platypus - R package for object detection and image segmentation

Michał Maj

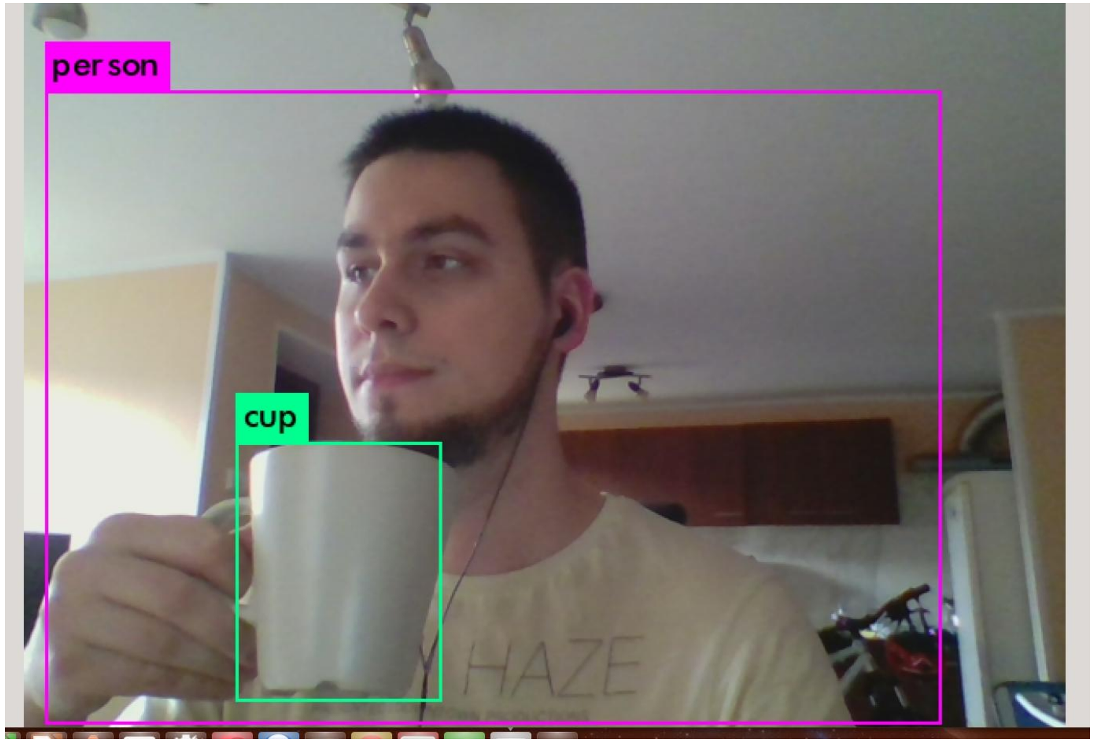
07.09.2020

Who am I

My name is Michał Maj:

- Linkedin https://www.linkedin.com/in/michal-maj116/
- Twitter @MichalMaj116

I am a Data Scientist at Billenium.



I am interested in machine/deep learning and statistics. I love new challenges and I'm always ready to help solving data science problems. I'm a big R language enthusiast and a co-organizer R Enthusiasts meetups in Gdańsk (https://www.meetup.com/Trojmiejska-Grupa-Entuzjastow-R/). Currently trying to become a deep learning expert!

My first big deep learning project - e-pionier

One of many projects from program called "e-pionier", financed by EU.

Goal:

Create a system that could predict:
- **patient medical condition (e.g urgent, stable) - 4 categories**
- **has pathological changes in the hip joints (e.g cysts, osteophytes) - 5 types of changes**
- **give doctors an easy to understand explanation of the model and prediction**

Steps

Step 0 :

- **collect the data (few thousands of DICOM X-ray images of hip joints)**
- **anonymization (remove from DICOMs personal patient info - name, address etc.)**
- **create a labels / descriptions**

Step 1 :

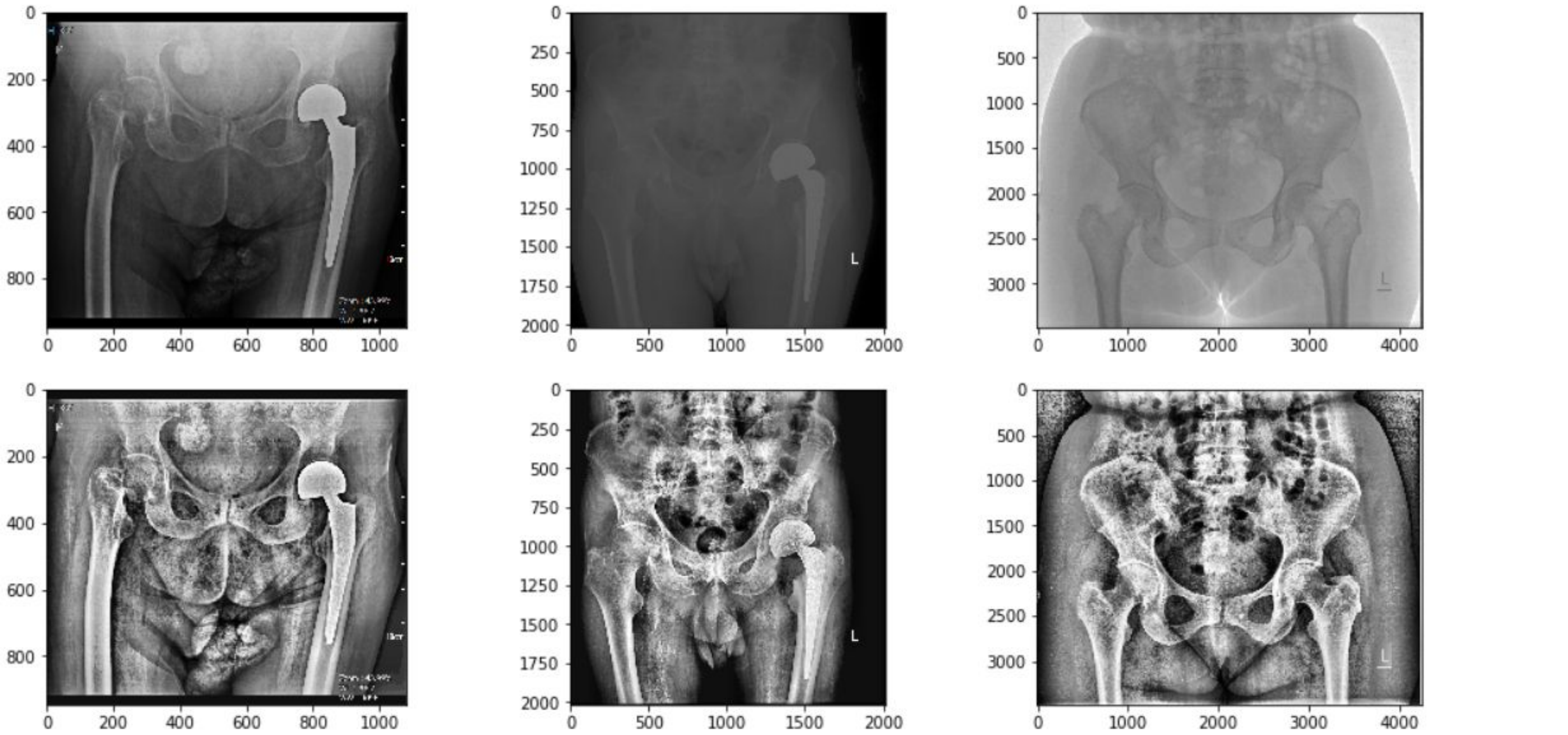
- **deep learning model(s) for pathological changes and patient condition (CLASSIFICATION)**

