Резюме мешка слов

Яндекс

- «Мешок слов» оказался очень эффективным иснтрументов для распознавания изображений
- Часто используются совместно и разреженная и плотная версии:
  - Мешок слов по характеристическим точкам
    - Bags of visual words
  - Плотные слова
    - Dense words (PhowGray, PhowColor)

- Реализация – библиотека VLFeat
  http://www.vlfeat.org/

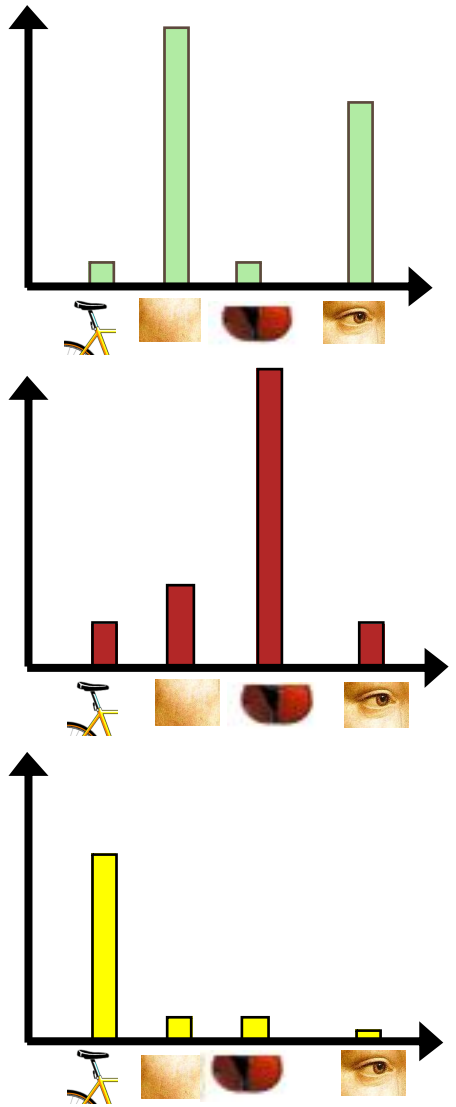
Резюме признаков

Яндекс

- Гистограммы – широко используемый метод описания распределения признаков в изображении (но не единственный)
- Гистограмму можем строить по регулярной сетке, а можем адаптивно, с помощью кластеризации
- Можем вычислить разные признаки:
  - Гистограммы цветов
  - Гистограммы градиентов
  - Гистограммы разреженных визуальных слов
  - Гистограммы плотных визуальных слов
- Каждый вид признаков можем считать по всему изображению, а можем по области
  - Часто используется пирамида разбиений из 3х уровней (1 + 4 + 16 гистограмм)

Категоризация изображений

- Описывать изображение научились, теперь нужно его категоризировать
  - Nearest Neighbour (+kNN)
  - SVM
  - Random Forest
  - Boosting
  - Neural Networks



- Простейший вариант – NN или SVM

Функции сравнения гистограмм

Яндекс

- Histogram intersection

(нормализованные гистограммы)

$$D(h_i, h_j) = 1 - \sum_{m=1}^{K} \min\left(h_i(m), h_j(m)\right)$$

- L1 distance

$$D(h_1, h_2) = \sum_{i=1}^{N} | h_1(i) - h_2(i) |$$

- χ^2 distance

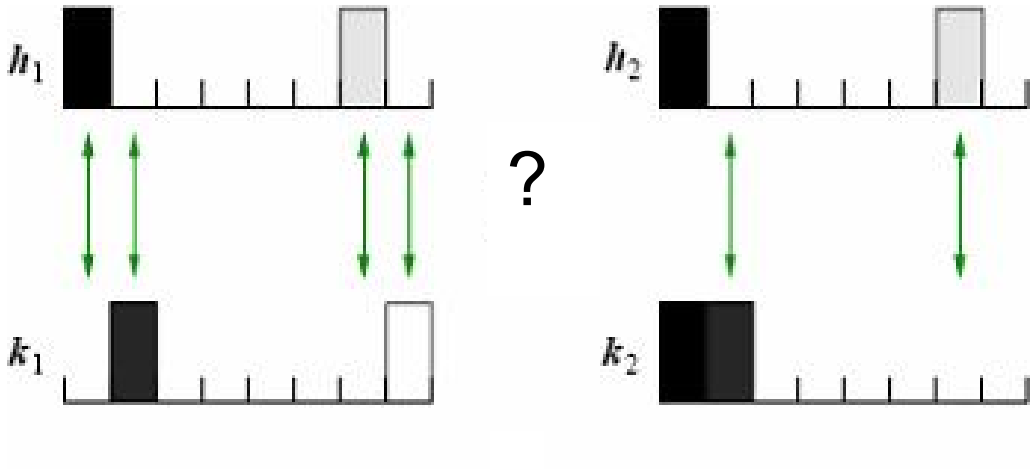
$$D(h_1, h_2) = \sum_{i=1}^{N} \frac{\left(h_1(i) - h_2(i)\right)^2}{h_1(i) + h_2(i)}$$

- Quadratic distance (*cross-bin*)

$$D(h_1, h_2) = \sum_{i,j} A_{ij} \left(h_1(i) - h_2(j)\right)^2$$

Jan Puzicha, Yossi Rubner, Carlo Tomasi, Joachim M. Buhmann: Empirical Evaluation of Dissimilarity Measures for Color and Texture. ICCV 1999

Проблемы сравнения гистограмм



- Все метрики сравнения гистограмм чувствительны к размеру ячеек
- Можно использовать более широкие ячейки, но при этом теряем «разрешающую способность»

Обучения классификатора

Яндекс

- Воспользуемся методом SVM
- Поскольку вектор-признак – гистограмма, нам потребуются специальные ядра
- Ядро пересечения гистограмм:

$$I(h_1, h_2) = \sum_{i=1}^{N} \min(h_1(i), h_2(i))$$

- Обобщенное Гауссово ядро:

$$K(h_1, h_2) = \exp\left(-\frac{1}{A} D(h_1, h_2)^2\right)$$

- *D* может быть евклидовым расстоянием, χ^2 , и т.д.

J. Zhang, M. Marszalek, S. Lazebnik, and C. Schmid, Local Features and Kernels for Classifcation of Texture and Object Categories: A Comprehensive Study, IJCV 2007

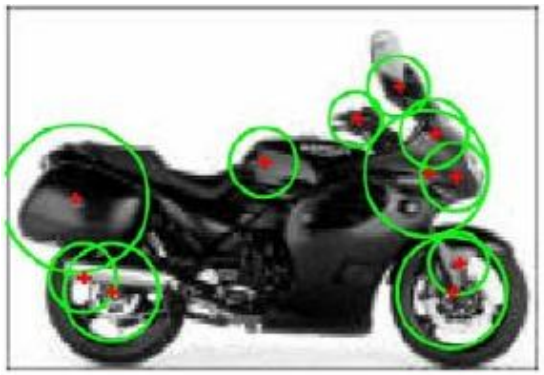
Резюме

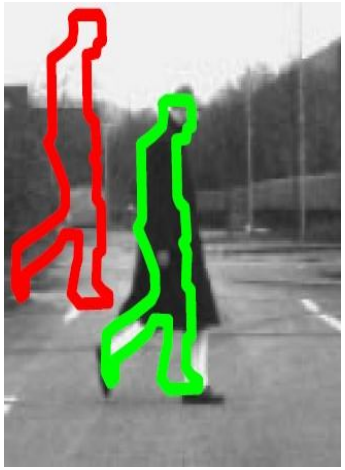
Яндекс

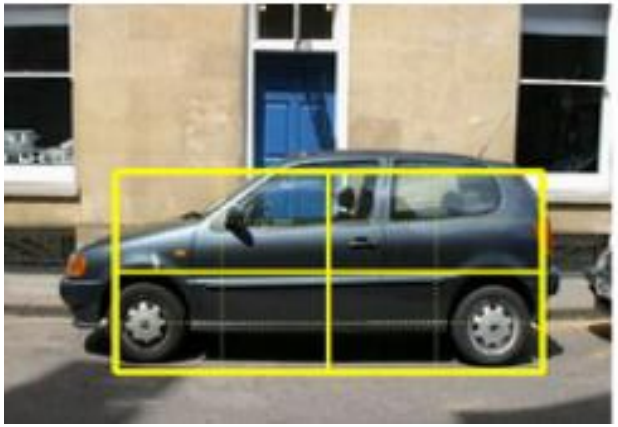
- Простая схема на основе машинного обучения
  - Придумываем вектор-признак для изображения
  - Обучаем классификатор на обучающей выборке
- В качестве признаков чаще всего используют гистограммы распределений цвета, градиентов, краёв, визуальных слов
- Любой метод классификации на основе машинного обучения можно использовать
- Текущие результаты не идеальны, задача в процессе исследования

Выделение объектов

Яндекс



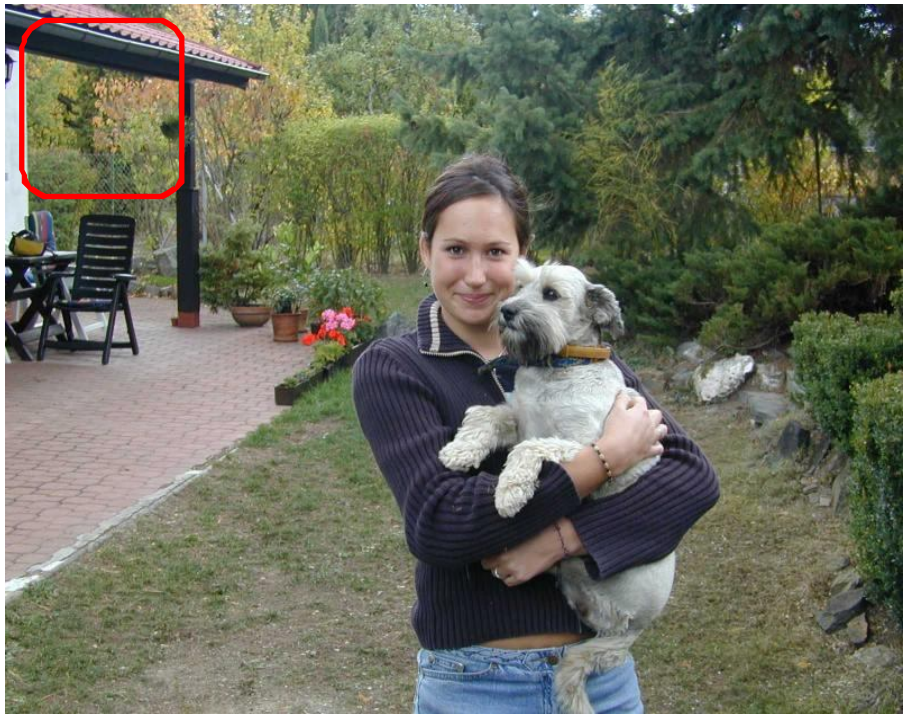




Скользящее окно

Яндекс

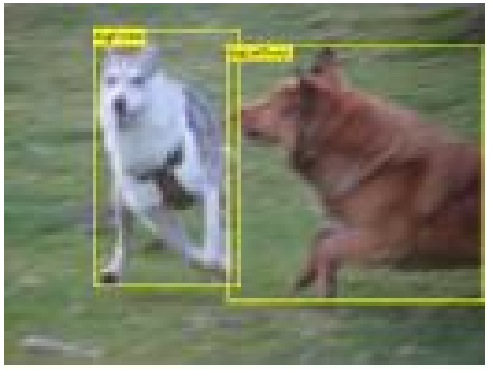
Сам принцип – очень хороший



- Рассматриваем только прямоугольную область (окно)
- Если выбрали точно, то в эту область попадёт только лицо
- Просмотрим все области, «сканируя» окном изображение

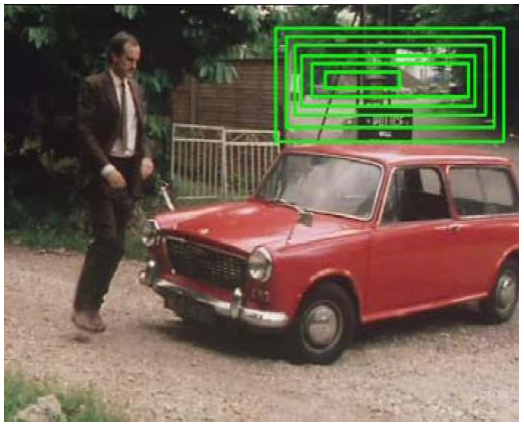
Проблемы скользящего окна

Разный размер объектов



Соотношение размеров окна

Перекрытие и плохое соответствие формы объекта окну



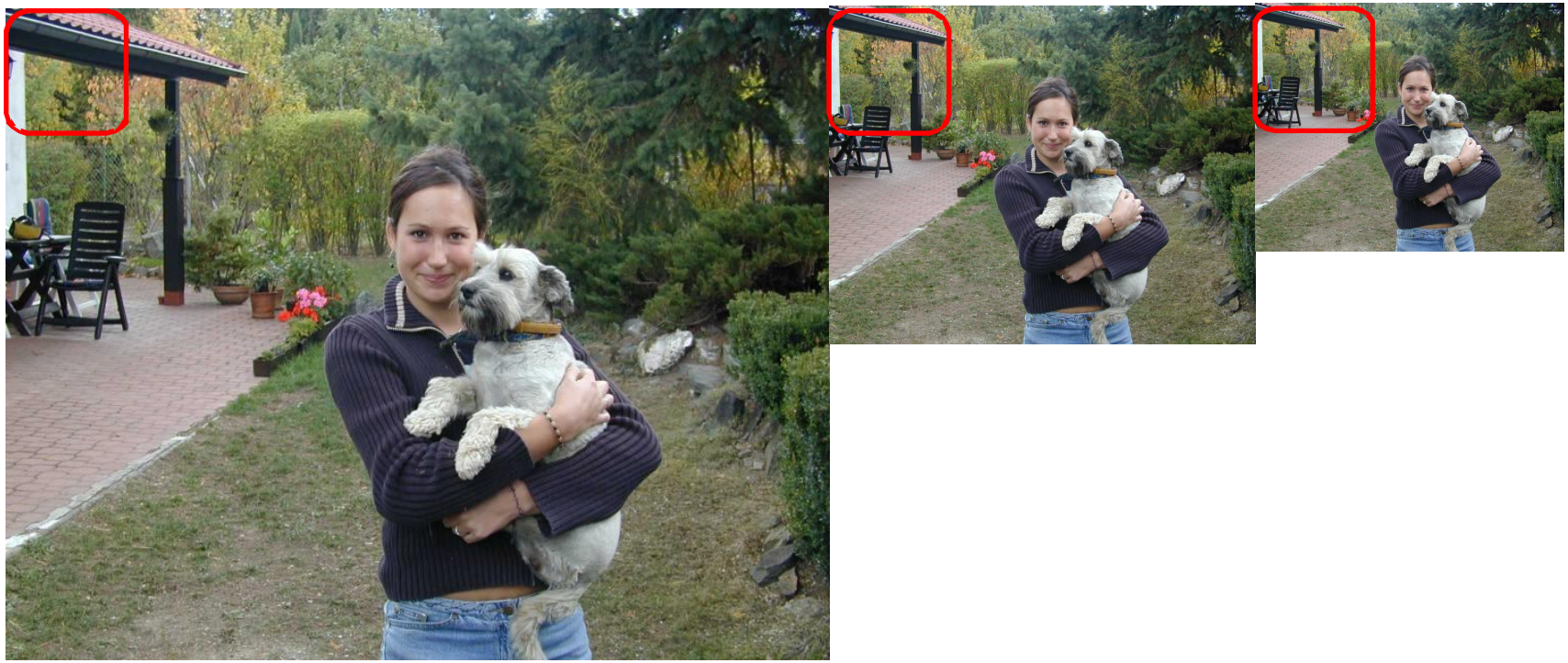
Множественные отклики

Разный размер объектов

Сканируем изображение рамками разного размера и разных пропорций

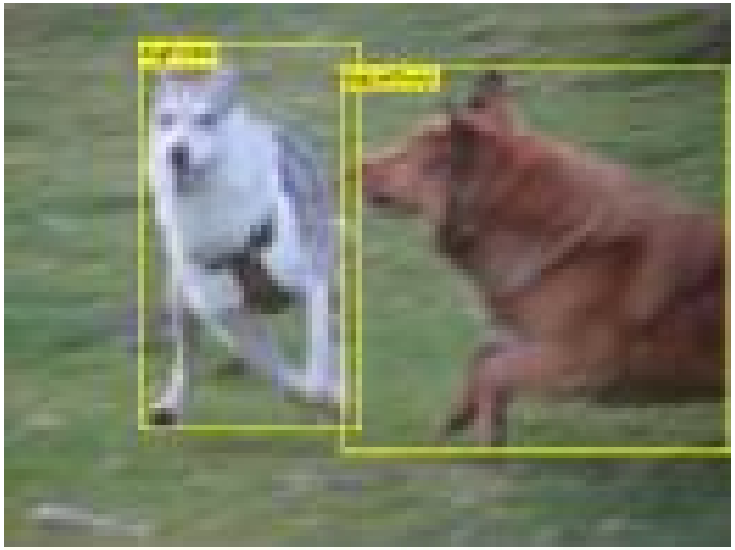
Разный размер объектов

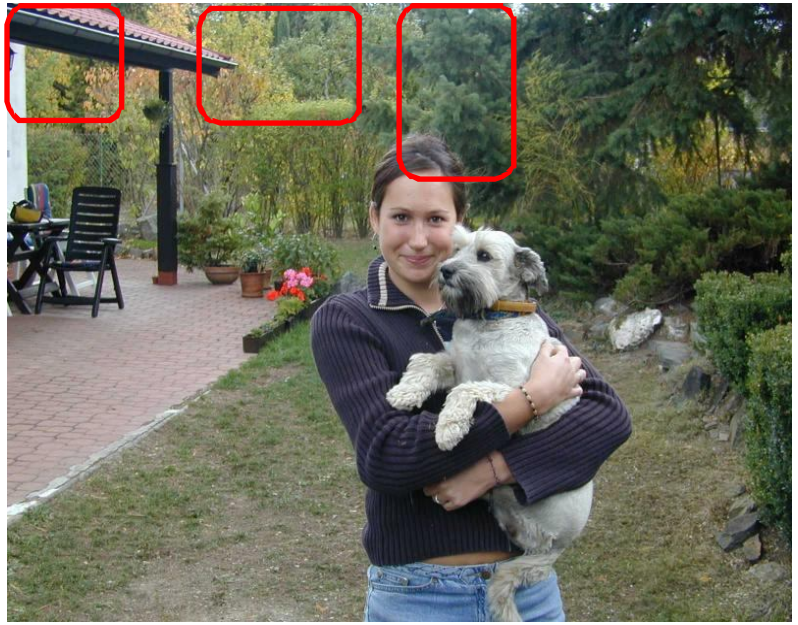
Яндекс



- Вместо изменения размеров рамки можем построить «пирамиду» изображений разных размеров и сканировать одной рамкой
- Сколько потребуется масштабов?
  - На практике до 50-70

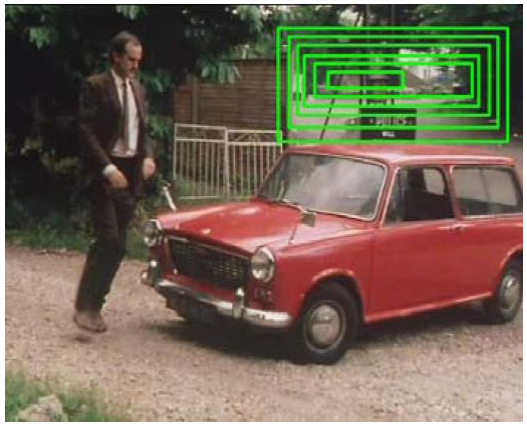
Пропорции объектов



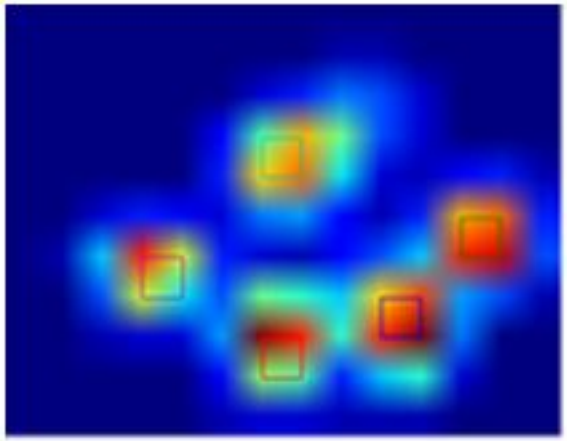


- Будем сканировать рамками разных пропорций
- С учетом масштабов нам придётся сканировать $N_{рамок}$ x $N_{масштабов}$ раз
- Поэтому общее количество сканирований может достигать 150-200

