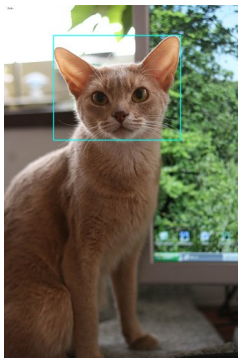
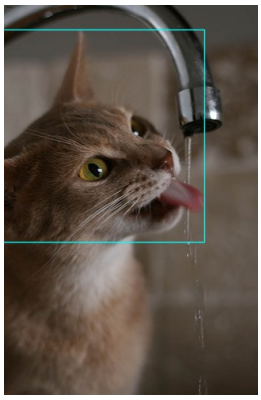
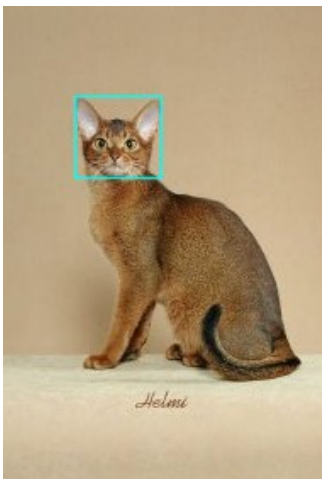
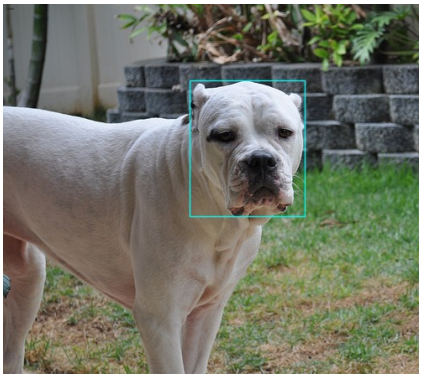
Вам прелагается решить следующую задачу:

Необходимо создать и обучить свёрточную нейронную сеть, которая локализует и классифицирует только 1 объект на изображении.
Датасет находится в той же папке, где и этот документ.

Датасет содержит 1037 изображений кошек и 2348 изображений собак, 3385 всего.




К каждому изображению в архиве cats_dogs_dataset.tar соответствует файл разметки RoI (region of interest) изображения. В данном случае это мордочка животного.

Файл разметки выполнен в следующей нотации.

class xmin ymin xmax ymax

Где *class* это id класса животного. В данном датасете таких класса всего два:

1 – кошка 2 – собака

A xmin ymin xmax ymax это абсолютные координаты bouding box’a на изображении, левого верхнего и правого нижнего углов соответственно.

Например, файлу Abyssinian_123.jpg соответствует файл Abyssinian_123.txt, в котором одной строкой записаны следующие цифры:

```
1 153 81 333 221
```

В данном датасете на любом изображении животное присутствует только одно и, как следствие, RoI тоже только один. Один - ни больше, ни меньше.

Для решения данной задачи Вам предлагаются следующие пути:
1) Сформировать собственную свёрточную нейронную сеть с пятью выходами и обучить её.
2) Применить transfer learning. Где embeddings уже обученной сети, например, mobilenet или inceptionv3, будут регрессироваться в полносвязаных слоях к пяти координатам.
3) Применить существующий general purpose object detector, такой как ssd, faster rcnn, yolo, retinanet и т.п. для детекции и классификации мордочки животного. Однако, мы не рекомендуем идти этим путём, разве только в ознакомительных целях.

Решение можно выполнить на любом языке программирования, с помощью любых фреймворков машинного обучения. В случае, если Вы справитесь с задачей не применяя их (фреймворки), это будет большим плюсом.

Будет плюсом, если Вы представите несколько вариантов решения. Допустим, обучите собственную свёрточную нейронную сеть и сделаете дополнительное решение с использованием transfer learning.

По каждому предложенному Вами решению необходимо прислать полученные метрики точности. Под метрикой точности понимаются два числа: mIoU (см. в ссылках ниже) и accuracy классификации в процентах. Так же необходим численный размер валидационного датасета и время на выполнение одного инференс прохода.

Например:

```
mIoU 75%, classification accuracy 94%, 0.09ms, 3000 train, 385 valid.
```

Эксперементируйте с размерностью слоёв, функциями активации, оптимизаторами, размерами входного окна и глубиной сети.

Ссылки и материалы:

Google colab, доступные вычислительные мощности от google, tesla k80:
https://medium.com/tensorflow/colab-an-easy-way-to-learn-and-use-tensorflow-d74d1686e309

Одна из популярных и довольно фундаментальных книг доступных в веб:
http://neuralnetworksanddeeplearning.com/

Тоже очень серьёзный ресурс:
http://cs231n.github.io/

Рекомендуем ознакомиться с разделом *Learn and use ML* на:
https://www.tensorflow.org/tutorials

Туториалы на tensorflow: https://github.com/Hvass-Labs/TensorFlow-Tutorials

Что такое mean Intersection over Union: https://www.pyimagesearch.com/2016/11/07/intersection-over-union-iou-for-object-detection/

Аугментация (!):

Процесс искуственного увеличения обучающей выборки:

https://medium.com/nanonets/nanonets-how-to-use-deep-learning-when-you-have-limited-data-f68c0b512cab

https://medium.com/nanonets/how-to-use-deep-learning-when-you-have-limited-data-part-2-data-augmentation-c26971dc8ced

В tensorflow за это отвечает пакет image: https://www.tensorflow.org/api_docs/python/tf/image

Так же есть очень интересный проект imgaug: https://github.com/aleju/imgaug

Касательно пункта 1 из предложенных вариантов решения:

В туториалах от Hvaas-Labs, первые два дадут хорошее представление, что такое свёрточная сеть:
https://github.com/Hvass-Labs/TensorFlow-Tutorials/blob/master/01_Simple_Linear_Model.ipynb
https://github.com/Hvass-Labs/TensorFlow-Tutorials/blob/master/02_Convolutional_Neural_Network.ipynb

Вам надо будет модфицировать свертночую сеть так, чтобы она имела большее входное окно, например 220 на 220 и имела 5 выходов, 1 из которых отвечал бы за классификацию, а остальные 4 за координаты. Координаты рекомендуем перевести в относительные (разделить на ширину и высоту соответственно).

Проходя туториалы по классификации рукописных цифр, рекомендуем заглянуть в https://github.com/zalandoresearch/fashion-mnist , где в разделе Benchmarks (табличка) сможете подсмотреть различные архитектуры сетей и варианты их оптимизации (batchnorm, dropout, shortcuts).

Касательно пункта 2

Самый простой путь, руководство к действию:
https://blog.keras.io/building-powerful-image-classification-models-using-very-little-data.html
https://jkjung-avt.github.io/keras-tutorial/

О transfer learning:
http://cs231n.github.io/transfer-learning/

Касательно пункта 3

В целом о детекции объектов:

https://leonardoaraujosantos.gitbooks.io/artificial-inteligence/content/object_localization_and_detection.html

О том как работает yolo:
https://machinethink.net/blog/object-detection-with-yolo/

Сама yolo:
https://pjreddie.com/darknet/yolo/

Яркий представитель другого вида детекторов:
https://github.com/rbgirshick/py-faster-rcnn