Step 2 :

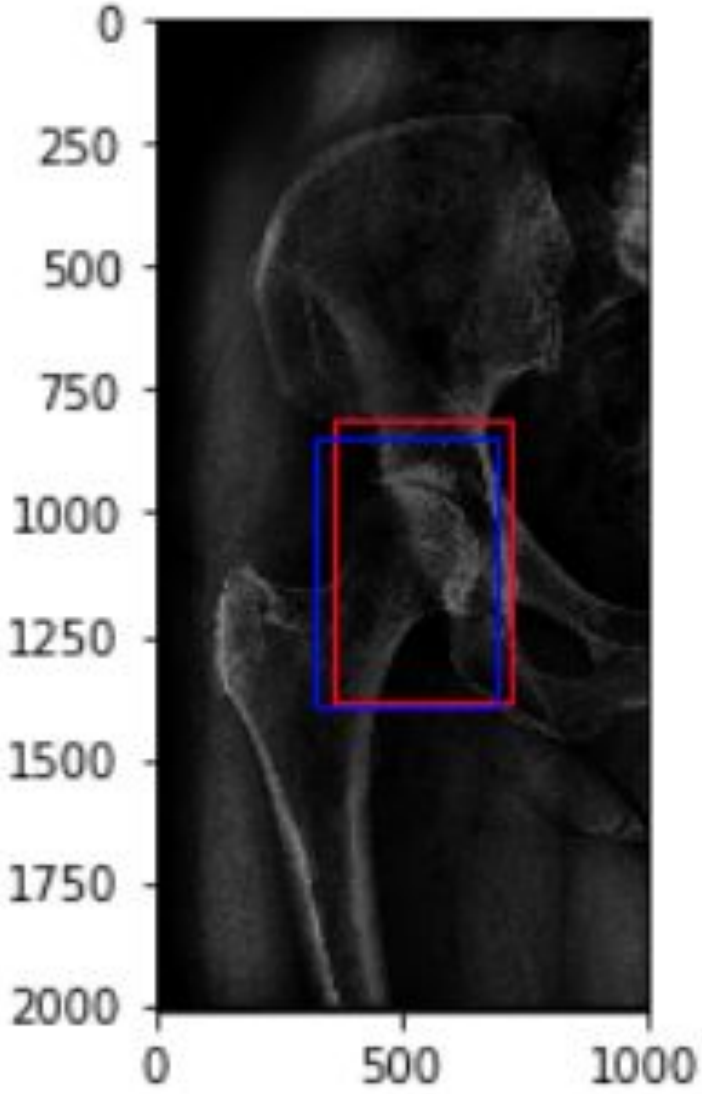
- **create a system that using above model(s) and patient information (e.g. age, sex, medical history) will give the final prognosis and explanation.**

