



Сервис по поиску домашних животных

22.10.2021

— Богомолов Кирилл, Данилов Максим, Скороходов Всеволод

МФТИ

Содержание

1. Как использовать сервис
2. Объем задач, решаемый сервисом
3. Детекция животного
4. Классификация животного
5. Классификация породы собаки и длины ее хвоста
6. Определение цвета
7. Определение наличие хозяина
8. Улучшение качества вырезанной фото с помощью NN
SuperResolution
9. Взаимодействие сервера с клиентом.
10. Количественные результаты
11. Технические характеристики системы, на которой обучали

Как использовать сервис

С сервисом можно ознакомиться по ссылке: <http://194.87.237.22:5000/>

1. Загрузить серию из фотографий
2. Нажать "Получить метки"
3. Подождать, пока нейросеть их обработает. Для ~10 фото обработка до 60 секунд



Найти моего питомца

**Найдите своего потерявшегося
животного всего за несколько
кликов**

С помощью сети камер городского видеонаблюдения
г.Москвы

Найти моего питомца





Заполните информацию о вашем питомце

После этого вы увидите результаты поиска

Город поиска	<input type="text" value="Москва"/>
Кого нужно найти?	<input type="text" value="Собака"/>
Цвет питомца	<input type="text" value="Бежевый"/>
Порода питомца (ближайшая)	<input type="text" value="Пудель"/>
Длина хвоста	<input type="text" value="Длинный"/>
Примерный адрес, где потерянся питомец <input type="text" value="Парк Горького, южный вход"/>	
<input type="checkbox"/> Находить также фото с хозяином	

[Найти моего питомца](#)

CSV файл с метками для тестового датасета лежит на гитхабе:

<https://github.com/kirrya95/leaders2021>

Объём задач, решаемый сервисом

Мы решали наиболее общую задачу: по серии снимков с камер видеонаблюдения находится собака, её цвет, порода, длина хвоста, с хозяином/без хозяина.

Наш сервис умеет:

1. По серии снимков распознавать собаку
2. Определять:
 - а) цвет
 - б) похожую породу
 - в) с хозяином / без хозяина
 - г) хвост: (длинный/короткий)
 - д) время съёмки, адрес камеры

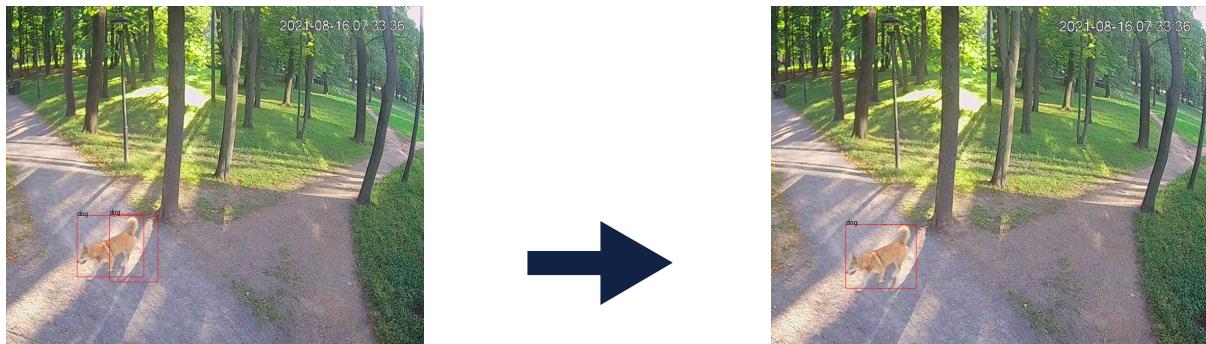
Технические подробности

1. Детекция животного

Для детекции мы попробовали несколько моделей: YOLOv5, Mask R-CNN ResNet-50 FPN, Faster R-CNN ResNet-50 FPN, Faster R-CNN MobileNetV3-Large FPN, RetinaNet ResNet-50 FPN. Но мы остановились на Faster R-CNN MobileNetV3-Large FPN ввиду ее скорости, качество детекции на всех моделях не сильно отличалось.

Мы заметили, что детектор не всегда корректно распознает объекты, но хорошо выделяют их, поэтому мы решили обрабатывать фрагменты с изображениями животных и скамейки.

