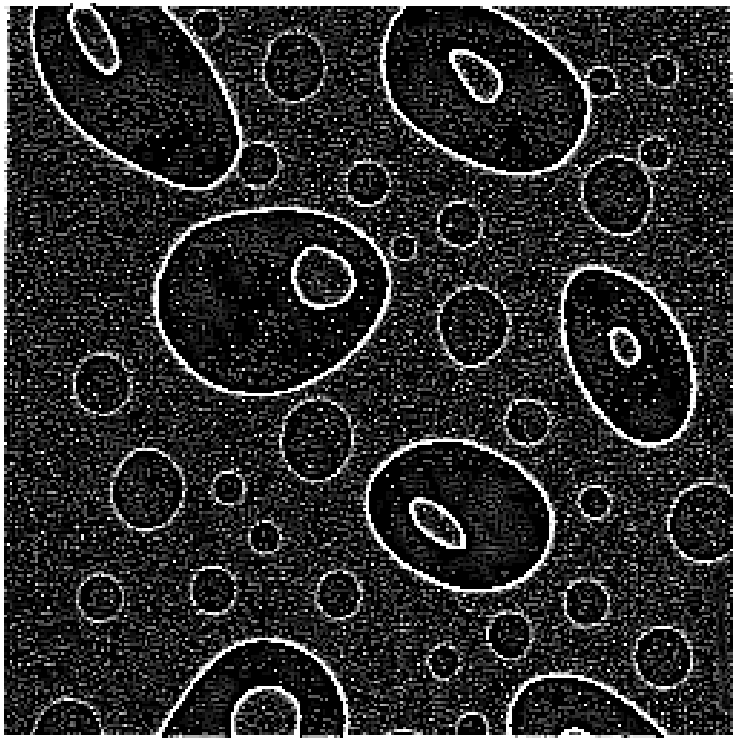
$$\begin{matrix} -1 & -1 & -1 \\ -1 & 8 & -1 \\ -1 & -1 & -1 \end{matrix}$$

Маска фильтра для вычисления круговой производной

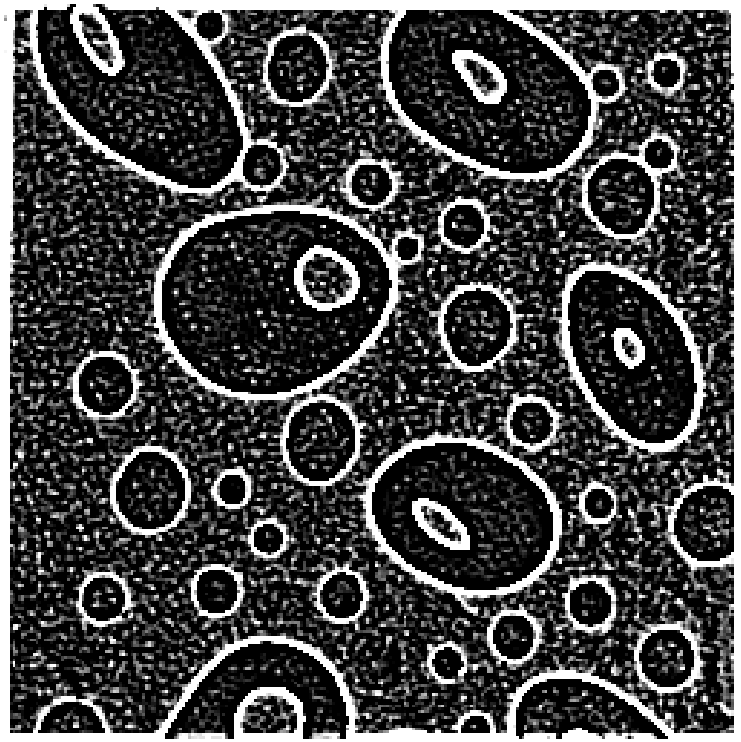
Такую маску можно интерпретировать как сумму разностей центрального элемента с каждым из 8 его ближайших соседей. Таким образом, в равной степени учитываются возможные перепады яркости во всех направлениях.

Оператор Лапласа

Пример работы оператора Лапласа.



(а) Лапласиан 3х3.



(е) Лапласиан 5х5

Оператор Марра

Контурные точки в операторе Марра определяются как точки пересечения нулевого уровня на отфильтрованном изображении. Получающиеся контуры не имеют разрывов.

Возможна масштабная настройка алгоритма путем выбора значения параметра σ. Как видно на рисунках на следующих слайдах, по мере увеличения значения параметра σ, оператор Марра выделяет все более и более крупные элементы изображения. При этом формируемые данным оператором контура продолжают сохранять характерную замкнутую форму.

Оператор Марра

Пример работы оператора Марра.

(а) Исходное изображение

(б) Оператор Марра (σ=1.0)

Оператор Марра

Пример работы оператора Марра.

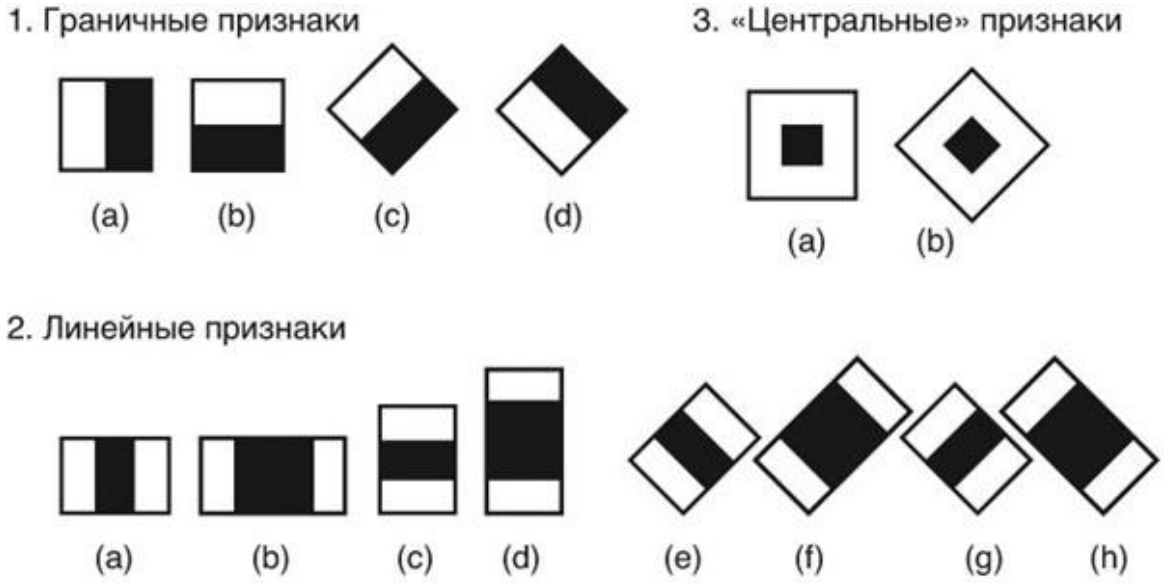
(в) Оператор Марра (σ=2.0)

а) Оператор Марра (σ=3.0)

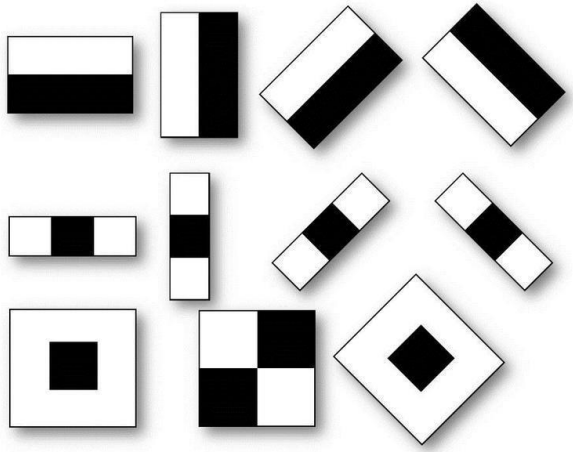
Выделение признаков на изображениях

Хаароподобные признаки

Метод Виолы—Джонса (2001 г.) — алгоритм, позволяющий обнаруживать объекты на изображениях в реальном времени. Алгоритм может распознавать различные классы изображений, но исходно предложен для обнаружения лиц



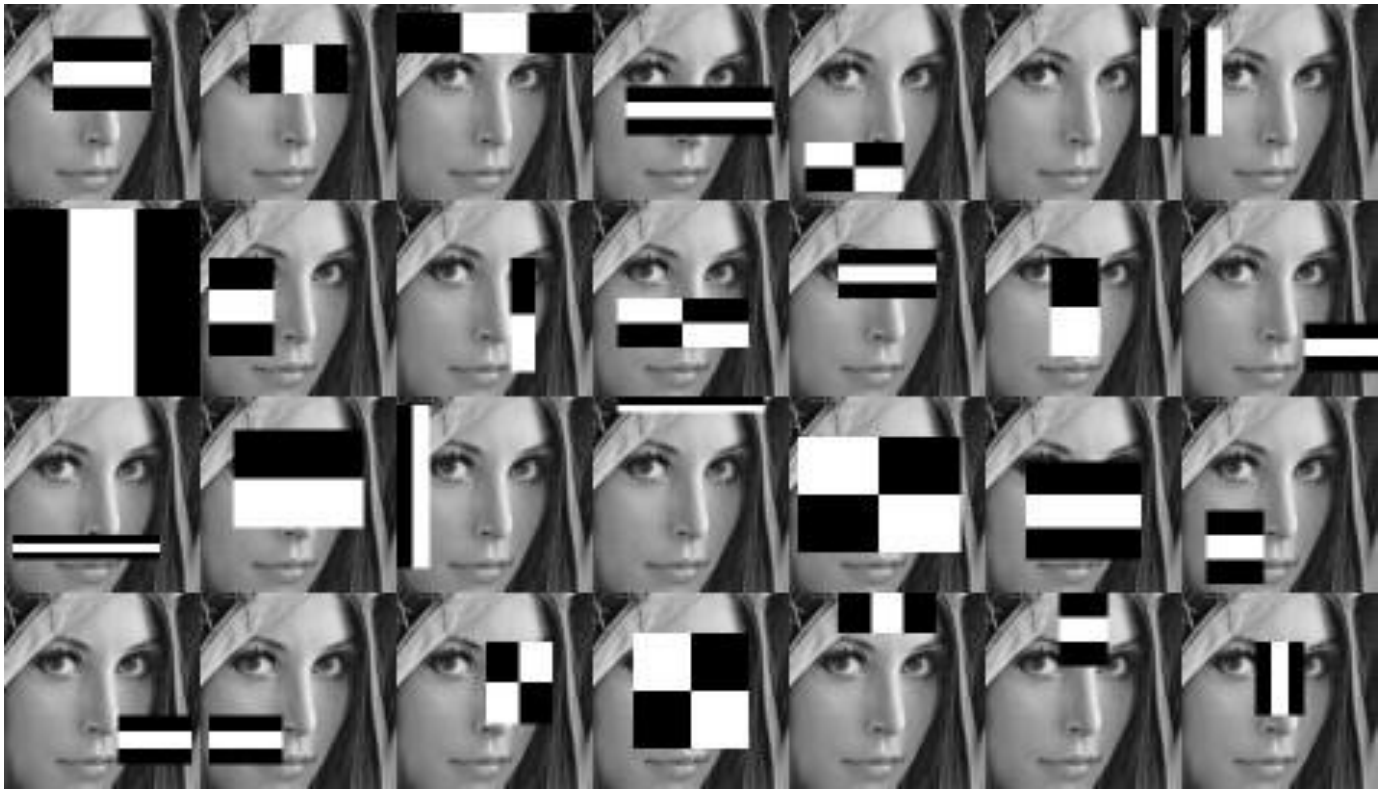
Маски различных обобщенных фильтров Виолы-Джонса
белые = +1
черные = -1



Признаки, используемые алгоритмом, опираются на суммирование пикселей из прямоугольных регионов и последующее взятие их разностей.

Хаароподобные признаки

Метод Виолы—Джонса (2001 г.) — алгоритм, позволяющий обнаруживать объекты на изображениях в реальном времени. Алгоритм может распознавать различные классы изображений, но исходно предложен для обнаружения лиц



Хаароподобные фильтры также не сглаживают изображение, а формируют в каждой его точке наобор *признаков* наличия особенностей изображения особого вида, полезных для обнаружения объектов (лиц)

Фильтры Габора

Для построения двумерного фильтра Габора применяется формула:

$$G(x,y) = \exp\left(-\frac{1}{2}\left[\frac{x_\phi^2}{\sigma_x^2} + \frac{y_\phi^2}{\sigma_y^2}\right]\right)\cos(2\pi\theta x_\phi)$$

$$x_\phi = x\cos(\phi) + y\sin(\phi)$$

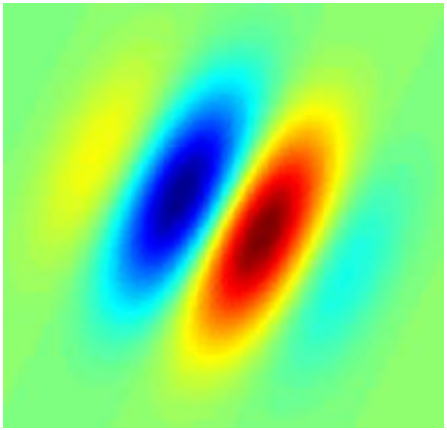
$$y_\phi = -x\sin(\phi) + y\cos(\phi)$$

где:

σ_x, σ_y - стандартные отклонения гауссова ядра, по осям x и y, определяющие растянутость фильтра по осям,

θ - частотная модуляция фильтра,

ϕ - пространственная направленность фильтра, определяющая его ориентацию относительно главных осей.