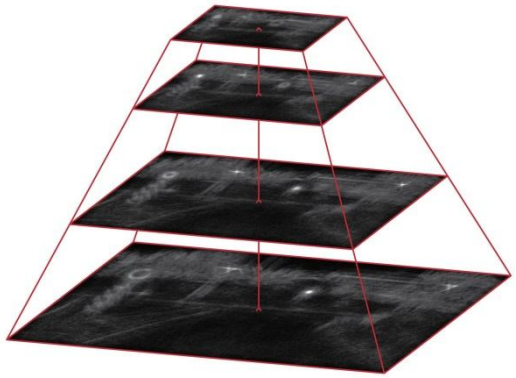
Множественные отклики



Множественные отклики



Карта откликов

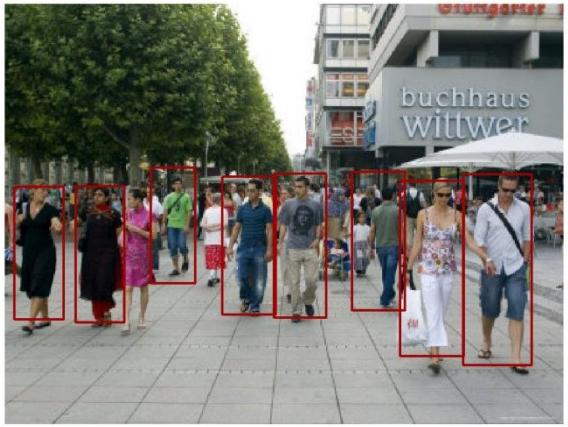


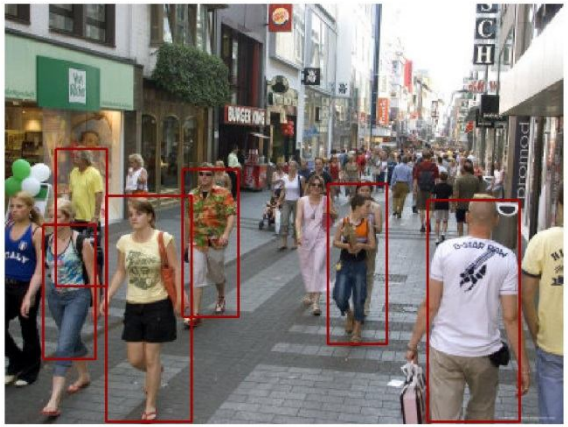
Пирамида изображений

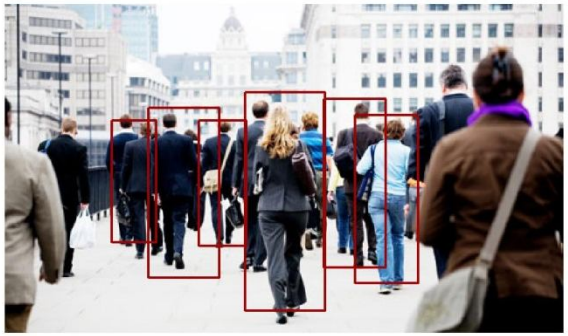
- В одном масштабе может быть множество откликов разной силы
- Мы выберем из них точки, в которых значение достигает локального максимума («Подавление не-максимумов»)
- При наличии нескольких масштабов нужно выбирать максимальные значения по 3х мерной области

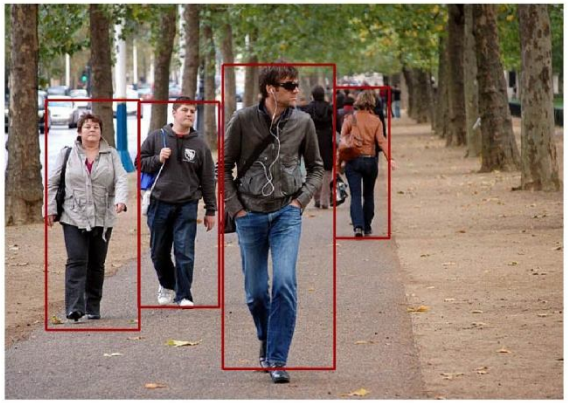
Перекрытия

Яндекс



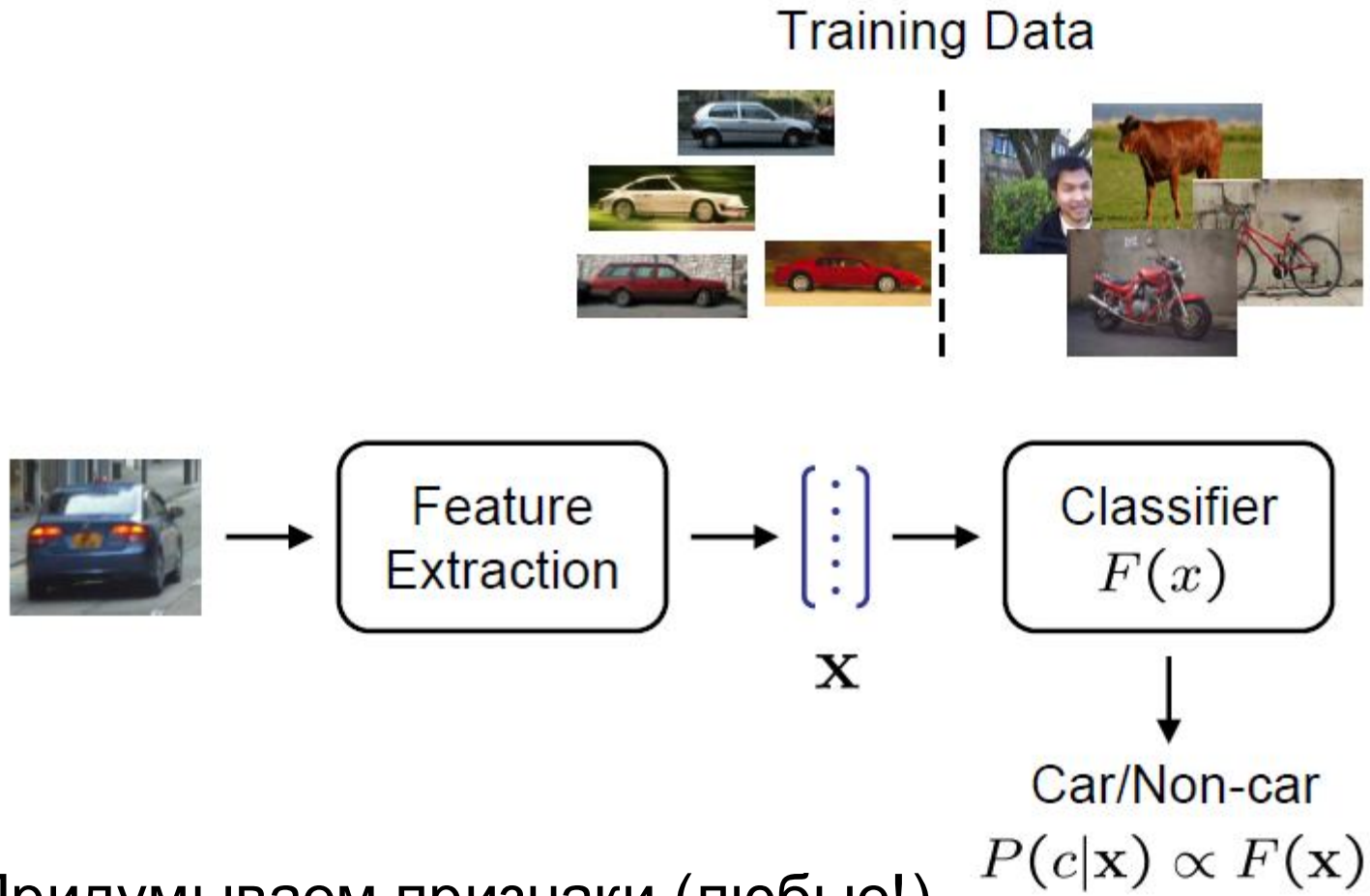






Самая сложная проблема!

Классификатор для окна



- Придумываем признаки (любые!)
- Обучаем классификатор (любой!)
- Можем взять мешок слов + SVM

Всё в кучу!

Яндекс

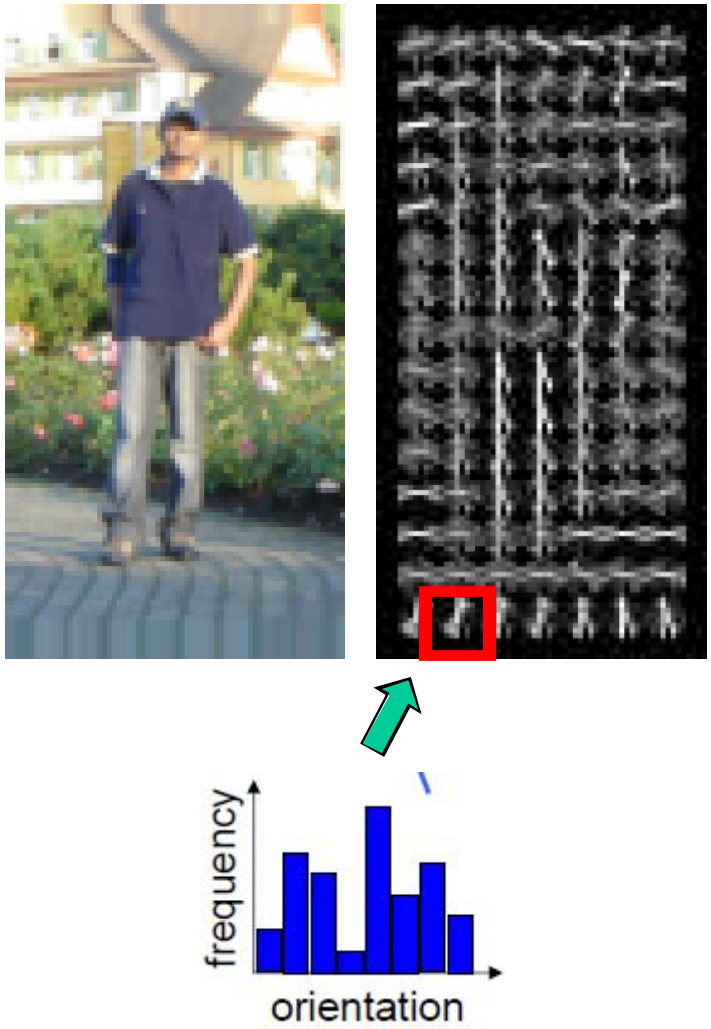
- Всё, до чего руки дотянутся:
  - Визуальные слова (разреженные)
  - Плотные визуальные слова
    - По серым изображениям
    - По цветным изображениям
    - На нескольких масштабах
  - Гистограммы ориентаций краёв Canny
- Забъём в нелинейный SVM с ядром *χ2-RBF*
- Кто первый успел – того и статья

Andrea Vedaldi, Varun Gulshan, Manik Varma, Andrew Zisserman **Multiple Kernels for Object Detection,** ICCV 2009

HOG

Яндекс

- Уберём промежуточный этап квантования фрагментов, и просто посчитаем большой SIFT для всего окна
  - Histogram of oriented gradient
- Идея была изначально предложена для пешеходов, но затем стала широко применяться
- Прямоугольное окно 64 x 128 пикселов разобьём на ячейки (cells) 8 x 8 пикселей
- В каждой ячейке посчитаем гистограмму ориентаций градиентов (8 корзин)



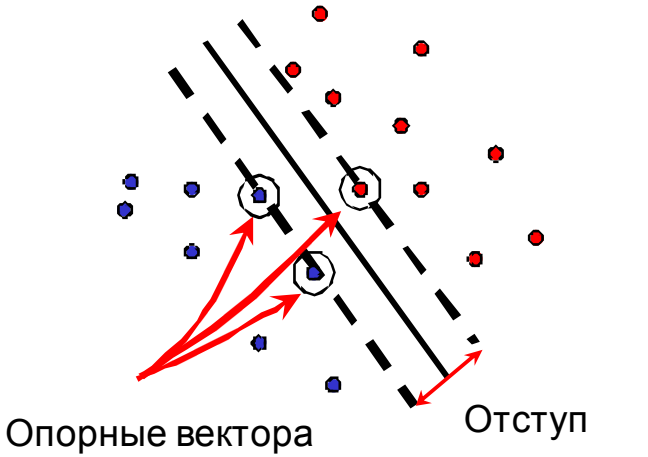
Navneet Dalal, Bill Triggs,Histograms of Oriented Gradients for Human Detection, CVPR 2005

Линейный SVM

• Обучаем бинарный линейный классификатор SVM :

$$f(x) = w^T x + b$$

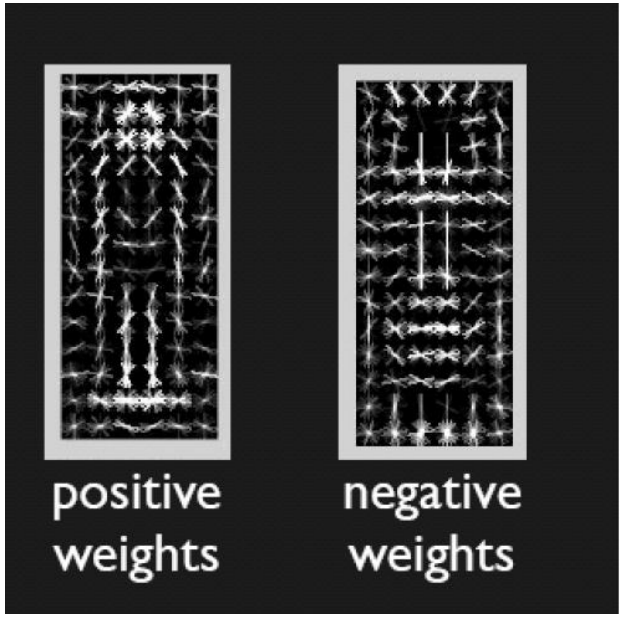
$$\mathbf{w} = \sum_i \alpha_i y_i \mathbf{x}_i$$



• Опорные вектора с положительными и отрицательными весами

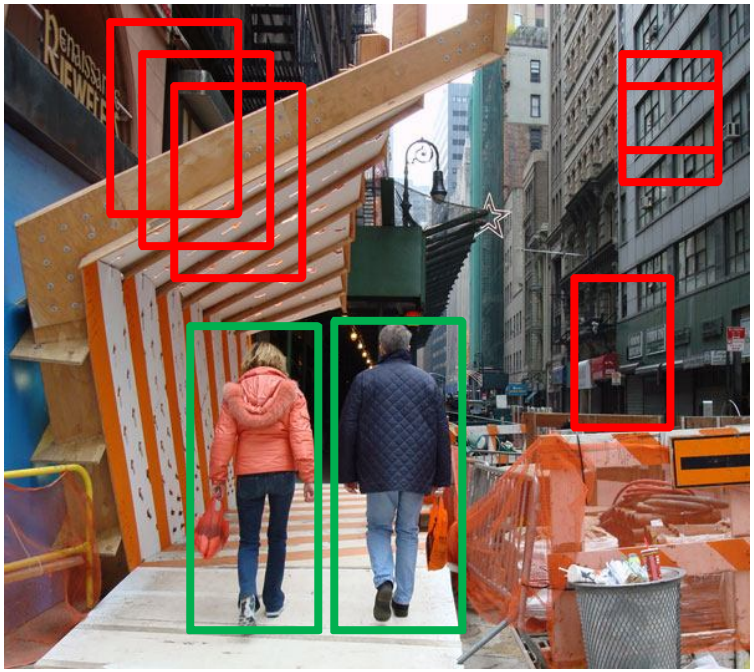
• Чем фактически в нашем случае являются x ?

• Каждый опорный вектор – это один «трудный» пример, конкретный пешеход или фон



Обучение детектора

- Сколько на изображении объектов «пешеход» и сколько фрагментов фона?



- Выделение объектов ассиметричная задача: объектов гораздо меньше, чем «не-объектов»
- Вдобавок, класс «не объект» очень сложный – нужно много разных данных для обучения
- Для SVM желательно одинаковое количество и фона, и объекта

Пример – поиск «торса»

Яндекс

- Хотим построить детектор «верхней части тела и головы»
- Воспользуемся схемой HOG + линейный SVM
- Данные
  - 33 фрагмента фильмов из базы Hollywood2
  - 1122 кадров с размеченным объектами
- На каждом кадре отмечены 1-3 человека, всего 1607 людей, это маловато



Положительные окна

Яндекс

Посмотрим, что отметили люди при разметке:

Внимание: похожие положение и ориентация!

Искаженные примеры

Яндекс

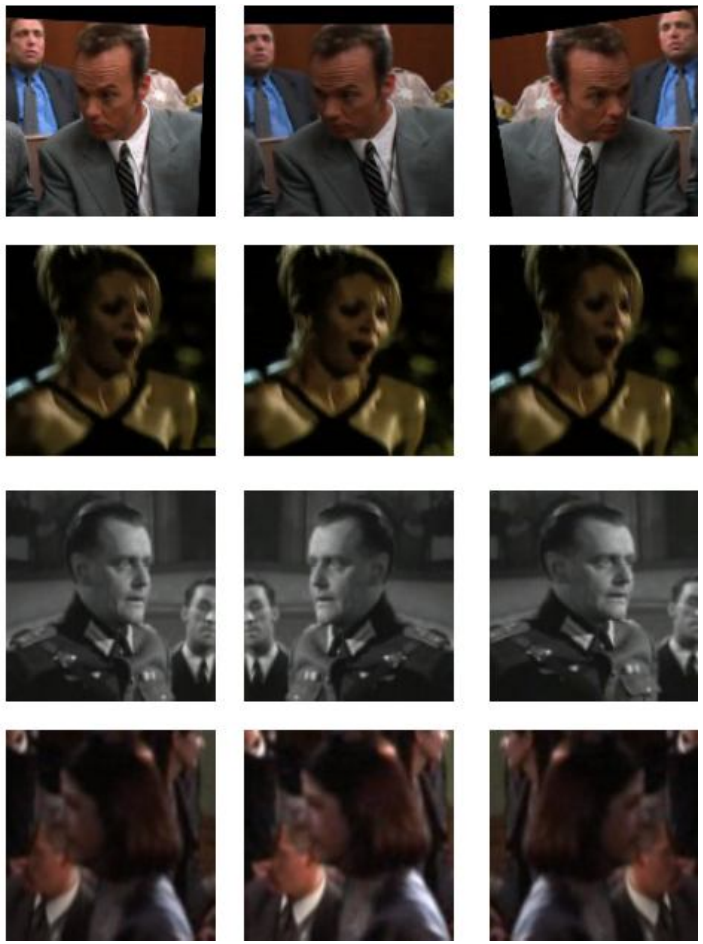
Давайте «размножим» данные, «пошевелив» их:

Небольшие сдвиги, отображения, повороты, изменения масштаба

Искаженные примеры

Яндекс

Из 1607 эталонных примеров получили ~32000 искаженных (jittered) примеров

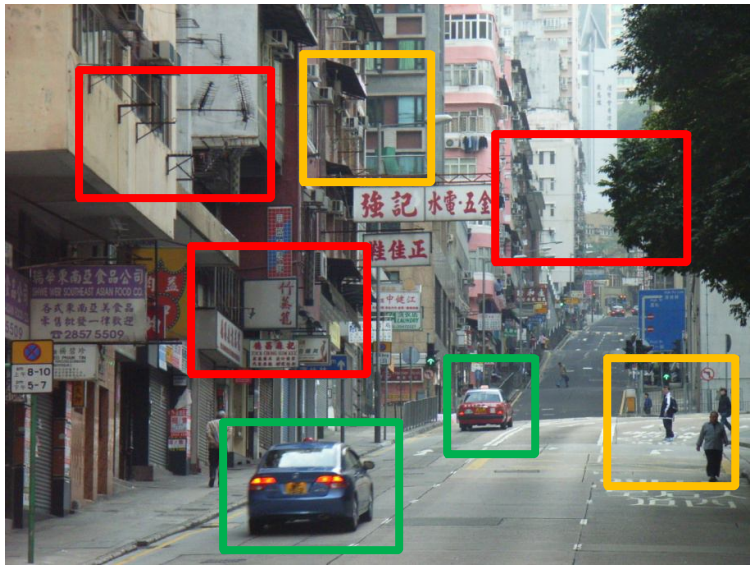


Сколько отрицательных примеров можно набрать из 1100 кадров?

- Гораздо больше 32k.

Вспомним SVM – нам нужны «трудные примеры» для фона. Как их найти, если мы всего можем выбрать ~32k для фона?