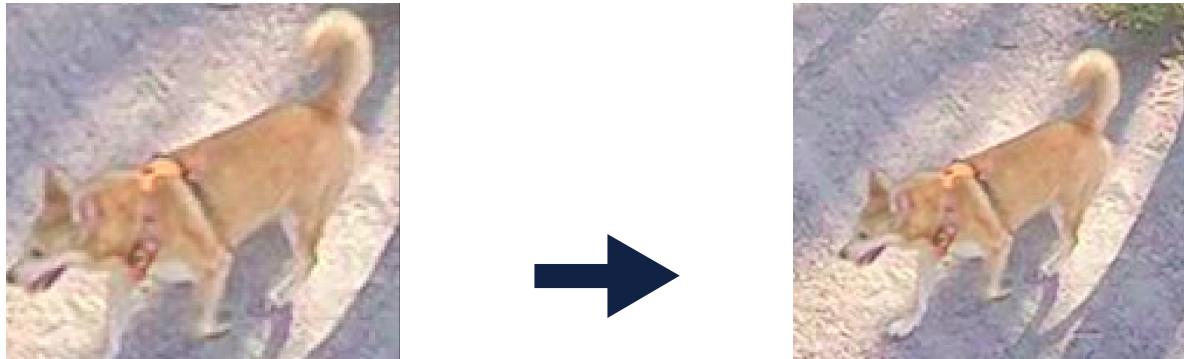
Одной из проблем стала то, что детектор несколько раз выделяет одну и ту же собаку. Для ее решения мы отсеивали слишком маленькие фрагменты и выбирали среди пересекающихся объект с максимальной площадью или с меткой "собака".



2. Классификация

2. В ходе проверки детектора на картинках из выданного датасета было замечено, что он детектирует собак не только к собакам, но к кошками и (???) скамейкам и людям. Так что мы решили установить классификатор для проверки фрагментов изображения.

Следующая проблема была в том, что детектор дает сильное ограничение на размеры детектируемых фрагментов, так что мы реализовали функцию, которая получает фрагменты изображения от детектора и увеличивает ширину окна в некоторое количество раз.



Для классификации собаки использовали модель CLIP (<https://github.com/openai/CLIP>) с токенами:

- 1) "an animal", "not an animal", "a person", "a bench", "a car" (для определения, что это животное)
- 2) "a dog", "a cat", "a bird", "a bench" (для определения, что это собака)

3. Классификация породы

3. Далее необходимо классифицировать породы собак. Для этого мы решили взять модель CLIP и токены для неё:

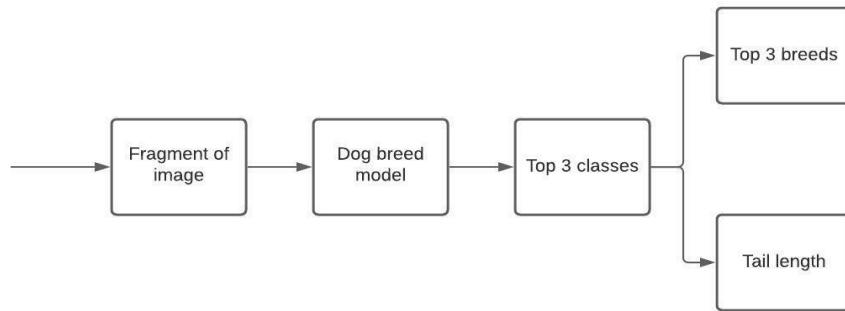
```
'a dachshund', 'a borzoi', 'a basset hound', 'a Pomeranian dog', 'a yorkshire terrier', 'a german shepherd', 'a labrador', 'a husky', 'a jack russell terrier', 'a caucasian shepherd dog', 'a corgi', 'a French Bulldog', 'a pug', 'a poodle', 'a golden retriever', 'a rottweiler', 'a cocker spaniel', 'a deutscher boxer', 'a pekingese', 'a sharpey', 'a bull terrier', 'a doberman', 'a st. bernard', 'a newfoundland', 'a border collie', 'a shiba inu', 'a dalmatian', 'a samoyed', 'a bichon frise', 'a toy terrier', 'a cavalier king charles spaniel'.
```

Всего **30** пород.

Следующей проблемой было определение длины хвоста собаки. Мешали 2 фактора:

- 1) Субъективность определения длины
- 2) Плохое качество картинки и, как следствие, сложности в обнаружении хвоста
- 3) Отсутствие размеченного датасета

Для решения второго пункта мы решили определять длину хвоста по породе собаки(в дальнейшем возможно улучшение или отказ от этого метода). Что касается первого пункта, от субъективности при определении длины хвоста мы избавиться не могли, так что сами разметили данные по длине хвоста в зависимости от породы.