Маска двумерного фильтра Габора в 2D и в 3D

Фильтры Габора

Обработка изображения фильтром Габора достигается путём усреднения значений обрабатываемого изображения по некоторой области в каждой точке. Соответственно, наложение фильтра Габора на изображение имеет вид:

$$I^{'}(x,y) = \frac{1}{n^2}\sum_{i=1}^{n}\sum_{j=1}^{n} I(x - \frac{n}{2} + i, y - \frac{n}{2} + j) \cdot G(i,j)$$

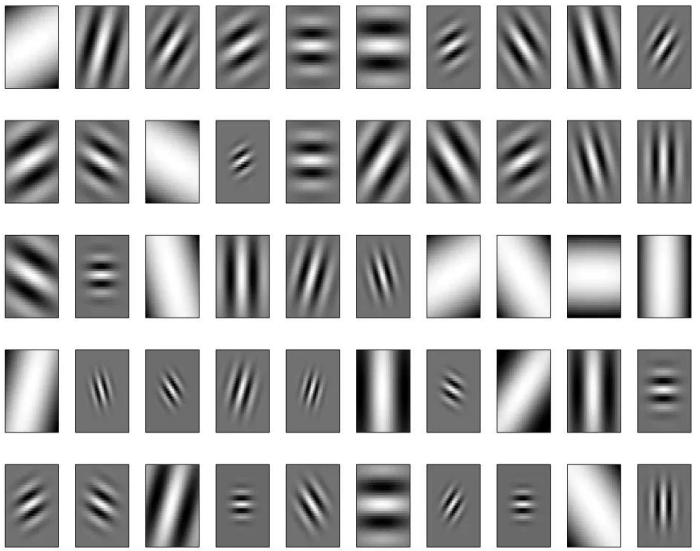
где:

$I(x,y)$ - интенсивность исходного изображения в точке (x,y),

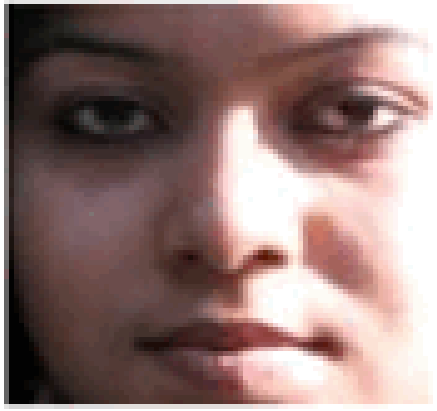
$I^{'}(x,y)$ - интенсивность нового изображения в точке (x,y),

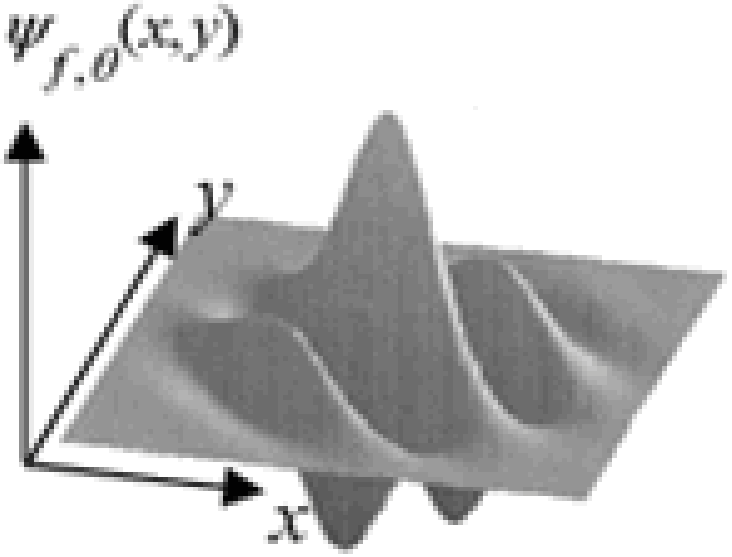
$G(i,j)$ - значение функции Габора, $i \in [0,n], j \in [0,n]$.

Фильтр Габора — линейный фильтр, маска которого определяется в виде гармонической функции, помноженной на гауссиан. При цифровой обработке изображений наборы таких фильтров применяются для построения поля направлений/ размеров элементов и границ изображений.

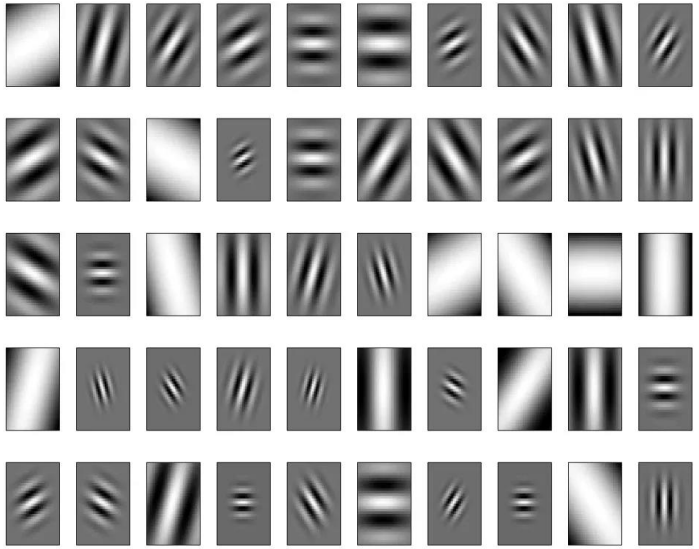


Фильтры Габора



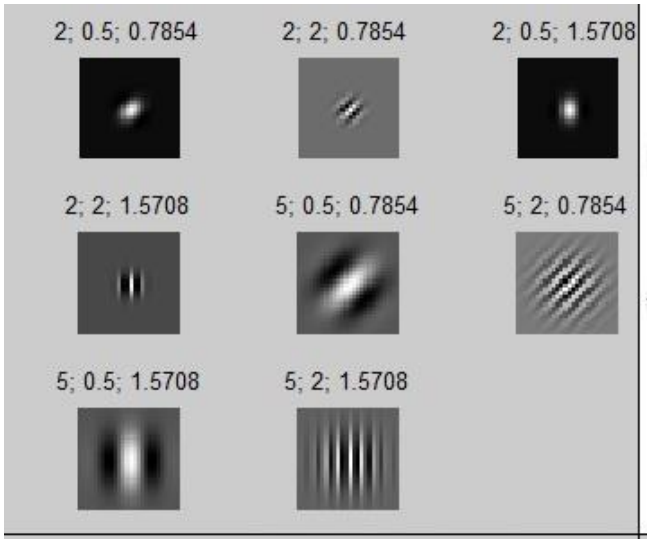


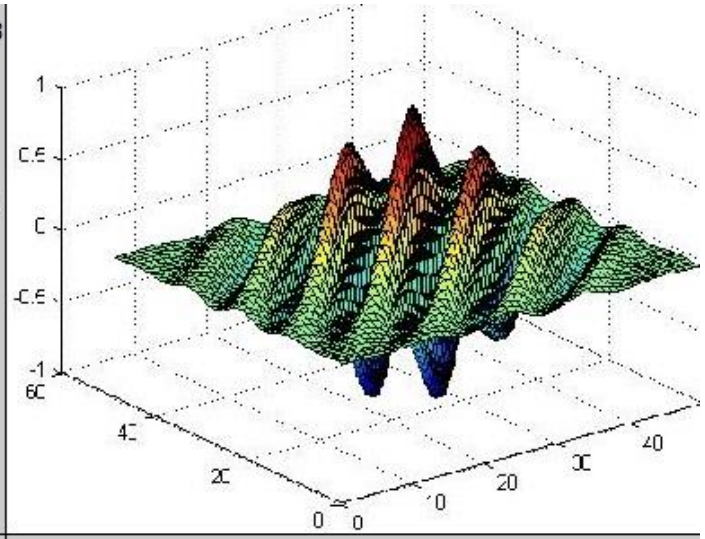
Фильтр Габора — линейный фильтр, маска которого определяется в виде гармонической функции, помноженной на гауссиан. При цифровой обработке изображений наборы таких фильтров применяются для построения поля направлений/ размеров элементов и границ изображений.

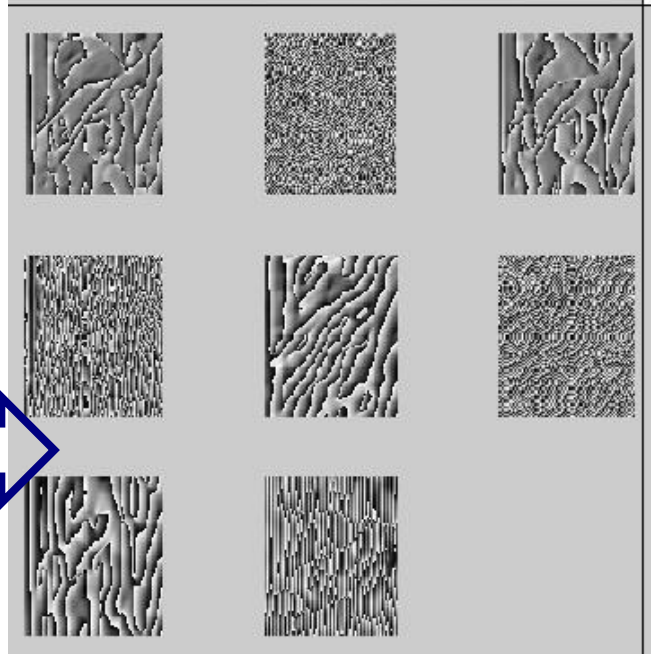


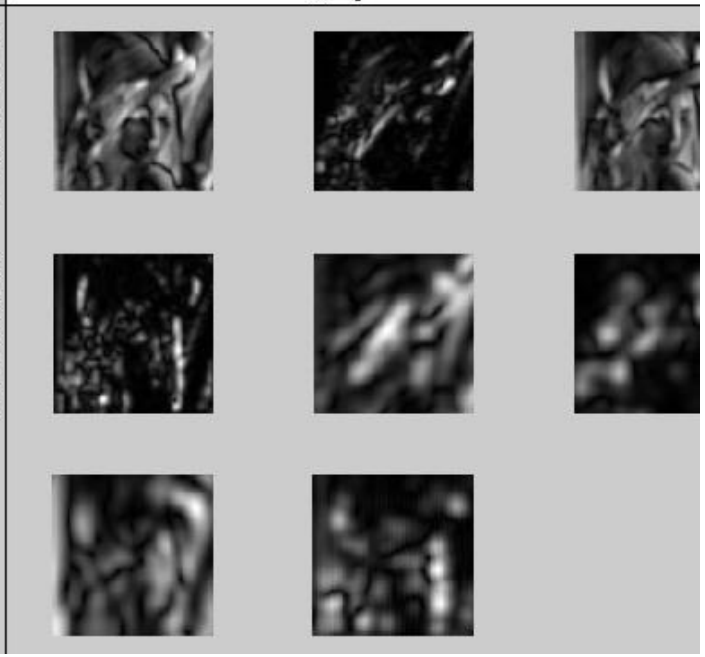
Фильтры Габора

Наборы (банки) фильтров Габора формируют наборы изображений (карт), характеризующих поля направлений/ размеров различных элементов текстуры изображений.







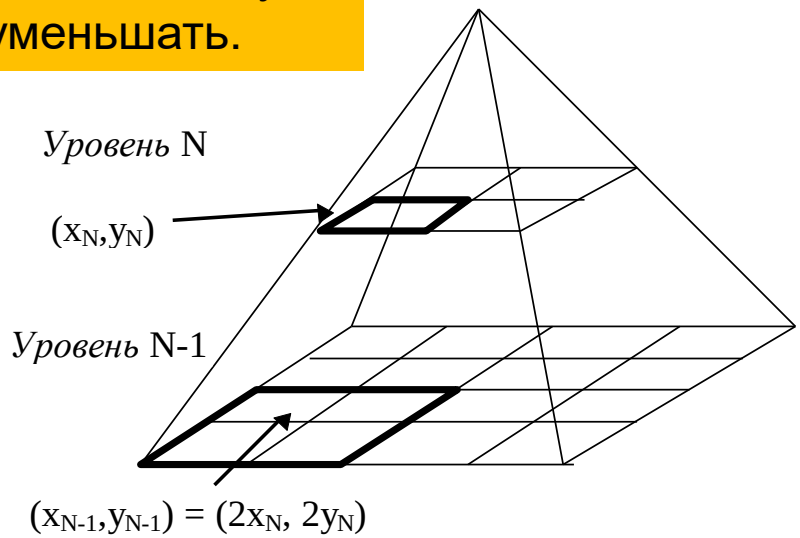


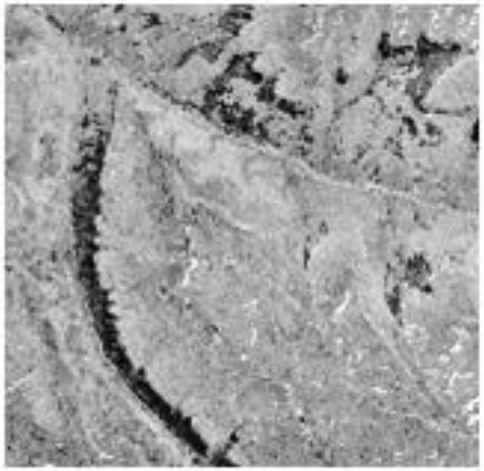
Разномасштабный анализ

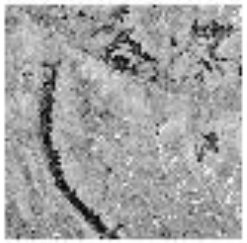
Анализ с использованием пирамиды изображений

Идея: если нужны крупные детали, можно не маску фильтра увеличивать, а изображение уменьшать.