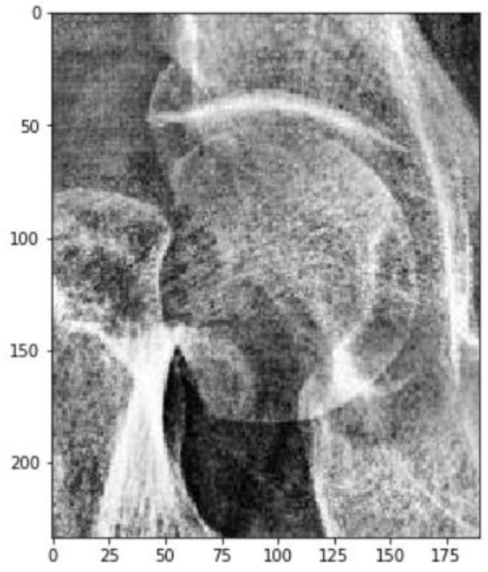
Step 0 - data collection and labeling

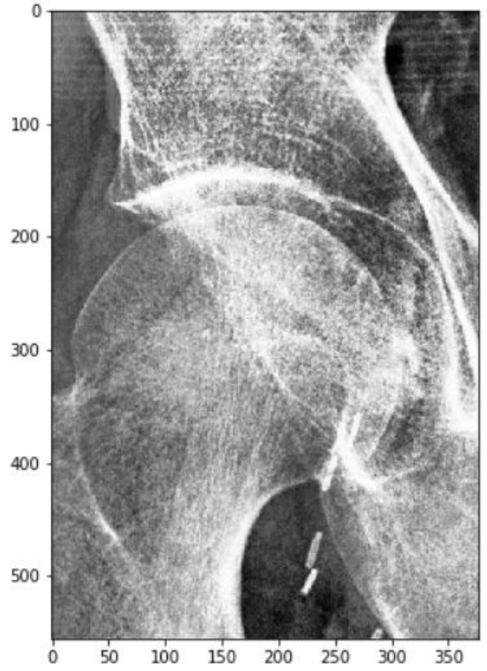
Step 1 - data processing



Step 1 - localization model





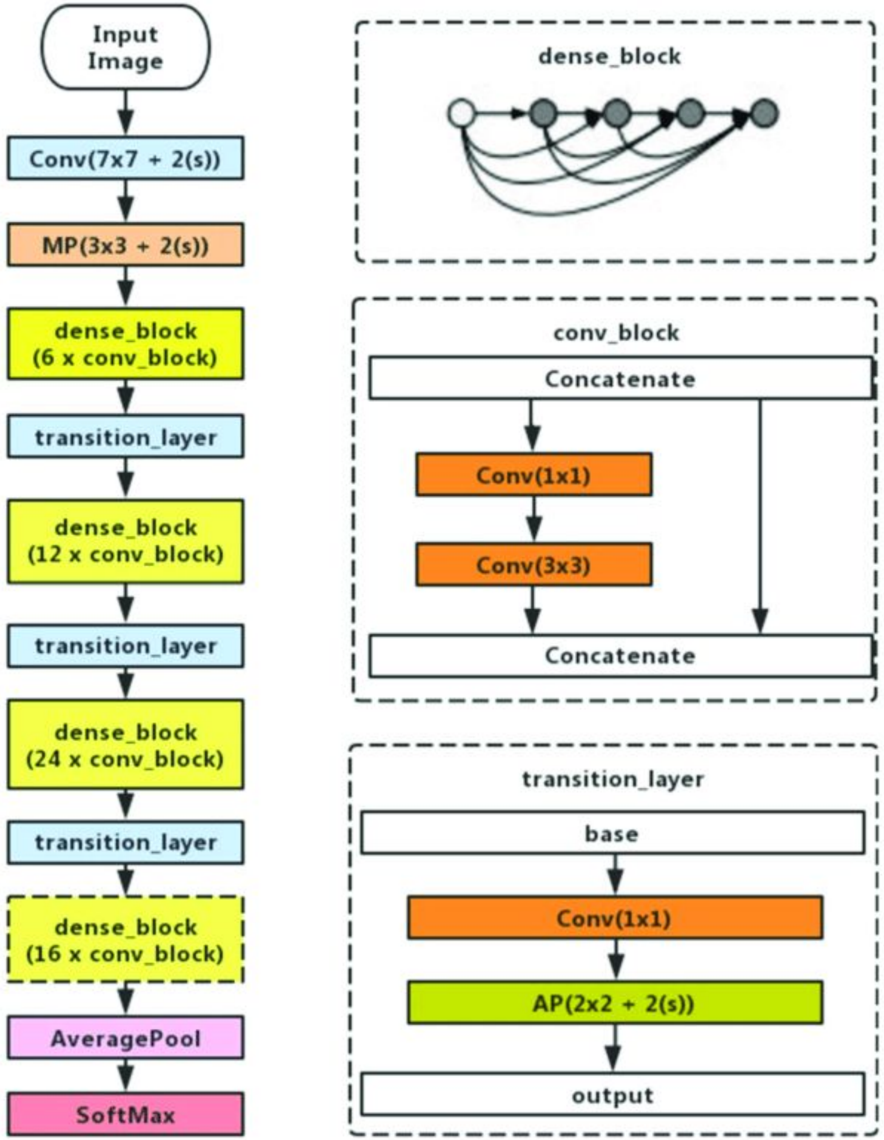


Step 1 - pathological changes classifications

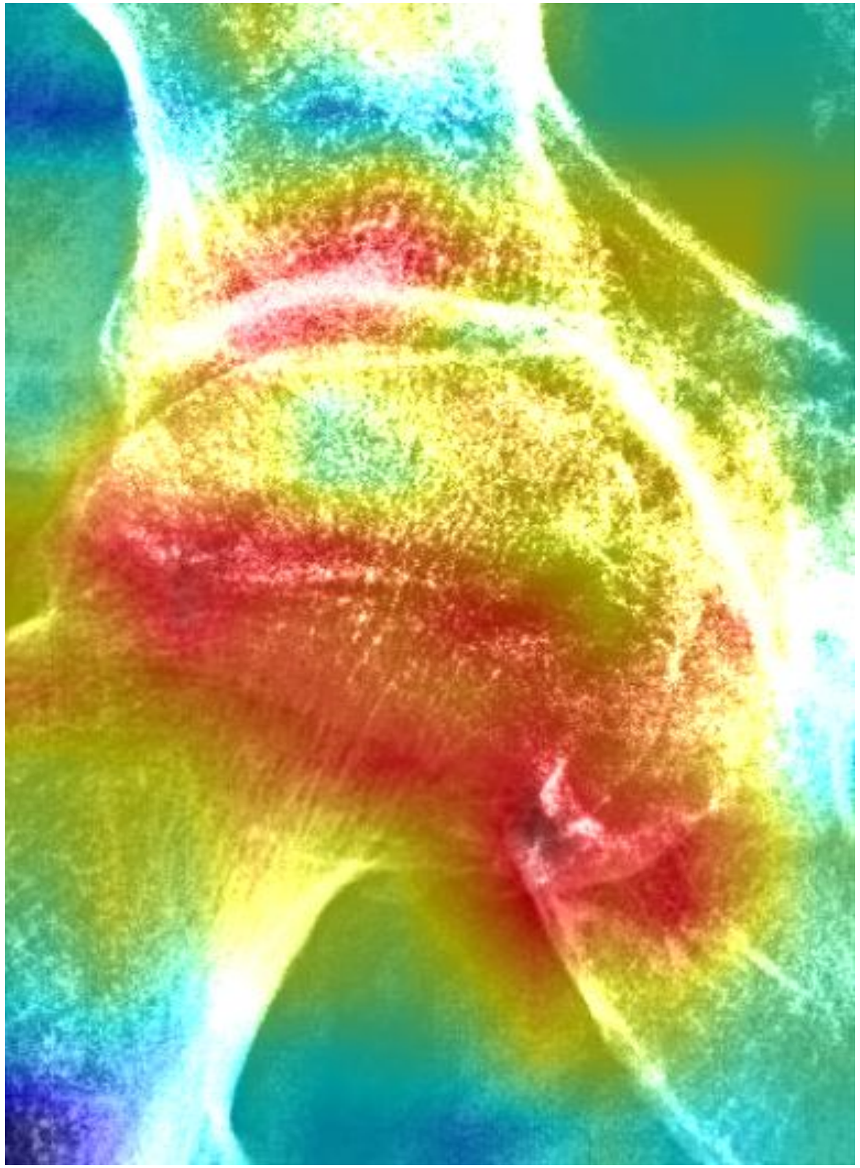
5 models:

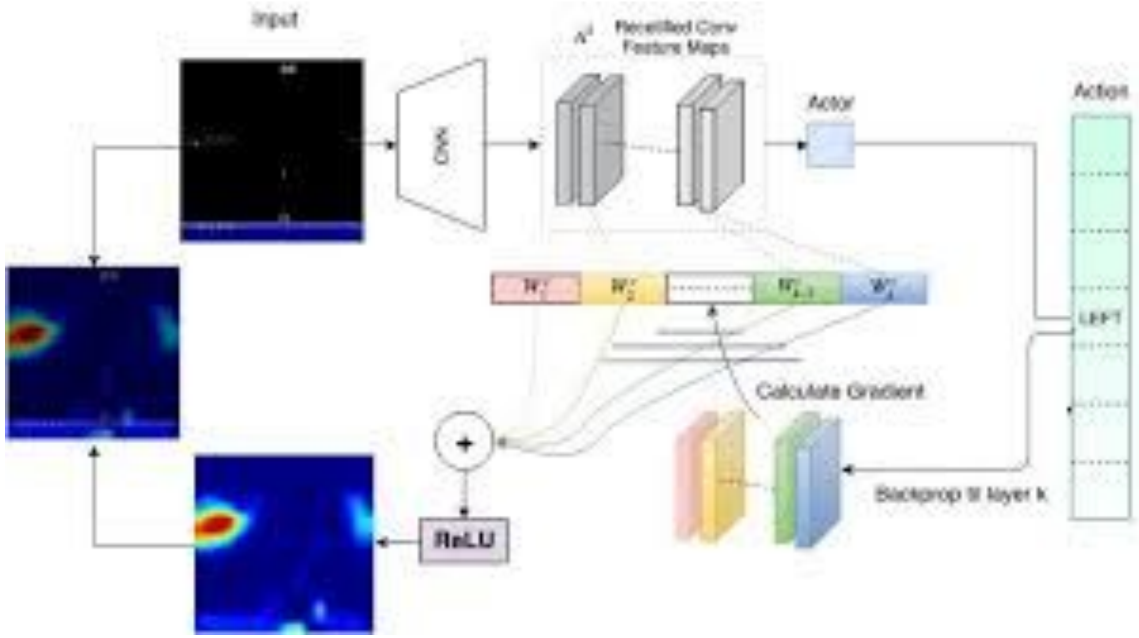
- **narrowing of the hip joint**
- **acetabulum sclerotization**
- **head sclerotization**
- **cysts**
- **osteophytes**