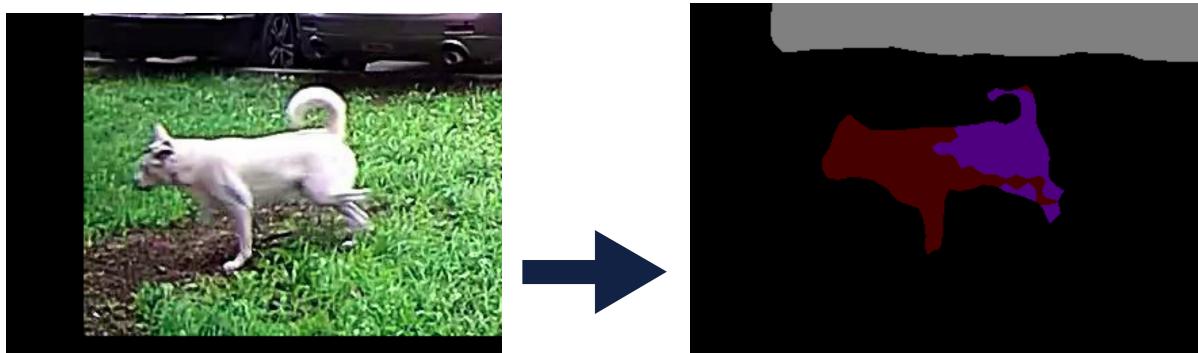
3. Определение цвета

Для определения цветов у нас было несколько идей, **но взяли пункт Г.**

а) Мы хотели выделить границы собаки на изображении с помощью Gradient Border и просто посчитать средний цвет выделенного контура. Однако пока от такой идеи пришлось отказаться. Во-первых, качество картинок довольно низкое, поэтому Gradient Border плохо находил границы, либо вообще не находил. Во-вторых, опять таки из-за плохого качества Gradient Border мог выделять границы других объектов(скорее всего, потому что эти объекты имели более простую форму и, соответственно, их можно было выделить), например, заборов, что тоже приводило к неправильному определению цвета.

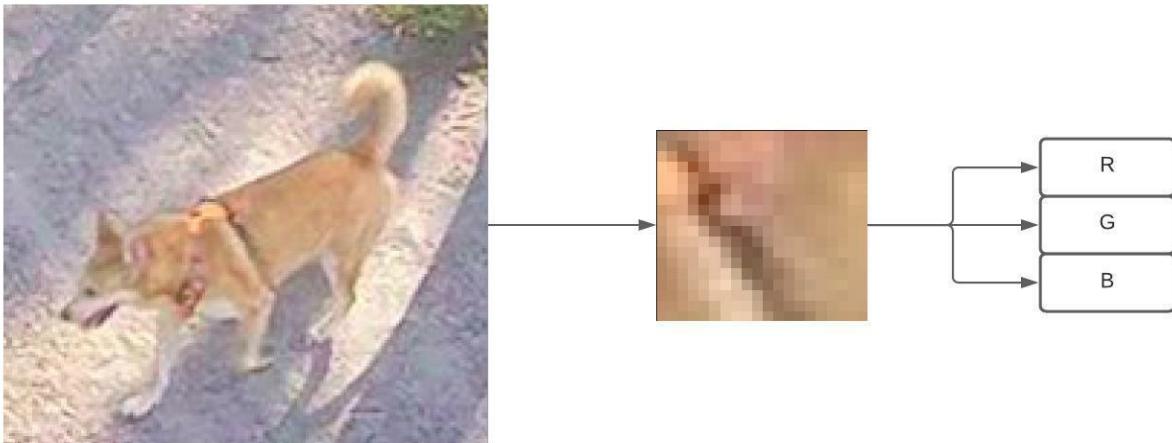


б) Мы пробовали решать это через семантическую сегментацию. Посмотреть на выделенную собаку и пройтись по этому фрагменту и уже там посчитать цвета.



Однако тут опять появились проблемы: из - за низкого качества ему проблематично распознавать собаку, так что он одну и ту же собаку мог на несколько разных выделить. К тому же мы не учитываем таким образом участки с тенями, которые сильно влияют на среднее значение. Использовали сегментационные модели: DeepLabV3 MobileNetV3-Large, DeepLabV3 ResNet101, FCN ResNet50.

в) Третий вариант был более простой, но и надежнее. Так как детектор выдавал нам фрагменты изображения с собаками, то всего скорее, если выделить небольшой квадрат в центре, то он будет полностью принадлежать собаке. Конечно, возможно такие выбросы, как тень или то, что собака пятнистая или разноцветная, поэтому возможно в дальнейшем придется эту модель улучшать. После получения этого квадрата, усредняем каждый из каналов RGB.