Пирамида из четырех уровней изображения



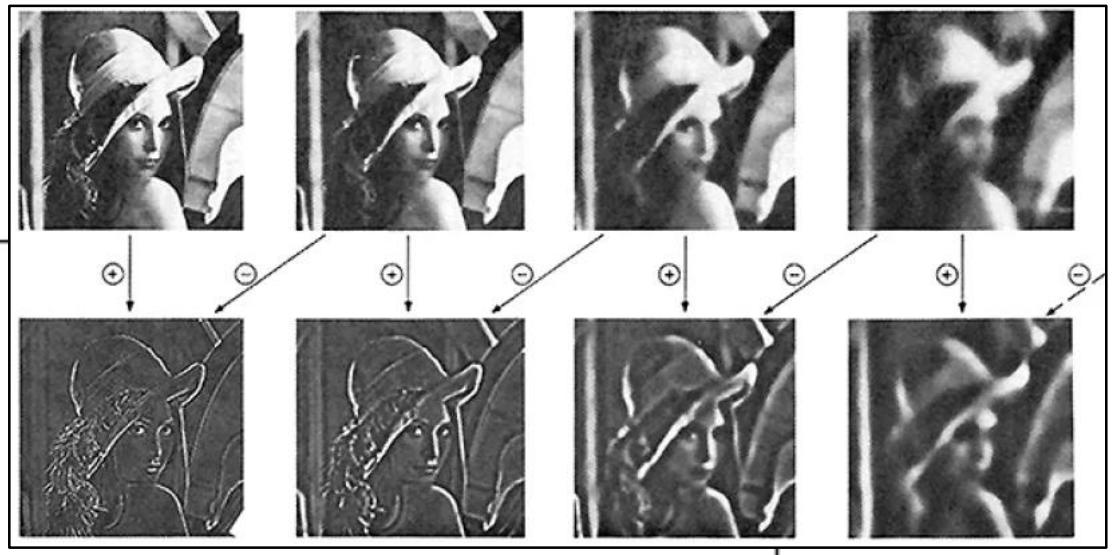


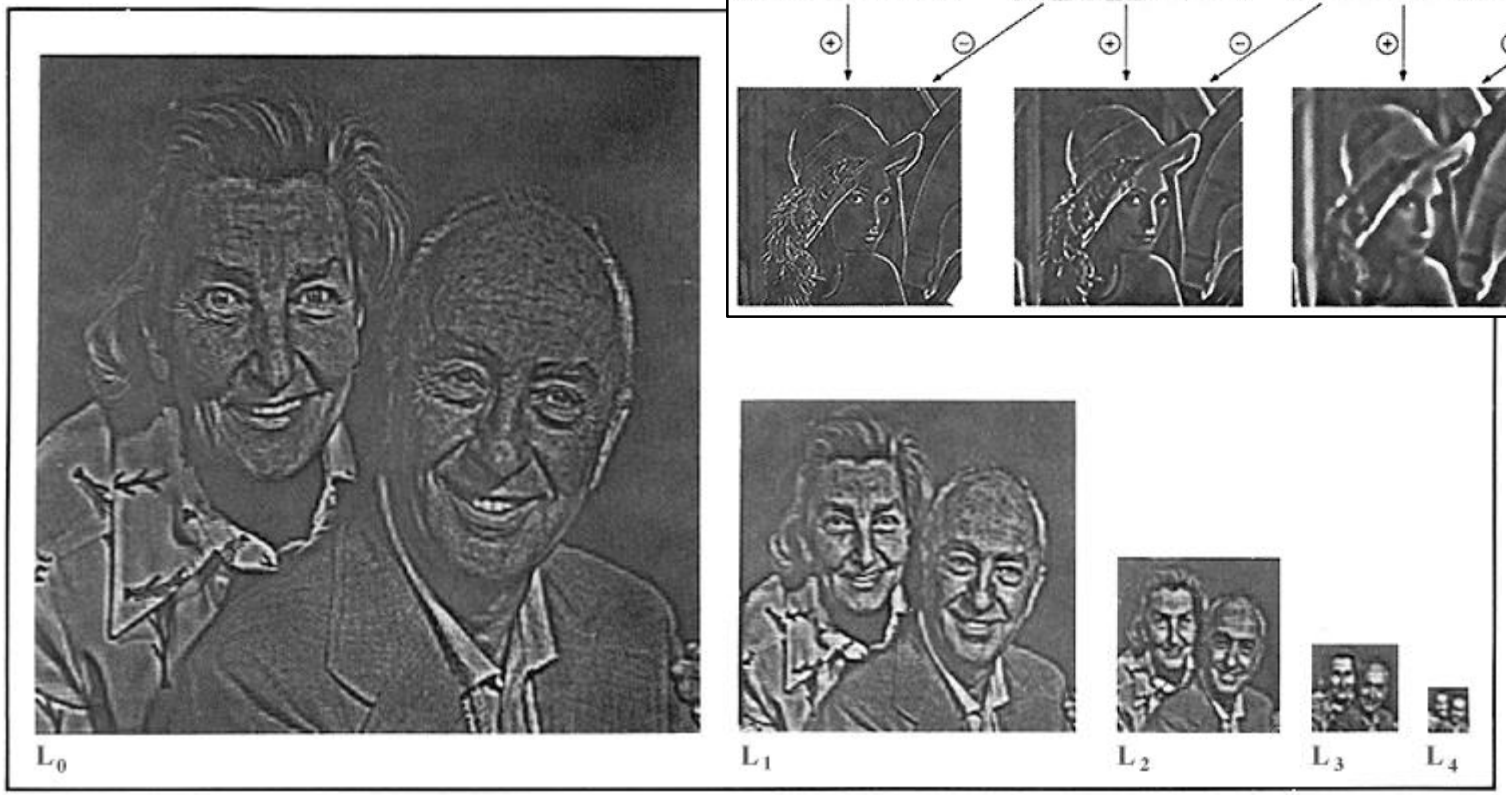


Разномасштабный анализ

Тогда фильтрами небольшого размера на каждом уровне пирамиды выделяются особенности (признаки) соответствующего масштаба

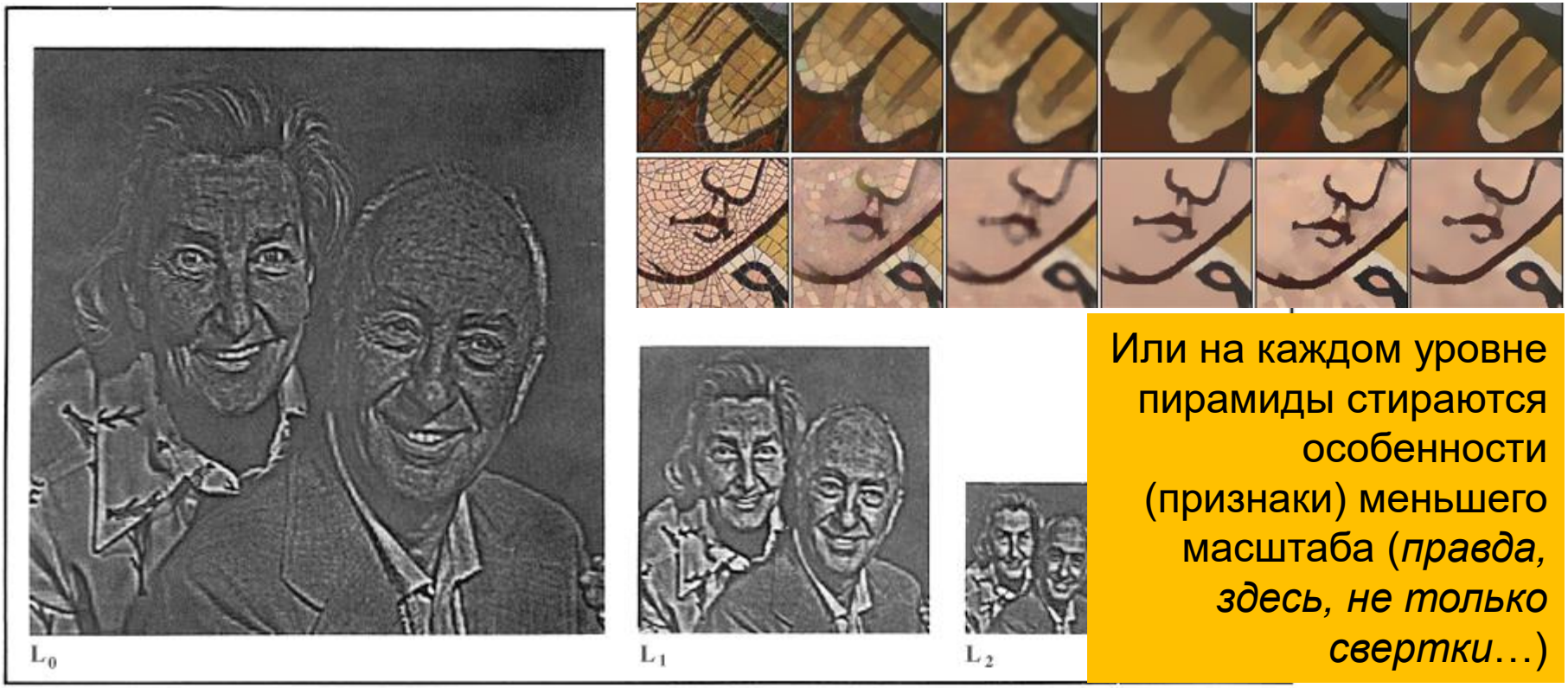
Анализ с использованием пирамид Гаусса и Лапласа





Разномасштабный анализ

Анализ с использованием пирамид Гаусса и Лапласа



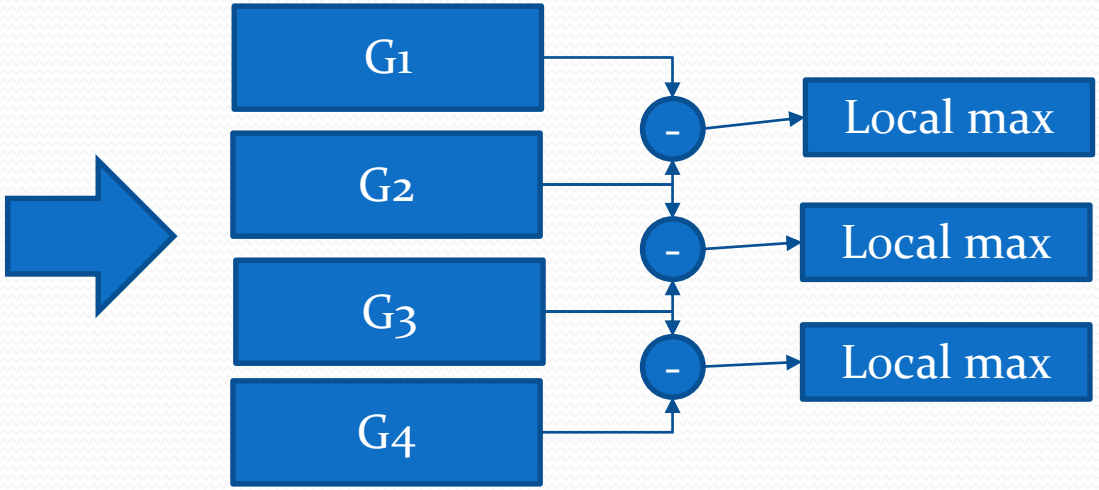
Или на каждом уровне пирамиды стираются особенности (признаки) меньшего масштаба (*правда, здесь, не только свертки*…)

Использование признаков для решения конечных задач

Пример задачи: стерео отождествление на основе характерных черт

Типы используемых характерных черт

Детектор особых точек DoG



$$D_i(x,y) = G_{i+1}(x,y) - G_i(x,y)$$

$$G_i\,(x,y) = \mathrm{I}(\mathrm{x},\mathrm{y}) \circ \frac{1}{\sigma_i} e^{-(x^2-y^2)/2\sigma_i^2}$$

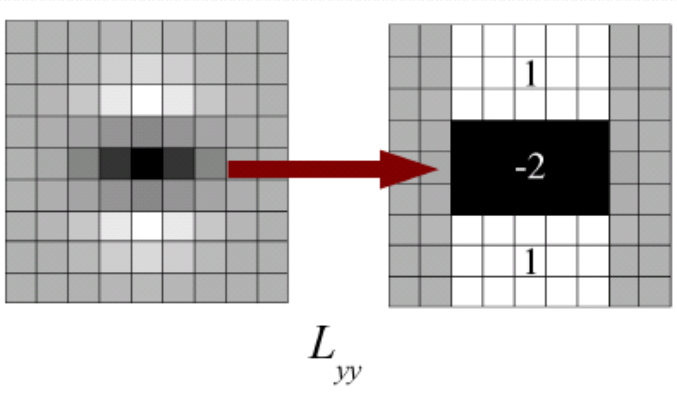
$$\sigma_1 < \sigma_2 \ldots . < \sigma_n$$

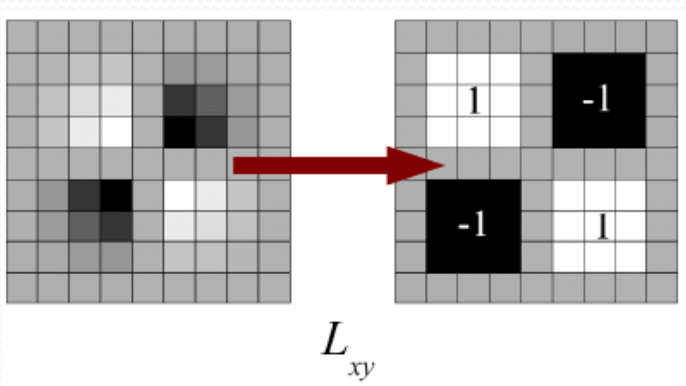
Детектор особых точек SURF

$$\mathrm{H}(x,y) = \begin{vmatrix} L_{xx}(x,y,\sigma) & L_{xy}(x,y,\sigma) \\ L_{xy}\,(x,y,\sigma) & L_{yy}(x,y,\sigma) \end{vmatrix}$$

$$L_{xx} = \frac{\partial^2 I}{\partial x^2}(x,y)\text{°G}(\sigma) \quad L_{xy} = \frac{\partial I}{\partial x\,\partial y}(x,y)\text{°G}(\sigma) \quad L_{yy} = \frac{\partial^2 I}{\partial y^2}(x,y)\text{°G}(\sigma)$$

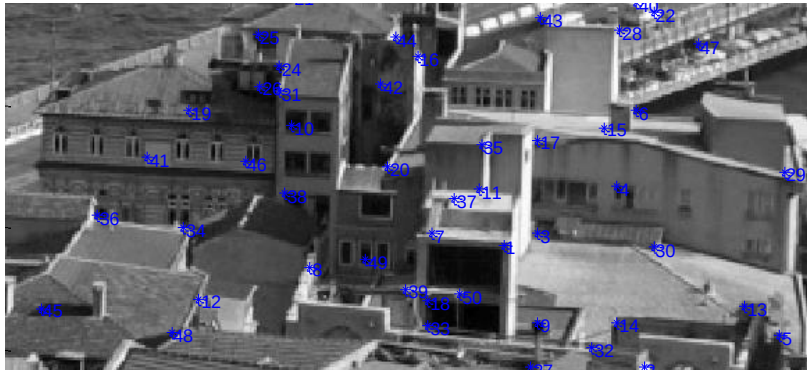
$$R(x,y) = D_{xx}D_{yy} - (wD_{xy})^{\,2}$$

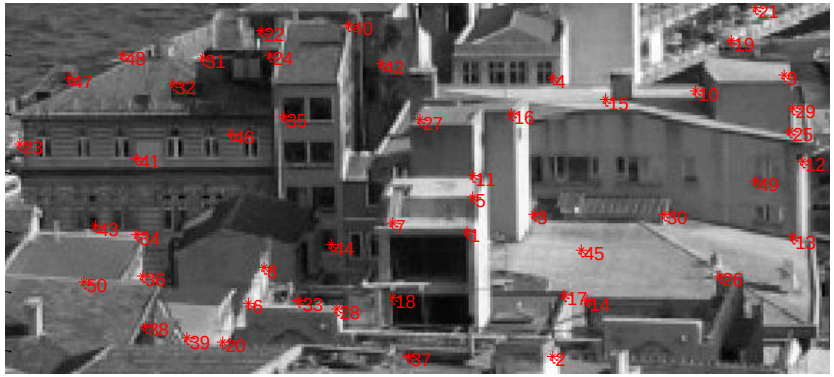




Стерео отождествление на основе характерных черт

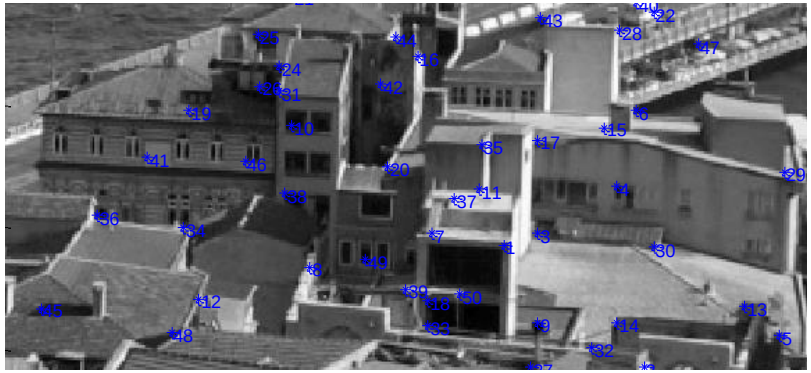
Первичное отождествление характерных точек (2/3 ложных пар)