Бутстраппинг (Bootstrapping)



- Выбираем отрицательные примеры случайным образом
- Обучаем классификатор
- Применяем к данным
- Добавляем ложные обнаружение к выборке
- Повторяем

- Смысл:
  - Ложные обнаружения для первого детектора – сложные (**hard negative**)
  - Пусть наша выборка фона будет маленькой, но сложной и представительной

I. Laptev "Improvements of Object Detection Using Boosted Histograms« BMVC 2006

Случайные фрагменты фона

Яндекс

Элементы выборки фона для первой итерации:

Первая стадия

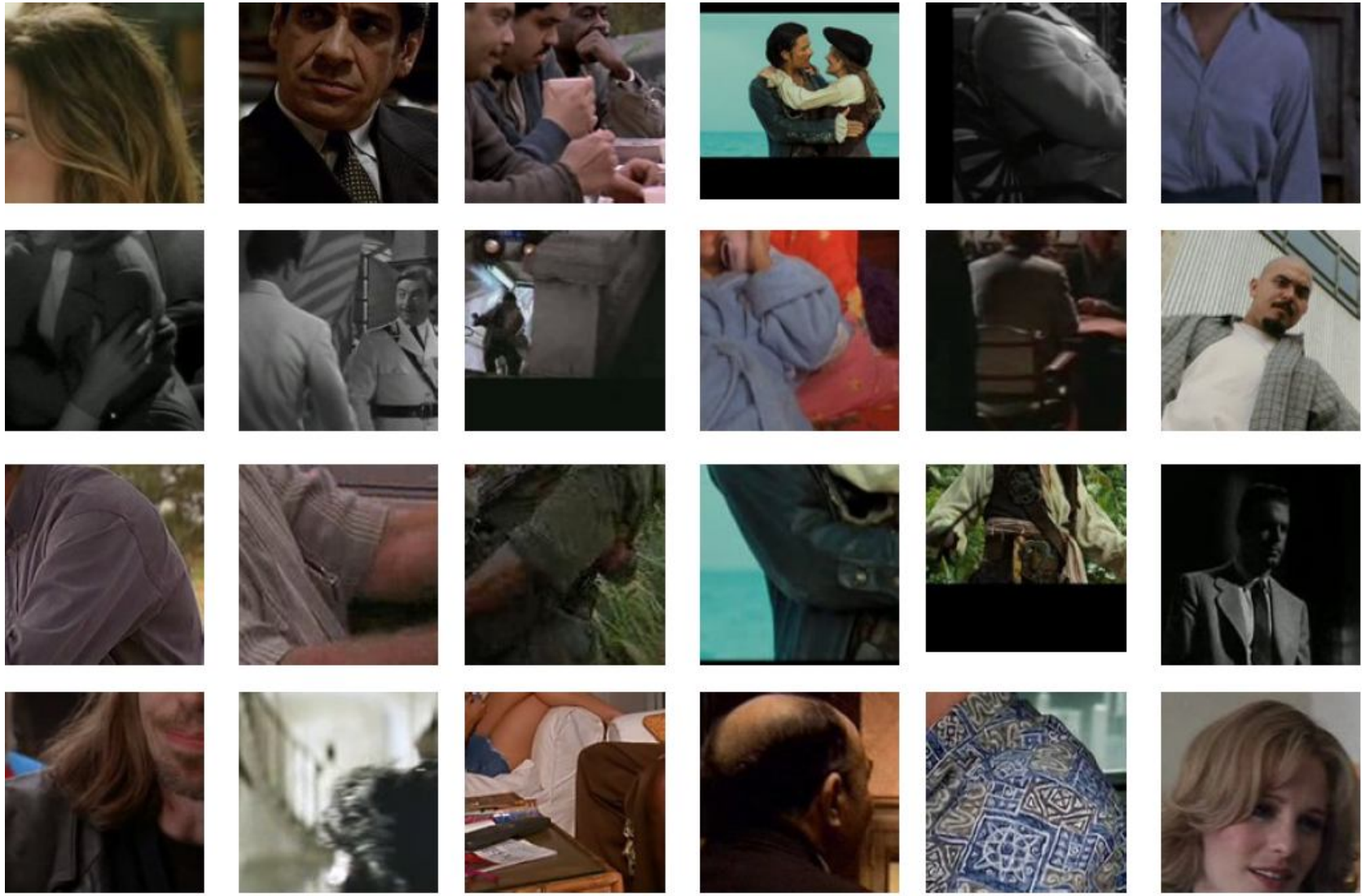
Яндекс

Трудный отрицательный пример

- Ищем ложные обнаружения с высоким рейтингом
- Используем их как трудные отрицательные примеры
- Затраты: # количество изображений x поиск в каждом

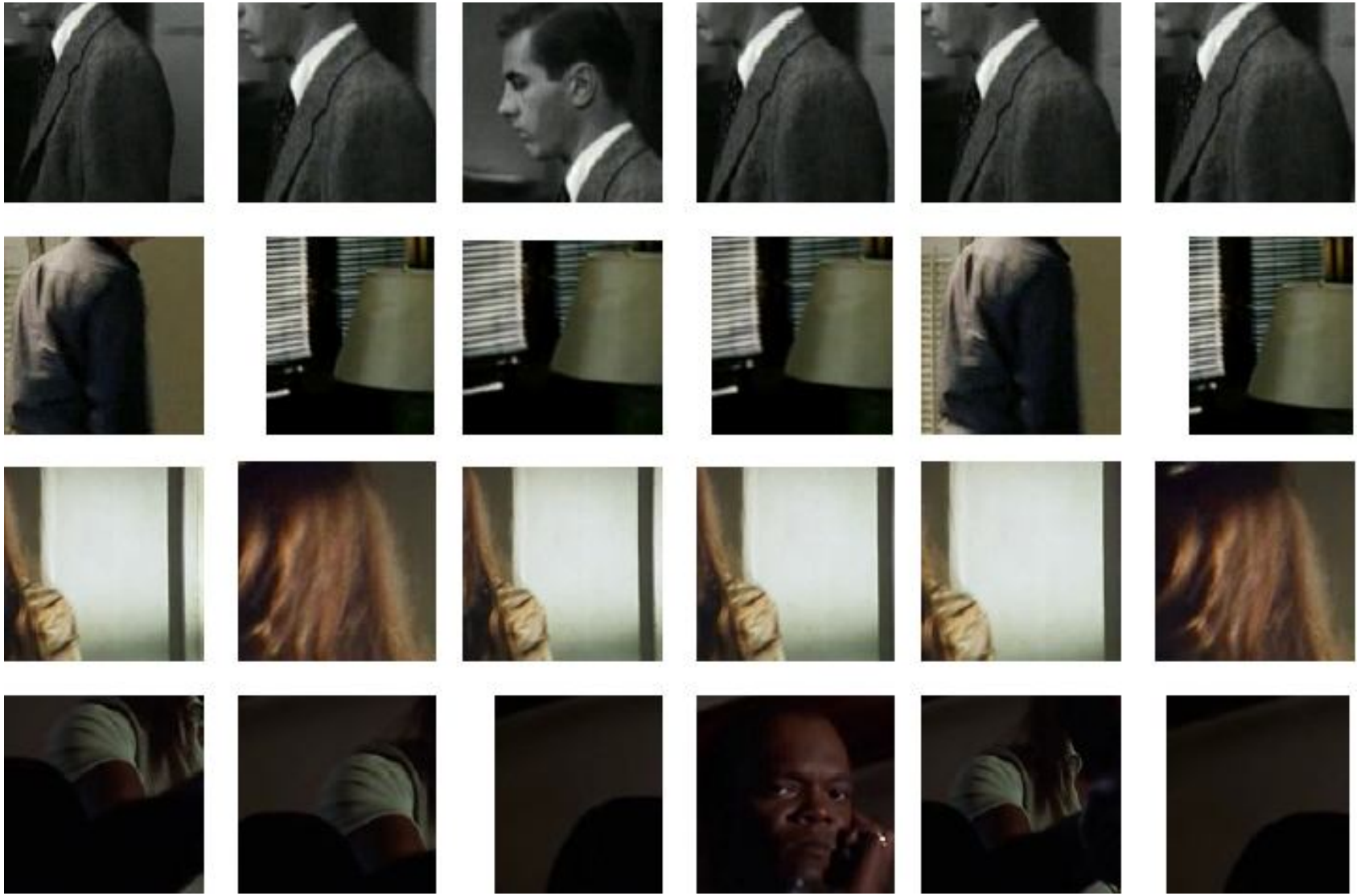
Трудные примеры

Яндекс



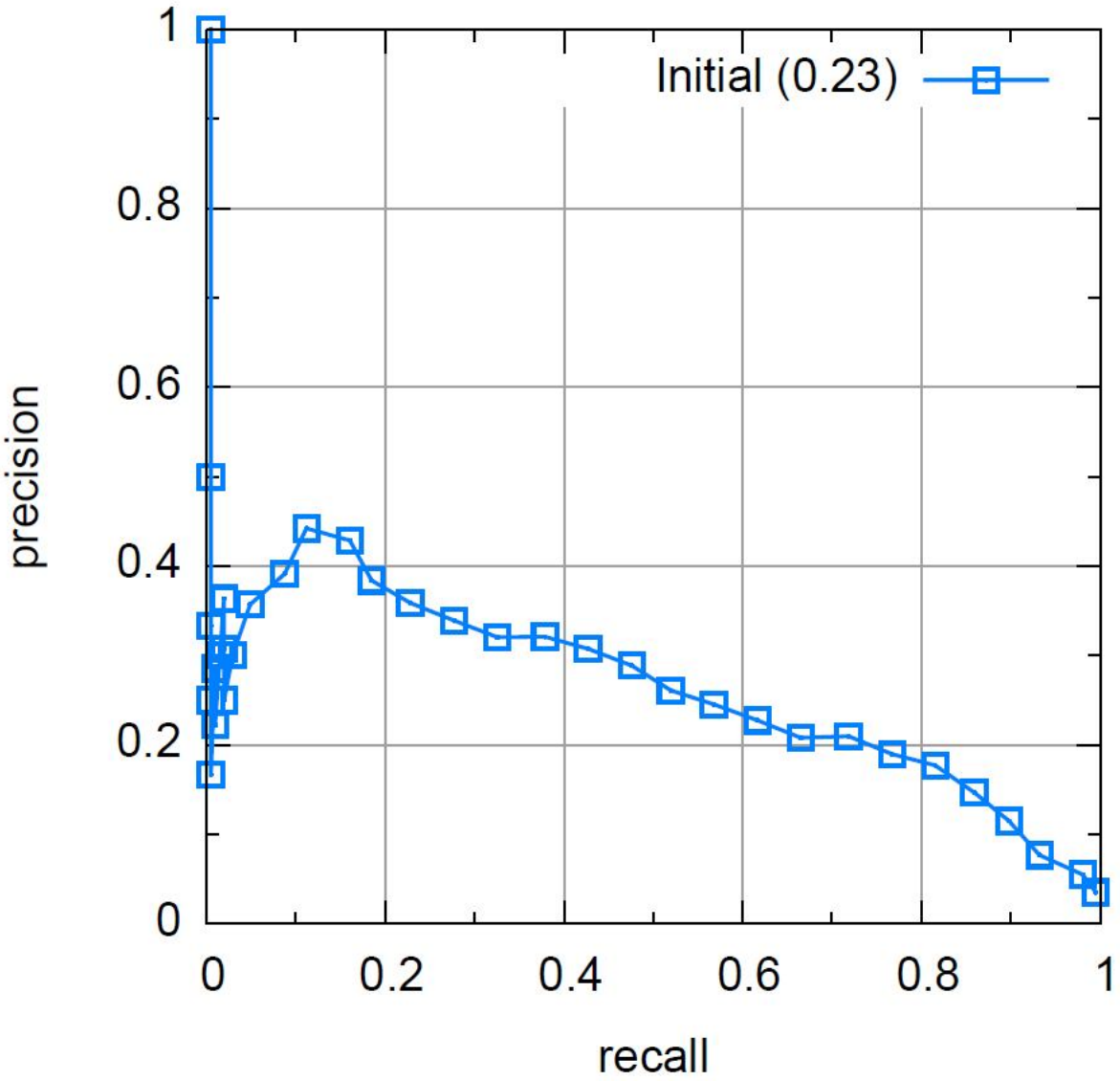
Трудные примеры

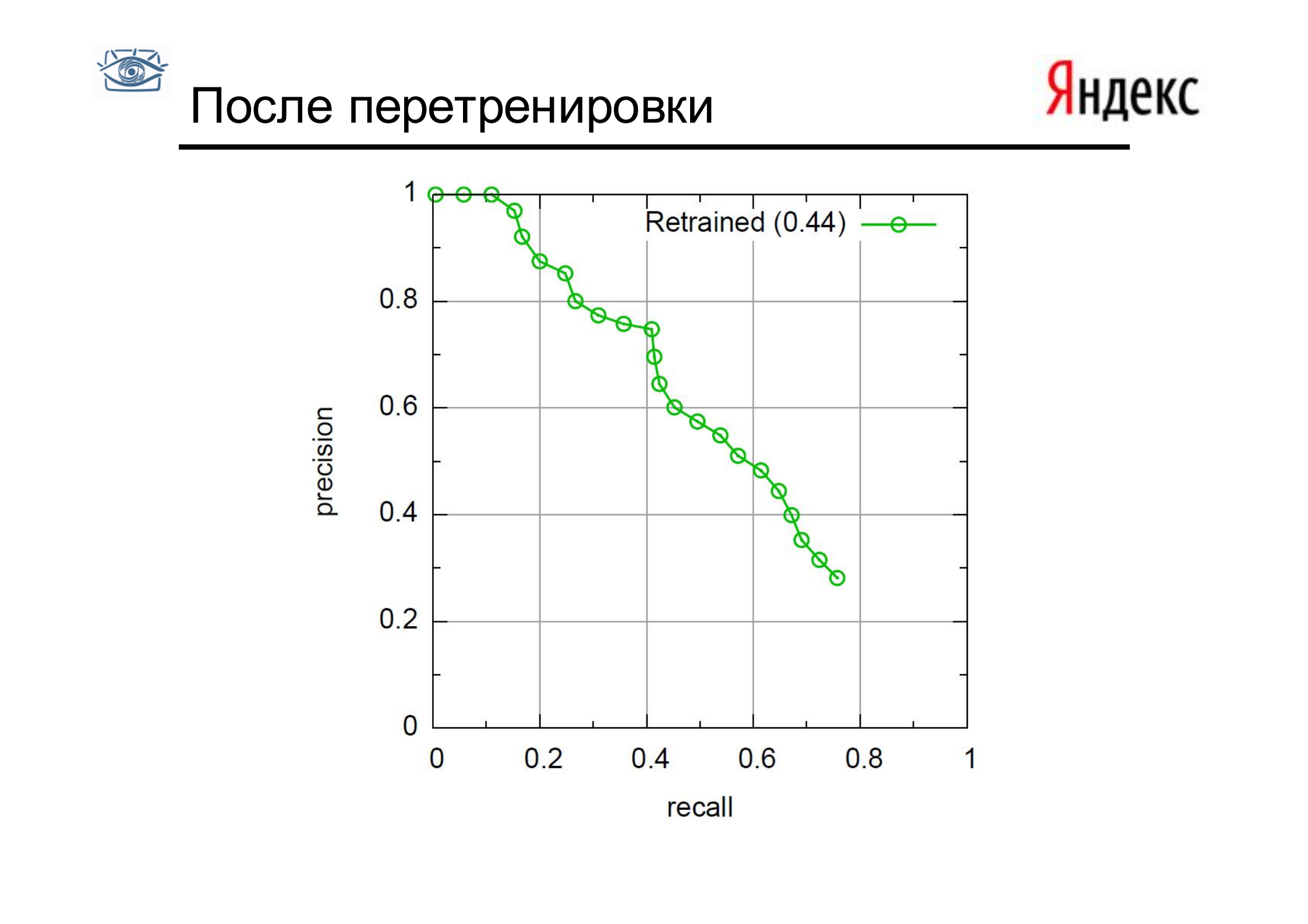
Яндекс



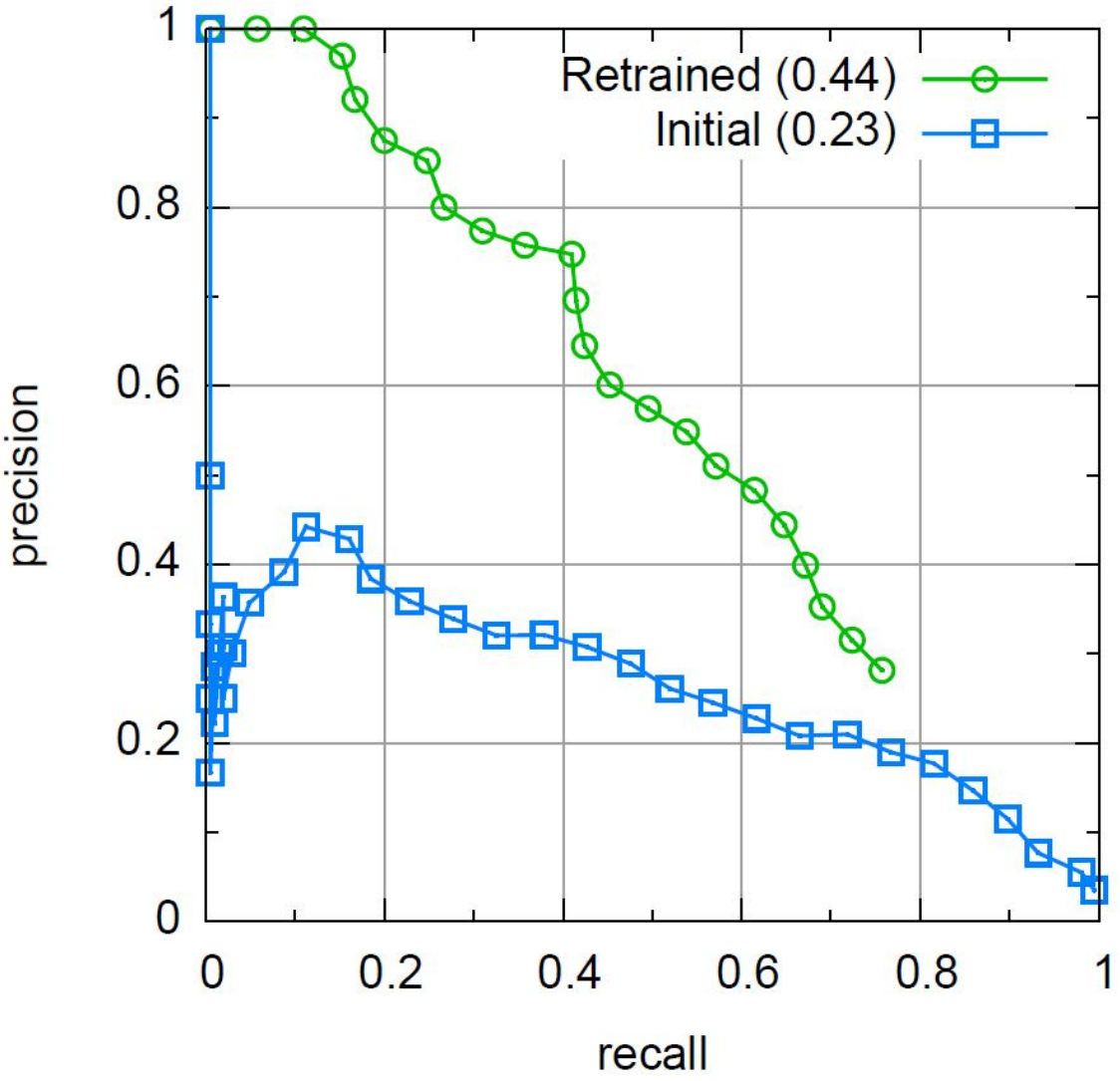
Измерение качества

Яндекс



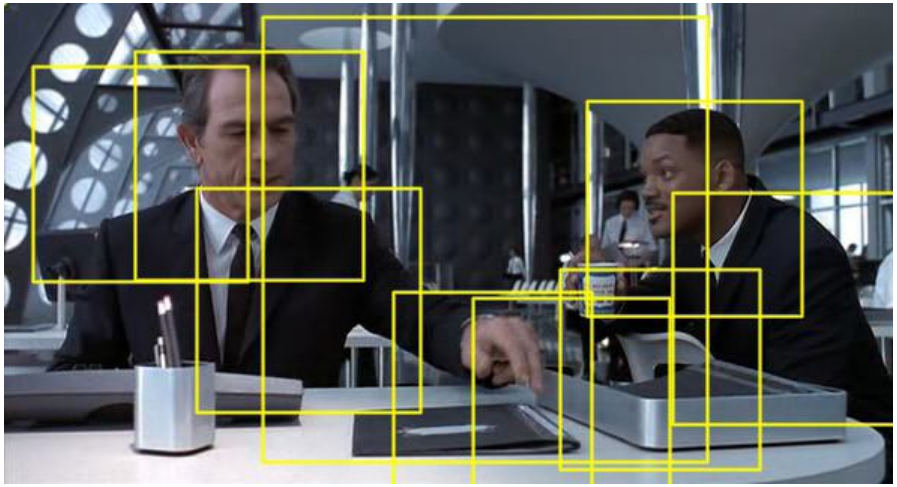

После перетренировки
Яндекс
Retrained (0.44)
precision
recall
0
0.2
0.4
0.6
0.8
1

