Use of neural networks in the diagnostic and prevention of endometriosis with the YOLO (You Only Look Once) algorithm

November 2021

Aurane Dibeu

Supervised by Professor Wei Zhou

Acknowledgment

One year ago, I joined the Ms Big Data and Business Analytics in the hope to improve my skills in data science by learning how to uncover business insights in mountains of data. I thought theses skills would help me land a job as a quantitative portfolio manager, little did I know at the time. After countless hackathons and hours building algorithms, I finally understood that those skills could be use in various disciplines other than finance.

Since I'm 16 years old, I unfortunately suffer from a chronic illness known as endometriosis. There is no cure, only an early diagnostic, hormonal treatments or surgeries can help reduce the level of pain. To finish my year at ESCP, I wanted to work on this subject. And I'm grateful that Professor Wei Zhou accepted and even offered to supervise the thesis.

I can now end my year with more hard skills on data science and more knowledge on the diagnostic and prevention of endometriosis.

Abstract

This paper reviews and test the possibility of using an object detection algorithm on a dataset composed of video recording of laparoscopies in order to detect lesions caused by endometriosis.

At the beginning, we focused on a review of literature in the field of endometriosis diagnosis from Antiquity to nowadays in order to investigate on the limitation of the diagnosis methodology and to highlight the consequences of it. Then in a second part, we focused on the data preprocessing of the GLENDA dataset and we tested two different models: Tensorflow 2 and YOLO to disclose whether or not computer vision could be of help to gynecology.

Results showed that even on a small dataset the training was successful yet the speed was an issue. For the moment, these models cannot be used in real time during a surgery but only after to assist the medical team in the treatment with images labeled for endometriosis.

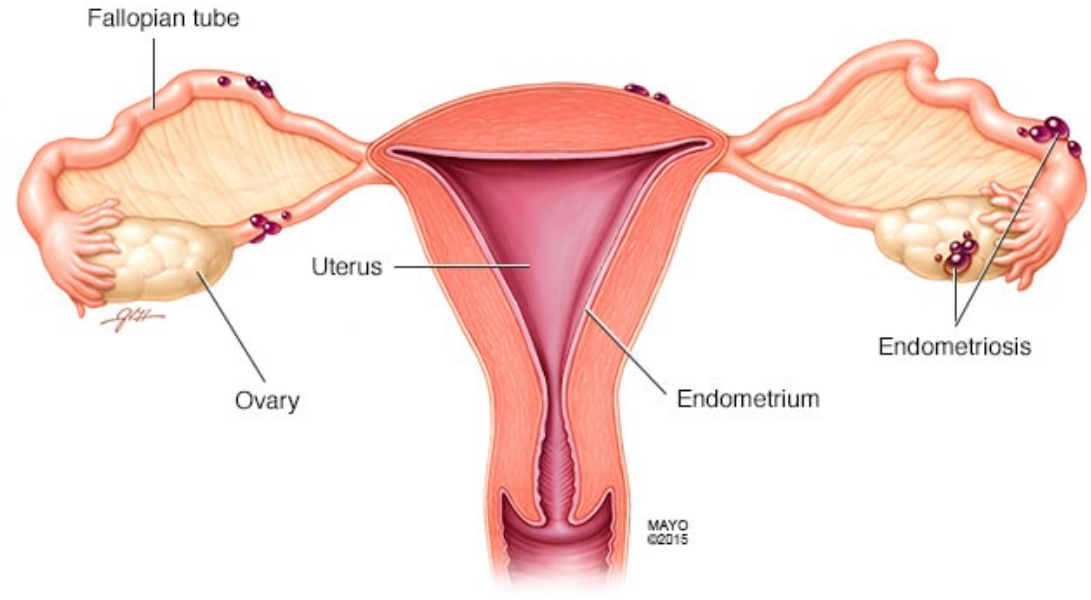
Keywords: ***endometriosis, yolo, object recognition algorithm, computer vision, neural networks, tensorflow, big data***

List of tables

Introduction

In 2017, researchers found out that having endometriosis led to a higher risk of having an ovarian cancer by 1,7 times [1]. It seems to be linked with the fact that endometriosis takes usually years to be diagnosed. The normalisation of pain, the bias towards thinking that it is normal to endure pain during menstruations and the 'low recognition of endometriosis at the general practitioner level' [2] explain the slowness of diagnostic.