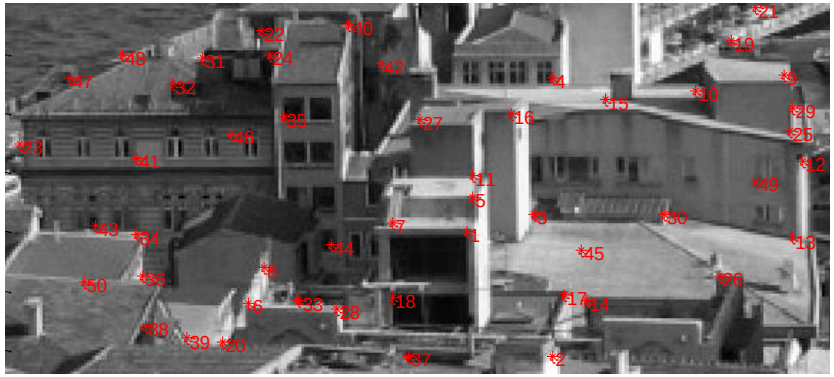




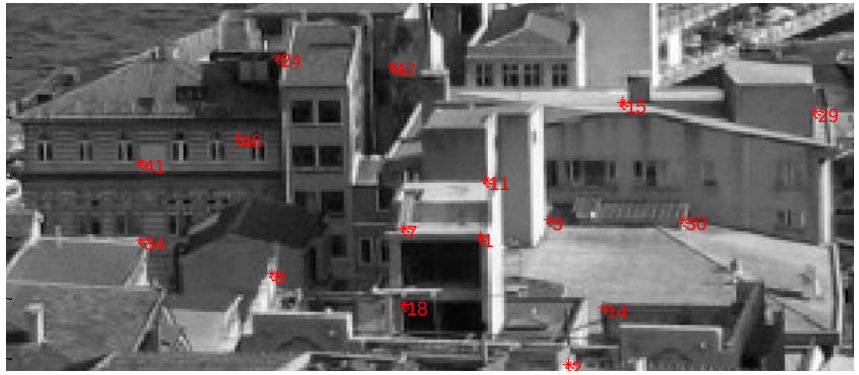
Стерео отождествление на основе характерных черт

Первичное отождествление характерных точек (2/3 ложных пар)





Геометрическая фильтрация пар точек (большинство ложных пар удалены)



Классический подход к задаче обнаружения и распознавания – обнаружение и распознавание объектов, основанное на их структурированных яркостно-геометрических моделях.

Делается человеком

1. Составление яркостно-геометрической конструкции (модели объекта), выбор признаков – *неформальный* элемент алгоритмизации
2. Поиск и локализация объекта распознавания – формальный элемент алгоритмизации

Делается машиной

Схема модельного подхода к обнаружению объектов

сенсоры

изображения

сегментация (выделение) характерных черт

робастные алгоритмы голосования или сопоставления (matching)

Характерные элементы (черты), используемые в иерархических алгоритмах обнаружения

Свойства (атрибуты) характерных черт

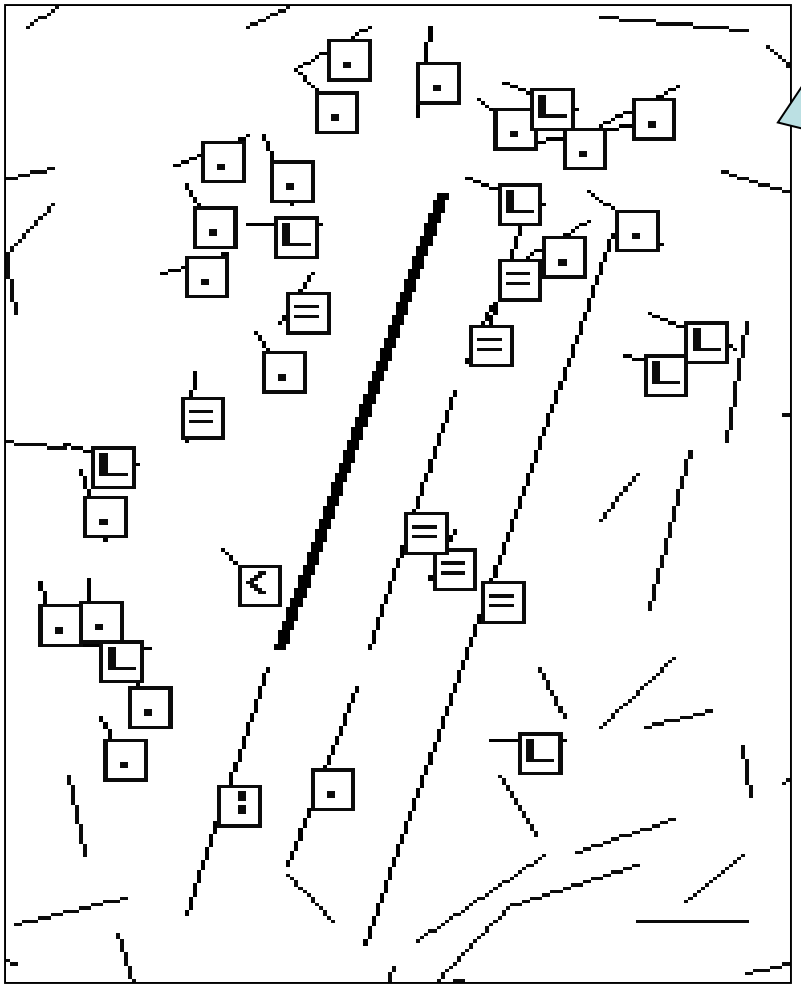
Характерные черты на изображении имеют следующие виды атрибутов:

1. **Положение**: Концы отрезка, центр отрезка, центр тяжести области, вершины многоугольников;
2. **Геометрические атрибуты**: Ориентация, длина, кривизна, площадь, периметр, ширина линии, минимальный и максимальный диаметр области, оси симметрии, число и положение особых точек, показатель компактности;
3. **Радиометрические атрибуты**: Контраст, статистика распределения яркости, знак и величина края, автокорреляция;
4. **Текстурные атрибуты**: Матрица смежности, показатель однородности, энергия, энтропия, статистика градиентов текстуры, результаты применения текстурных фильтров, моменты;
5. **Топологические атрибуты**: Связность, соседство, общие точки, пересечение, параллельность, перекрытие, включение;
6. **Цветовые/многозональные атрибуты**: вектор атрибутов для каждого канала;
7. **Динамические атрибуты**: атрибуты статических и движущихся объектов;
8. **Временные атрибуты**: функции изменения атрибутов со временем.

Пример задачи: Обнаружение зданий системой технического зрения автономного ЛА

Иерархический анализ признаков – от простых элементов к сложным структурам

Примеры атрибутов характерных черт



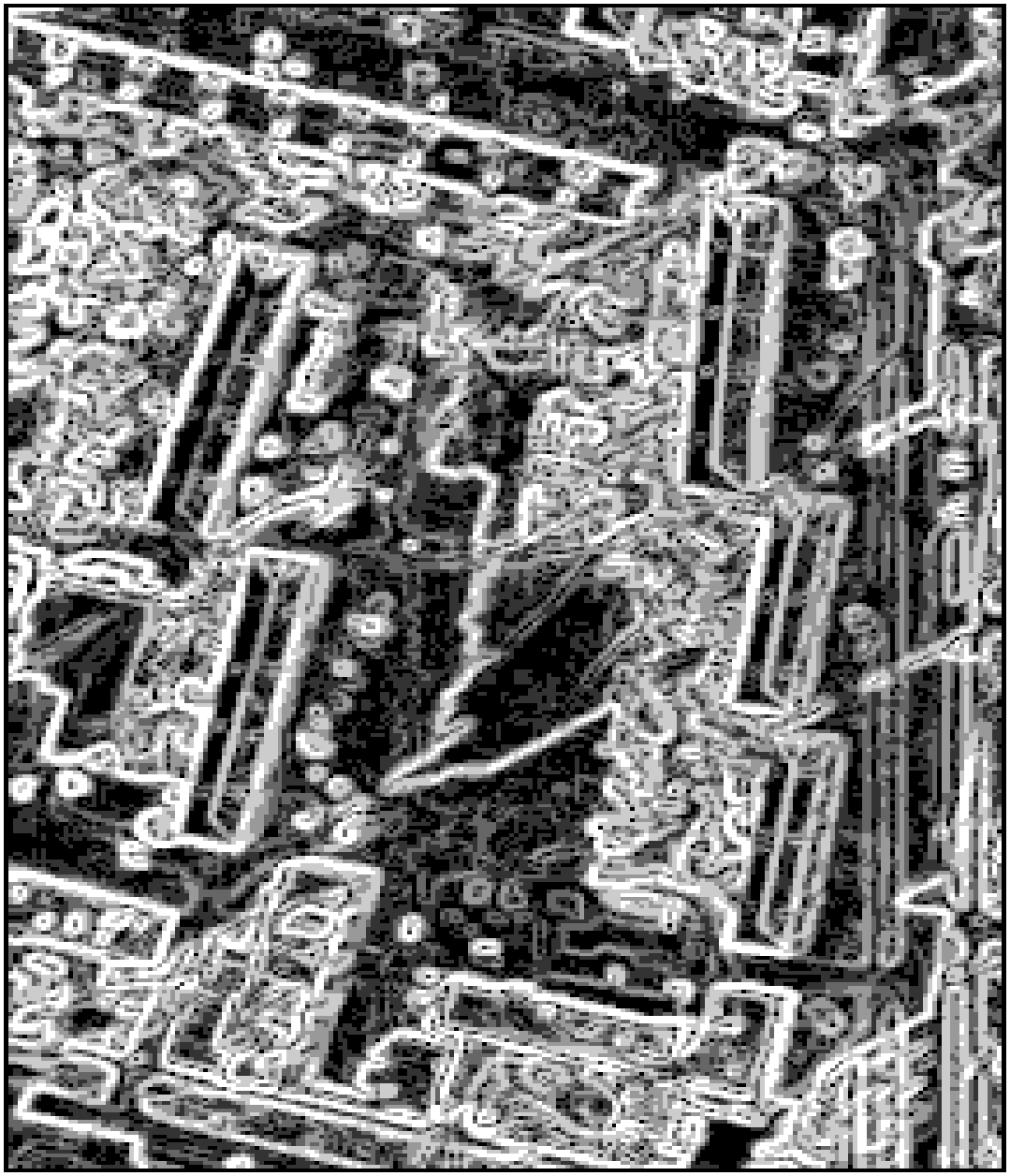
Исходная информация

‘.’ - близость;
‘!’ - коллинеарность;
‘=‘ - параллельность;
‘L’- перпендикулярность; ‘<‘ – угол.

Выбранный отрезок
Другие линии

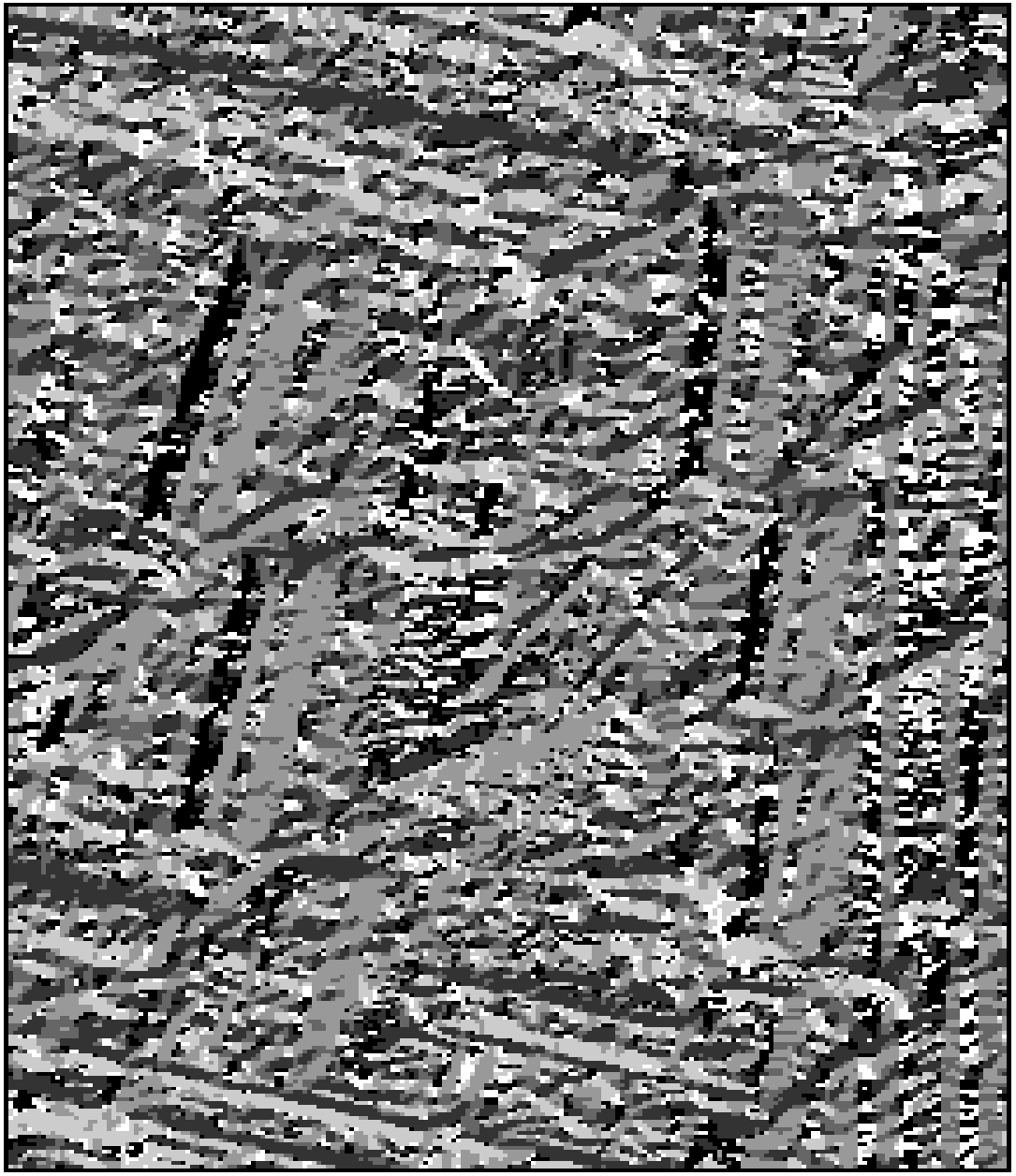
Пример автоматического обнаружения зданий с летательного аппарата