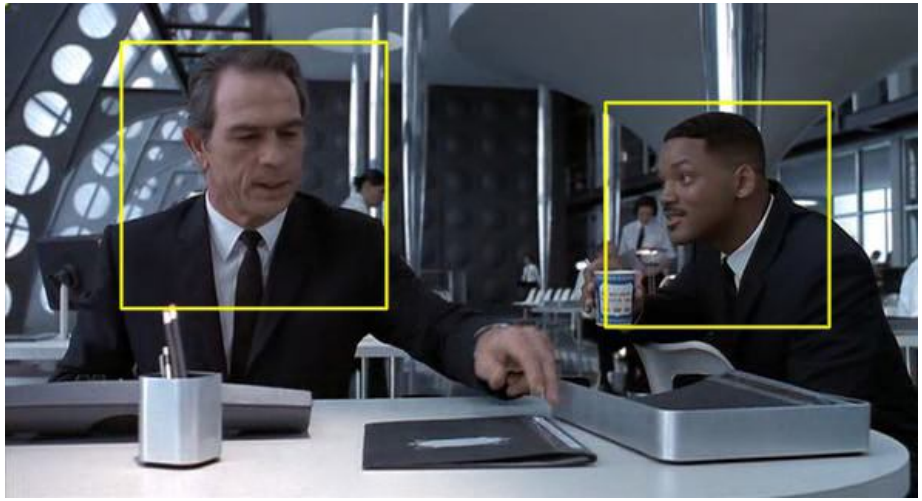
Сравнение

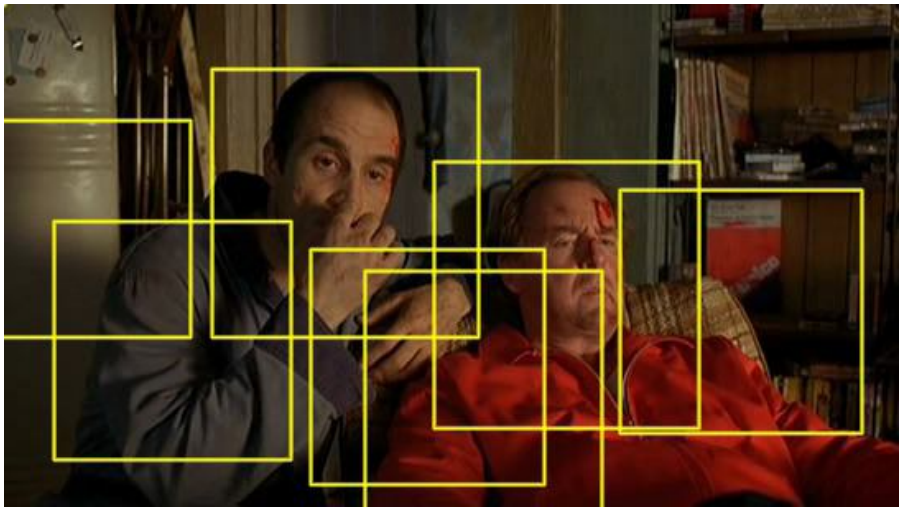


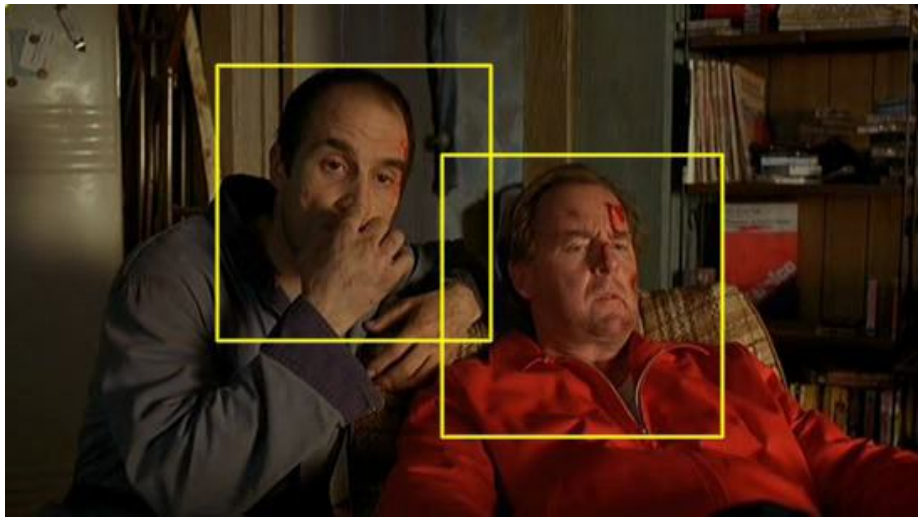
Сравнение

Яндекс



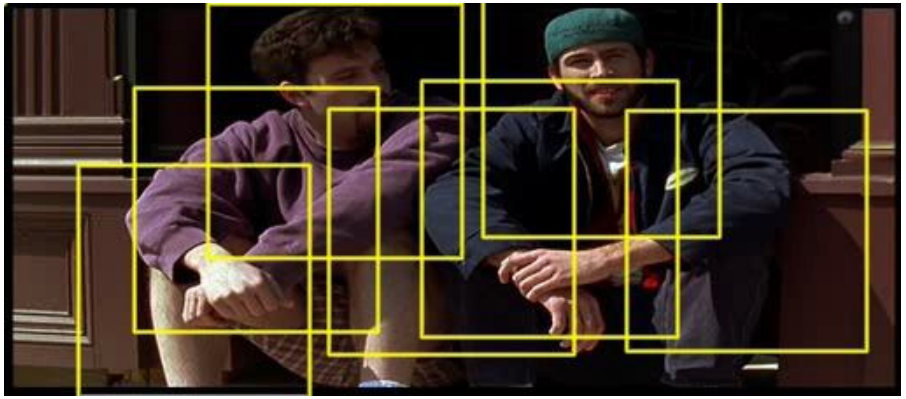


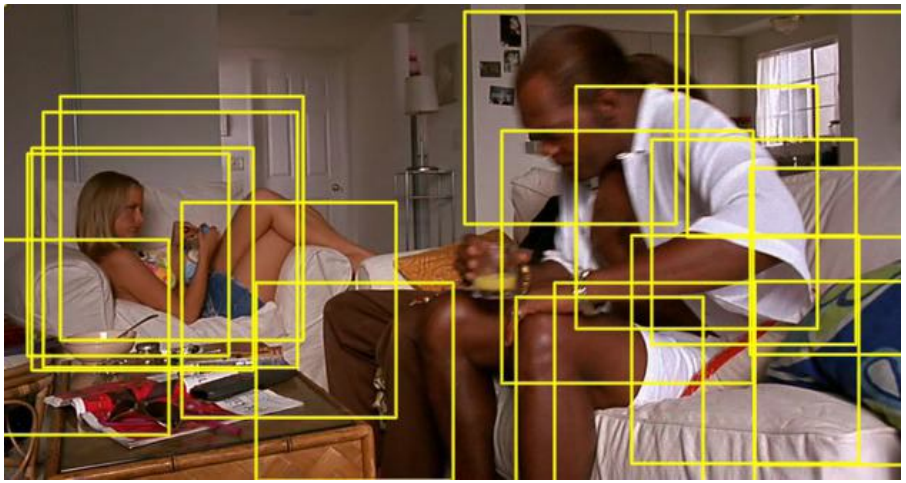




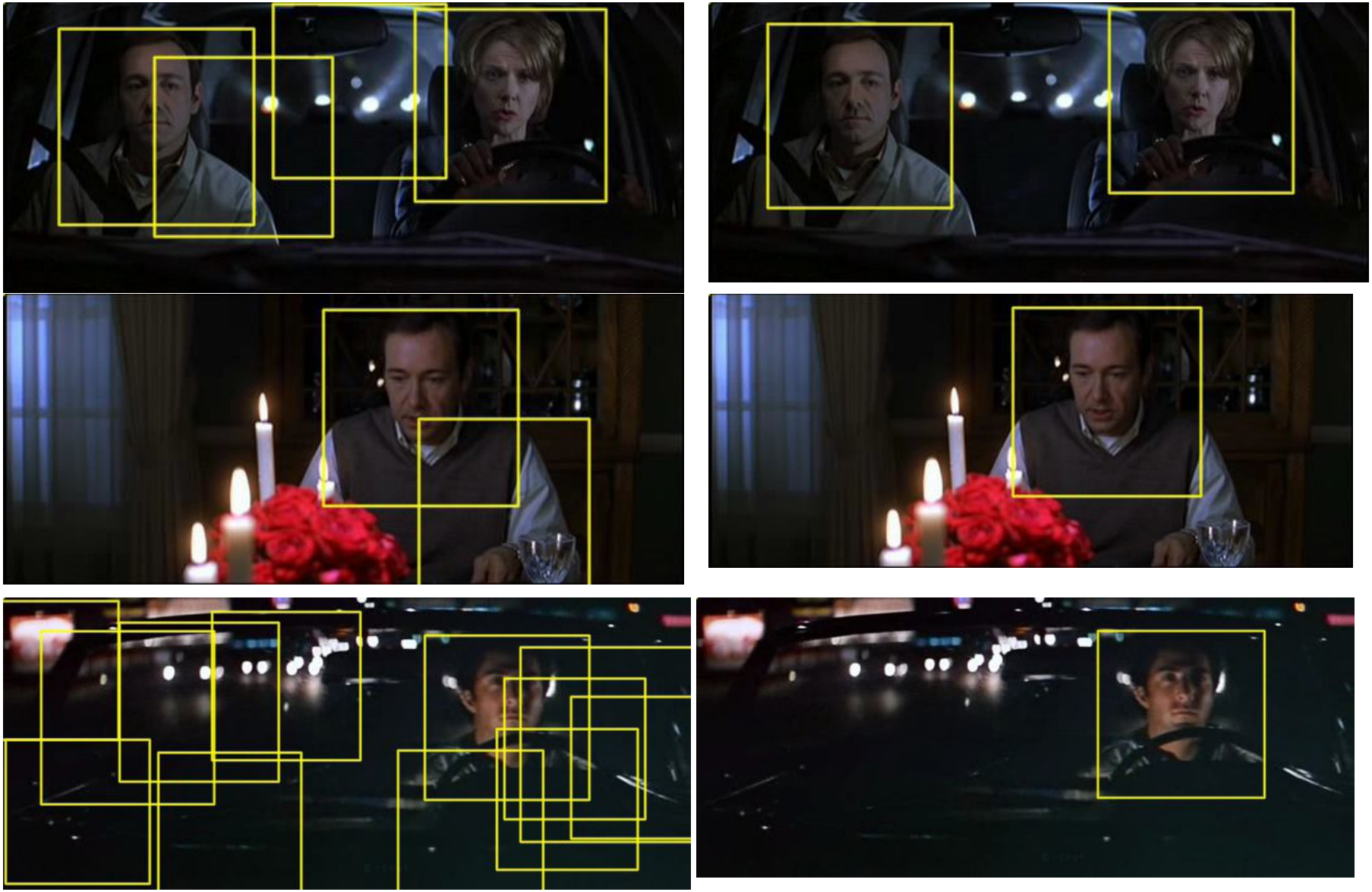
Сравнение

Яндекс





Сравнение



Резюме алгоритма

Яндекс

- Используем скользящее окно
- Вычисляем вектор-признак на основе HOG
  - Разбиваем окно на ячейки
  - В каждой ячейке считаем гистограмму ориентации градиентов
- Обучаемый линейный SVM
- Для обучения:
  - Размножаем (шевелим) эталонные примеры объектов
  - Используем схему bootstrapping для выбора примеров фона
    - На первой стадии берём случайные окна для фона
    - На следующих стадиях выбираем ложные срабатывания детектора как «трудные» примеры

Требования к детектору

- Для изображения в 1МП нужно просмотреть порядка 1М окон (например, для случая лиц)
- На одном изображении обычно 0-10 лиц

- Чтобы избежать ложных обнаружений (false positives) ошибка 2го рода должна быть ниже 10^{-6}
- Нужно быстро отбрасывать ложные окна

Детектор Violo-Jones

Яндекс

- Основополагающий метод для поиска объектов на изображении в реальном времени
- Обучение очень медленное, но поиск очень быстр
- Основные идеи:
  - *Признаки Хоара в качестве слабых классификаторов*
  - *Интегральные изображения* для быстрого вычисления признаков
  - *Бустинг* для выбора признаков
  - *Каскад (Attentional cascade)* для быстрой отбраковки окон без лица

P. Viola and M. Jones. *Rapid object detection using a boosted cascade of simple features.* CVPR 2001.

P. Viola and M. Jones. *Robust real-time face detection.* IJCV 57(2), 2004.

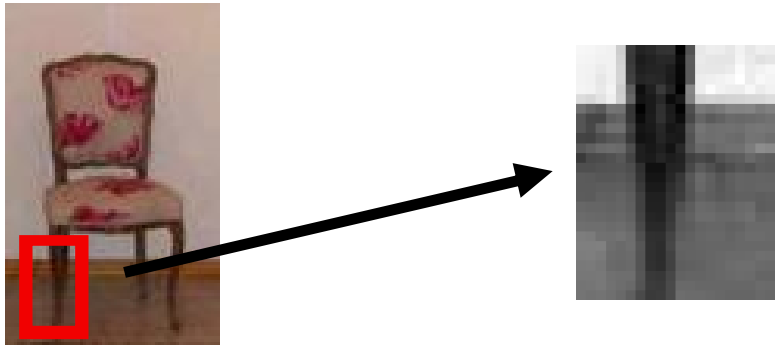
Вспомним сопоставление шаблонов

Найти стул в изображении

Что если воспользоваться маленькими фрагментами?
Искать не стул, а часть стула?

Части объектов

Что если воспользоваться маленькими фрагментами? Искать не стул, а часть стула?

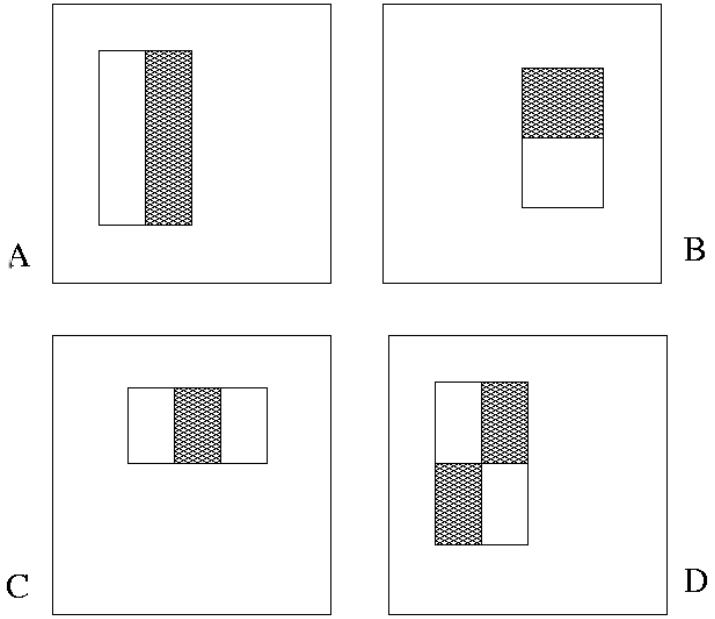


Не так плохо, срабатывает на ножках

Признаки Хоара

Яндекс

“Прямоугольные фильтры”



Value =

∑ *(pixels in white area)* –
∑ *(pixels in black area)*

Слабые детекторы/классификаторы

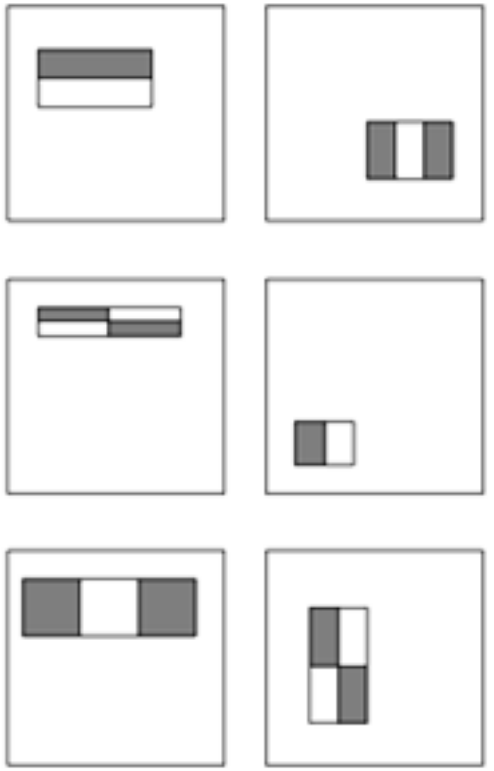
- Определяем слабые классификаторы на основе прямоугольных признаков

Значение признака

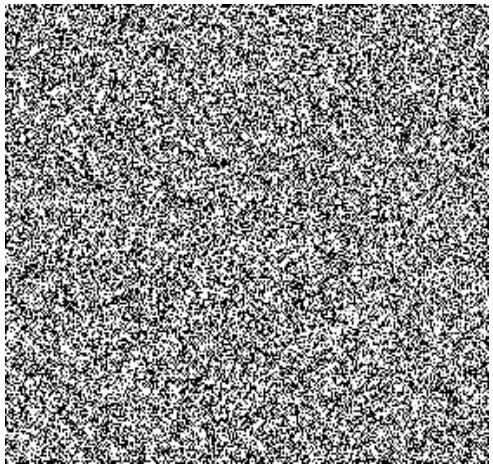
$$h_t(x) = \begin{cases} 1 & \text{if } f_t(x) > \theta_t \\ 0 & \end{cases}$$

окно

порог

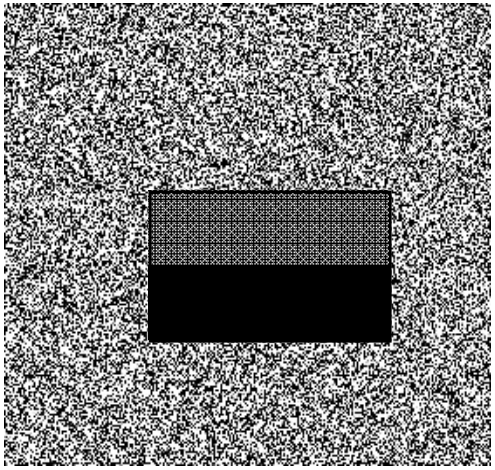


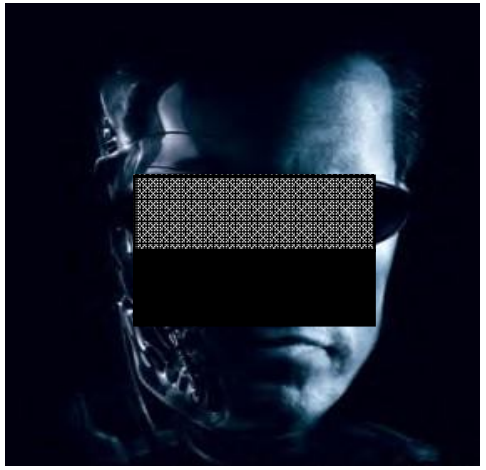
Пример



Source

Result

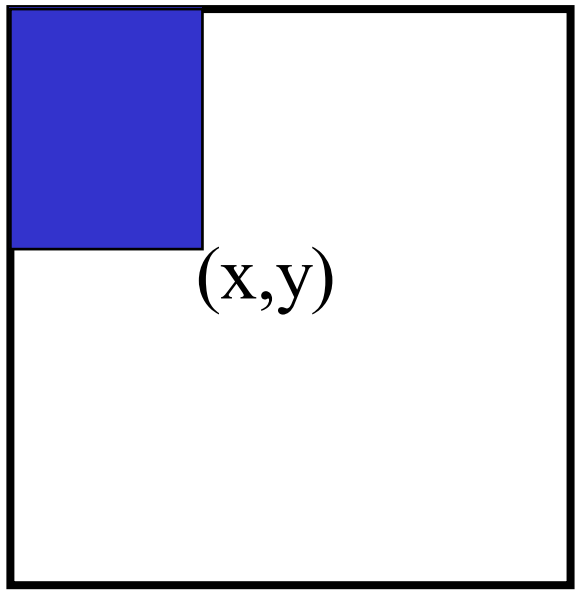




Интегральные изображения

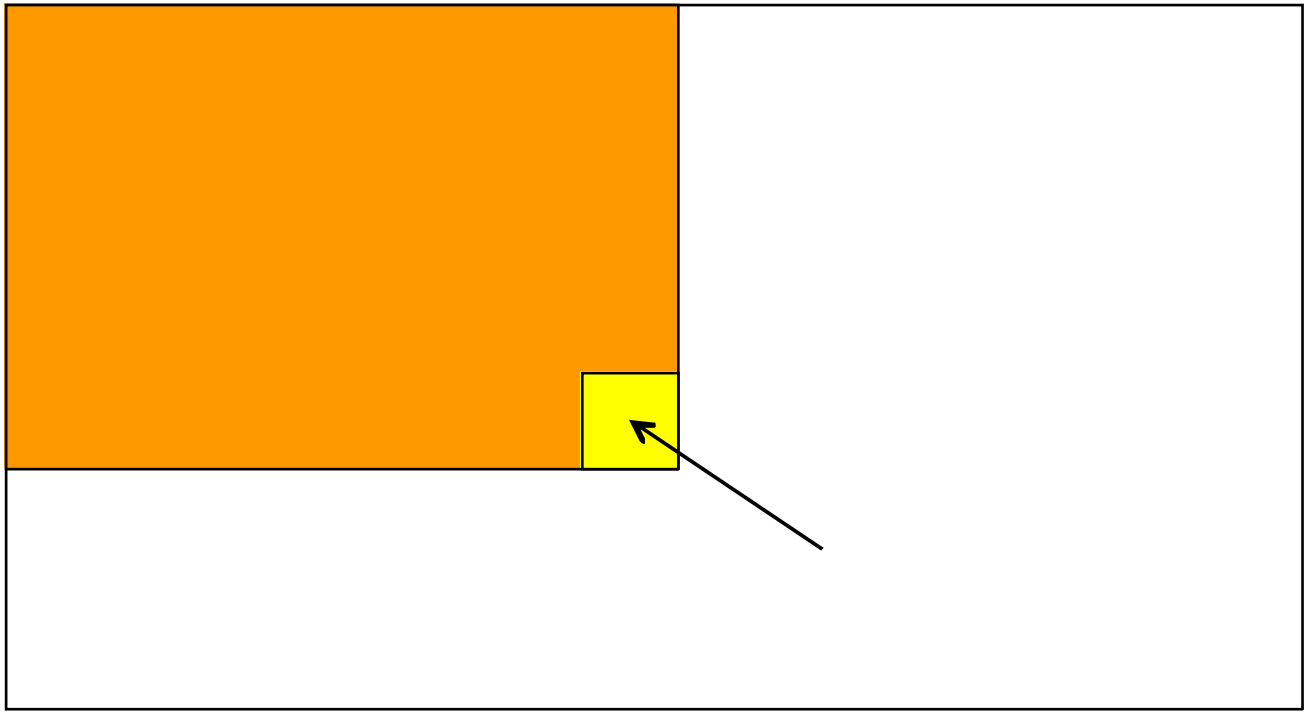
Яндекс

- Значение каждого пиксела (x,y) равно сумме значений всех пикселов левее и выше пикселя (x,y) включительно
- Интегральное изображение рассчитывается за один проход

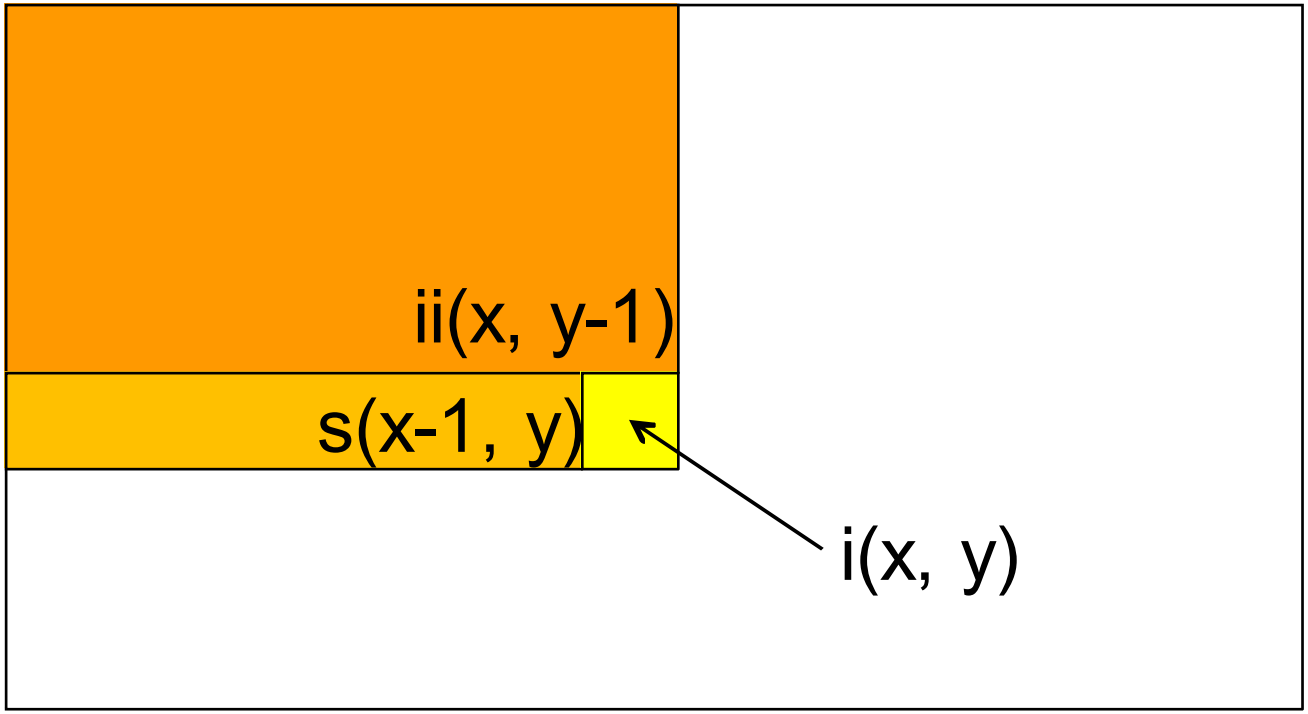


Вычисление интегрального изображения

Яндекс



Вычисление интегрального изображения



- Сумма по строке: s(x, y) = s(x–1, y) + i(x, y)
- Интегральное изображение: ii(x, y) = ii(x, y−1) + s(x, y)