Endometriosis is a gynecological condition that touches around 176 millions people. It can happen at any age but it usually happens to women between puberty (around 15 years old) and menopause. It is a long term condition, indeed it is related to menstruations and therefore has significant implications in the health and life of women enduring this disease. Menstruation is the name behind the monthly bleeding affecting women. The body of a women is always preparing for pregnancy. Hormones will develop the eggs in the ovaries and also make the lining of the uterus thick to hold a baby and provide nutrients. However, if you aren't pregnant the lining falls down and becomes period to exit the body.



For patients suffering from endometriosis, the lining will start growing on other places. On the picture above we can see that part of the endometrium went on the ovary and the fallopian tubes but it can also go to any other organs such as lungs or the heart. The migration of the endometrium cells won't affect their functioning, when hormones will drop because the uterus isn't holding a child, these cells will start to bleed. Unfortunately the endometriosis won't leave the body as periods but will damage the organs by irritating it which will lead to inflammation, scars and pain.

The symptoms vary, some women won't even notice and other will have pain in the lower abdomen that will worsen during periods, some will have pain during and/or after sexual intercourse, going to the toilets. Besides, as the endometrium cells are outside their usual place, they will damage organs and lead to more difficulty in getting pregnant due to all the inflammation and scar tissues. It can also unfortunately lead to infertility as it can damage the fallopian tubes. The origin of endometriosis is unknown. One of the oldest theory is the Sampson's theory, the migration of endometrial cells would be caused by a reversed period flow. Instead of exiting the body, part of of the menstruation's blood would go back and exit the uterus. In this blood would be the endometrial cells that would now bleed on other organs at the drop of hormones happening every month. It might also be related to genetics as having a close member of the family suffering from endometriosis increases the risk of having endometriosis.

The diagnostic can be done via ultrasound and laparoscopy. Nonetheless most of the women aren't heard and the struggle to get a diagnostic dwells on the difficulty to be taken seriously. Lena Dunham explained "Many will be dismissed as having mere menstrual pain or, worse yet, some modern version of hysteria" and Lakshmi said, "If I had been diagnosed at 16 or 26 or even 32, I would have gained valuable time. I would have been able to be more present for my family and friends … [had] a greater capacity to advance professionally, and I would have also had a greater capacity for

intimacy."[3]. Indeed, having endometriosis will lead to an average loss of 10 hours of work per week [4], a significant loss that affects women desire in career, family in addition to the chronic pain.

Treatment of symptoms consists in decreasing the flow of period which will avoid having the migrated cells to bleed and cause damage to the organs. This treatment is usually composed of hormones that prevents menstruations. Another possibility is a surgical procedure where we will remove the endometriosis and the cysts. However, removing these cells doesn't mean that the patient will never suffer endometriosis again since recurrence is prevalent. The only eventual definitive treatment is a hysterectomy which is the complete removal of the womb and the ovaries.

Diagnosis is a crucial step in the treatment of endometriosis because depending of the information, the medical team will select a treatment. Indeed depending on the type of endometriosis and the severity of it, the treatment will differ.
Ultrasound is an helpful tool for identifying ovarian cysts nevertheless it can usually miss the other endometriosis sites. That is why, laparoscopy is the gold standard in the diagnosis of endometriosis. Yet, during the surgery the detection of endometriosis is very difficult. On the image below we can see how complex it is to distinguish endometriosis characterized by the red lesions from the healthy tissue.

Peritoneal endometriosis with unclear extent and spread: red lesions with typical vascular patterns next to fibrotic peritoneal defects.

That is where computer vision and object detection can help by automating the detection of endometriosis via the recording of the laparoscopy. Computer vision designates a sub-field of artificial intelligence. From images or video, models will discover meaningful insights. In another way, artificial intelligence can be understood as giving a brain to a machine and computer vision as giving eyes to a machine. In a fraction of the time that a human being needs, computer vision and artificial intelligence can processes information faster and sometimes even better that its human counterparts.

