# 0. Pick some tools: OpenCV and Tensorflow ?

# 1. Pick some domain of expertise:

A. Data Gathering and processing

1. TODO: bounding boxes etc.
2. Dataset handling (manual method needed): <https://colab.research.google.com/github/tensorflow/docs/blob/master/site/en/tutorials/load_data/images.ipynb>
3. Code for (compressions, rotations, stretches and even color changes., etc). using OpenCV/Tensorflow. <https://www.tensorflow.org/tutorials/images/data_augmentation>

B. Augmentation/ Data Tricks and methods for enhancing results

C. Model productization

D. Classic methods evaluation. From quality to speed.

E. Deep methods evaluation: Faster R-CNN, Yolo, SSD. From quality to speed. | Later GANs

F. Evaluation of medical papers and the methods used – this must be done always for each of the A-E steps ! (possibly not C).

# 2. State of the art in computer science

A. Classic methods <https://lilianweng.github.io/lil-log/2017/10/29/object-recognition-for-dummies-part-1.html> and Open CV algorithms:

A) Check the image processing algorithms from here:

<https://docs.opencv.org/master/d7/da8/tutorial_table_of_content_imgproc.html>

especially

* **Smoothing Images**,
* Erosion and Dilating (probably we’ll need **Dilating**) **,**
* **Image Pyramids** for zooming small regions !
* **Histograms:** Why ? we can argue that those areas has some red inside while the others don’t . We are looking for this kind of patterns in the input features !
* **Anisotropic image segmentation** by a gradient structure tensor (check the image proof in the doc) or [**Image Segmentation with Distance Transform and Watershed Algorithm**](https://docs.opencv.org/master/d2/dbd/tutorial_distance_transform.html).

**Research question:** Could **Dilating** + this kind of segmentation mark better the ‘red’ zone ? What if we apply **Histograms** after ?

* Feature Matching algorithms from OpenCV (<https://docs.opencv.org/master/d5/d6f/tutorial_feature_flann_matcher.html> and all other, SIFT, SURF, etc).

The features should be different with any combination of above. Keep a dictionary of results versus methods !

B. Deep learning path: Parts 2-4 from above blog.

C. The probable model we’ll end up using

**Faster R-CNN:** <https://towardsdatascience.com/faster-r-cnn-object-detection-implemented-by-keras-for-custom-data-from-googles-open-images-125f62b9141a>

BUT in combination with many classic methods for preprocessing / augmentation etc from 2.1

D. During development we need to keep a dictionary of attempts and results similar to: <https://colab.research.google.com/github/tensorflow/docs/blob/master/site/en/tutorials/keras/overfit_and_underfit.ipynb#scrollTo=LqG3MXF5xSjR>

# 3. State of the art in medical field - TODO

Someone must take the papers (the two from Cristi and each one that we talked about in the project proposal) and write a document containing:

* All the image processing techniques
* All tricks that they used
* All methods used overall.

We’ll then try to reuse existing ideas in our work.

# 4. Tasks – Part 1, a reasonable model

1. Dataset processing and pipelines.

1.1 Clean the dataset and provide a csv file with (file\_path, bounding boxes with locations of the pattern we are interested in). Gold standard: <https://storage.googleapis.com/openimages/web/download.html>

Automatize from medical rectangles to bboxes coordinates.

1.2 Improve the bounding box using OpenCV – make the bounding box as small as possible, but not cutting all the features outside it.

1.3 Provide a programmatic API to download the dataset. E.g.: <https://colab.research.google.com/github/tensorflow/docs/blob/master/site/en/tutorials/images/classification.ipynb>

1.4 Write some demonstration code in a jupyter notebook that uses PIL and / or OpenCV to :

- Visualize images

- Draw bounding box locations with the interesting patterns.

2. Try the classic methods defined in 3.A

- Take one-two pictures from the dataset (until we have Task 1 in place working) and :

- As much as possible follow the methods in order

- Write in the document the results and show pictures, draw plots such that we can all evaluate them !

- Keep the dictionary of results and methods that I was talking about.

- **Note**: This can be done in parallel in Task 1 for prototyping, but to test the final results WE NEED Task 1 finished !

- Output: should have a method to identify and match the patterns between images !

3. Data augmentation methods

3.1 Try common techniques like <https://www.tensorflow.org/tutorials/images/data_augmentation>, and provide source code in jupyter notebook that prove the systems.

3.2. Implement the common tricks used in the medical science image vision and put them together like a library. Use as much open source code as you can !

4. Medical Science Image processing review.

- See Chapter 3 above

- Write down a document clearly explaining processing tricks, methods etc.

**Motivation for classical methods:**

* **We always need to compare the methods**
* **Our problem is somehow different from deep learning, We’ll need those methods for sure**