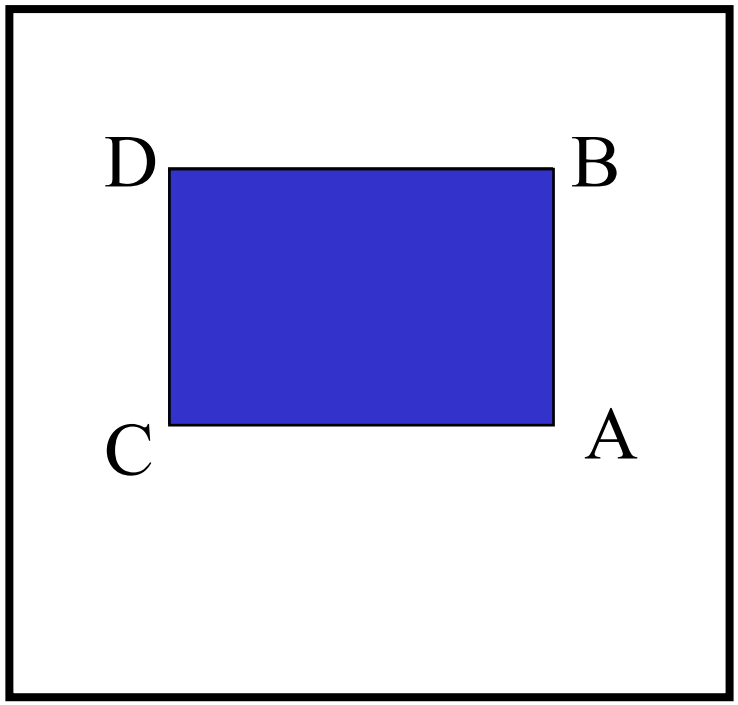
MATLAB: ii = cumsum(cumsum(double(i)), 2);

Вычисление суммы в прямоугольнике

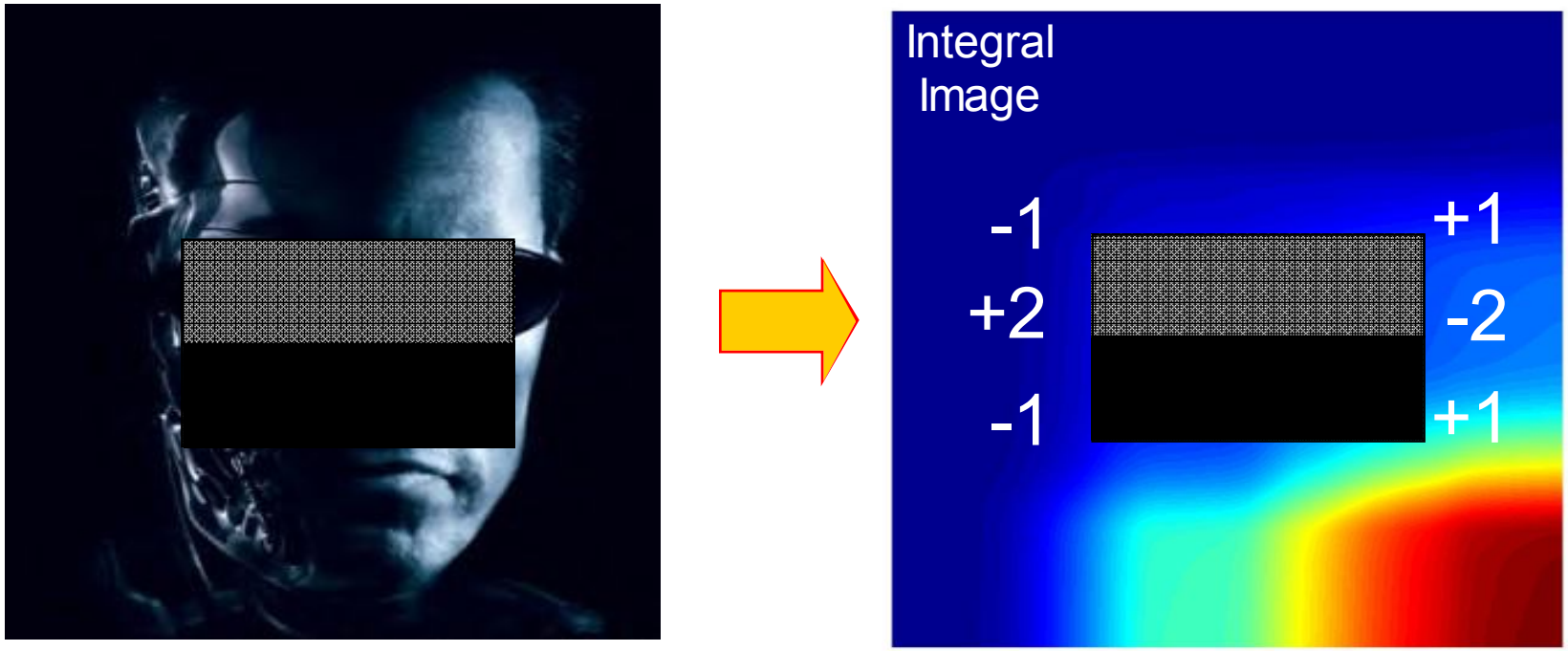
Яндекс

- Пусть A,B,C,D – значения интегрального изображения в углах прямоугольника
- Тогда сумма значений пикселов в исходном изображении вычисляется по формуле:

  sum = A – B – C + D
- 3 операции сложения для любого прямоугольника



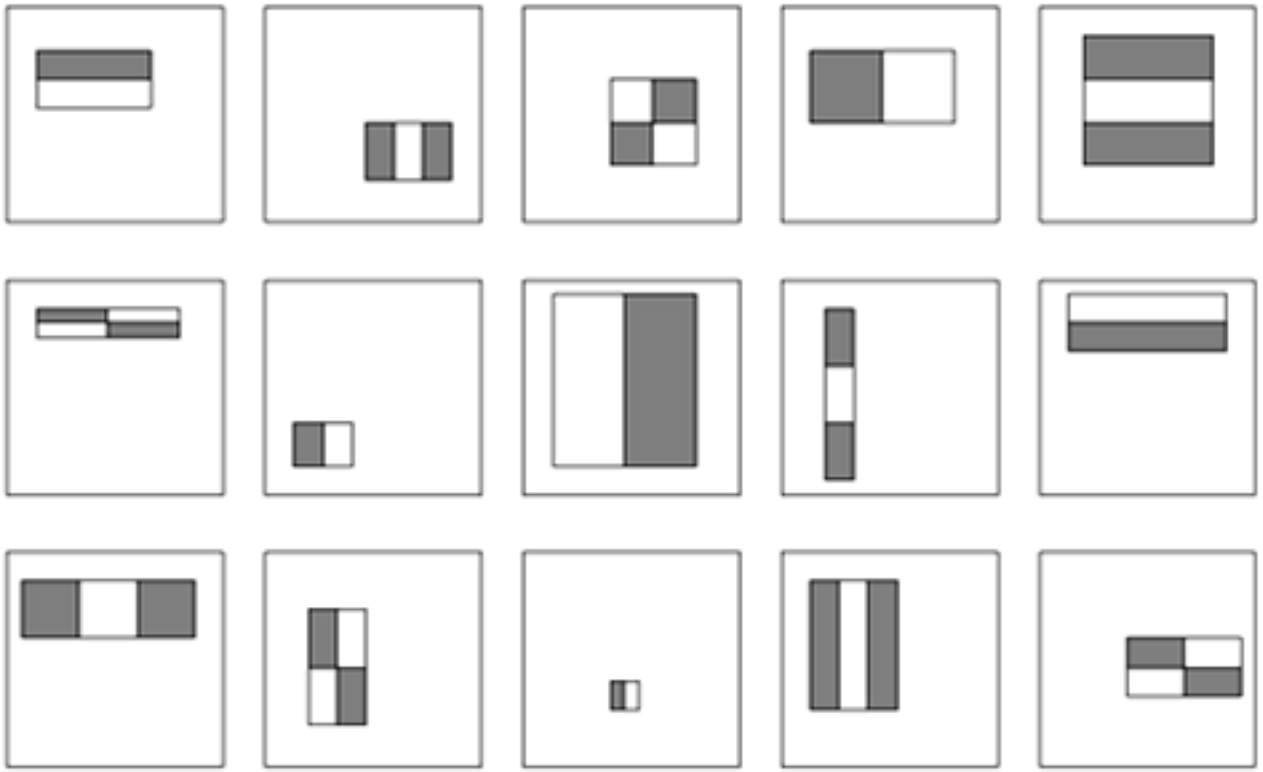
Пример



Количество признаков

Яндекс

Для окна поиска 24x24 пиксела, число возможных прямоугольных признаков достигает ~160,000!



Выбор признаков

Яндекс

- Для окна поиска 24x24 пиксела, число возможных прямоугольных признаков достигает ~160,000!
- В процессе поиска вычислять все признаки нереально
- Хороший классификатор должен использовать лишь маленькое подмножество всевозможных признаков
- Будем выбирать нужные признаки и строить из них линейные комбинации с помощью бустинга

Y. Freund and R. Schapire, A short introduction to boosting, *Journal of Japanese Society for Artificial Intelligence*, 14(5):771-780, September, 1999.

Бустинг

Яндекс

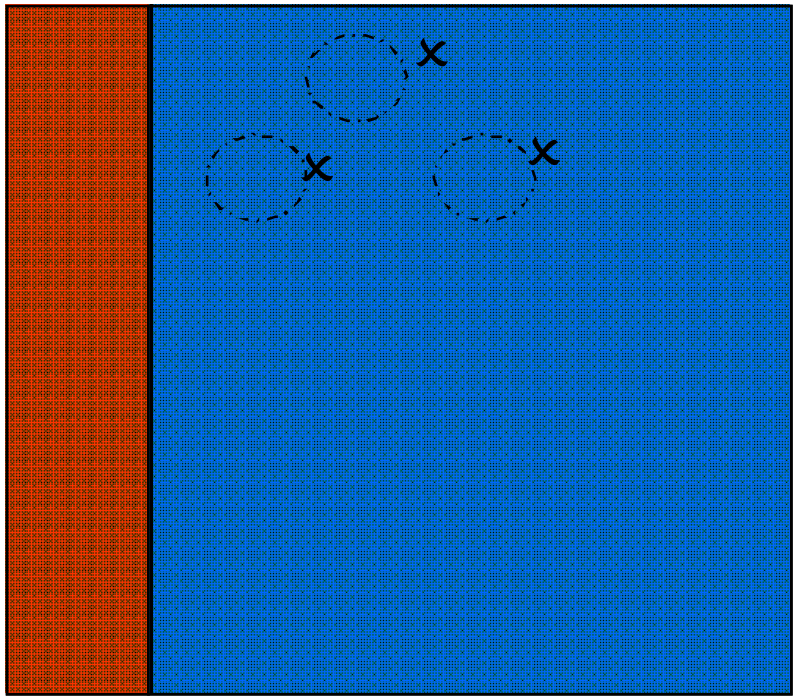
- Бустинг – схема классификации, основанная на комбинировании слабых классификаторов в более точный комитетный
  - Слабый классификатор должен быть лучше монетки
- Обучение состоит из нескольких этапов усиления (*boosting rounds)*
  - На каждом этапе выбираем слабый классификатор, который лучше всех сработал на примерах, оказавшихся трудными для предыдущих классификаторов
  - «Трудность» записывается с помощью весов, приписанных примерам из обучающей выборки
  - Составляем общий классификатор как линейную комбинацию слабых классификаторов

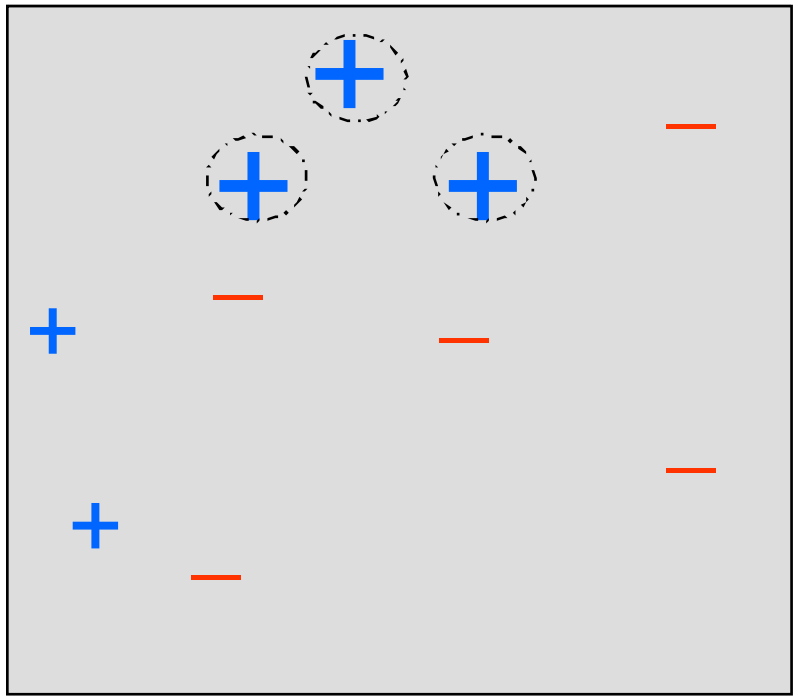
Y. Freund and R. Schapire, A short introduction to boosting, *Journal of Japanese Society for Artificial Intelligence*, 14(5):771-780, September, 1999.

Пример хорошего классификатора

Яндекс

Итерация 1 из 3



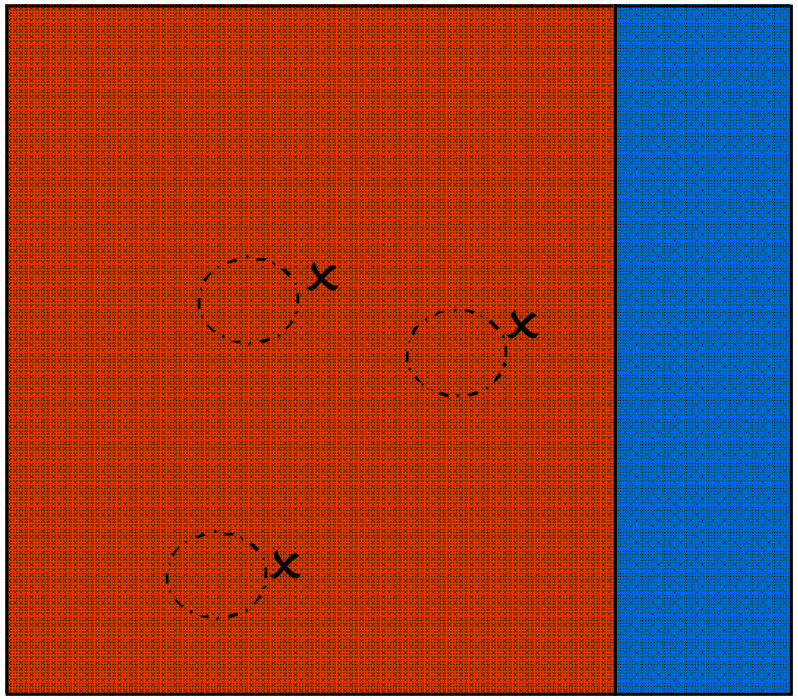


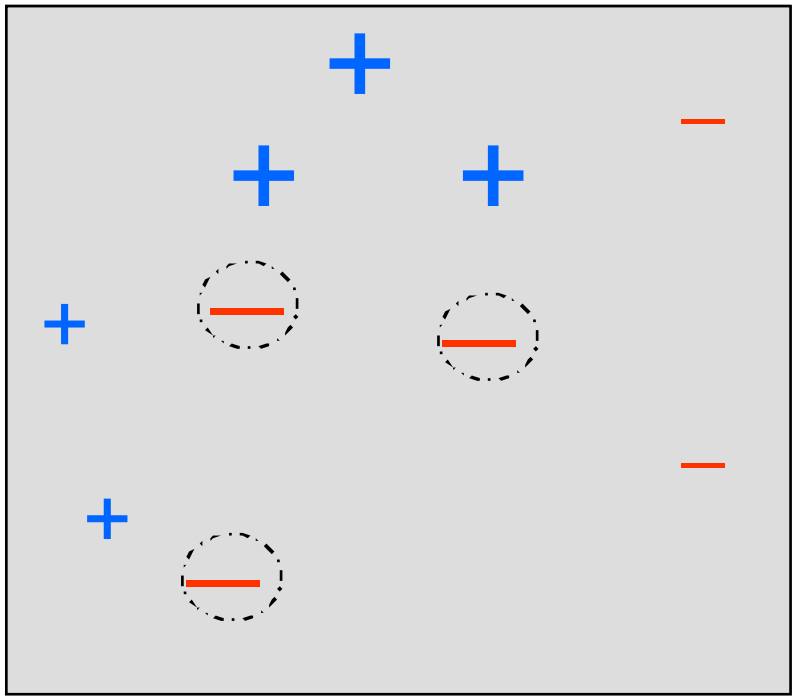
h_1 $\varepsilon_1 = 0.300$

$\alpha_1 = 0.424$

D_2

Итерация 2 из 3



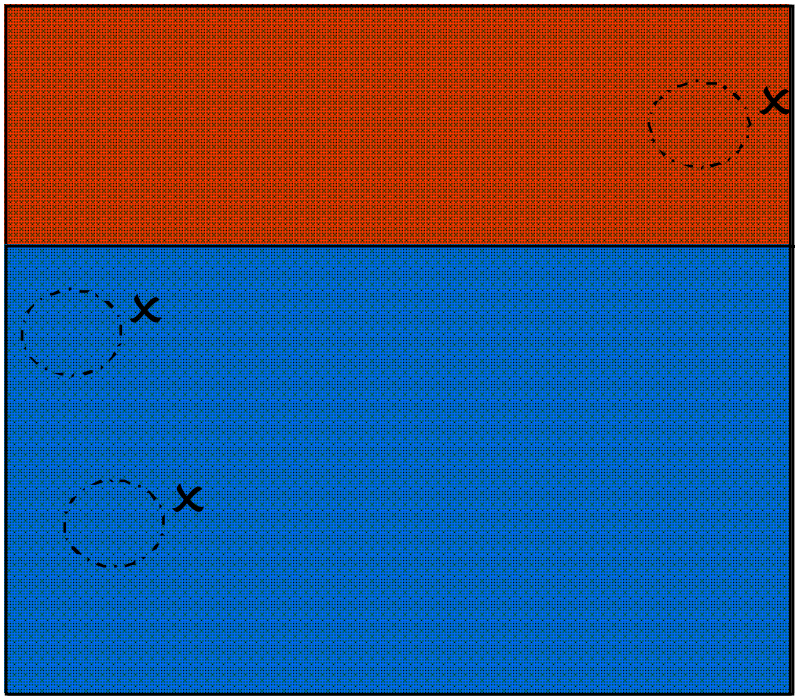


$\varepsilon_2 = 0.196$ h_2

$\alpha_2=0.704$

D_2

Итерация 3 из 3



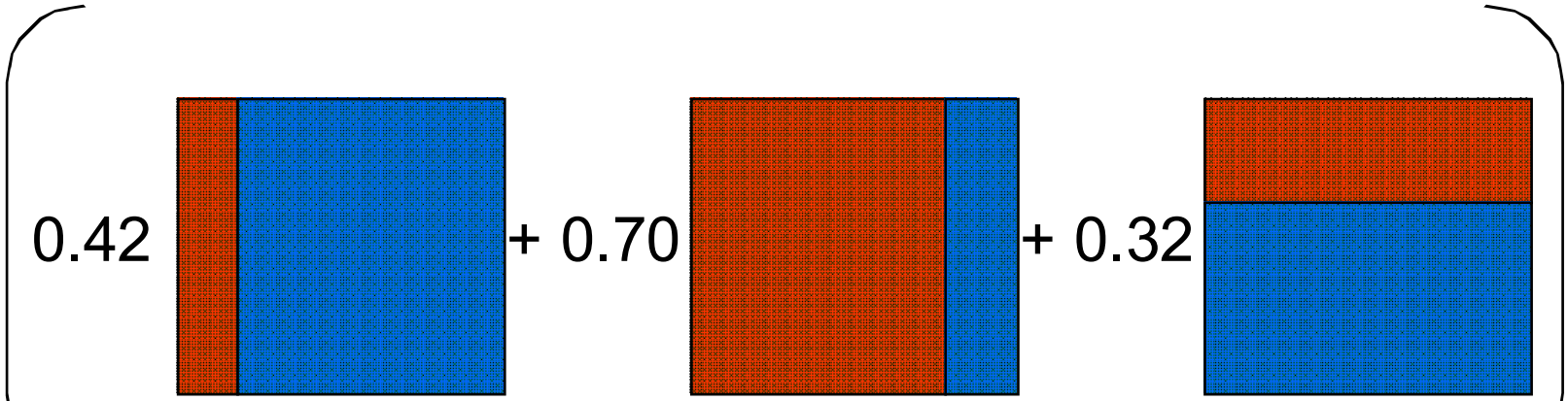
h_3

СТОП

$\varepsilon_3 = 0.344$

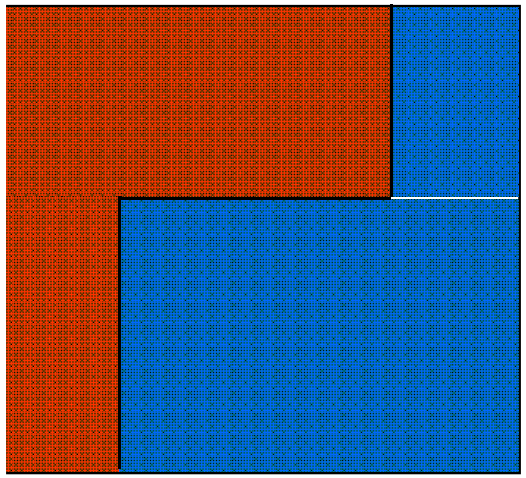
$\alpha_2 = 0.323$

Конечная гипотеза



$$H_{final} = sign[0.42 \cdot h_1(x) + 0.70 \cdot h_2(x) + 0.72 \cdot h_3(x)]$$

$$h_i(x) = \begin{cases} -1, x \to \Theta_0 \\ +1, x \to \Theta_1 \end{cases}$$



AdaBoost (Discreet)