How can object detection algorithms help detect endometriosis during a laparoscopy?

In a first part, we will review the current diagnosis methodology for endometriosis to highlight the challenges that computer vision can help solve. In a second part, we will use a model on a laparoscopy dataset to test if we can help increase the accuracy rate in the detection of endometriosis thanks to algorithms.

I. State of art

1.1 History of endometriosis diagnosis

In ‘endometriosis: ancient disease, ancient treatments’ [5], we learn that allusions to endometriosis goes back at least to Antiquity. Indeed, a Hipprocraptic chapter titled ‘Diseases of Young girls’ in relates a disease related to adhesions from the endometrium. The treatments were various and highly inefficient: suppositories made of urine or castor oil for instance.It wasn't until a century later that the idea that this pain could be caused by some sort of inflammation in the uterus would see the light with Soranus. The cure during Antiquity was pregnancy, indeed when a woman is pregnant she doesn't have period and therefore cannot suffer from endometriosis. However the link between menstruations and endometriosis (or what was called at the suffocation of the womb) wasn’t established until later.

In the Middle Ages, the diagnosis of endometriosis wasn't facilitated by the context. The Dark Ages were plagued by a doubt around science and the comeback in Western imagination of the supernatural. Nonetheless, a significant breakthrough was made at the time: the discovery of the link between period and suffocation of the womb by Aetius of Amida. Most of the advancements in the diagnosis of endometriosis were made via autopsies. The diagnosis on living human beings was particularly difficult since until the 18th and 19th centuries, suffering from the suffocation of the womb wasn't perceived as suffering from a disease such a common cold but could be seen as being possessed by a demon or a sign of being a witch. As there were no treatments, women suffered an unimaginable pain. It was so unbearable that a lot of the time, the testimonies of women enduring this disease are relating women fainting.

FIGURE 20

Engraving from 1642 representing folkloric conceptions of women's illness, including hysteria and other disorders that may have been endometriosis or other gynecologic conditions. (Courtesy of the U.S. National Library of Medicine. Engraving by Hendrick Hondius, titled "Pilgrimage of the Epileptics to the Church at Molenbeek, 1642.")

Nezhat. Endometriosis in history. Fertil Steril 2012.

The idea of women being possessed became less and less believed however endometriosis was still seen as psychological issue: hysteria. Diagnosis was also only done via autopsies. It wasn't until the 19th century, that a group known as the French pioneers were able to find evidence of endometriosis being a specific disease and not a symptom of another medical condition. They did so by making autopsies but also by analyzing via a microscope their findings. Surgery was quite painful for women, doctors used scissors or their bare hands to remove the cysts from their uterus. Treatments were still archaic and particularly cruel (curetage, twits and tear off method, excision).

The use of laparoscopy was mainly due to the research of Raoul Palmer in the 20th century.

1.2. Laparoscopy as the standard in endometriosis diagnosis

In 1971, Hasson introduced the open laparoscopy. The idea was to see more clearly the lesions caused by endometriosis to establish a diagnostic. The vision was done by the surgeon's eyes. It means that during a surgery a surgeon would open the lower abdomen of a women and try to see with his/her eyes the lesions. The arrival of video during these surgeries only happened at the end of the 1970s. Camran Nezhat is the one that brought video into the operating room. It was a considerable breakthrough, as it was a less invasive procedure than an open surgery, and it led to an increase in accuracy. Before a surgeon could only trust his/her eyes and now with the camera they could record the surgery, see the images after the surgery and most importantly zoom on some parts of the image. Moreover as years passed, the cameras improved and became smaller which decreased the necessary cuts during a laparoscopy and decreased the risks (infection, death).

