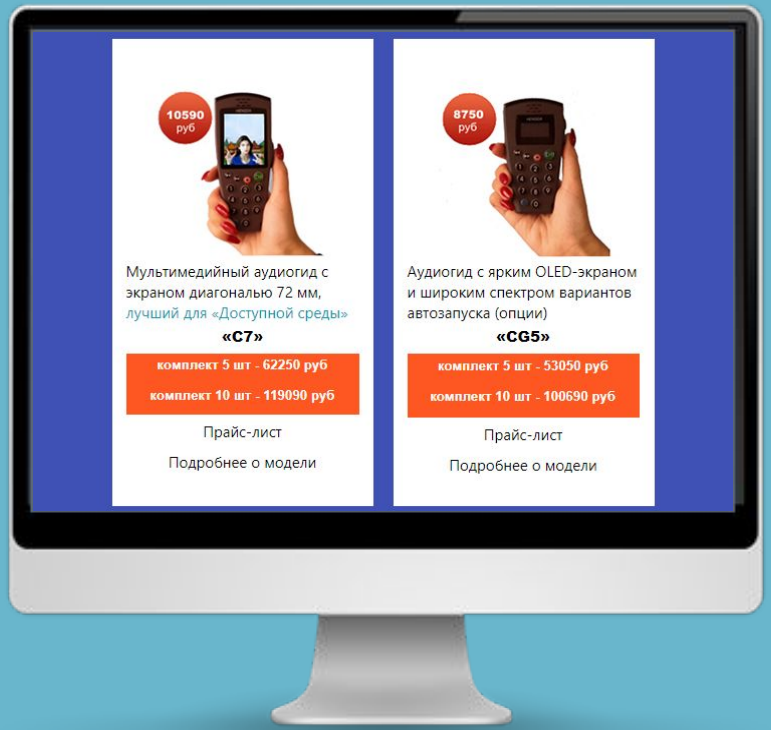
PHONE PAINT DETECTOR

Царикович Артур, г. Слуцк
Анастасия Кривленя, г. Слуцк

Главные проблемы

Деньги



Большие затраты на покупку комплектов **аудиогидов**

Эстетика

Порча атмосферы и **ухудшение** впечатления от картин и заведения

Безопасность

В настоящее время **более 23%** троянских программ и вирусов передаются **через QR-коды**.

Лю Цинфэн, председатель провайдера облачного сервиса распознавания голоса iFlytek

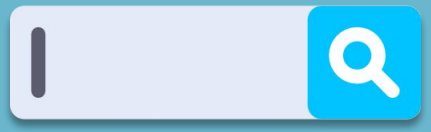
Есть ли решение?

Предоставить **информацию** с помощью **самой картины**

Что нужно для пользования:

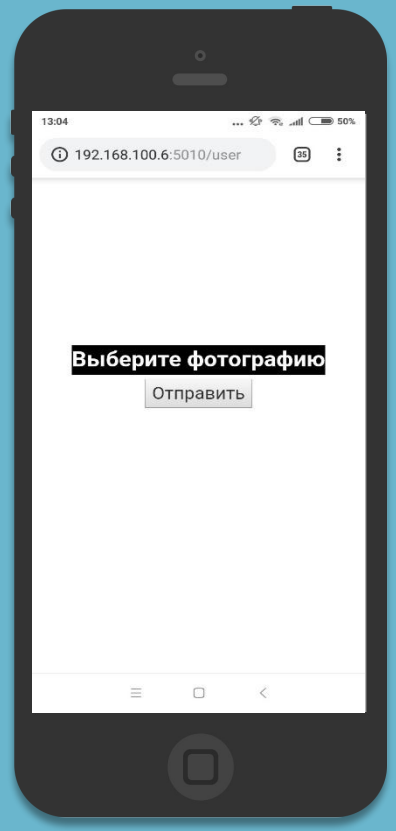
- Небольшие вычислительные мощности (обязательно с камерой)