Fine-tuning using CheXNet (DenseNet121): https://stanfordmlgroup.github.io/projects/chexnet/

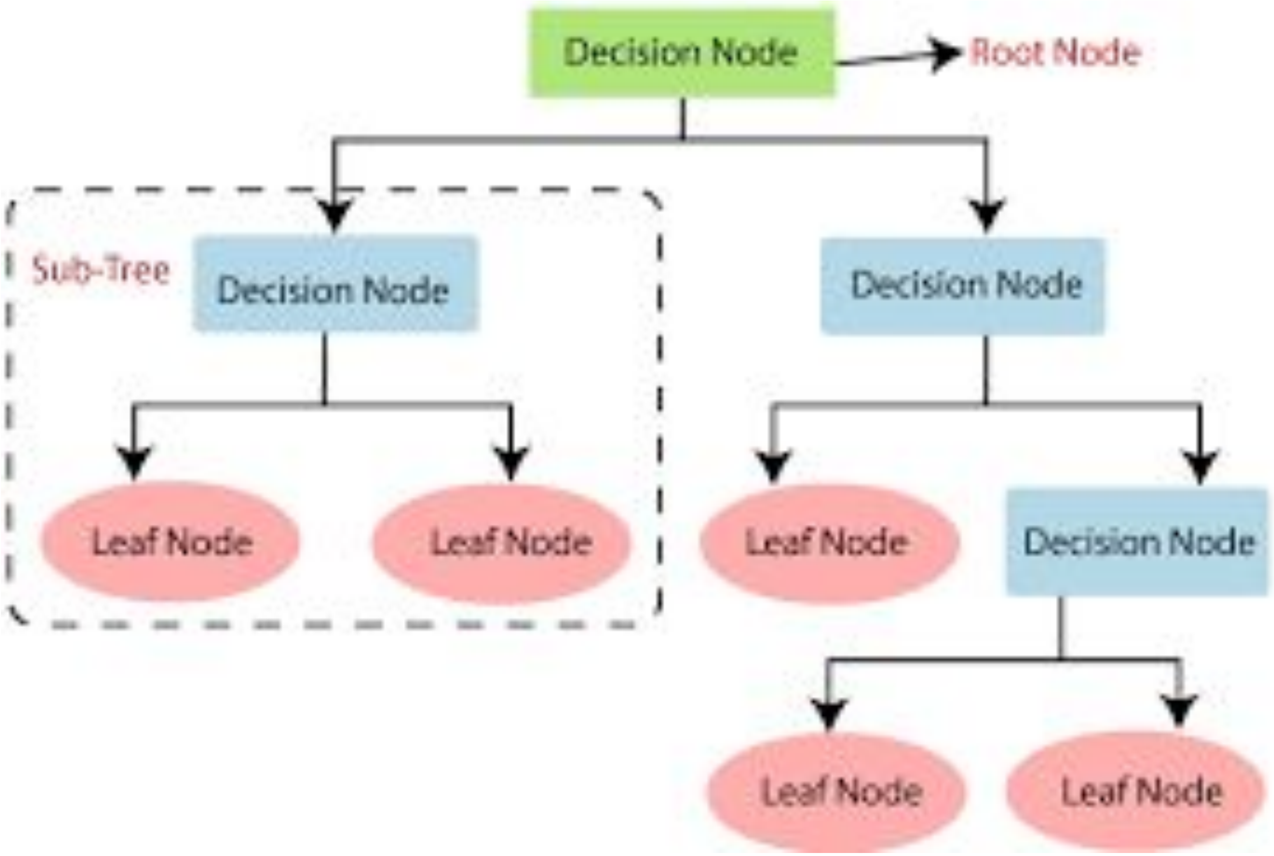


Step 2 - model explanations





Step 2 - patient condition



My second big deep learning project - platypus

"platypus" is an R package for object detection and image segmentation.

With "platypus" it is easy to create advanced computer vision models like YOLOv3 and U-Net in a few lines of code.

For more details and examples visit: https://github.com/maju116/platypus

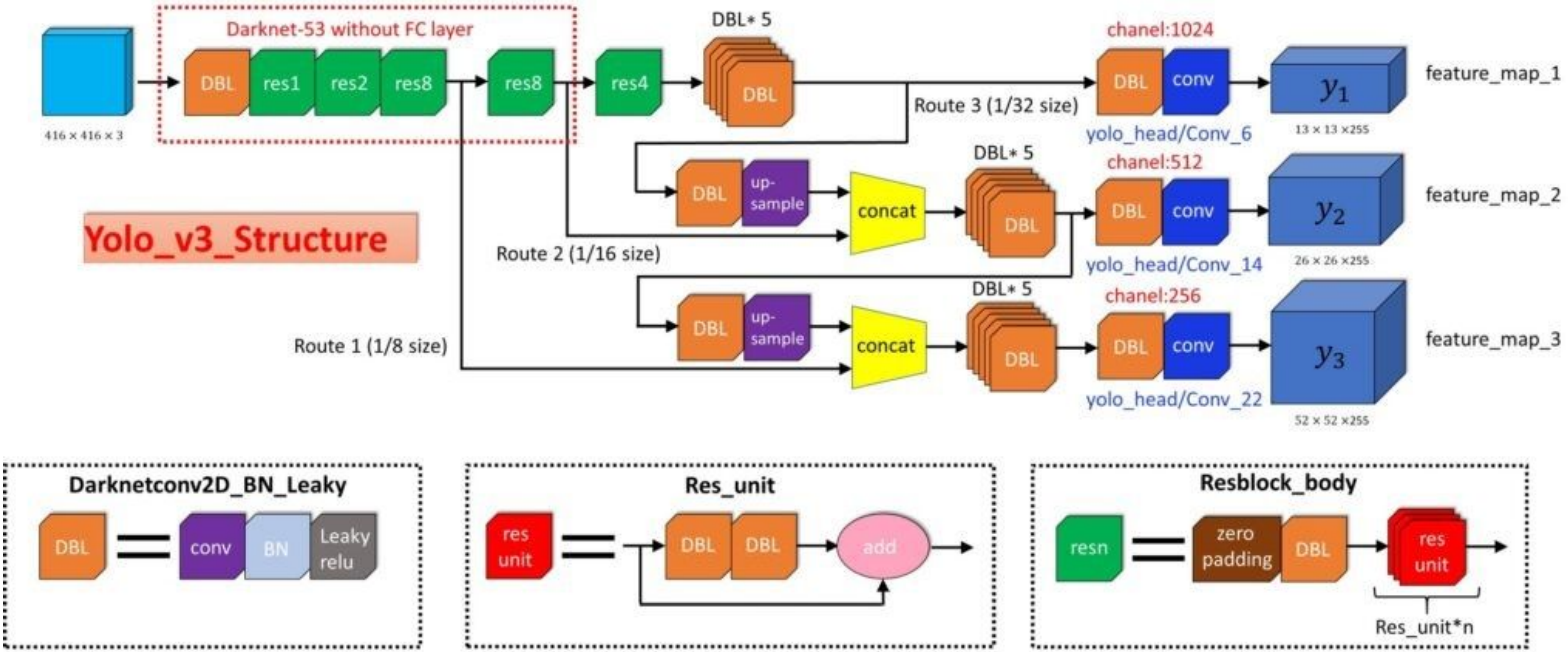
Object detection with YOLOv3

There are a few different algorithms for object detection and they can be split into two groups:

1. **Algorithms based on classification. They are implemented in two stages. First, they select regions of interest in an image. Second, they classify these regions using convolutional neural networks. This solution can be slow because we have to run predictions for every selected region. A widely known example of this type of algorithm is the Region-based convolutional neural network (RCNN) and its cousins Fast-RCNN, Faster-RCNN and the latest addition to the family: Mask-RCNN. Another example is RetinaNet.**

2. **Algorithms based on regression – instead of selecting interesting parts of an image, they predict classes and bounding boxes for the whole image in one run of the algorithm. The two best known examples from this group are the YOLO (You Only Look Once) family algorithms and SSD (Single Shot Multibox Detector). They are commonly used for real-time object detection as, in general, they trade a bit of accuracy for large improvements in speed.**

Object detection with YOLOv3



Object detection with YOLOv3

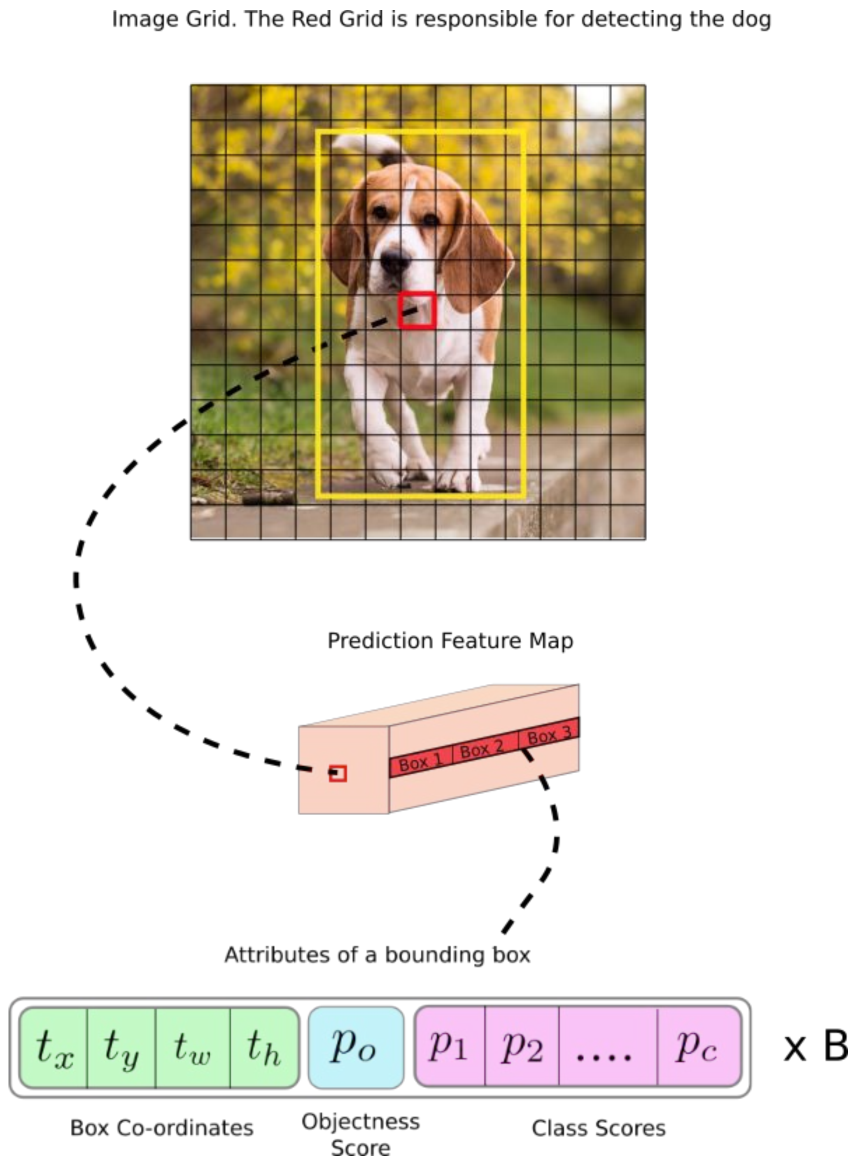


Image is divided into a grid. In YOLOv3 we have 3 different grids with different resolutions (small, medium and large objects).