1.3 Limitation

When women undergo a laparoscopic exploration, it has usually been a decade since they have been waiting for a diagnosis. The difficulty doesn't only reside in the struggle to get heard and redirected towards a skilled practician. It resides in the difficulty itself to detect endometriosis. As we have seen in the introduction, endometriosis can be red but it can also be black, yellow or bluish with a large number of nuances. And it can also be non pigmented at all and therefore extremely difficult to distinguish between healthy tissue, even with a camera. Laparoscopy has challenges that can be resolved by breakthrough in imaging technologies [6] and perhaps in the introduction of computer vision.

II. Model Testing

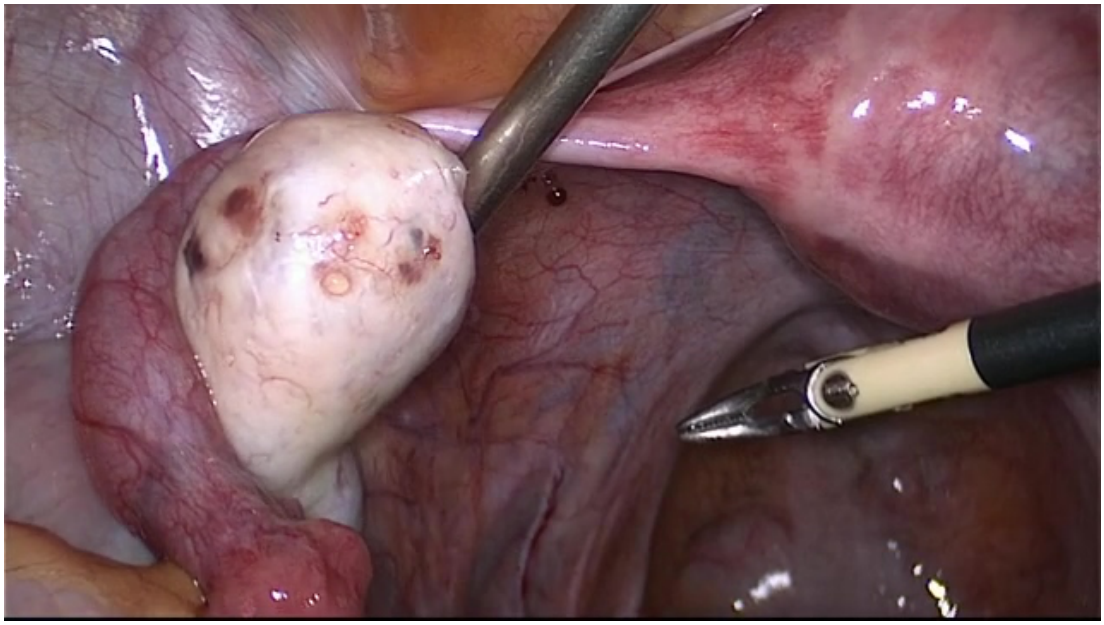
2.1 Presentation of the dataset: GLENDA

The subject of this thesis is to test if computer vision and object recognition algorithm can help increase the detection of endometriosis during laparoscopic exploration. In order to do so, I selected the Gynecologic Laparoscopy ENdometriosis DAtatset [7]. It is a dataset composed of over 350 images labelled for endometriosis lesions. The sources are more than 100 videos of laparoscopies. The authors of this dataset wanted to respond to the challenges of laparoscopy by providing data science researchers with an extensive dataset composed of both annotated endometriosis lesions and more than 13000 unlabelled images. The labels have been done by a medical team that agreed to characterize as endometriosis everything that corresponded as endometriosis in the classifications rASRM [8] and Enzian [9] systems. A shared definition formalizes the process. For this paper, I worked on v1.5.