исходное изображение



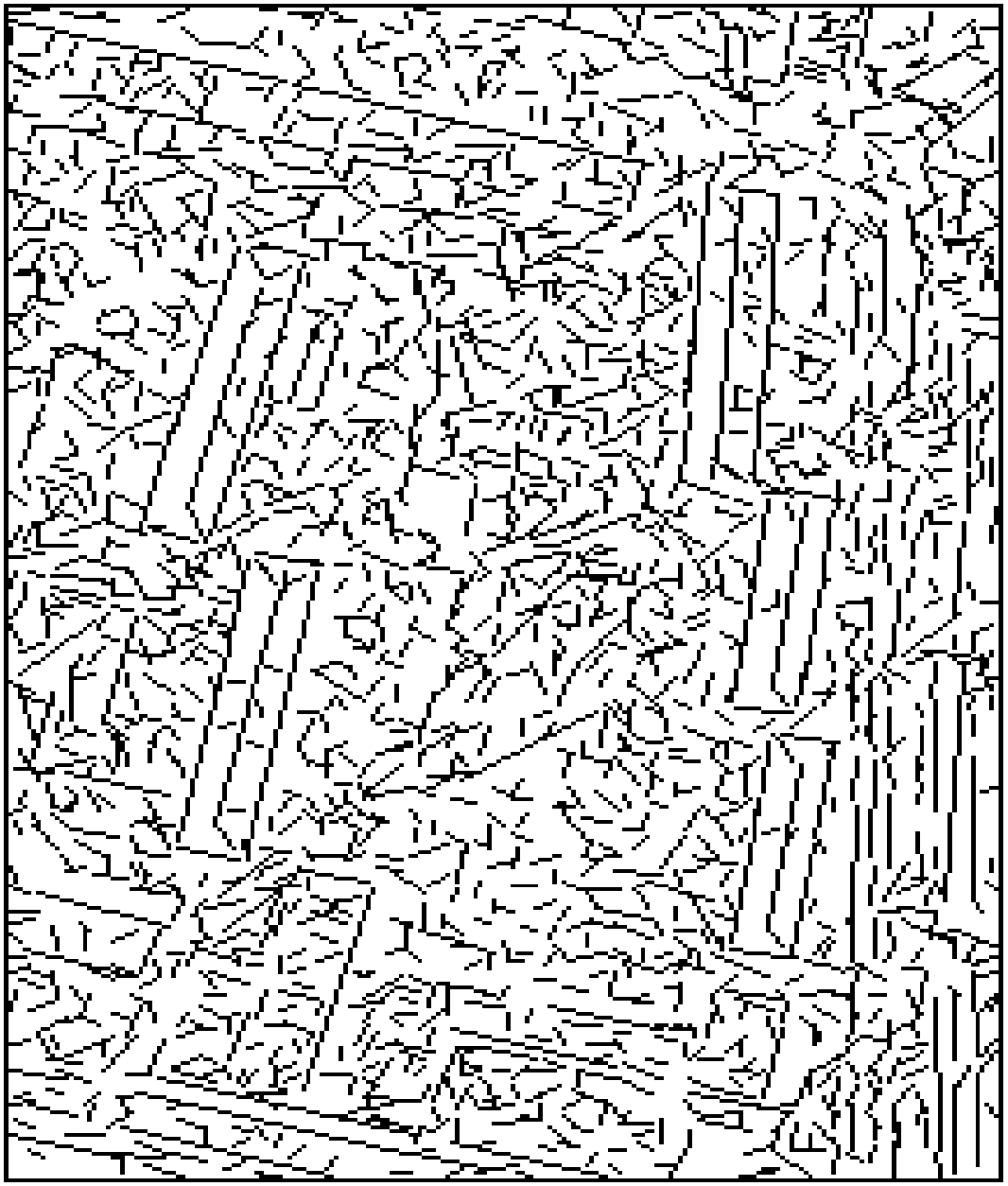
Пример автоматического обнаружения зданий с летательного аппарата

К о н т у р н ы й п р е п а р а т



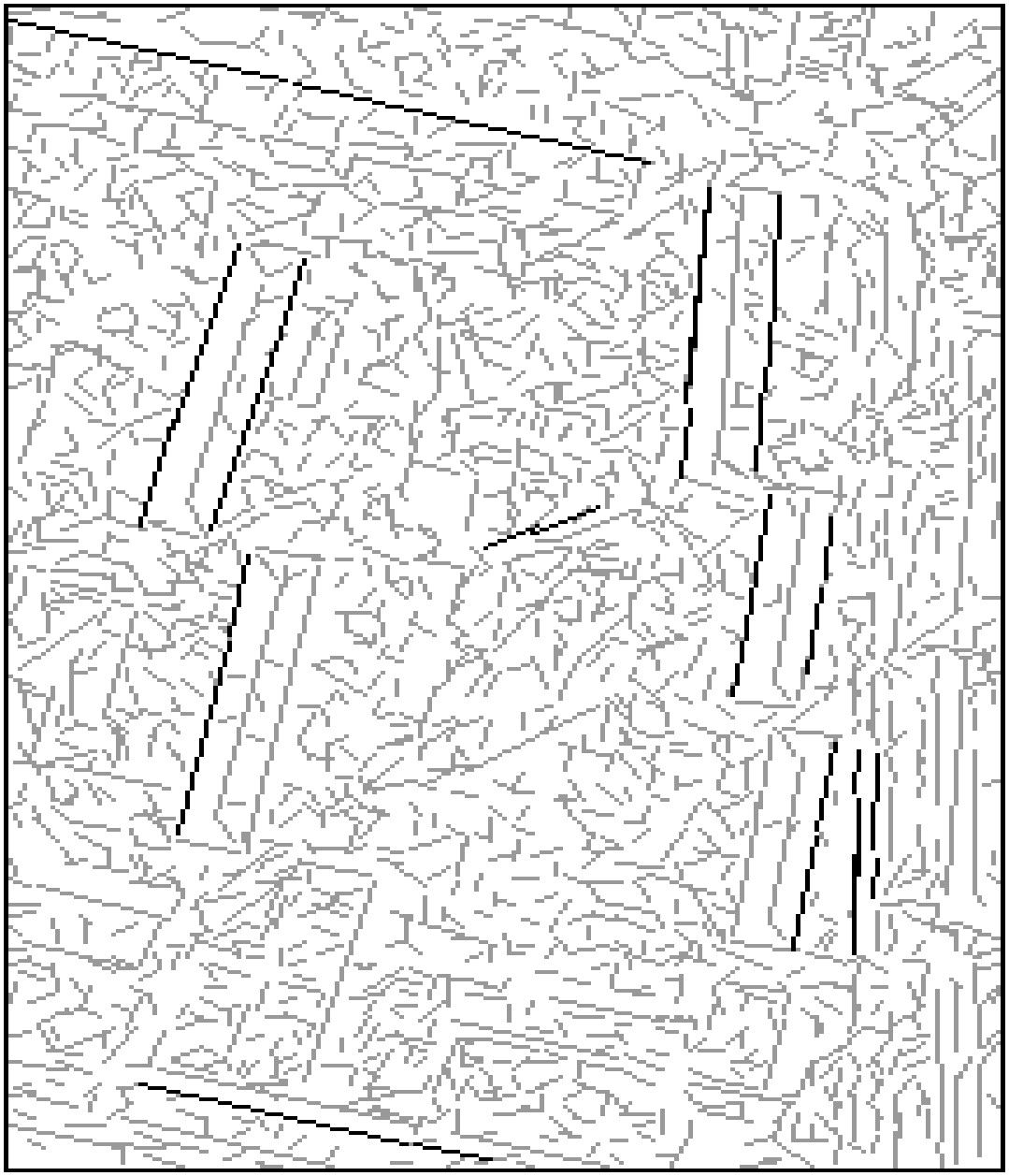
Пример автоматического обнаружения зданий с летательного аппарата

Выделение характерных черт – областей поддержки линеаментов



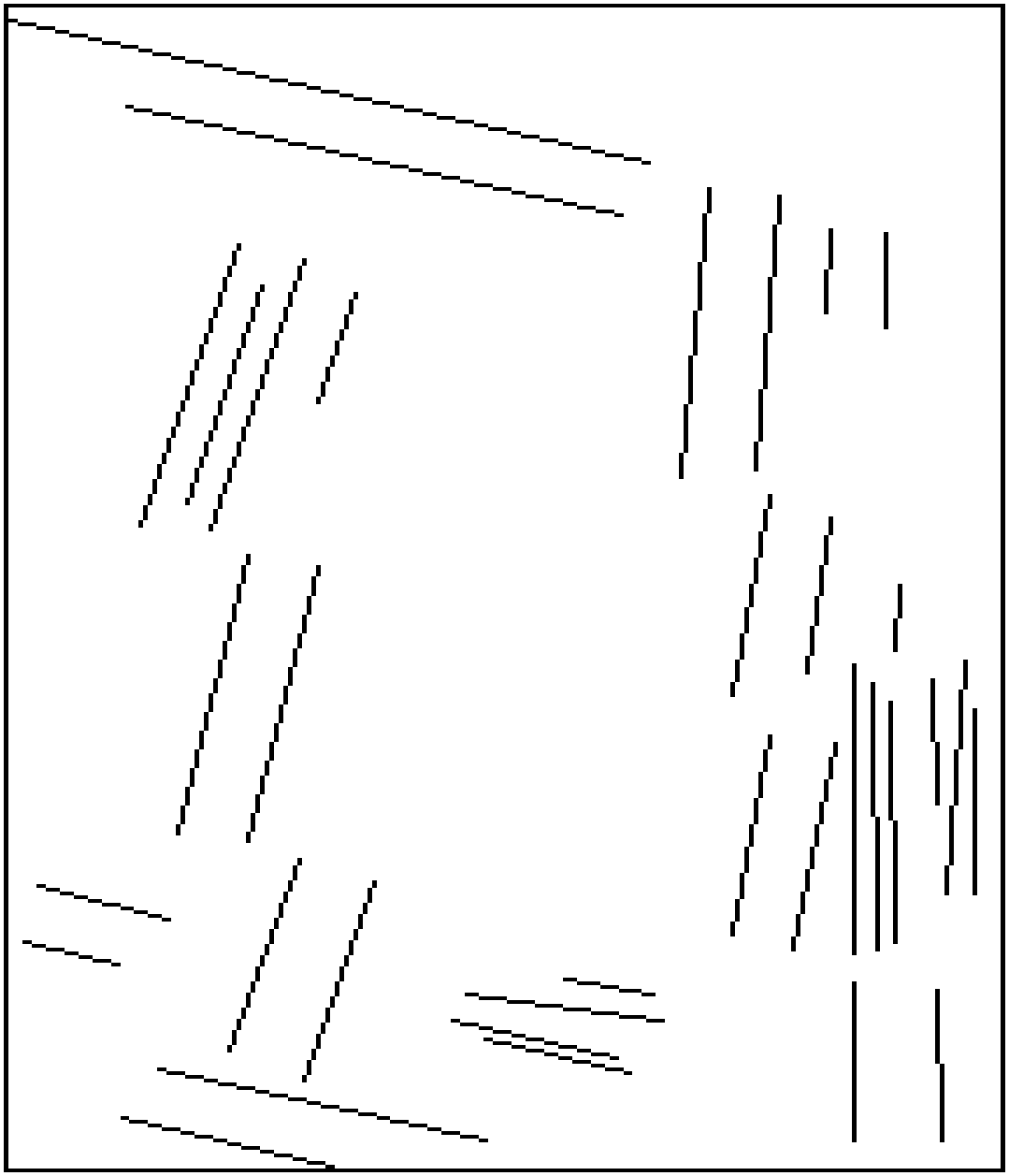
Пример автоматического обнаружения зданий с летательного аппарата

Выделение первичных линеаментов



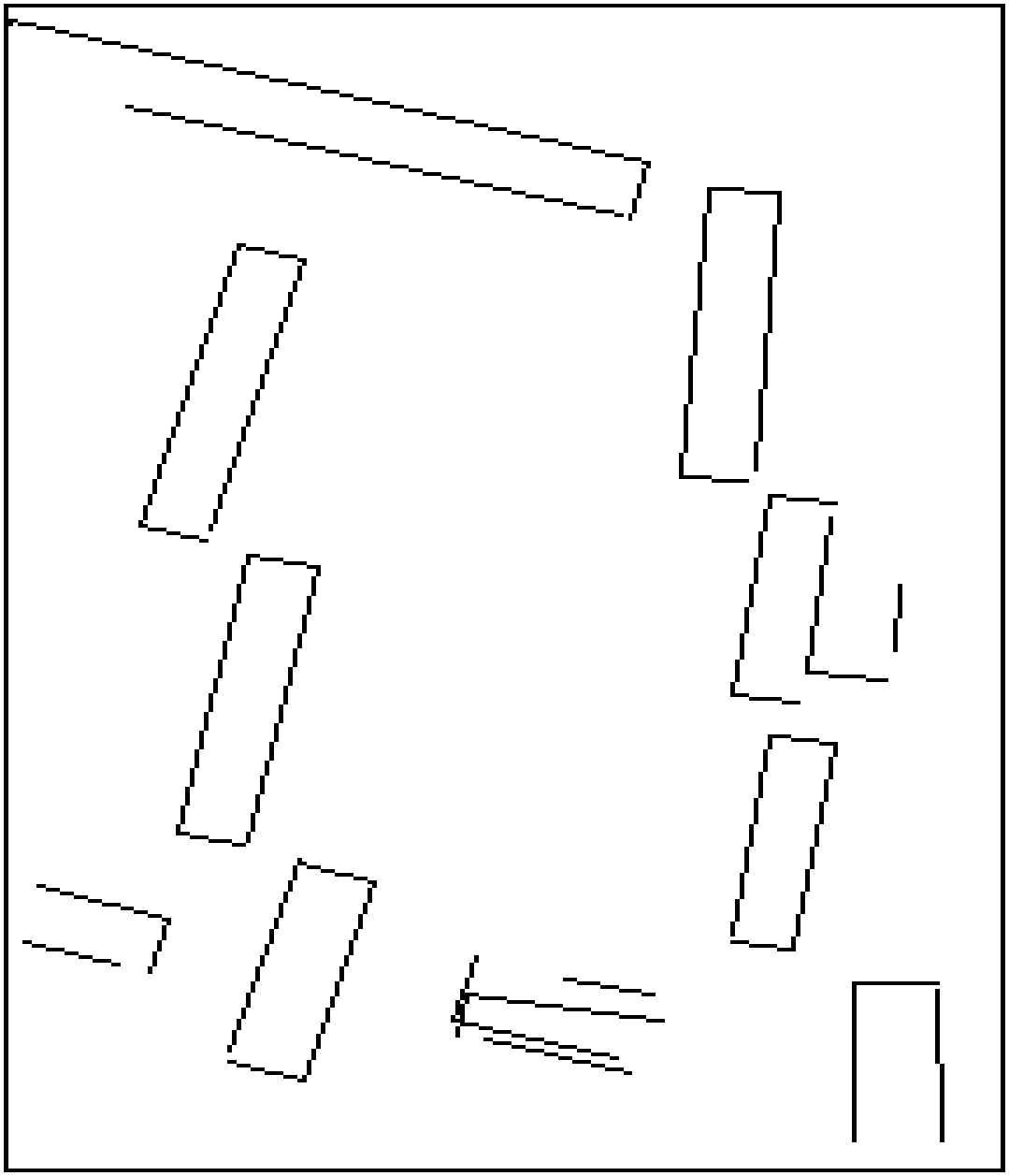
Пример автоматического обнаружения зданий с летательного аппарата