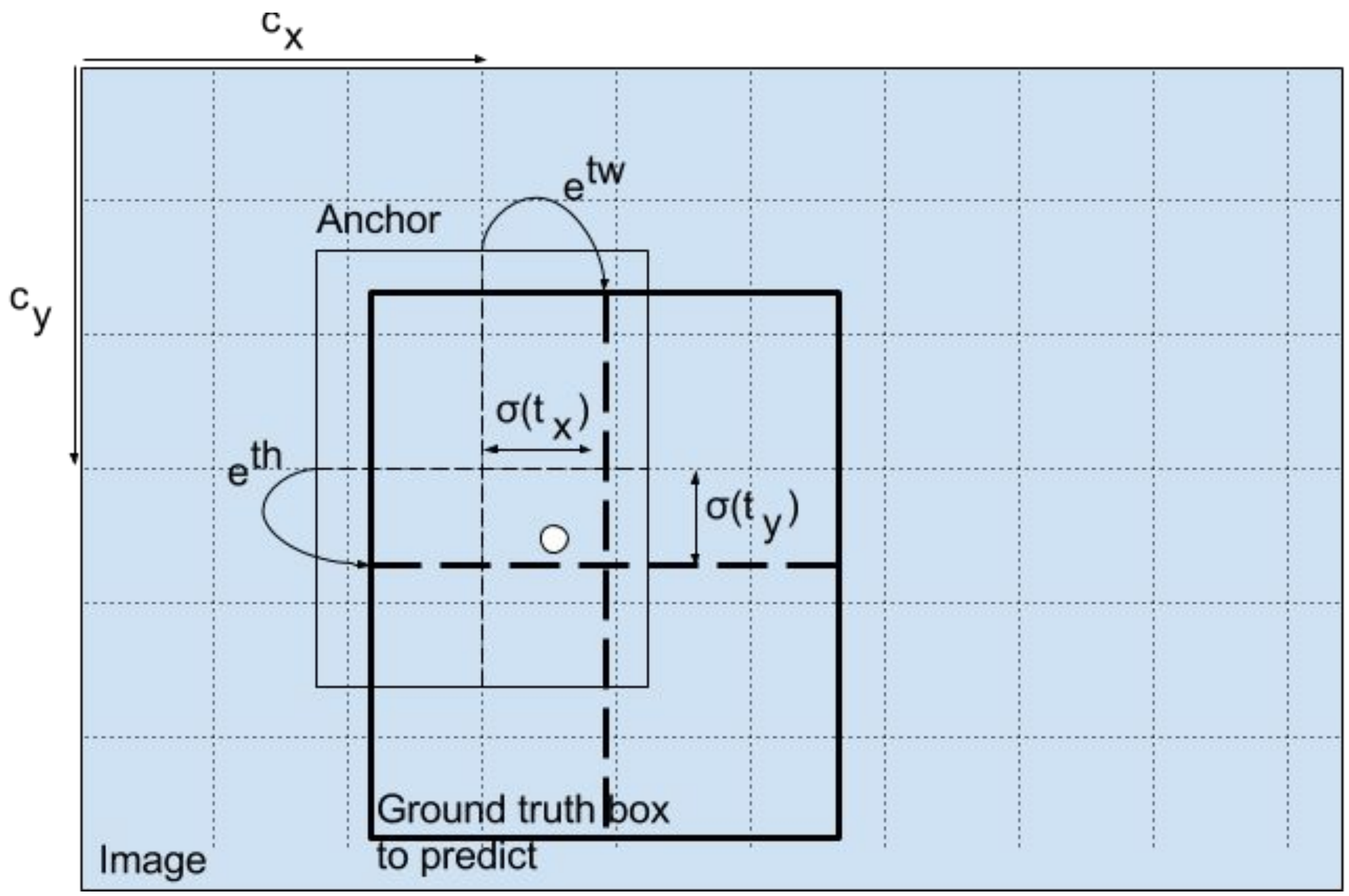
Each cell is responsible for detecting N boxes (objects) - 3 in this example.

For each box we are predicting 3 things:

- **probability that there is an object**
- **class probabilities**
- **box coordinates (how are they different from ANCHOR BOX)**

$$b_x = \sigma(t_x) + c_x$$
$$b_y = \sigma(t_y) + c_y$$
$$b_w = p_w e^{t_w}$$
$$b_h = p_h e^{t_h}$$

Object detection with YOLOv3



Object detection with YOLOv3

$$\lambda_{coord} \sum_{i=0}^{S^2} \sum_{j=0}^{B} 1_{i,j}^{obj} [(t_x - \hat{t}_x)^2 + (t_y - \hat{t}_y)^2 + (t_w - \hat{t}_w)^2 + (t_h - \hat{t}_h)^2]$$

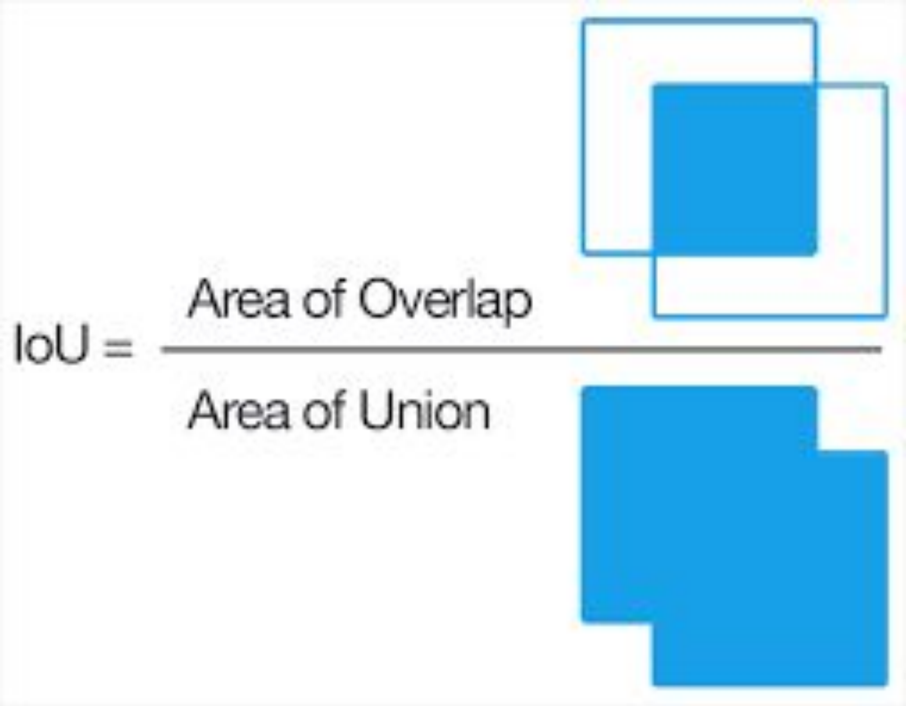
$$+ \sum_{i=0}^{S^2} \sum_{j=0}^{B} 1_{i,j}^{obj} [-log(\sigma(t_o)) + \sum_{k=1}^{C} BCE(\hat{y}_k, \sigma(s_k))]$$

$$+ \lambda_{noobj} \sum_{i=0}^{S^2} \sum_{j=0}^{B} 1_{i,j}^{noobj} [-log(1 - \sigma(t_o))]$$

Loss functions is made up from 4 partial losses:

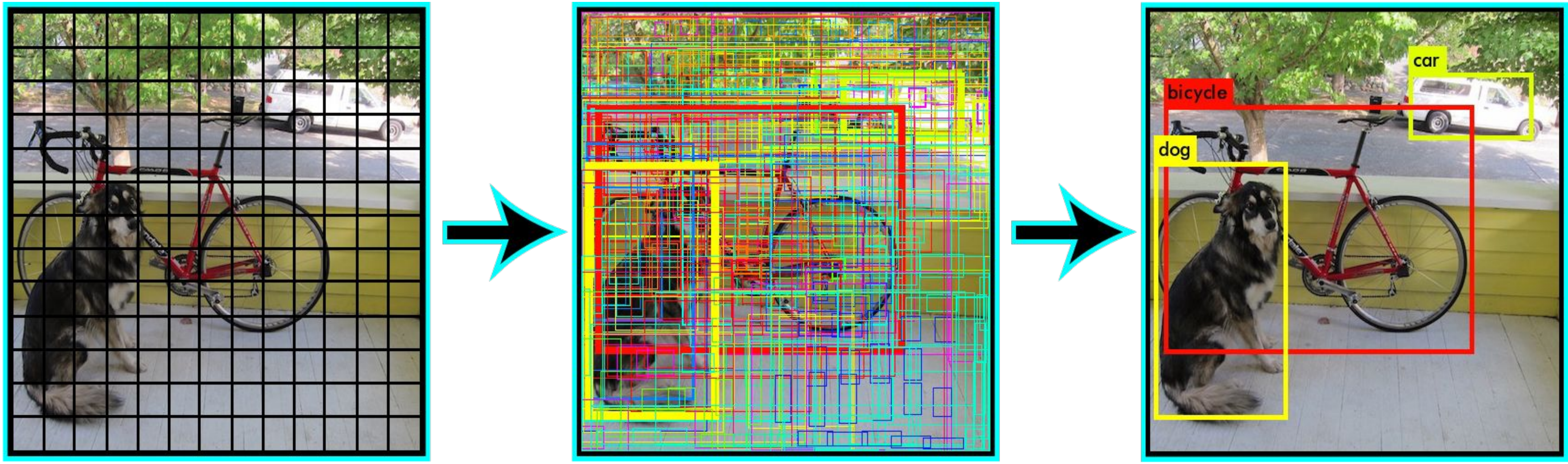
- **coordinates loss (MSE)**
- **objectness loss (BCE)**
- **non-objectness loss (BCE)**
- **class loss (sum of BCE)**