Далее нужно было по RGB описанию вернуть цвет. Мы взяли самые популярные и известные цвета и определяли ближайший к полученному RGB цвет. Как оказалось использовать евклидово расстояние не очень хорошо в силу различного восприятия человеком различных цветов. Поэтому для подсчета расстояний мы пользовались библиотекой colormath.

г) Но мы решили использовать CLIP с токенами: "a black dog", "a white dog", "a brown dog", "an orange dog", "a chocolate dog", "a liver dog", "a red dog", "a gold dog", "a cream dog", "a fawn dog", "a blue dog", "a gray dog", "a grey dog".

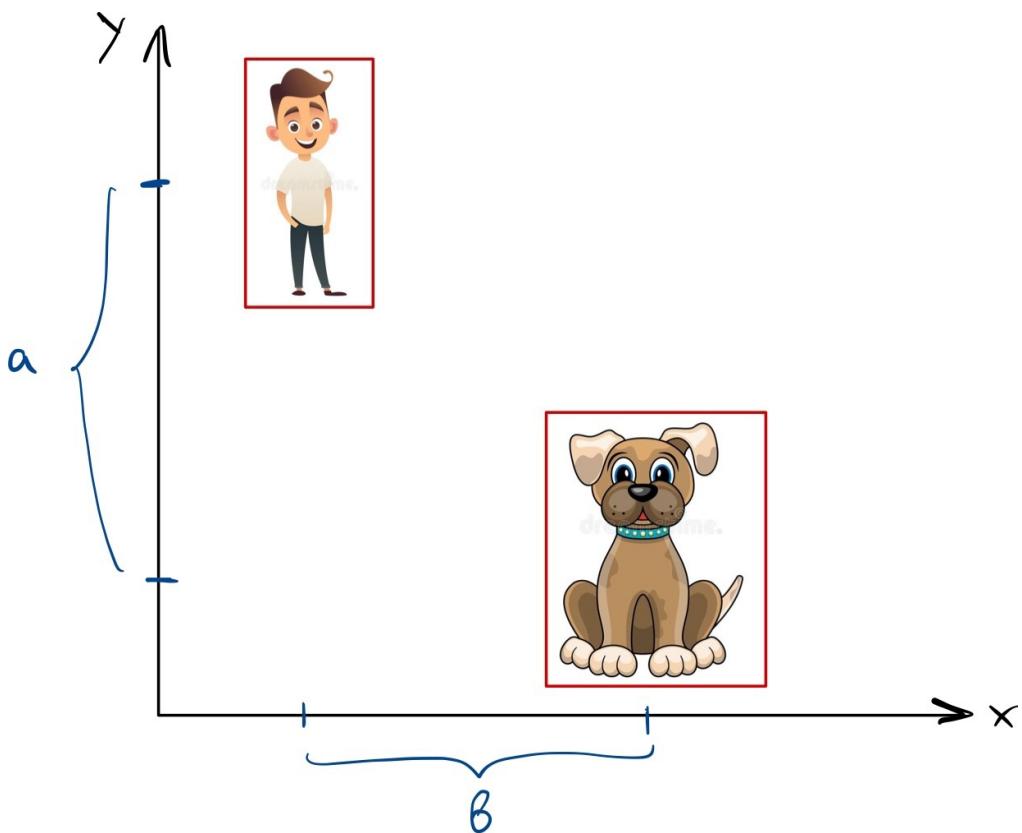
5. Наличие хозяина

Для определения наличия хозяина на фотографии необходимо было определить есть ли на фото человек. С этой задачей хорошо справлялся детектор, так как человек достаточно большой, чтобы успешно его детектировать и классифицировать. За неимением лучшего мы пытались измерять расстояния между людьми и собаками. Из-за перспективы проблематично вычислять его точно, поэтому здесь мы ограничились лишь качественными оценками.

Помимо непосредственного вычисления расстояния между центрами прямоугольников (назовём расстояние a), в которых находятся собака и человек, мы это делаем на нормировочный коэффициент равный высоте человека, так как в среднем человек в несколько раз по размеру превышает собаку. Значит если он оказался маленьким относительно собаки, то он находится далеко и, вероятно, не является хозяином данной собаки.