- Зайти на сайт веб-приложения

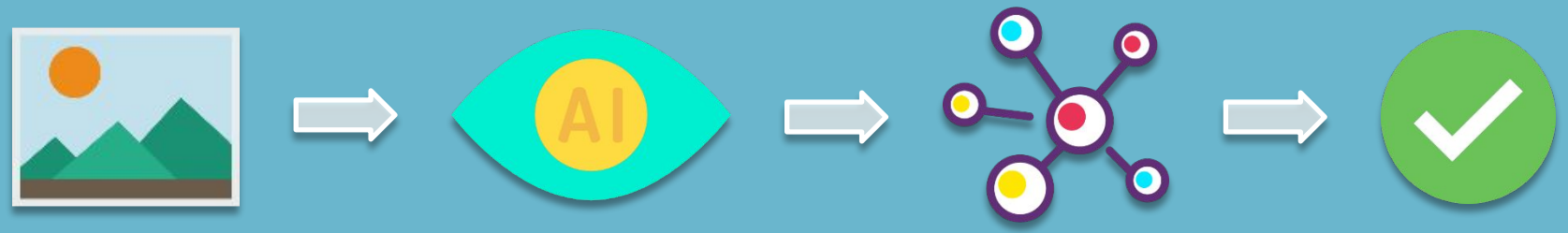


Как это работает:

1. Регистрация
2. Добавление и удаление
3. Распознавание



Распознавание картины происходит с помощью **уникальных признаков**



Алгоритмы

Поиск картины:

1. Алгоритм поиска прямоугольников

2. Своя нейронная сеть

Наш алгоритм

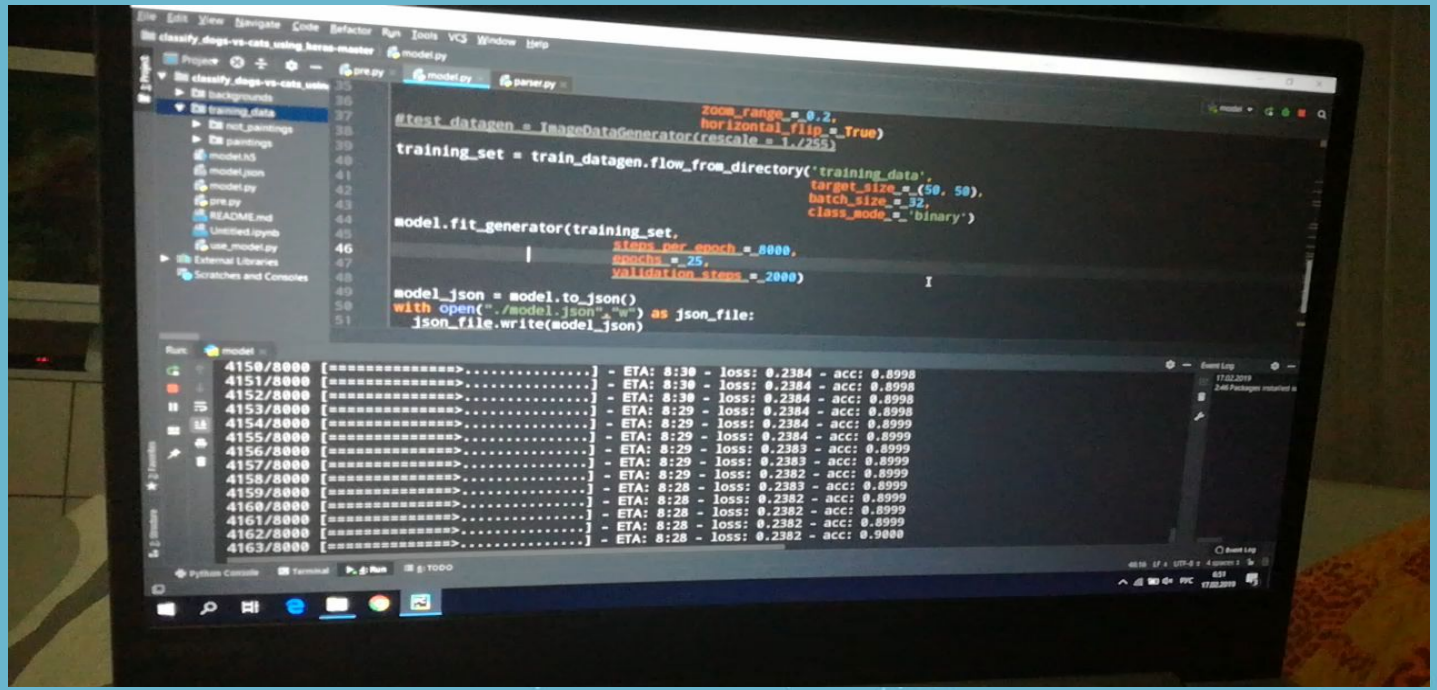
Оптимизация метода поиска картин с помощью алгоритма поиска прямоугольников заключается в использовании машинного обучения.