Object detection with YOLOv3



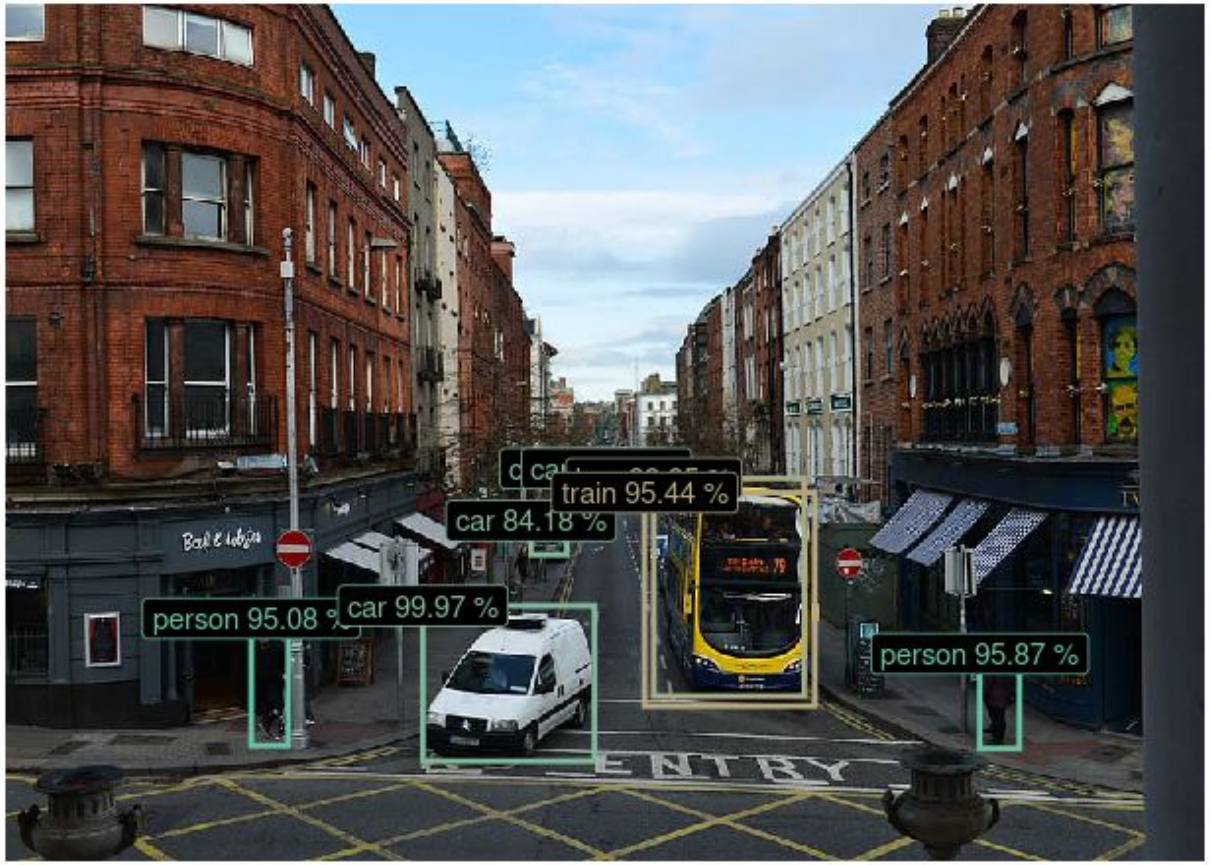
To check how good is our single prediction we can use Intersection over Union (IoU) metric.

Object detection with YOLOv3



In the final prediction we are showing only boxes with high objectness probability. We are also removing boxes with high IoU (for the same class)