Теперь разберёмся с горизонталью. Мы заметили, что в среднем собаки "шире" человека. Поэтому в данном случае в качестве нормировочного коэффициента выступит ширина собаки (это связано ещё и с тем, что, как правило, маленькие собаки отходят от хозяина на меньшее расстояние). (Расстояние по горизонтали назовём b)

Из двух полученных величин берем максимальную и сравниваем с какой-то предельной. Мы решили взять предельную равную 3,0.



6. Улучшение качества фрагмента изображения

Мы решили, что на сайте было бы хорошо выводить фрагмент изображения, на котором обнаружена собака. Однако, как было видно выше, картинки имеют достаточно плохое качество. Чтобы визуально смотрелось лучше, мы решили улучшить качество картинки с помощью NN Super Resolution (https://github.com/Saafke/FSRCNN_Tensorflow/blob/master/models/FSRCNN_x4.pb) .

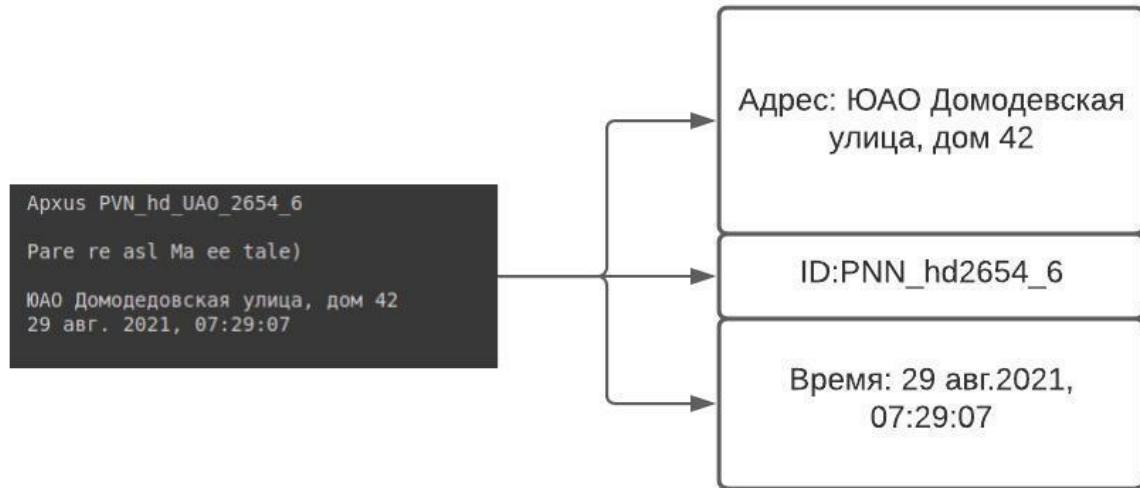


7. Определение времени съемки и адреса камеры

Для определения времени и адреса мы решили считывать информацию непосредственно с фотографии, для этого была необходима модель для выделения текста с картинки. Мы решили воспользоваться OCR из opencv, однако он работает достаточно медленно (~30 сек на одной картинке), поэтому мы решили отказаться от этой модели.

Далее мы решили воспользоваться функциями библиотеки pytesseract. Они хоть и используют более простые методы, работают быстро и достаточно хорошо для решения нашей задачи. Так как камеры имеют примерно одинаковый формат, для извлечения необходимых нам меток (время, адрес и ID камер) из детектированного текста, мы воспользовались регулярными выражениями.





9. Взаимодействие сайта с сервером и моделью

Технологии:

1. Страницы:

1. html + tailwindcss
2. JS, Ajax, jquery

2. Backend:

1. Flask (python)
2. Работа с csv через pandas

3. Geocoder для получения координат по адресу

1. через api



Как происходит взаимодействие? Есть один сервер 1, который постоянно обрабатывает камеры и отправляет csv файл и вырезанные фрагменты на сервер 2, на котором происходит взаимодействие с сервером.

10. Результаты

Тестовый датасет для породы собак:

<https://www.kaggle.com/jessicali9530/stanford-dogs-dataset>

Однако в нём были слишком хорошие картинки по сравнению с реальными, поэтому мы ухудшили качество. Наш датасет можно найти по ссылке

Тестовый датасет для собака/не собака:

<https://www.kaggle.com/c/dogs-vs-cats/data>

Качество на нём мы опять же подпортили

На породы топ 1: в среднем 70%

На кошка/собака: 99%

Подробнее смотреть в файле validation.

11. Технические характеристики системы

Обучали на Гугл колабе в основном, поэтому довольно сложно определить истинную конфигурацию машины. Все результаты можно посмотреть в наших ноутбуках.