1. **Заготовка изображений картин и фонов**
   - Алгоритм получения данных с сайтов

2. **Генерация данных**
   - Алгоритм генерации данных, заключающийся в накладывании изображений картин в случайных масштабах и под случайным углами на случайные фоны.

3. **Создание и обучение нейронной сети**
   - Корректировка нейронной сети vgg16
   - Распределение данных по двум категориям: 1 – картина, 0 – картина на фоне

Переобучение нейронной сети:



Распознавание картин:

1. Нейронная сеть VGG16
2. База данных

KNN

Сравнивается каждый пиксель двух изображений D, L одинакового размера w*h и находится сумма разностей их значений.

$$d(D,L) = \sum_{i=1}^{w*h}{}_{j=1} = |D(x_i, yj) - L(x_i, yj)|$$

Таким образом, если размер изображений равен w*h, то скорость алгоритма поиска расстояния d(D,L) составляет O(2*w*h).

Поиск похожего изображения p по минимальному значению v_i расстояния можно представить так:

$$\mathrm{p} = \min(v_i)$$

Тогда скорость поиска похожего изображения при количестве кандидатов **n** равно O(n).

Общая сложность алгоритма равна O(2*w*h + n) и память M(k^2*w*h*n), где k – «приемлемый» масштаб изображения.

Features detection (alg. SIFT)

Обнаружение экстремумов в масштабном пространстве

довательных Гауссово-размытых изображений. Затем ключевые точки
принимаются как максимумы / минимумы разности Гаусса, которые встречаются в разных масштабах

$$D(x, y, \sigma) = L(x, y, k_i\sigma) - L(x, y, k_i\sigma)$$

Где $L(x, y, k_i\sigma)$ это свертка исходного изображения $I(x, y,)$ с размытием по Гауссу $G(x, y, k_i\sigma)$ в масштабе $k_i\sigma$ т.е.

$$L(x, y, k\sigma) = G(x, y, k\sigma) * I(x, y)$$

Интерполяция соседних данных для точного положения

масштаба. $D(x,y, \sigma)$ с ключевой точкой кандидата в качестве источника.

$$D(x) = D + \frac{\partial D^T}{\partial x} x + \frac{1}{2} x^T \frac{\partial^2 D}{\partial x^2} x$$

где D и его производные оцениваются в ключевой точке кандидата и $x = (x,y,\sigma)^T$

Ориентация назначения

Для образца изображения в масштабе величина градиента и ориентация предварительно вычисляются с использованием разностей пикселей:

$$m(x, y) = \sqrt{(L(x+1, y) - L(x-1))^2 + (L(x, y+1) - L(x, y-1))^2}$$

$$\theta(x, y) = atan2(L(x, y+1) - L(x, y-1), L(x+1, y) - L(x-1, y))$$

Обнаружение экстремумов в масштабном пространстве занимает O(w*h* $\mathbf{V(\sigma)}$), при размере изображения равное w*h и вариативности масштабов V($\boldsymbol{\sigma}$).

На интерполяцию соседних данных для точного положения тратится O(n*$\boldsymbol{\sigma}$) времени, где n – количество ключевых точек, найденных на предыдущем этапе.

На ориентацию назначения требуется еще O(n*4*2), где n – количество ключевых точек, найденных на предыдущем этапе.

Сопоставление ключевых точек соответствует требует O(n*m) времени, где m – количество ключевых точек второго изображения.

Таким образом, пренебрегая многими операциями, общее время составляет O(w*h* $\mathbf{V(\boldsymbol{\sigma}) + \boldsymbol{n}*\boldsymbol{\sigma}}$ + 8n + n*m) и память M(k*E_i), где k – количество картин, E_i - вектор признаков i-й картины

Neural network (alg. VGG16)

Каждый пиксель изображения $L(x_i,y_j)$ умножается на вес соответствующего w_{ij} нейрона.

$$v_{ij} = L(x_i, y_j) * w_{ij}$$

Время равно O(w*h).