Яндекс

Пусть есть набор *{h}* – слабых классификаторов

Пусть *T*: $(x_1, y_1), \ldots, (x_m, y_m)$ где $x_i \in X, y_i \in \Theta = \{-1, +1\}$

Инициализируем $D_1(i) = 1/m$

Для $k = \overline{1, K}$

1. Обучим h_k с минимальной ошибкой

$$\varepsilon_k = \Pr_{i \sim D_k}[h_k(x_i) \neq y_i]$$

Ошибка h_t рассчитывается с учётом D_t

2. Рассчитаем вес гипотезы

$$\alpha_t = \frac{1}{2} \ln\left(\frac{1-\varepsilon_k}{\varepsilon_k}\right)$$

Вес Адаптируется. Чем больше ε_k тем меньше α_k

3. Для всех i = 1 to m

$$D_{k+1}(i) = \frac{D_k(i)}{Z_k} \times \begin{cases} e^{-\alpha_k} & \text{if } h_k(x_i) = y_i \\ e^{\alpha_k} & \text{if } h_k(x_i) \neq y_i \end{cases}$$

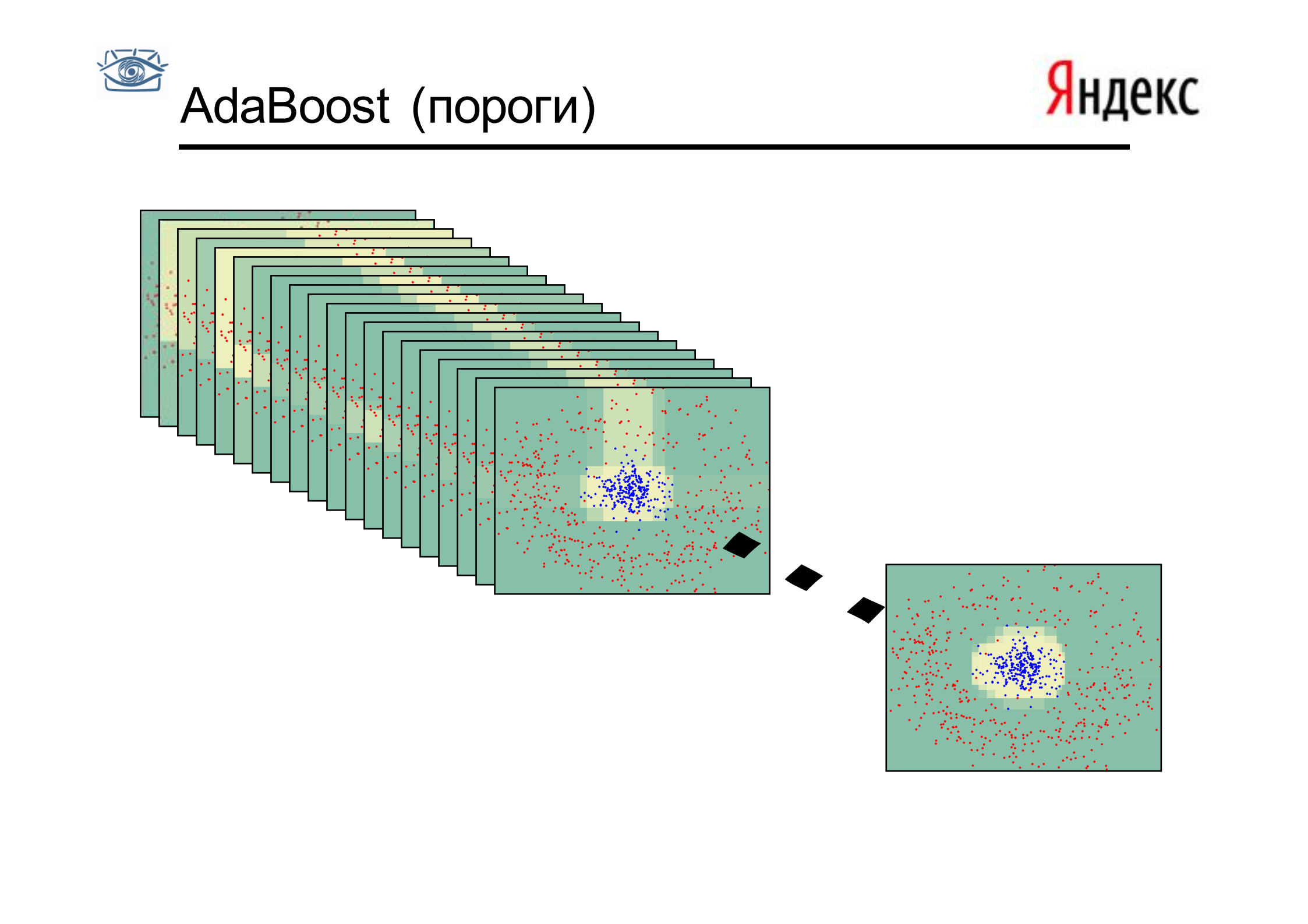
Увеличиваем вес примера, если на нём алгоритм ошибается

Z_t нормализующий коэф.

Результат:

$$H(x) = \operatorname{sign}\left(\sum_{k=1}^{K} \alpha_k h_k(x)\right)$$

Линейная комбинация


AdaBoost (пороги)
Яндекс

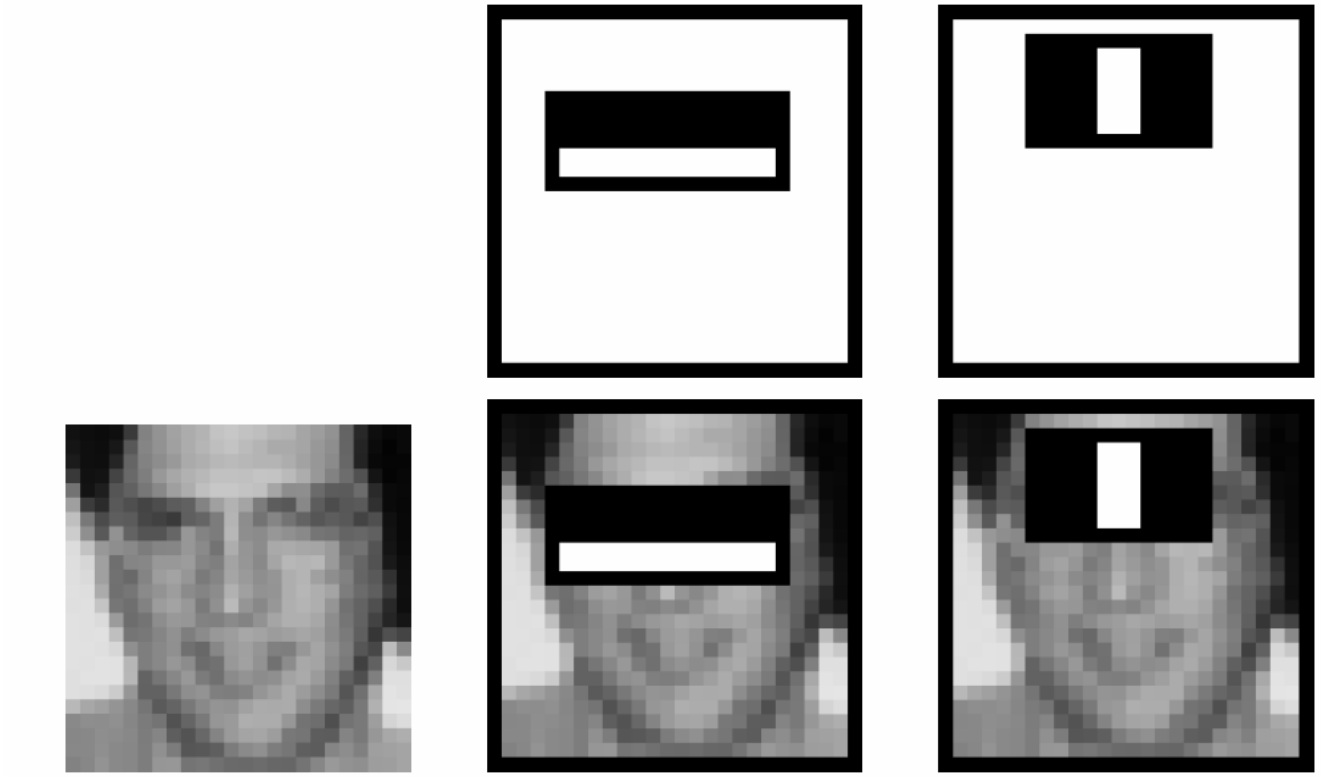
Бустинг для поиска лиц

- Определяем слабые классификаторы на основе прямоугольных признаков
- Для каждого этапа бустинга:
  - Вычисляем каждый прямоугольный признак на каждом примере
  - Выбираем наилучший порог для каждого признака
  - Выбираем наилучший признак / порог
  - Перевзвешиваем выборку
- Вычислительная сложность обучения: $O(MNK)$
  - M этапов, N примеров, K признаков

Бустинг для поиска лиц

Яндекс

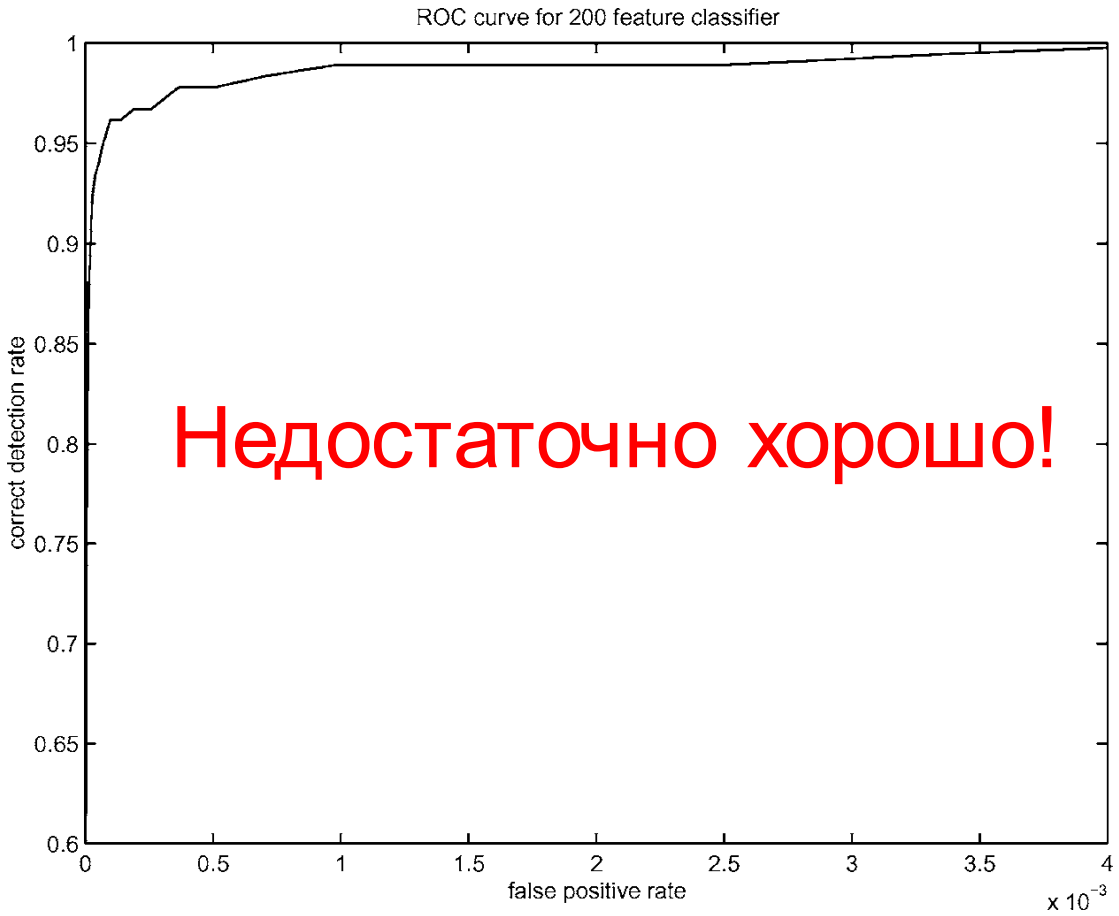
- Первые два признака, выбранные бустингом:



Эта комбинация признаков дает 100% detection rate и 50% false positive rate

Бустинг для поиска лиц

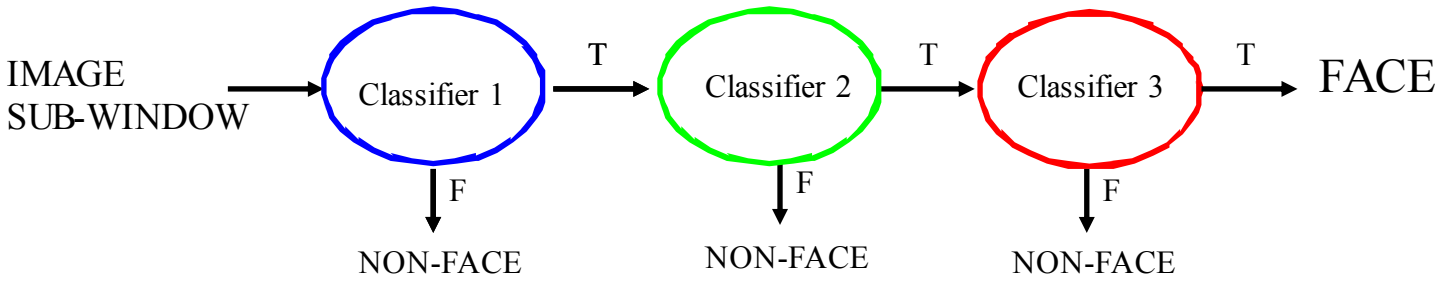
- Классификатор из 200 признаков дает 95% точность и долю ложноположительных срабатываний 1 из 14084



ROC-кривая

Каскад (Attentional cascade)

- Начинаем с простых классификаторов, которые отбрасывают часть отрицательных окон, при этом принимая почти все положительные окна
- Положительный отклик первого классификатора запускает вычисление второго, более сложного, классификатора, и т.д.
- Отрицательный отклик на любом этапе приводит к немедленной отбраковке окна



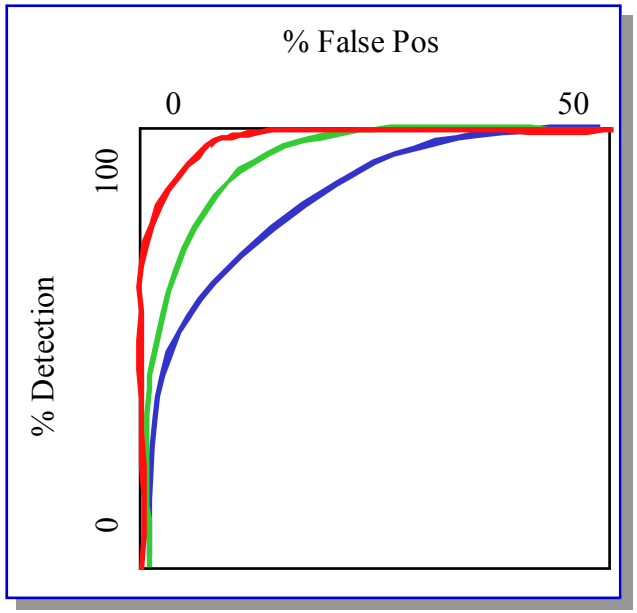
Каскад

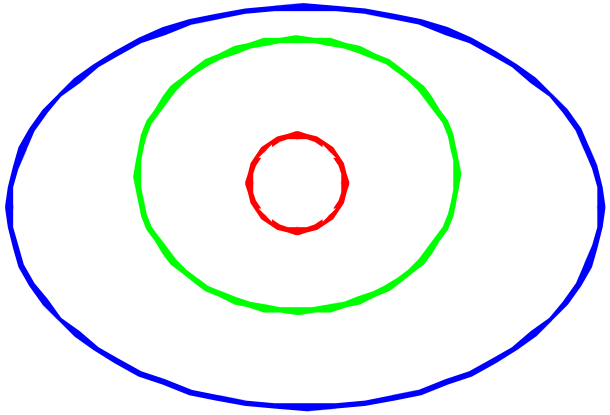
- Медленные классификаторы применяются только к некоторым окнам ⇒ существенное ускорение
- Управляем сложностью/скоростью классификатора:
  - Количество опорных векторов [Romdhani et al, 2001]
  - Количество признаков [Viola & Jones, 2001]
  - Видом ядра SVM [Vedaldi et al, 2009]

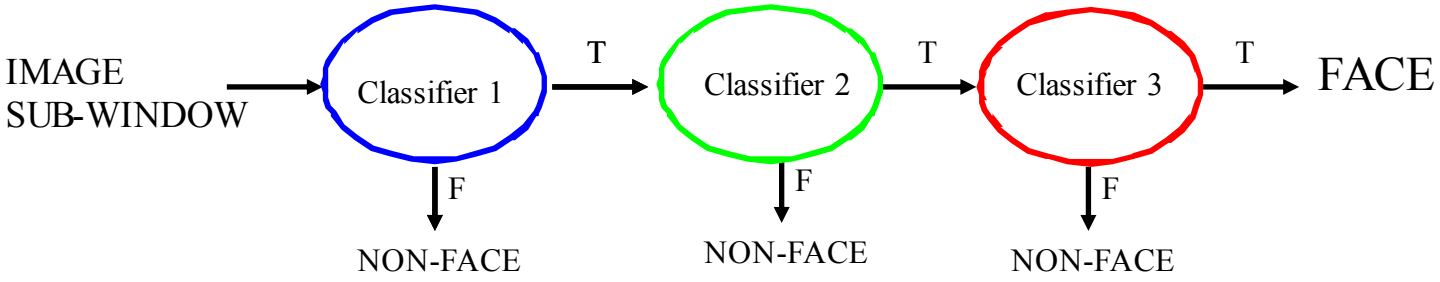
Каскад

Цепочка классификаторов с каждым уровнем становится более сложной, ошибка второго рода постоянно снижается

ROC-кривая

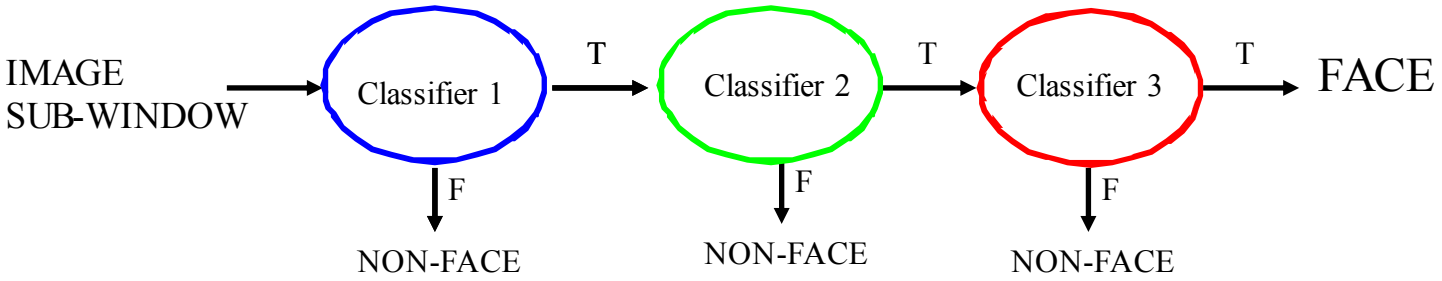






Каскад

- Уровень обнаружения и доля ложноположительных срабатываний каскада можно оценить как произведение соответствующих уровней ошибок каждого этапа
- Уровень обнаружения 0.9 и доля ложноположительных срабатываний порядка 10^{-6} достигается с помощью каскада из 10 этапов, если на каждом этапе уровень обнаружения примерно равен 0.99 ($0.99^{10} \approx 0.9$) и доля ложноположительных примерно 0.30 ($0.3^{10} \approx 6 \times 10^{-6}$)



Обучение каскада

Яндекс

- Задаем требуемые значения уровня обнаружения и доли ложноположительных срабатываний для каждого этапа
- Добавляем признаки до тех пор, пока параметры текущего этапа не достигнут заданного уровня
  - Приходится понижать порог AdaBoost для максимизации обнаружения (в противоположенность минимизации общей ошибки классификации)
  - Тестирование на отдельном наборе (*validation set)*
- Если общий уровень ложноположительных срабатываний недостаточно низок, добавляем очередной этап
- Ложные обнаружения на текущем этапе используются как отрицательные примеры на следующем этапе