Выделение крупных и/или параллельных линеаментов



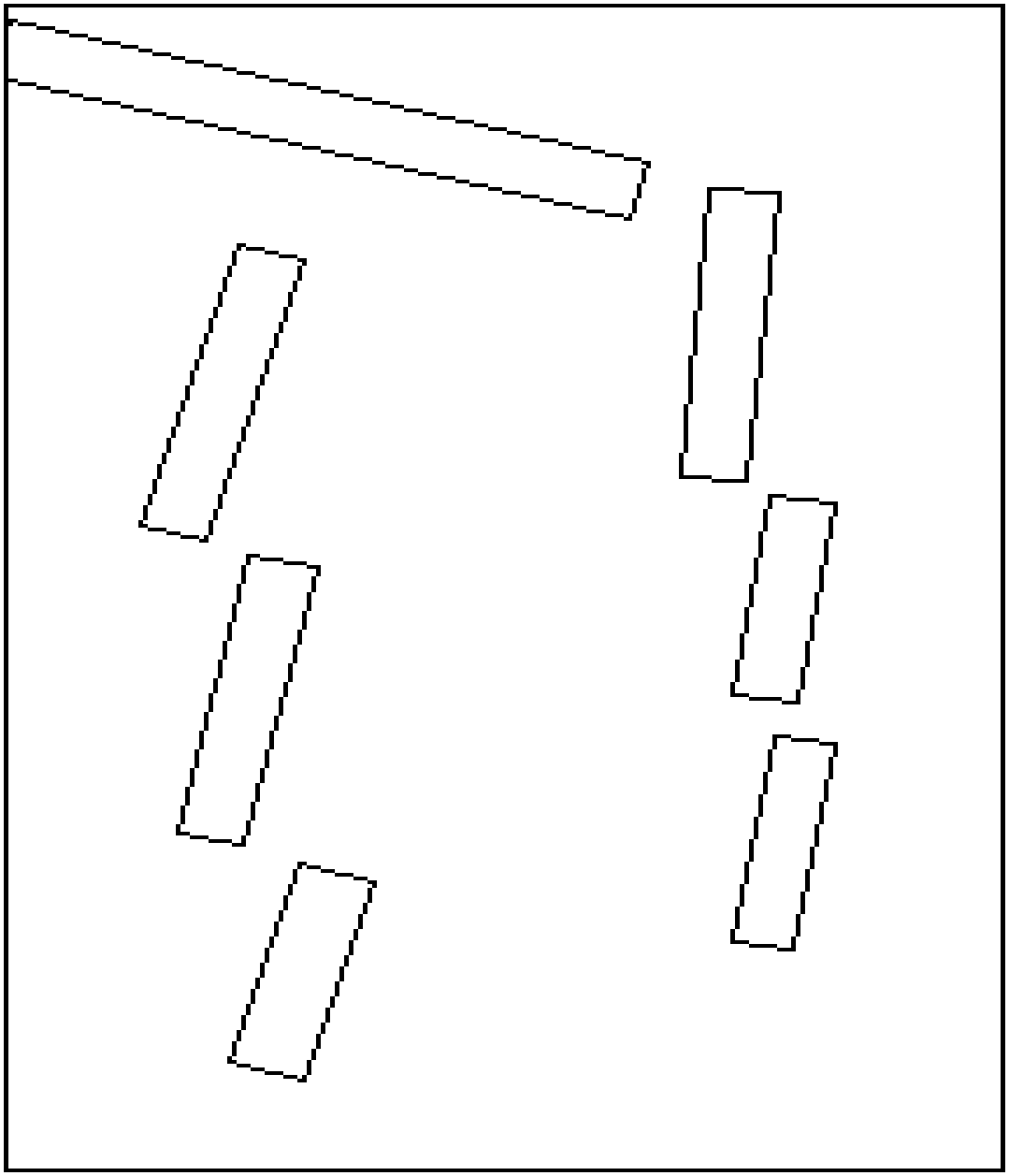
Пример автоматического обнаружения зданий с летательного аппарата

Фильтрация по размеру



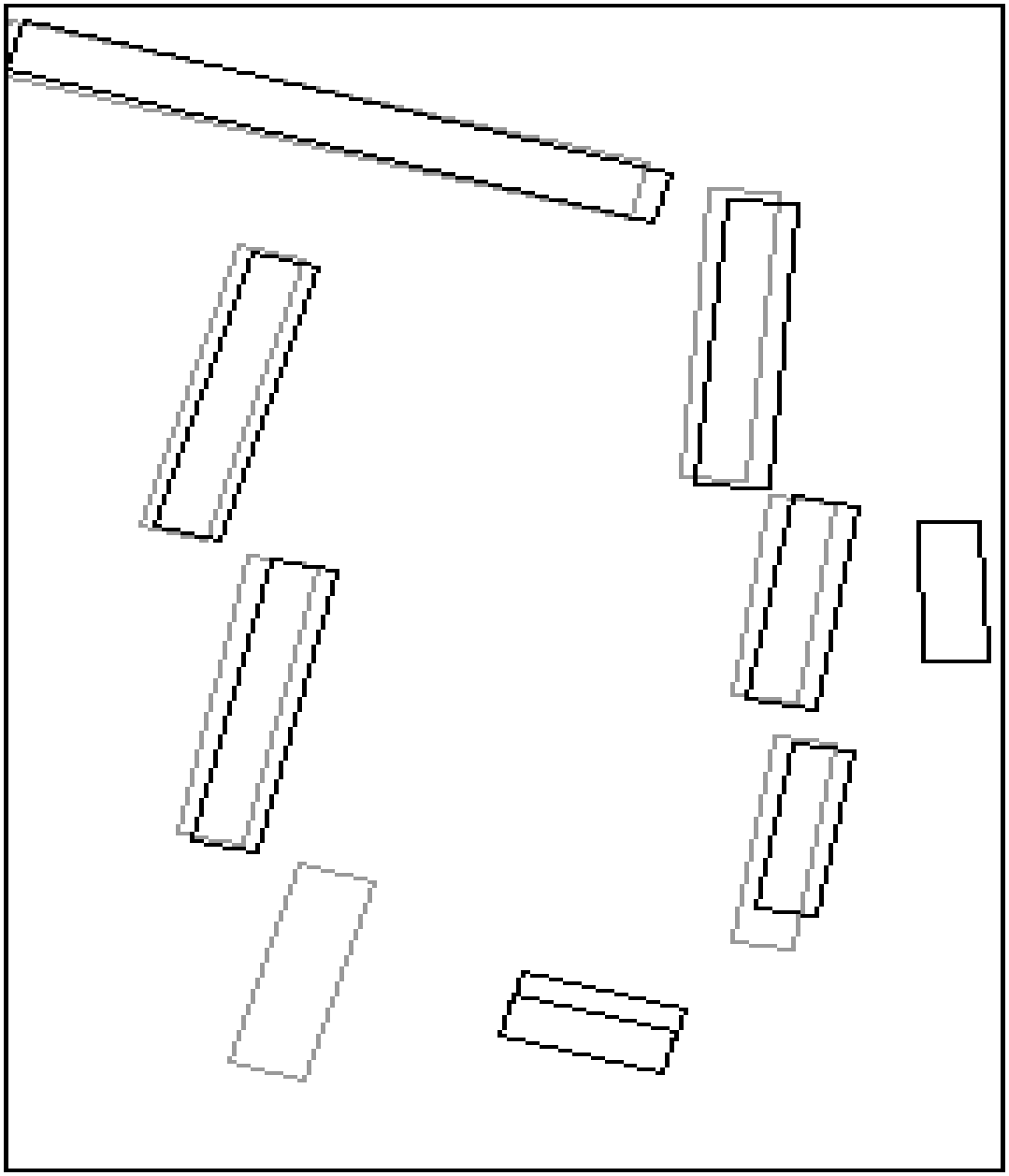
Пример автоматического обнаружения зданий с летательного аппарата

Поиск П-образности



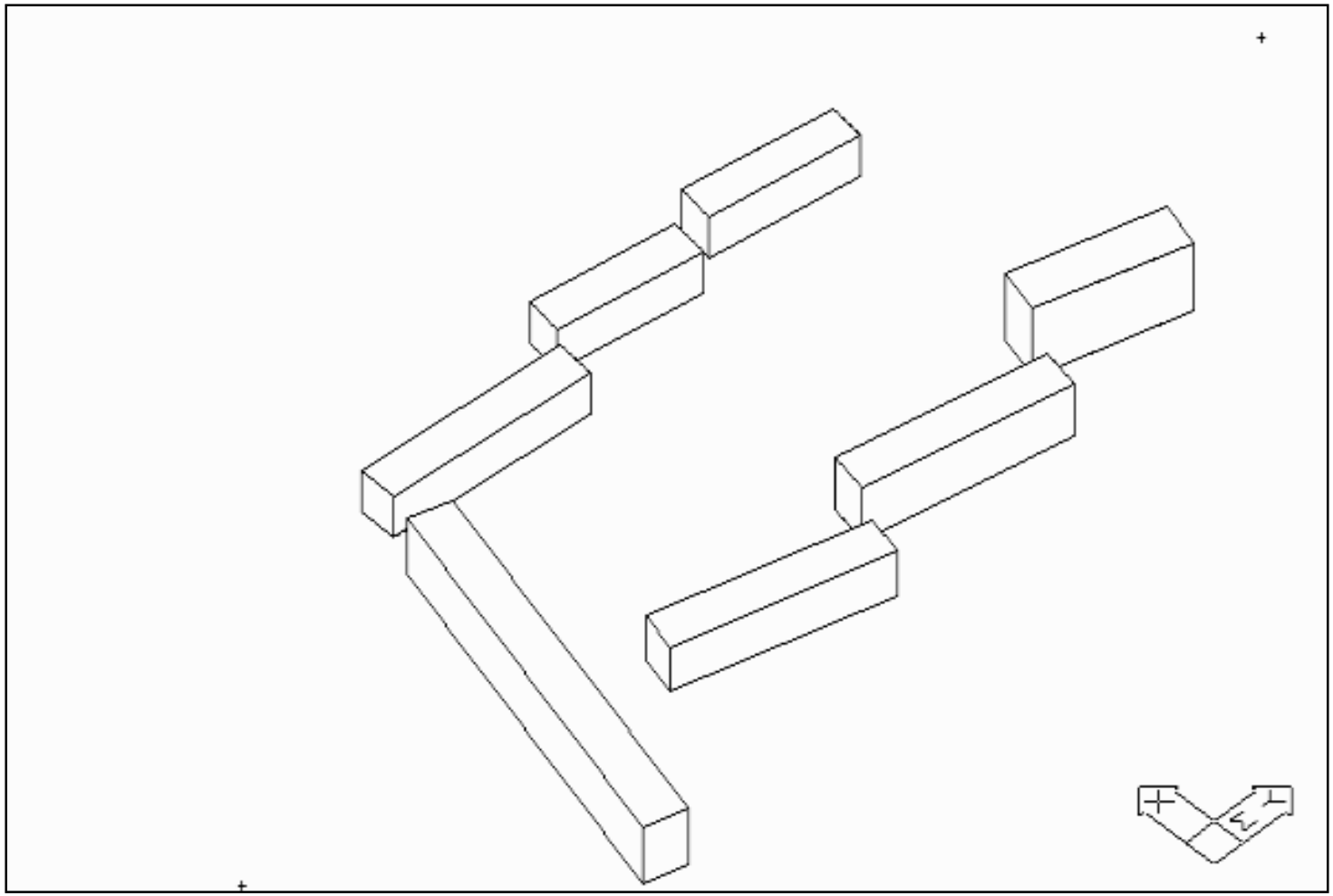
Пример автоматического обнаружения зданий с летательного аппарата

Окончательная 2D-модель



Пример автоматического обнаружения зданий с летательного аппарата

Межкадровое соответствие



О к о н ч а т е л ь н а я 3D м о д е л ь

Формирование признаков в классическом компьютерном зрении (выводы)

Все разработанные человеком детекторы структур и дескрипторы признаков так или иначе обладают следующими свойствами:

- **Локальность** (обработка не всего изображения, а «окна»)
- **Многомасштабность** («окна» признаков разного размера)
- **Линейность** - многие базовые признаки вычисляются как простые линейные свертки в окне (линейные фильтры с заданными масками)
- **Инвариантность к сдвигу** («окно» фильтра «бежит» по изображению, вычисляя в каждом положении окна одно и то же – свертку с одной и той же фиксированной маской)
- **Наборы признаков** - нужны банки разнообразных фильтров для одновременного получения не одного, а целого вектора признаков
- **Иерархические признаки** – признаки более высокого уровня «собираются» из признаков более низкого уровня

Сверточные нейронные сети и глубокое обучение

Соединяем идеи машинного обучения и нейронных сетей (обучение, автоматическое формирование признаков, структура персептрона) с идеями компьютерного зрения (локальность, свертки, многомасштабность, иерархия признаков)

Convolution networks, Deep learning, Image Recognition

Задача распознавания визуальных образов

2011+: Результаты автоматического распознавания изображений лучше, чем у человека

2011: First Superhuman Visual Pattern Recognition

twice better than humans
three times better than the closest artificial competitor
six times better than the best non-neural method

Jürgen Schmidhuber

The list of won competitions in Computer Vision:

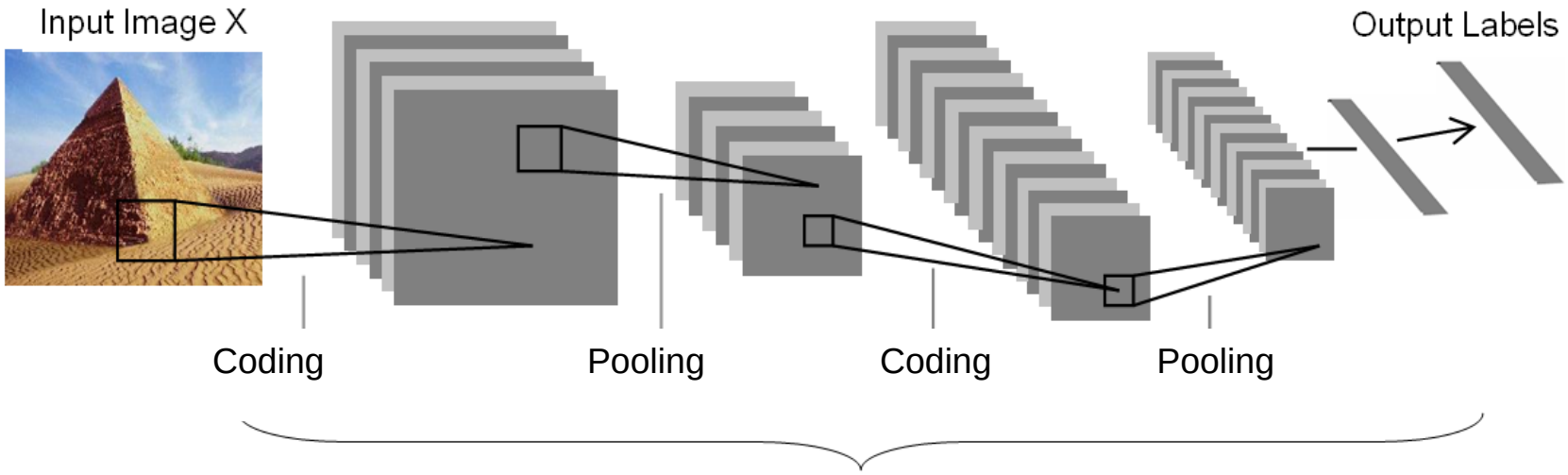
9. MICCAI 2013 Grand Challenge on Mitosis Detection
8. ICPR 2012 Contest on Mitosis Detection in Breast Cancer Histological Images
7. ISBI 2012 Brain Image Segmentation Challenge (with superhuman pixel error rate)
6. IJCNN 2011 Traffic Sign Recognition Competition (only our method achieved superhuman results)
5. ICDAR 2011 offline Chinese Handwriting Competition
4. Online German Traffic Sign Recognition Contest
3. ICDAR 2009 Arabic Connected Handwriting Competition
2. ICDAR 2009 Handwritten Farsi/Arabic Character Recognition Competition
1. ICDAR 2009 French Connected Handwriting Competition. Compare the

Список тестов, в которых данный алгоритм показал такие результаты

http://people.idsia.ch/~juergen/deeplearning.html

Convolution networks, Deep learning, Image Recognition

Сверточные сети, глубокое обучение – современная реинкарнация персептронов



Архитектура “Coding + Pooling” = “Кодирование + Объединение ”

- Иерархическое выделение признаков при обучении и/или самообучении на сверхбольших выборках изображений
- Много десятков уровней (кодирование + объединение данных)
- Инвариантность к сдвигу изображения (линейные фильтры)
- На верхних уровнях – собственно классификатор (полносвязный)

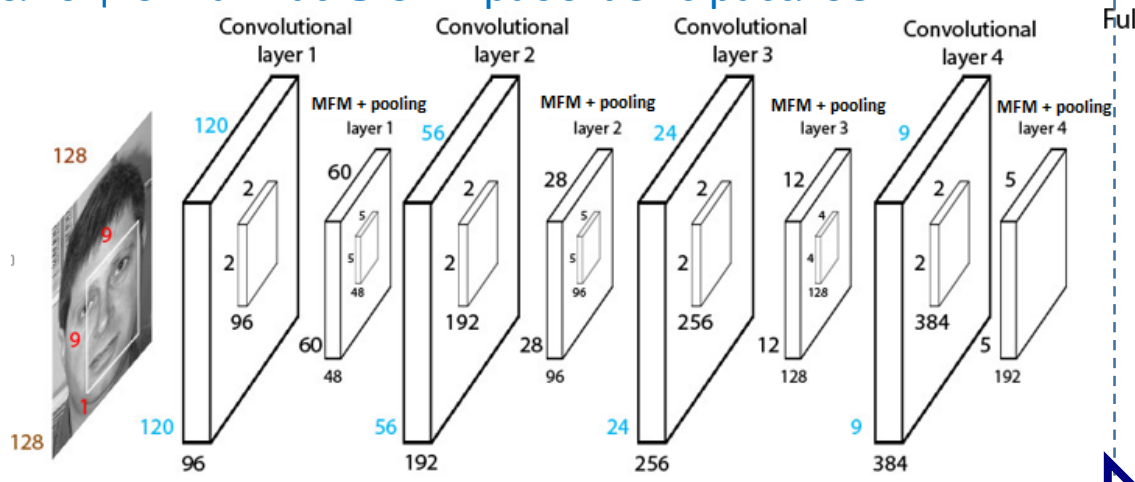
Сверточный нейрон это обучаемый линейный фильтр (веса = элементы маски фильтра) + нелинейность

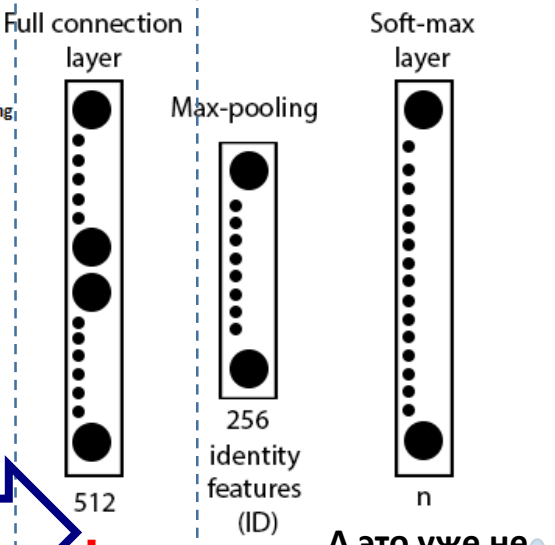
Идеи Deep Learning

(Kai Yu, 2012)

Интерпретация CNN как процесса эволюции расслоения базового многообразия

Конволюционная часть CNN работает с расслоениями





От локальных признаков… к признакам изображения

А это уже не признаки, а классы

ConvID

DeepID

Вырожденное расслоение = = Базовое многообразие

Углубление расслоения

Вырожденное расслоение = = вектор признаков

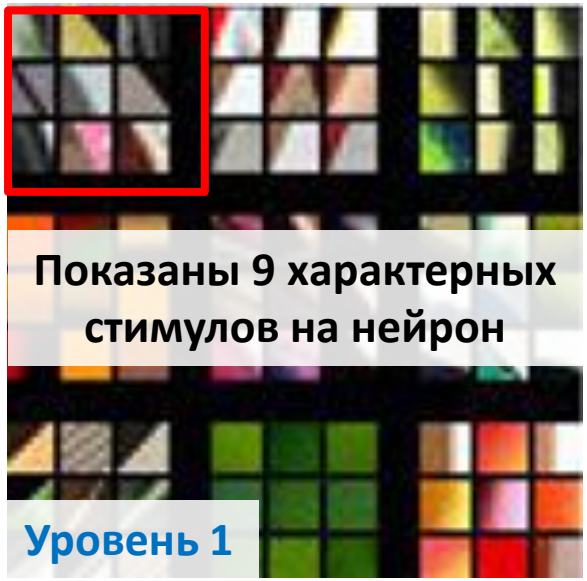
Снижение размерности вектора признаков

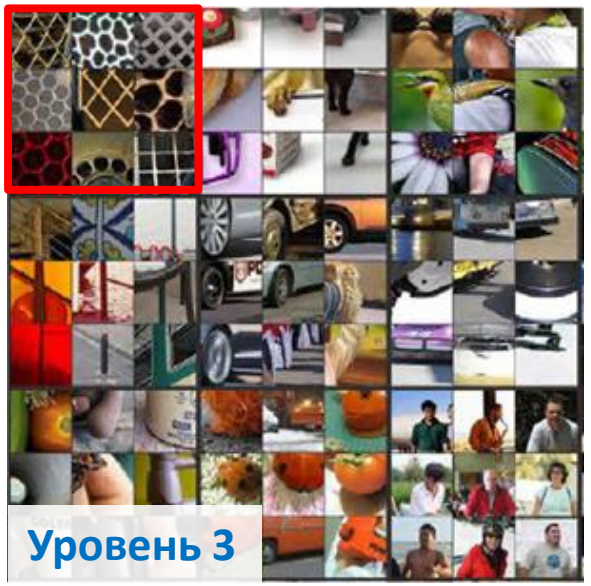
Компактификация (сжатие) базового многообразия

Полносвязная часть CNN (персептрон) работает с векторами

Глубина вектора признаков = размерности векторного подпространства

Convolution networks, Deep learning, Image Recognition





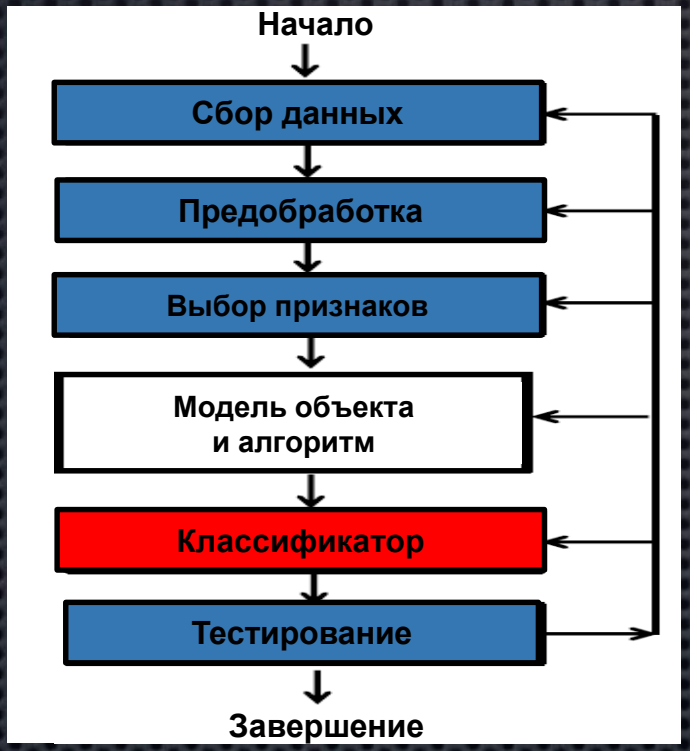
Какие элементы изображения распознают нейроны разных уровней:

чем выше слой сети, тем выше уровень абстракции

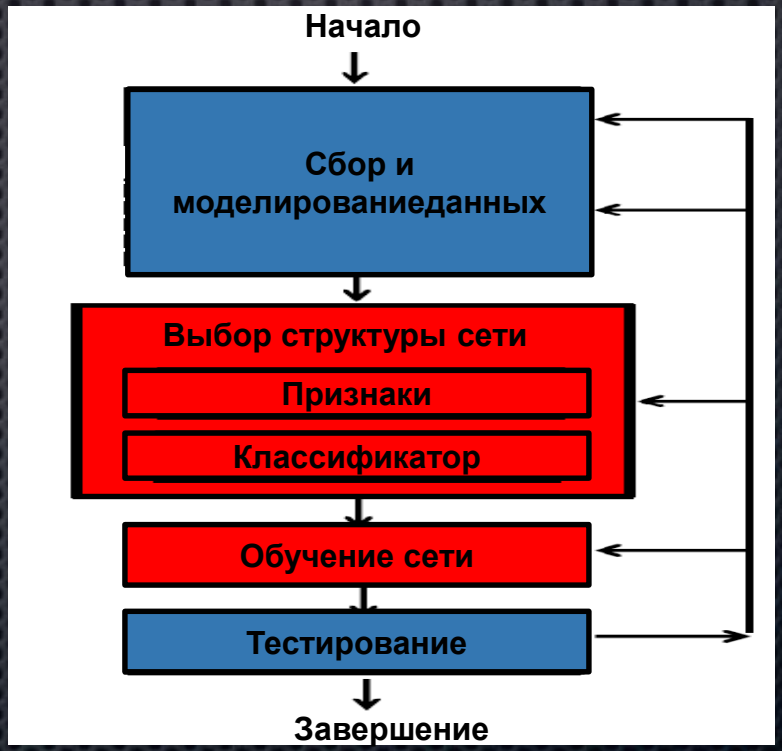
Visualizing and Understanding Convolutional Networks, Matthew Zeiler and Rob Fergus, ECCV'14

Глубокие конволюционные нейронные сети – новое поколение алгоритмов обработки и анализа изображений

- Глубокое обучение это **автоматизированная технология разработки алгоритмов обнаружения и распознавания**