Object detection with YOLOv3



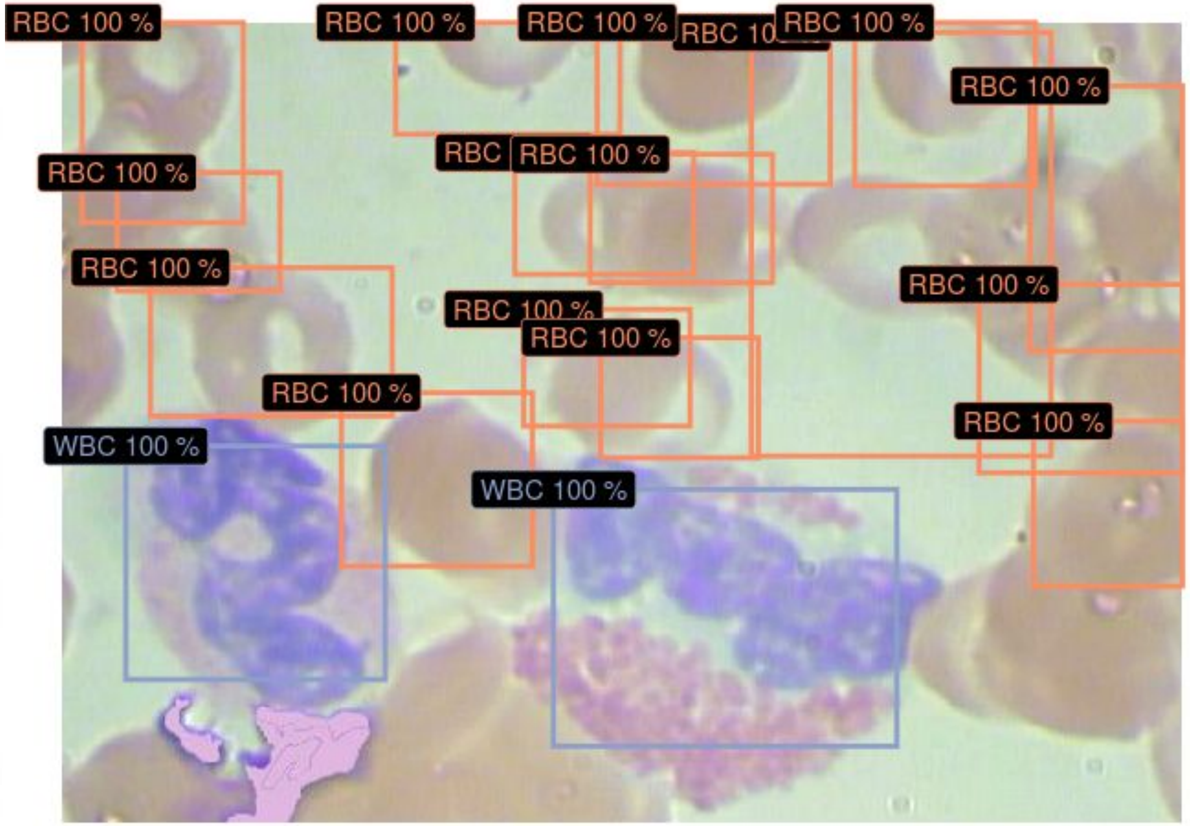


Image segmentation with U-Net

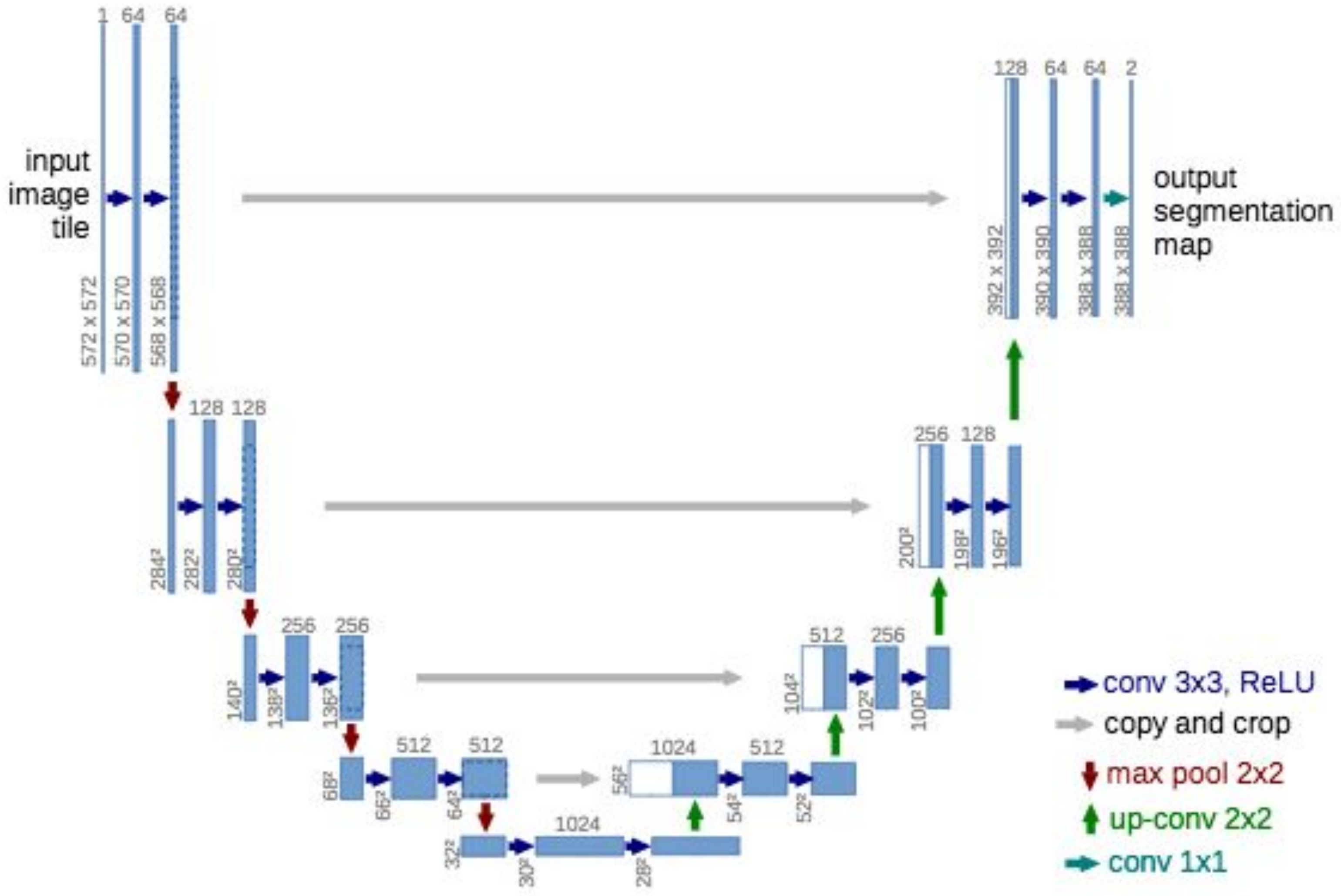
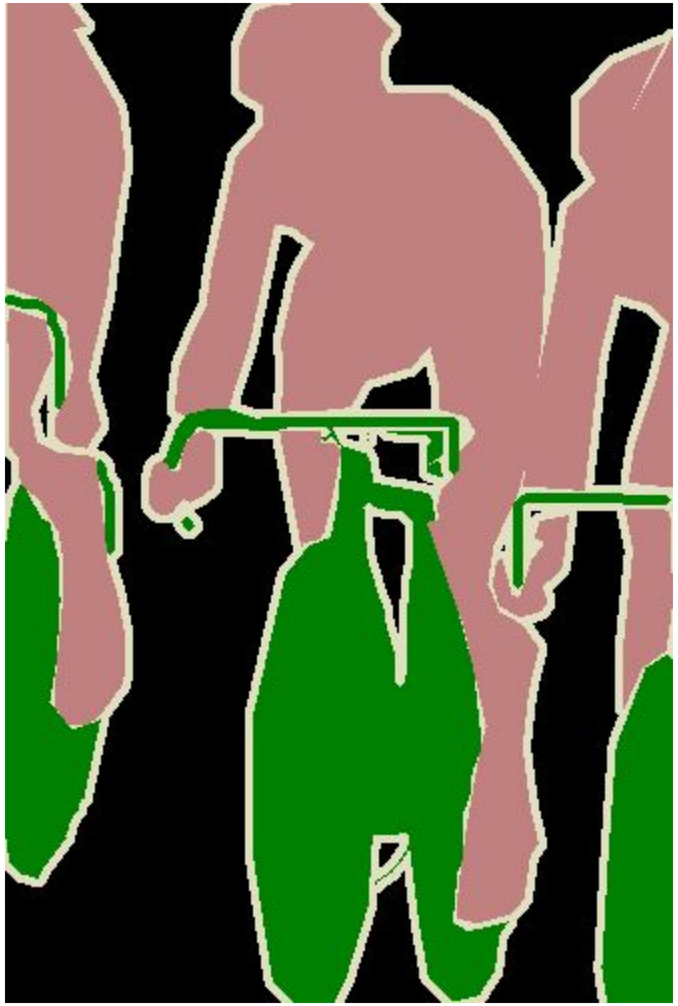


Image segmentation with U-Net



Questions ?