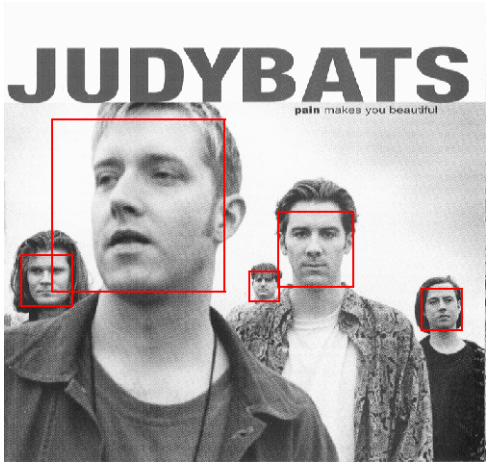
Тренировочная выборка

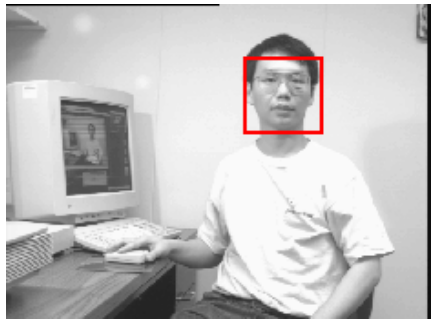
- 5000 лиц
  - Все фронтальные, уменьшенные до 24x24 пикселов
  - Все нормированы
- 300M отрицательных примеров
  - 9500 изображений без лиц
- Большая изменчивость
  - Разные люди
  - Освещение
  - Поза лица

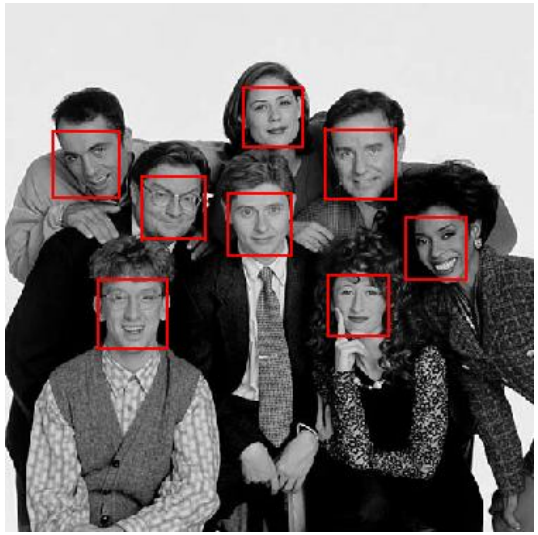
Производительность системы

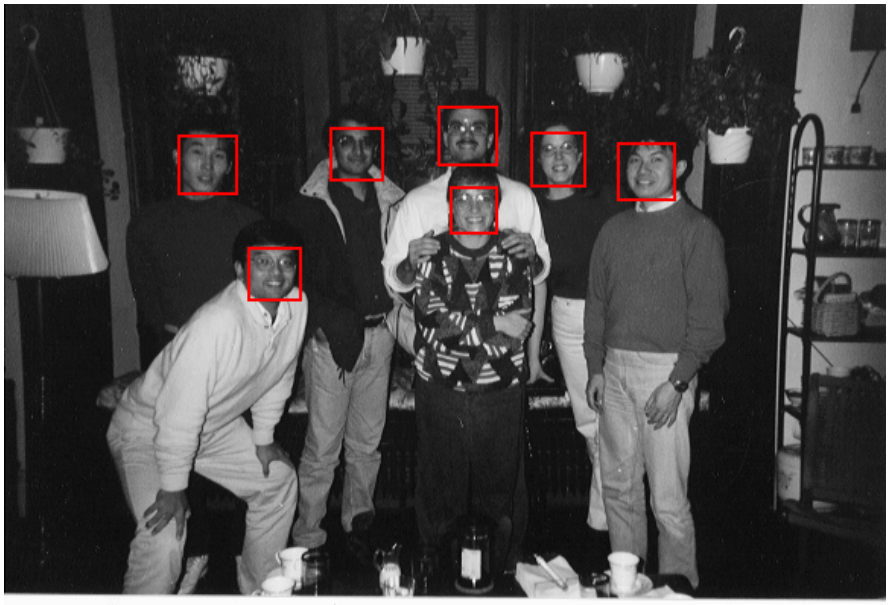
- Обучение: “недели” на 466 MHz Sun рабочей станции
- 38 этапов, всего 6061 признаков
- В среднем 10 признаков оцениваются для каждого окна на тестовой выборке
- “На 700 Mhz Pentium III, детектор лиц обрабатывает одно изображение 384x288 пикселов за 0.067 секунды”
  - 15 Hz
  - В 15 раз быстрее сравнимого по точности предшествующего метода (Rowley et al., 1998)

Пример работы



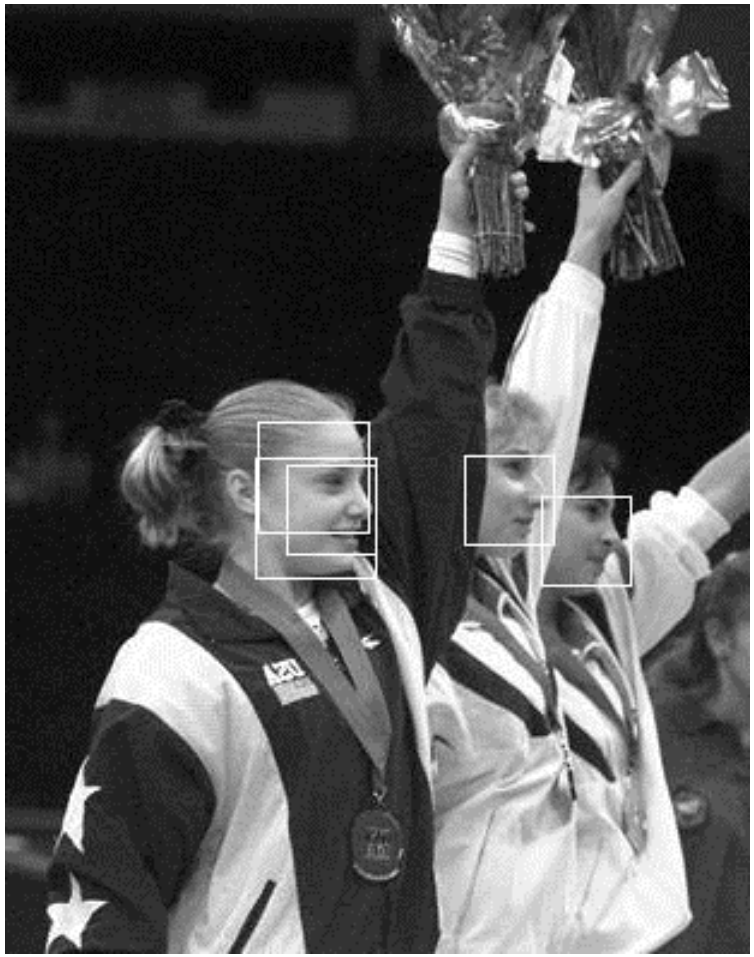






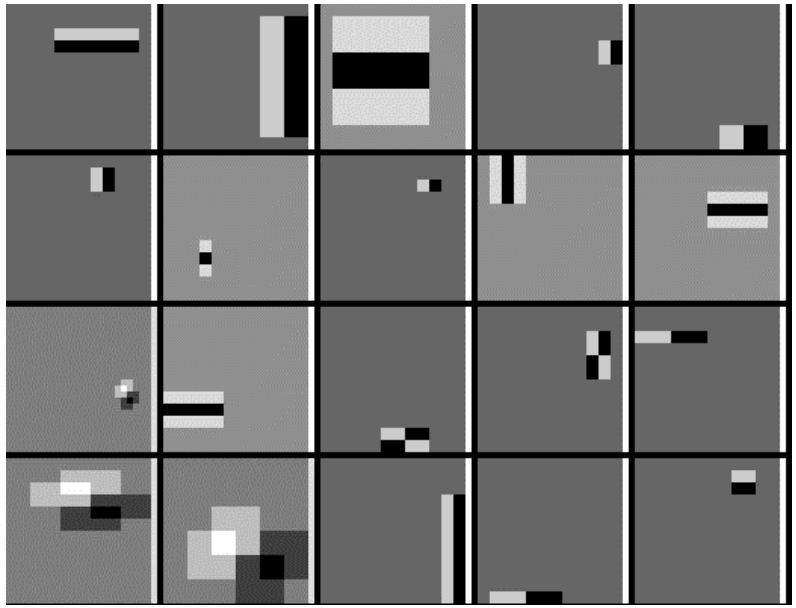
Поиск профилей

Яндекс



Признаки для поиска профилей

Яндекс



Резюме: детектор Viola-Jones

Яндекс

- Прямоугольные признаки как слабые классификаторы
- Интегральные изображения для быстрого вычисления признаков
- Бустинг для выбора признаков
- Каскад классификаторов для быстрого выбраковки отрицательных окон

Недостатки Viola-Jones

Яндекс

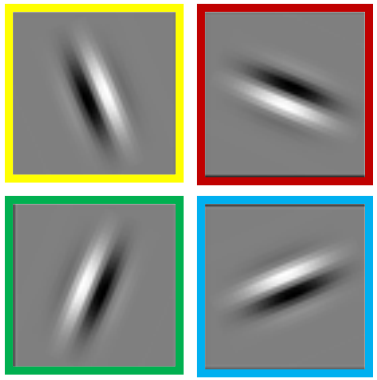
- Малое количество используемых каналов изображения и признаков
  - Серые изображения, фильтры Хоара
- Недостаточная точность
  - 10^{-6} – слишком много ложных срабатываний
- Множественные отклики у правильных обнаружений
- «Дискретность» - возможность завершения только в конце этапа
- Очень долгое время обучения

Набор каналов

Яндекс

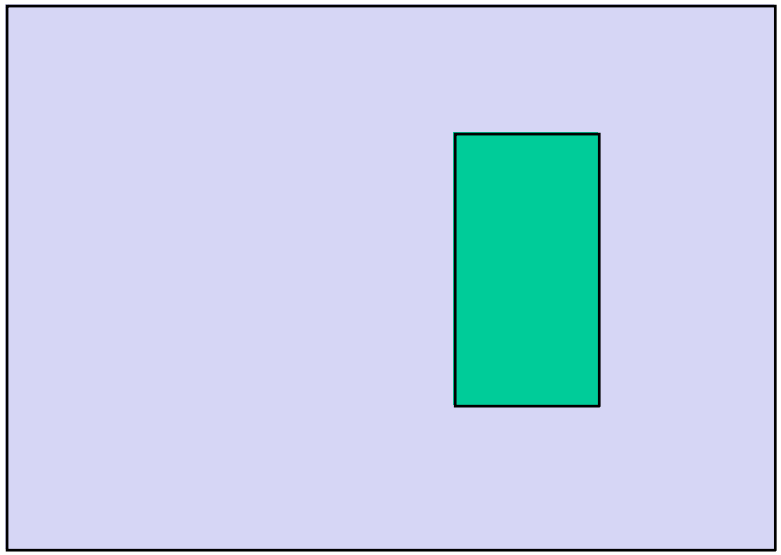
- Цвет и градиенты очень показательны для многих классов
- Нужно строить большее количество каналов
  - LUV
  - Норма градиента + отклики по направлениям

P. Dollár, Z. Tu, P. Perona and S. Belongie Integral Channel Features BMVC 2009

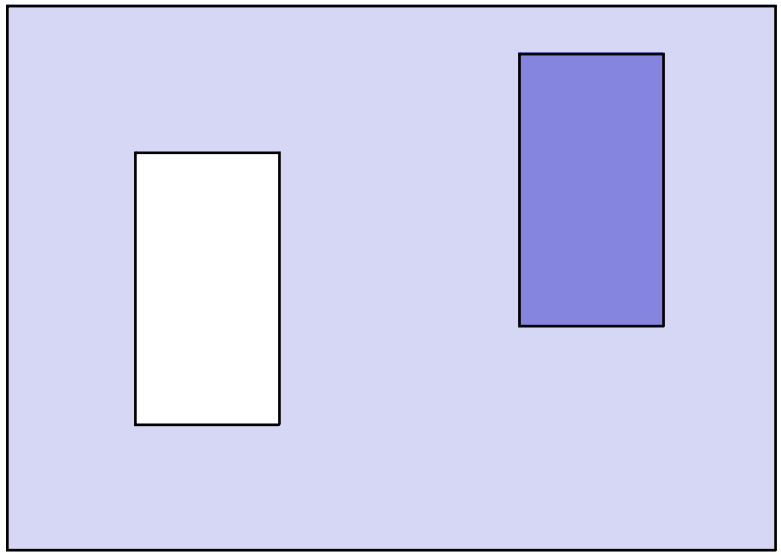


Прямоугольные признаки

Яндекс



Прямоугольная область



Диполи

P. Dollár, Z. Tu, P. Perona and S. Belongie Integral Channel Features BMVC 2009

Учёт масштабов

N models, 1 image scale

(a) Naive approach

1 model, N image scales

(b) Traditional approach

1 model, N/K image scales

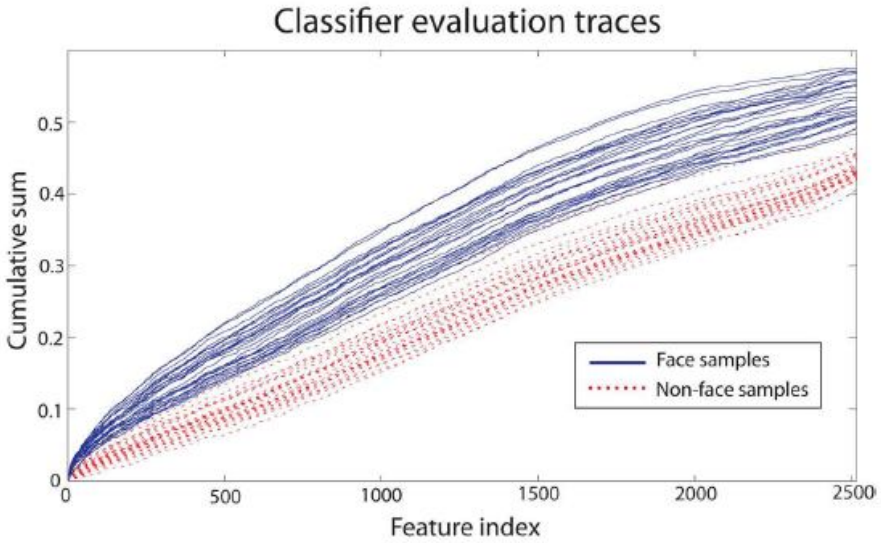
(c) FPDW approach

N/K models, 1 image scale

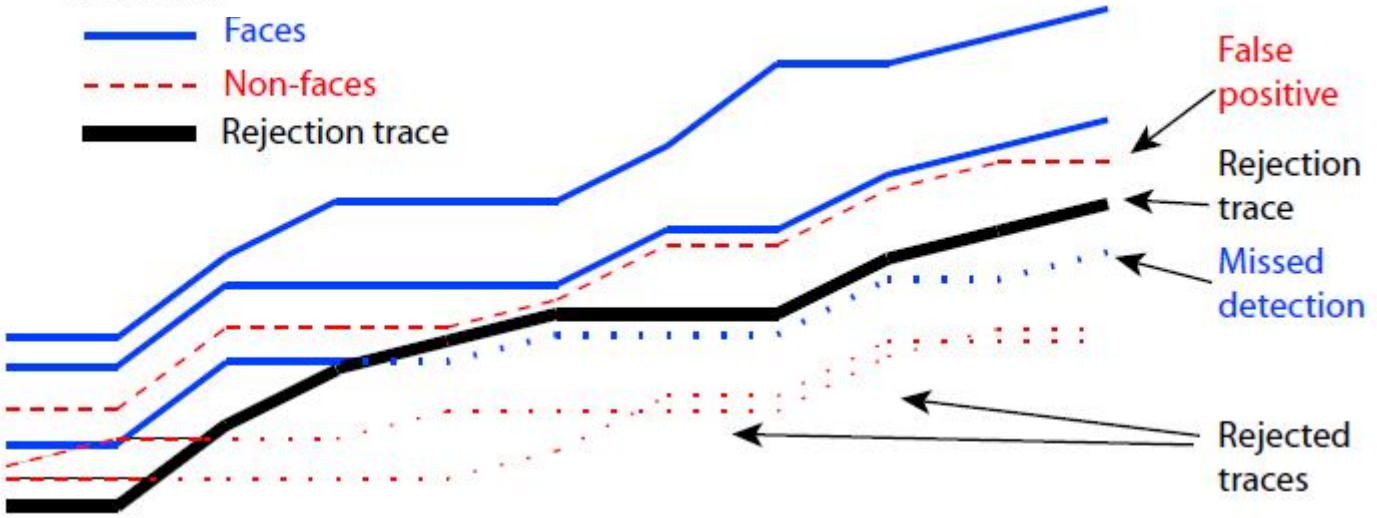
(d) Our approach

P. Dollár, S. Belongie and P. Perona The Fastest Pedestrian Detector in the West BMVC 2010, R. Benenson,M. Mathias, R. Timofte, L. Van Gool Pedestrian detection at 100 frames per second, CVPR 2013

Мягкий каскад



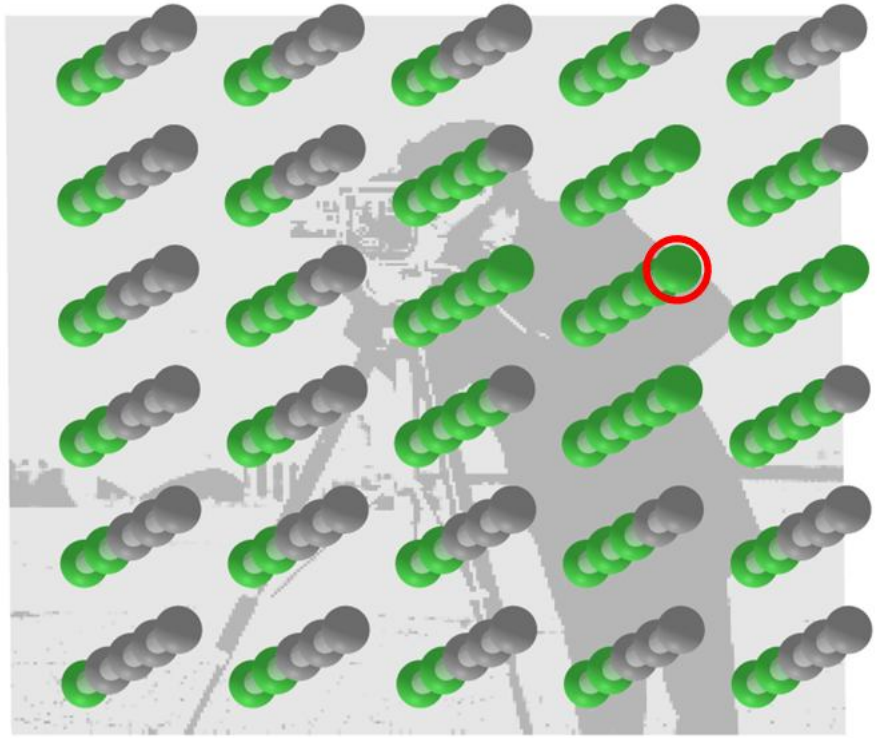
Теперь можем остановиться на любом слабом классификаторв

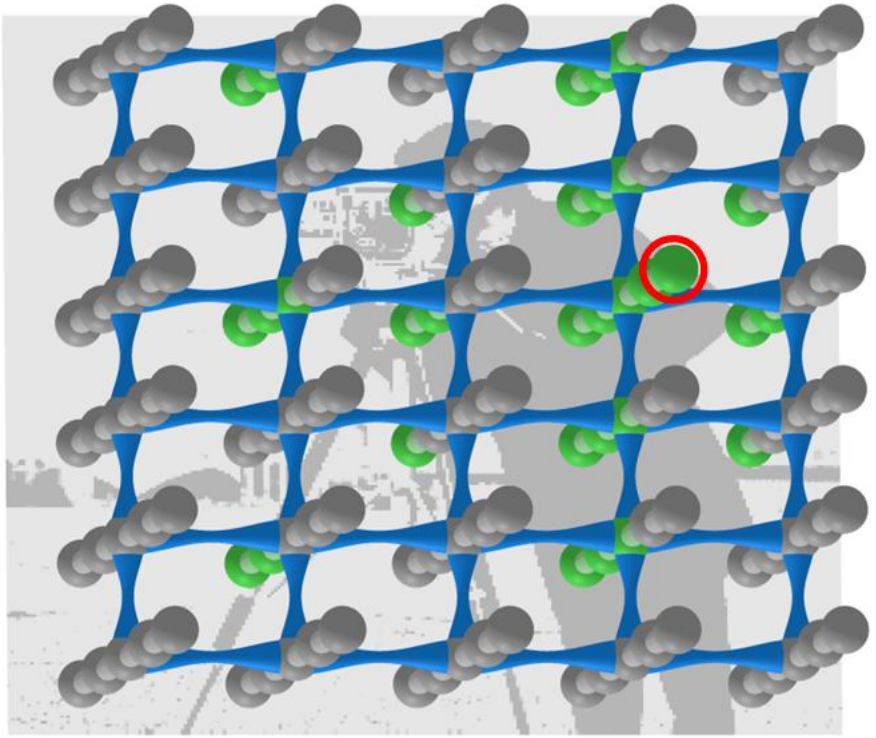


Bourdev, L., Brandt,J. Robust object detection via soft cascade. CVPR 2005

«Crosstalk cascades»

Яндекс





Идея – нужно «подавлять» каскады в тех областях, где скорее всего не будет локального максимума

P. Dollár, R. Appel and W. Kienzle Crosstalk Cascades for Frame-Rate Pedestrian Detection. ECCV 2012,

Резюме

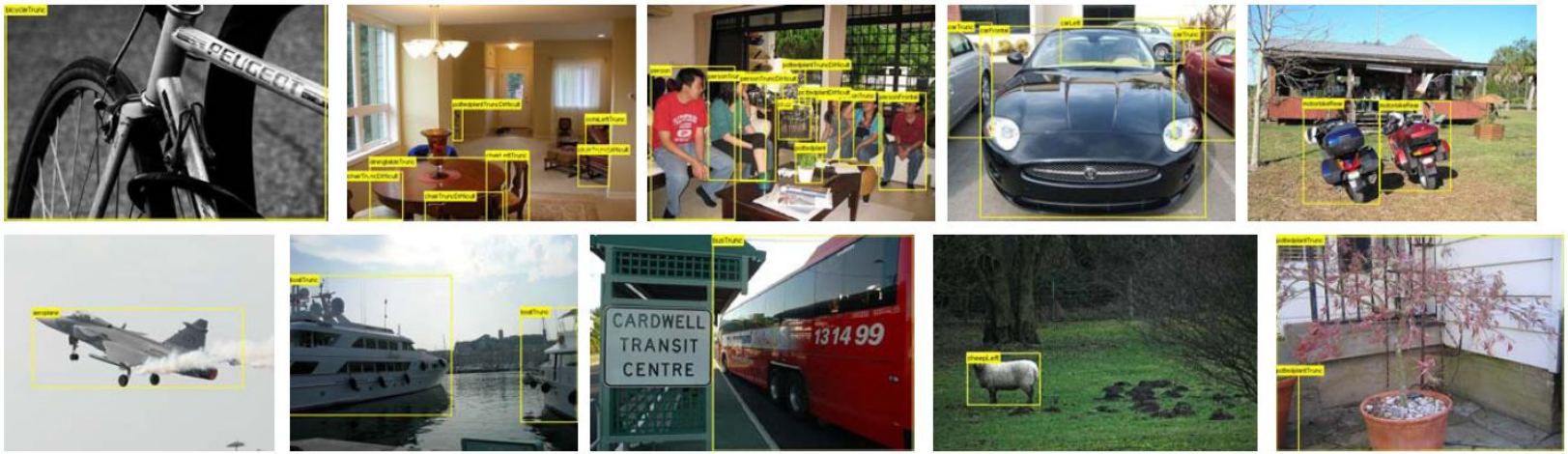
Яндекс

Развитие по всем направлениям!