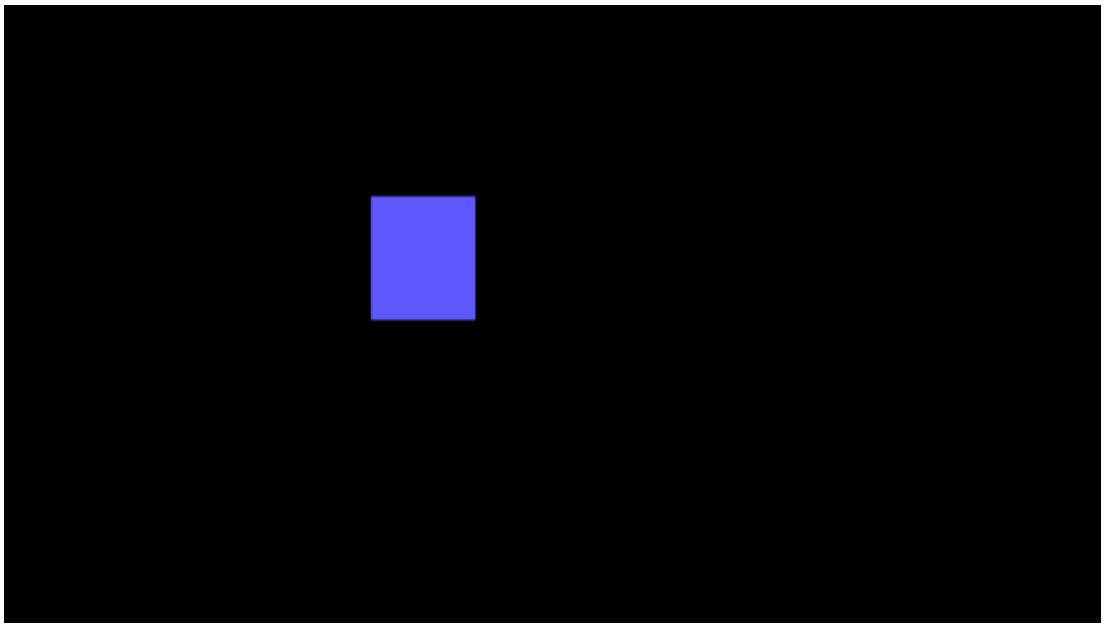
The dataset was composed of five different classes:

- peritoneum which is a membrane lining the abdomen and also covers organs;
- ovary which is a reproductive organ producing ovules;
- uterus which is an organ located in the pelvis;
- deep infiltrating endometriosis which is an invasive and aggressive form of endometriosis located at least 5 mm under the peritoneum.

The pictures were 640×360 and upon downloading the dataset I had two files one with the surgery pictures made from the video recordings and the frames.



Example of a picture (c_1_v_(video_7.mp4)_f_36)



Example of the related annotation (c_1_v_(video_7.mp4)_f_36)

As you can see the picture and frame are two separate files. Therefore, I had to merge both the picture and the annotation in order to train my model later. To do so I wrote the Python script below:

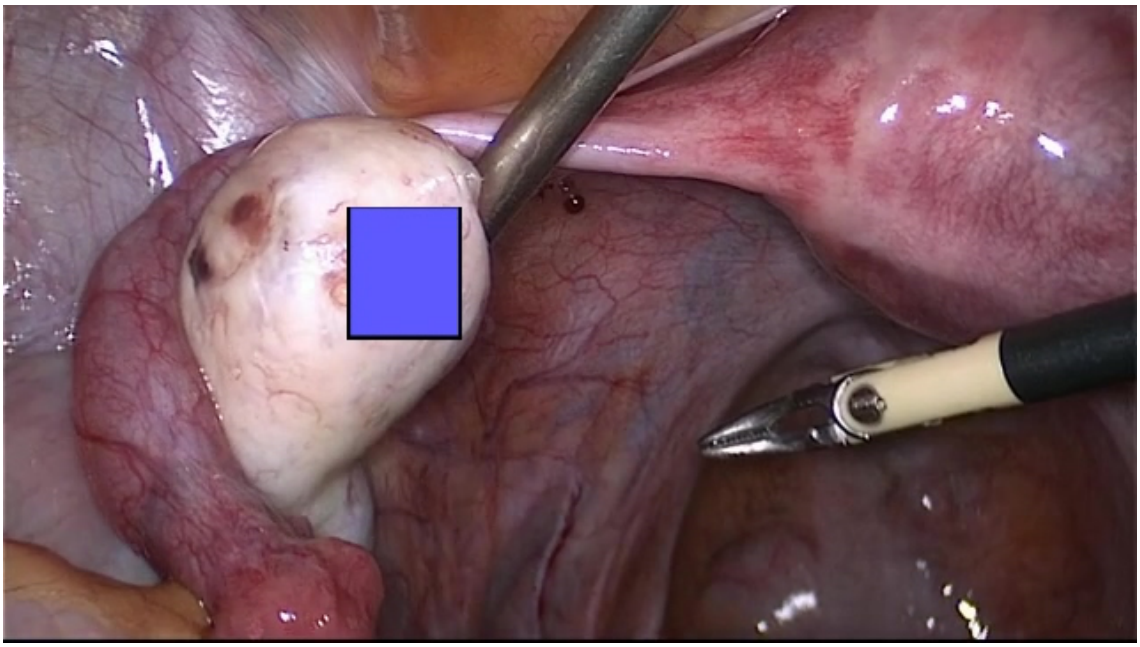
Entrée [7]:
```
try:
    from PIL import Image
except ImportError:
    import Image

background = Image.open("c_1_v_(video_7.mp4)_f_36.png")
overlay = Image.open("c_1_v_(video_7.mp4)_f_363.jpg")

background = background.convert("RGBA")
overlay = overlay.convert("RGBA")

new_img = Image.blend(background, overlay, 0.5)
new_img.save("new.png","PNG")
```