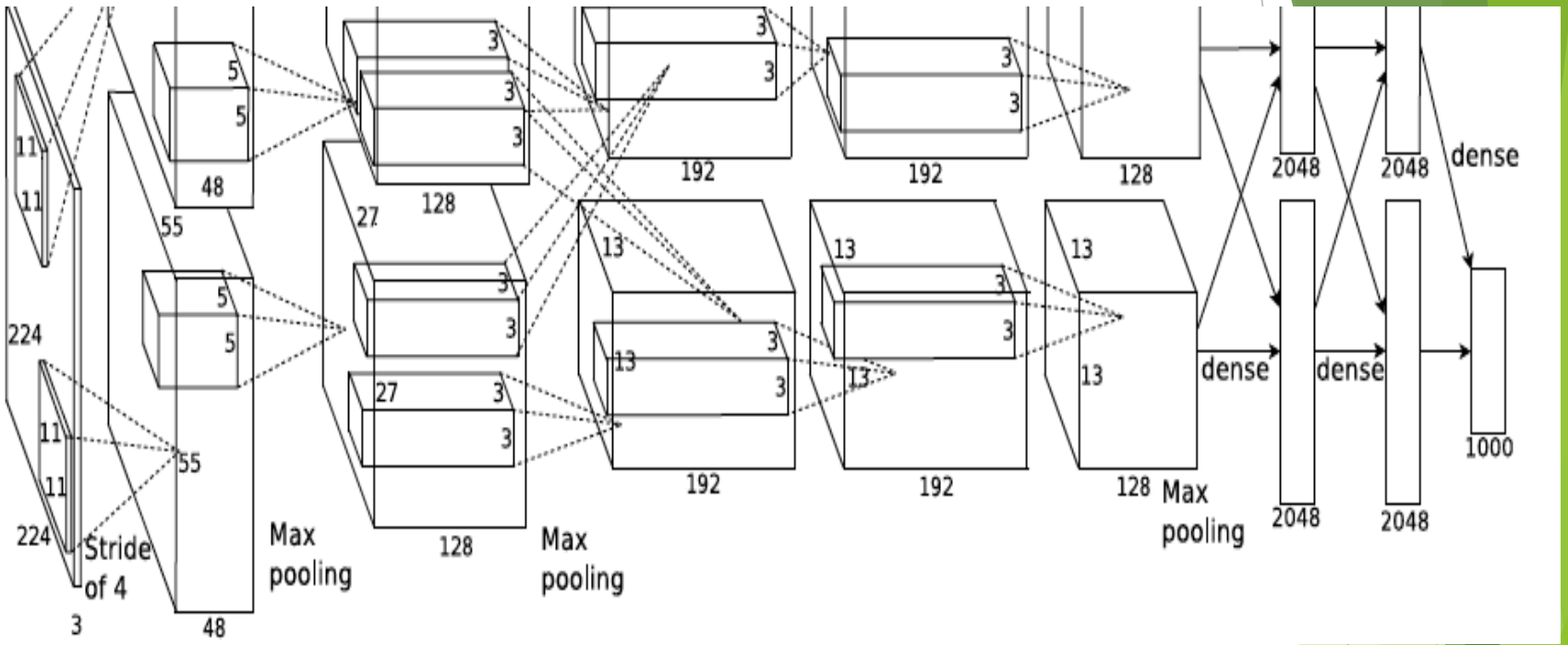


Этапы разработки классического алгоритма КЗ 2-го поколения (признаки и модели создает человек)



Этапы разработки алгоритма КЗ на основе глубокой конволюционной сети (все элементы формируются автоматически)

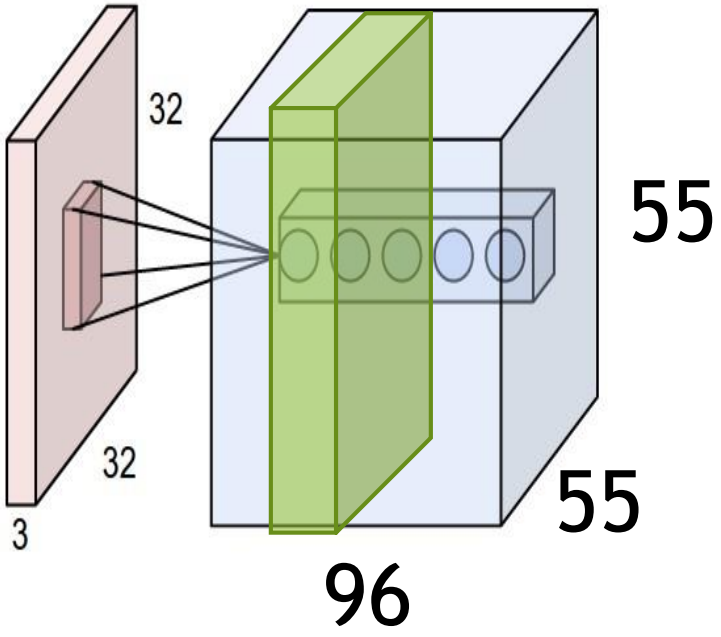
Convolutional Neural Networks (CNN)



Krizhevsky et al. [NIPS 2012]

7 hidden layers, 650,000 neurons, 60,000,000 parameters

Конволюционный слой

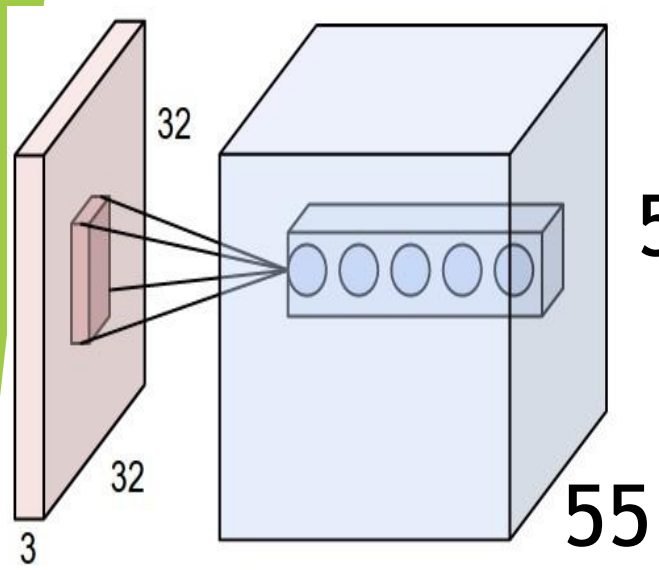


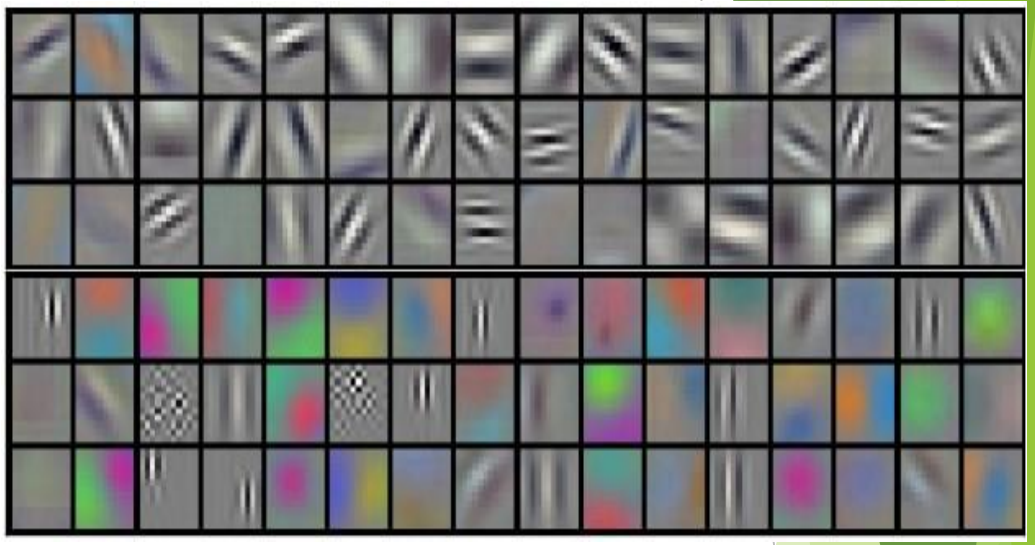
Всего 55*55*96 нейронов т.е. 290400 нейронов.

Если каждый нейрон связан с окрестностью 11x11 то всего 105,705,600 связей.

Все нейроны в пределах одного слоя глубины имеют одинаковые параметры (веса синапсов и пороги)

Фильтры

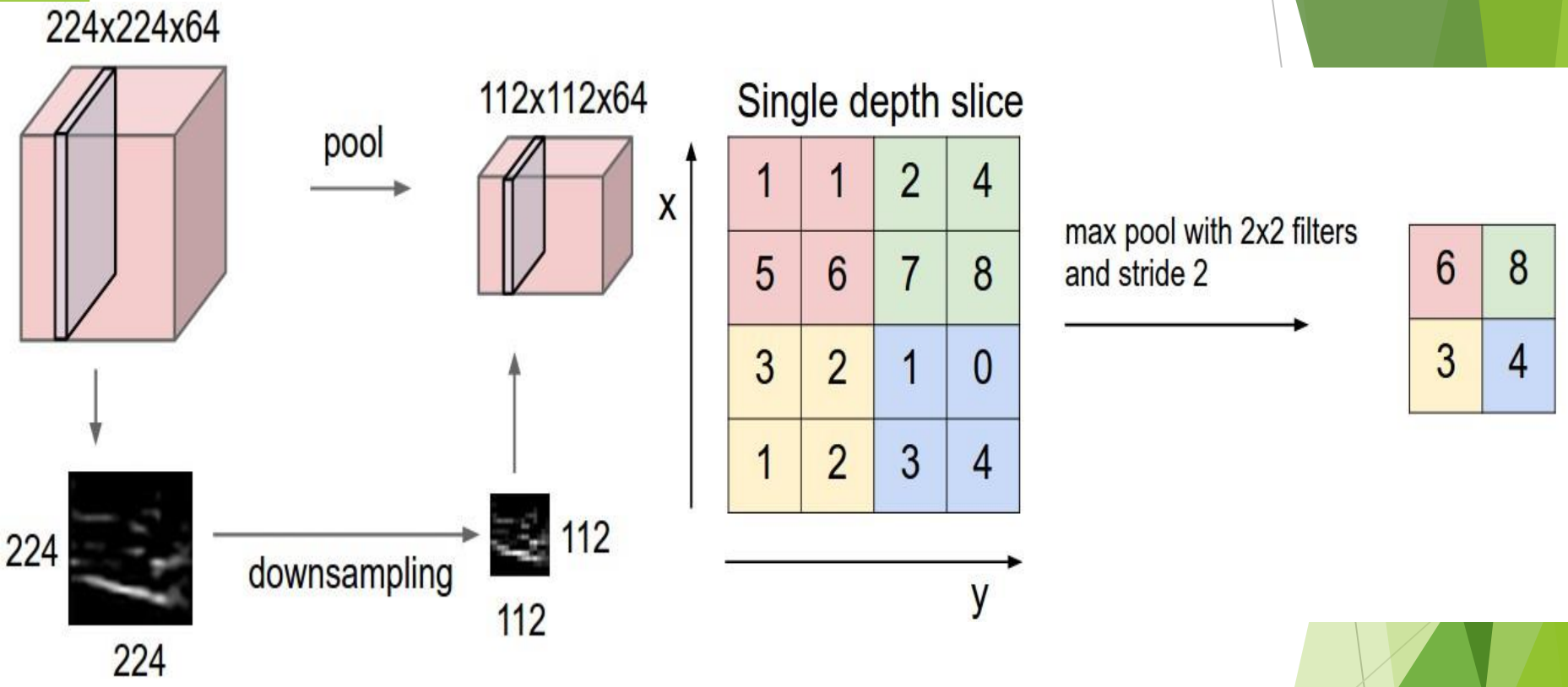




В данном случае есть только 96 уникальных наборов параметров

Каждый такой набор называется *фильтром* или *ядром*

Пулинг

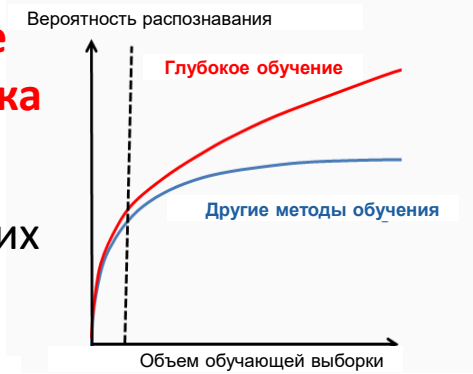


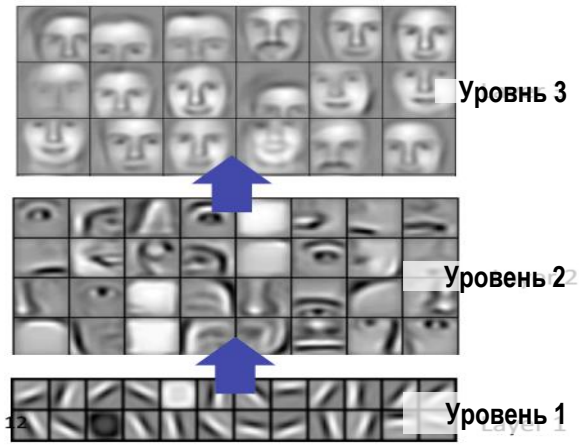
Слой пространственного объединения нужен для снижения размерности и уменьшения точности пространственной локализации признаков

Автоматическое обнаружение и распознавание объектов *на базе глубоких конволюционных нейронных сетей (с 2011)*

+ С 2011 г. - **распознавание образов на уровне человека или выше** (superhuman)

+ Обучение на сверхбольших объемах данных



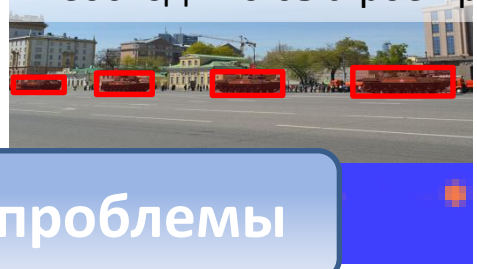


+ Иерархическое обучение с повышением абстракции данных от уровня к уровню

- **Нужны огромные обучающие выборки**
- Длительное моделирование и обучение

Постановка задачи обнаружения и Распознавания объектов заданного типа → Формирование «комплекта обучения» на основе реальных и модельных данных → Обучение алгоритмов обнаружения и глубокой нейронной сети

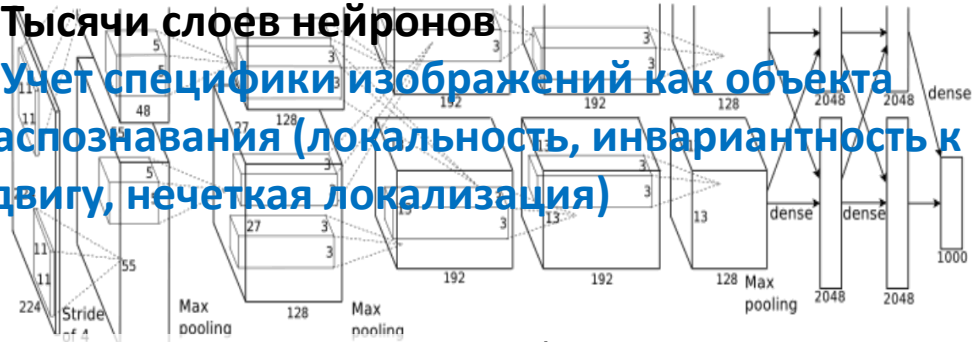
- **Ресурсоемкость, низкая скорость**
- Необходимо быстрое предобнаружение



Достоинства и **проблемы**

+ Тысячи слоев нейронов

+ Учет специфики изображений как объекта распознавания (локальность, инвариантность к сдвигу, нечеткая локализация)



Типовая структура глубокой конволюционной сети

- Необходимость эффективных алгоритмических реализаций
- Необходимость создания нового поколения нейропроцессоров

Спасибо за внимание!