Pascal VOC (2005-2012)

- PASCAL Visual Object Classes (VOC) Dataset and Challenge
- 20 классов:
  - aeroplane, bicycle, boat, bottle, bus, car, cat, chair, cow, dining table, dog, horse, motorbike, person, potted plant, sheep, train, TV
- Реальные изображения из flickr, не фильтровались по качеству

PASCAL
Pattern Analysis, Statistical Modelling and Computational Learning



Размер выборки

Яндекс

- Same size as VOC2011.

	Training	Testing
Images	11,540	10,994
Objects	27,450	27,078

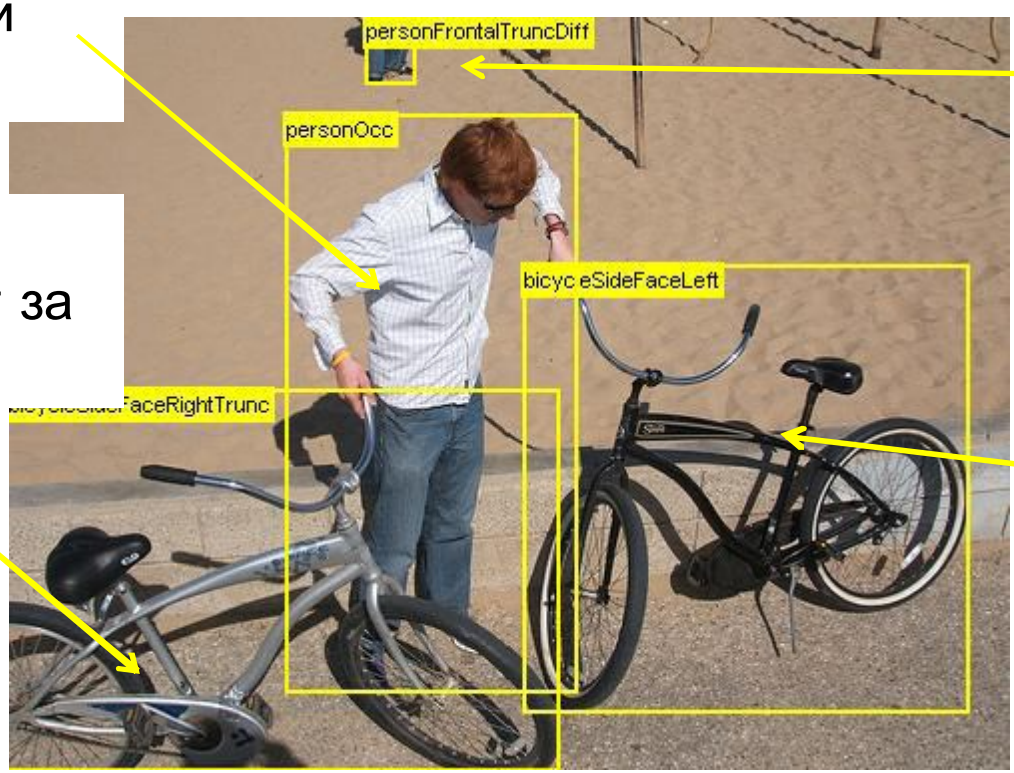
- Minimum ~600 training objects per category
- ~2,000 cars, 1,500 dogs, 8,500 people
- Approximately equal distribution across training and test datasets

Аннотация

- Полная аннотация всех объектов
  - В одну сессию по инструкции

Перекрытый
Объект перекрыт другим внутри рамки

Обрезанный – объект выходит за пределы рамки



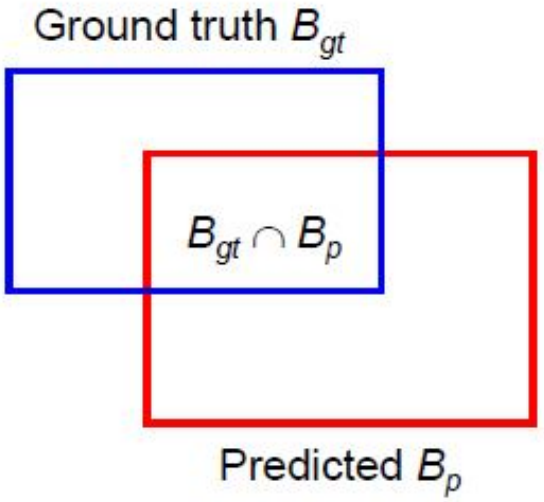
Сложный
Не участвует в оценке

Положение
Смотрит налево

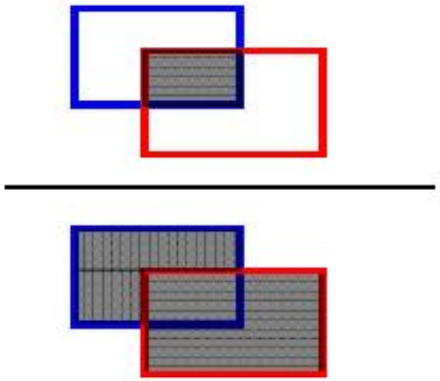
Оценка локализации

Яндекс

- Area of Overlap (AO) Measure

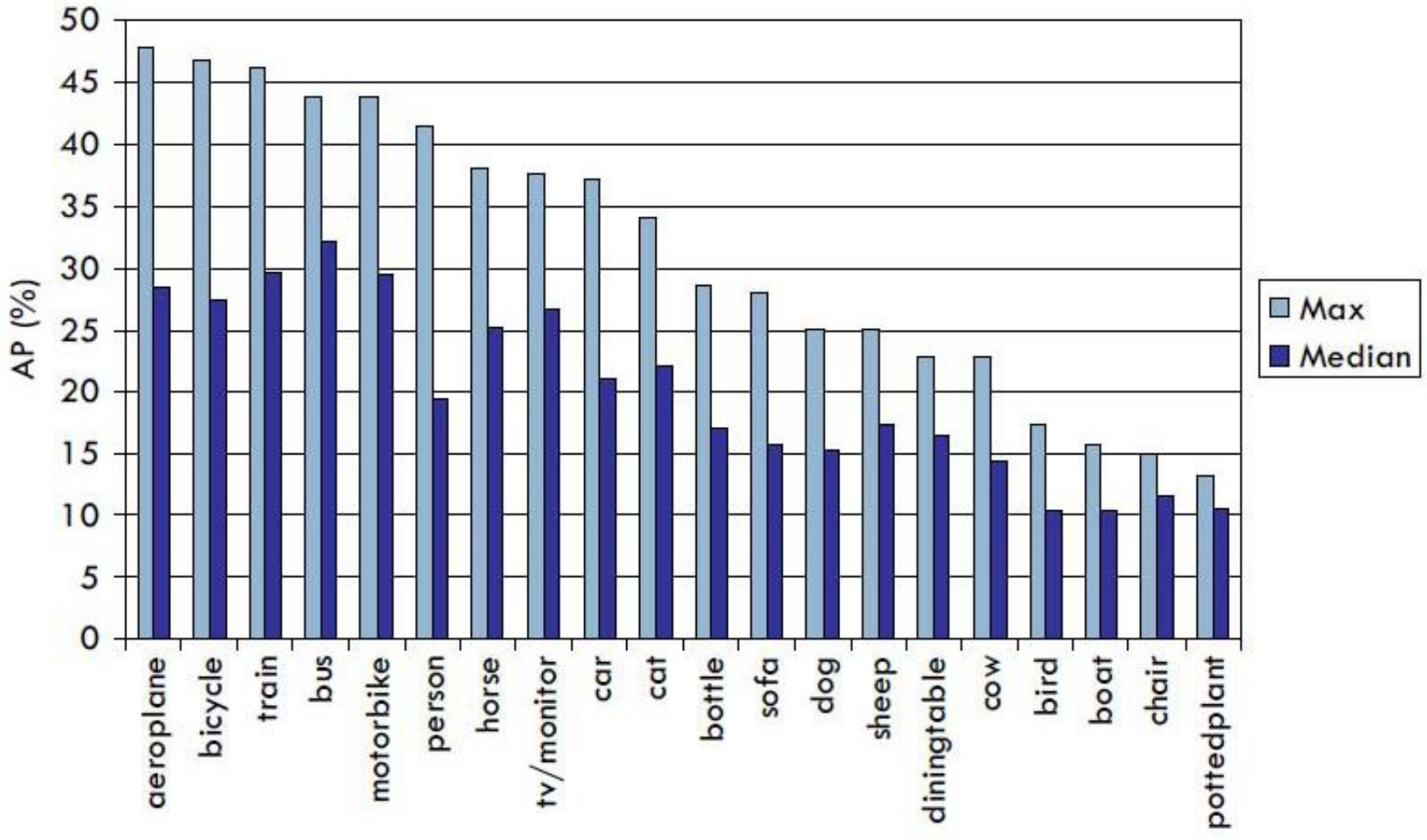


$$AO(B_{gt}, B_p) = \frac{|B_{gt} \bigcap B_p|}{|B_{gt} \bigcup B_p|}$$

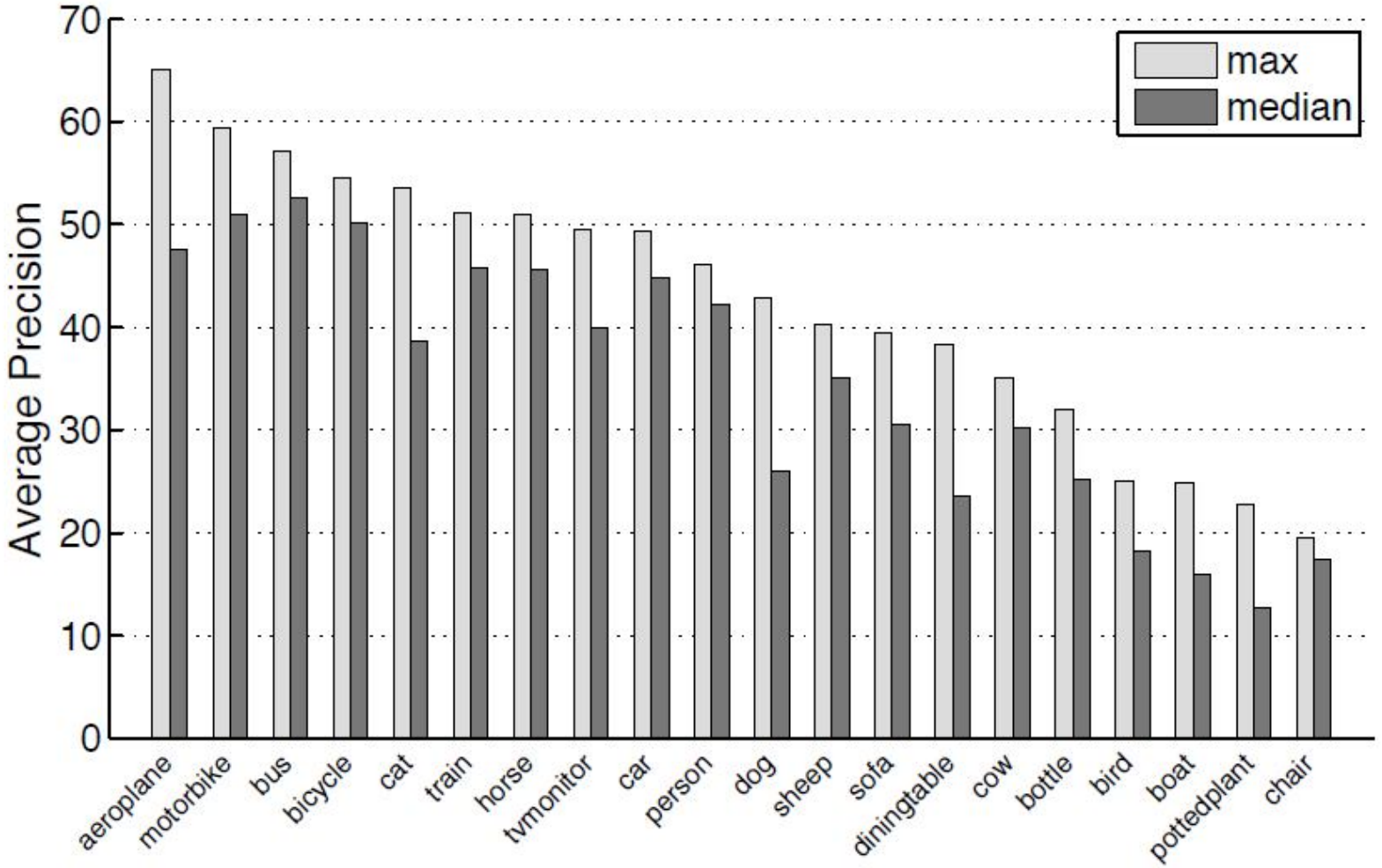
Detection if  > Threshold

50%

Точность (2009 год)



Точность (2012 год)



IMAGENET

Яндекс

14M изображений

Будет 1000 на каждую категорию

http://www.image-net.org

Large-scale visual recognition (2012)

Яндекс

Task 1: Classification

Car

- Predict a class label
- 5 predictions / image
- 1000 classes
- 1,200 images per class for training
- Bounding boxes for 50% of training.

Task 2: Detection (Classification + Localization)

classification Car

- Predict a class label and a bounding box
- 5 predictions / image
- 1000 classes
- 1,200 images per class for training
- Bounding boxes for 40% of training.

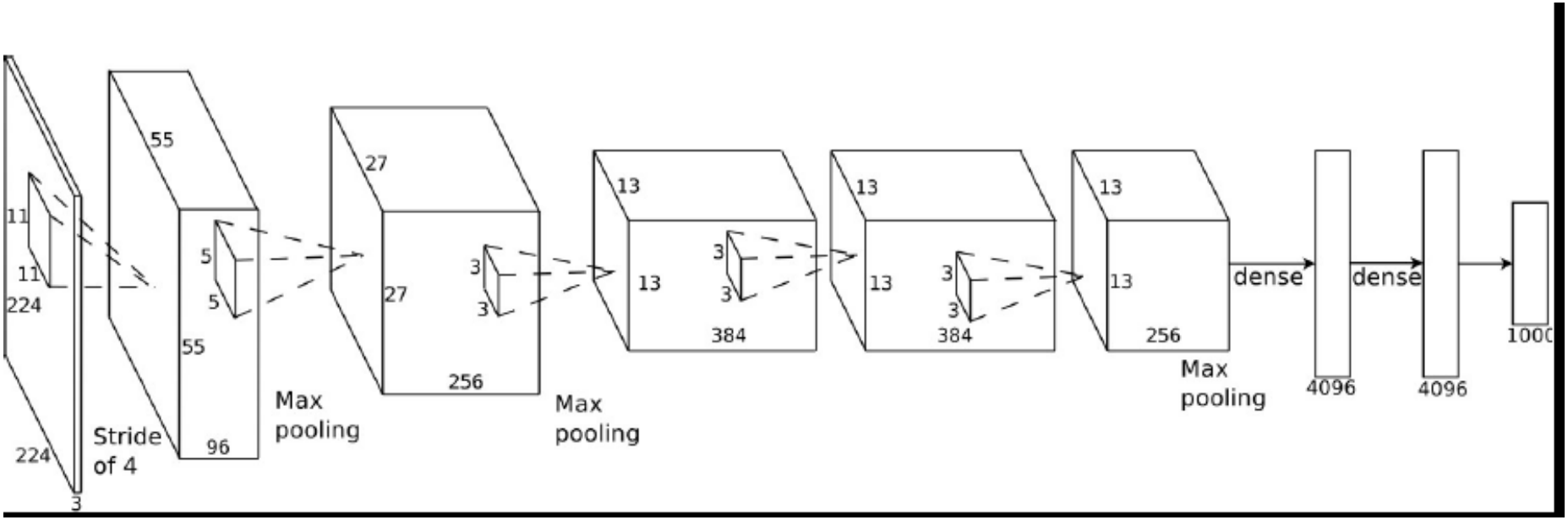
Task 3: Fine-grained classification

classification Walker hound

- Predict a class label given a bounding box in test
- 1 prediction / image
- 120 dog classes (subset)
- ~200 images per class for training (subset)
- Bounding boxes for 100% of training

1.2M изображений для обучения

Нейросети – победители (2012)



Нейросеть с 7 скрытыми слоями, обученная за неделю на 1 рабочей станции с 2 GPU

SuperVision

Alex Krizhevsky, Ilya Sutskever, Geoffrey Hinton

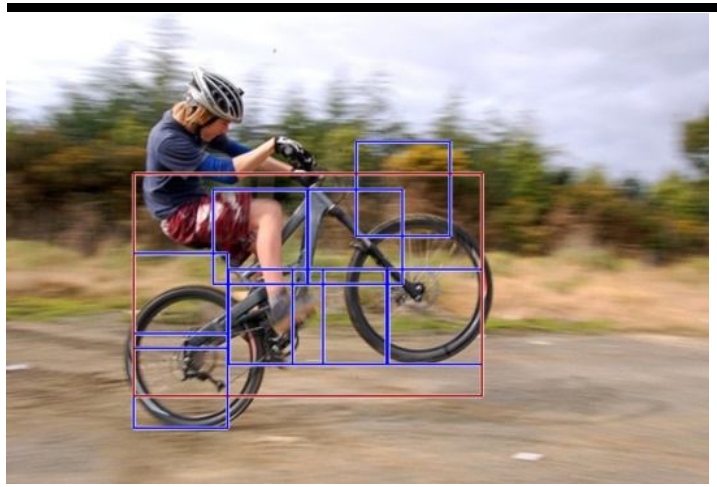
University of Toronto

Резюме выделения

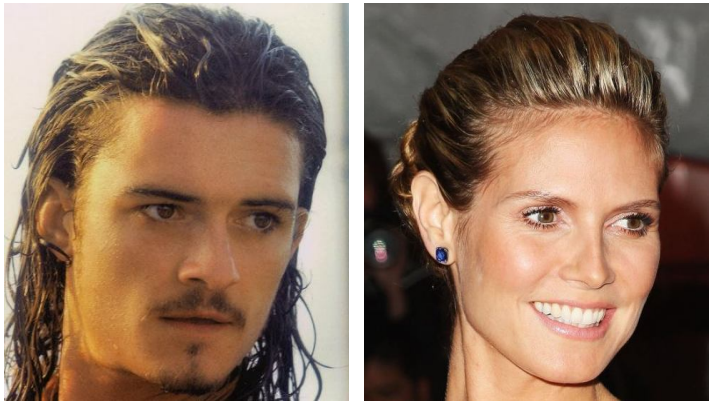
Яндекс

- Скользящее окно – основной способ сведения задачи выделения объектов к задаче категоризации изображений
- Каскад классификаторов – основной способ повышения скорости и точности работы
- Нужно уделить большое внимание построению хорошей выборки для обучения (jittering, bootstrapping)

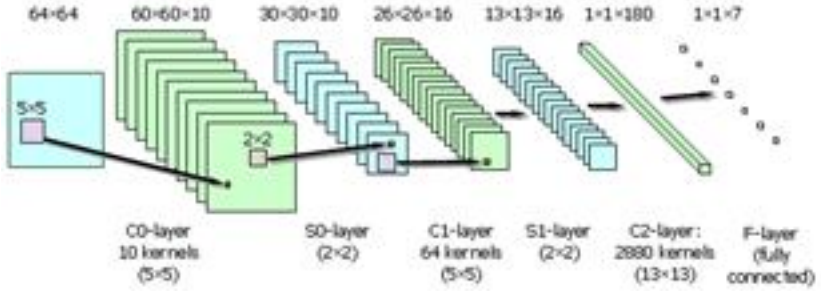
Завтра



Модели из набора частей



Лица



Многослойные нейросети

Большие коллекции

Thank you for evaluating AnyBizSoft PDF Splitter.

A watermark is added at the end of each output PDF file.

To remove the watermark, you need to purchase the software from

http://www.anypdftools.com/buy/buy-pdf-splitter.html