Сравнение каждого i-го признака матриц признаков v1 и v2 с помощью евклидового расстояния:

$$S_{ij} = \sqrt{(v1_i\,[0] - v2_i[0])^2 + (v1_i\,[1] - v2_i[1])^2}$$

Поиск минимального значения r среди S_{ij} $m = \min(S_{ij})$

Random Forest

Каждый пиксель изображения $L(x_i, y_j)$ проходит через каждое дерево и умножается на его вес w_k.

$$v_{ij} = L(x_i, y_j) * w_n$$

Время равно O(**k**^{2}w*h*n), где n - количество деревьев.

Затем также, с помощью евклидового расстояния находится дистанция, где I и j – два и зображения.

$$S_{ii} = \sqrt{(v1_i\,[0] - v2_i[0])^2 + (v1_i\,[1] - v2_i[1])^2}$$

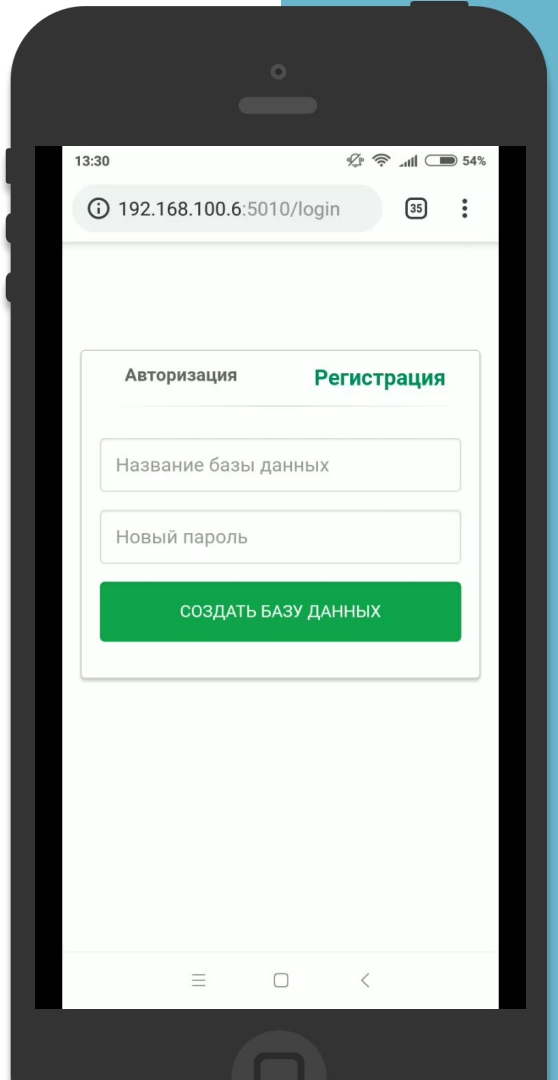
Сопоставление точек соответствует требует O($k^4w^2h^2$) времени.

Таким образом, общее время составляет O(k^2*w*h*n +k^4*w^2h^2) и память M(k^2*n*w*h_i).

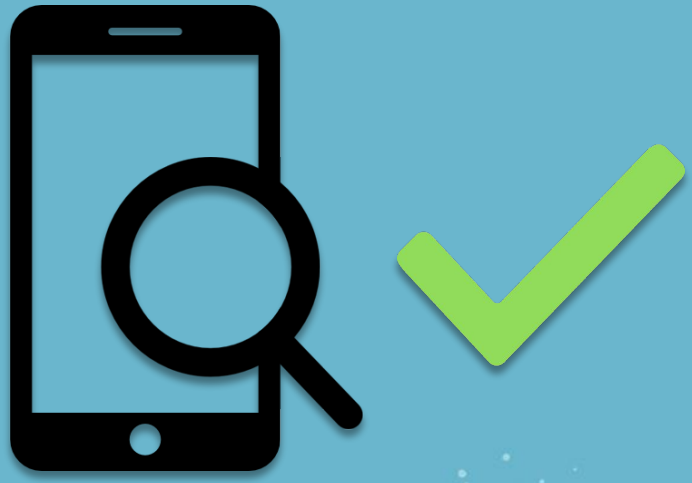
Таблица скорости работы (сек) соответствующих алгоритмов

	KNN	Feature Search	Neural Networks (VVG16)	Random Forest
Speed (sec)	5,6 – 5,8	10,0 – 10,1	7,8 – 8,0	8,0
Database memory (kB)	79 600	390 710	98 900	390 710
Conclusion	Есть недостаток	Не оптимален	Оптимален	Не оптимален

Практическая часть



В результате…



Спасибо за внимание!