It produced the results below:



It wasn't ideal at all because the annotation actually covered the endometriosis lesion. It posed the issue of how can I build a model to detect endometriosis if the lesion is covered by a colorful square.

The strategy I chose is to label for a second time the images. I run a loop (python script) to merge the picture and the annotation together and then I labelled the original picture via the python library ‘labelimg’. It led to the same annotation as below but with a clear square that didn't prevented the endometriosis lesion.

Due to computing limitation, I decided to work with two classes: peritoneum and ovary. I selected 10 images for each classes. The elimination of the remaining classes (uterus and deep infiltrating endometriosis) were arbitrary. 20 images seemed like a good compromise since models usually required more than 24 hours of training for only 20 images and pass a certain number of hours, Google Colab shut down the notebook.

2.2 A model with Tenserflow 2

Tensorflow is a machine learning platform full of ressources of various kinds: tools, libraries, documentation, models etc… It has been built by Googlers in 2015 to help data enthusiasts leverage their knowledge with all an extensive library of tools in machine learning and deep neural network. It can be used by various languages such as Python or Java but in this thesis I have only used Python. Tensorflow 2 is just an upgraded version of Tensorflow 1 with variables and checkpoints.

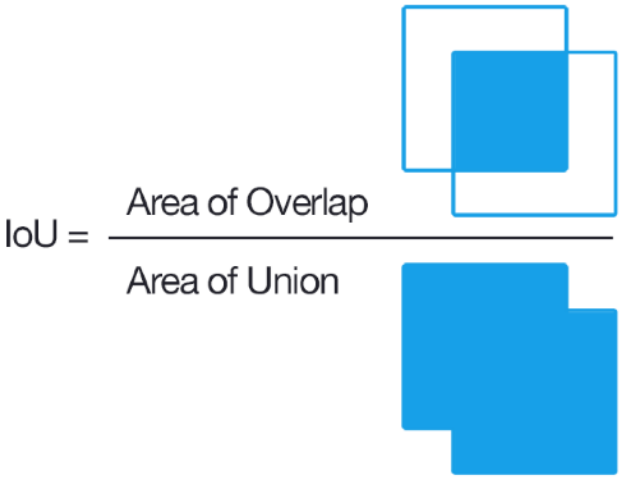
$$\mathrm{MAP} = \frac{\sum_{q=1}^{Q} \mathrm{AveP(q)}}{Q}$$

Mean average precision formula given provided by Wikipedia

I used the Tensorflow object detection API. First, API stands for Application Programing Interface. It allows communication between computer or programs via a

set of functions. It helps extract information that our model will need. In this case, it helped me build and train my single object detection model. More than a dozen of models are available on the Tensorflow object detection API, the best strategy to select the adequate model is to evaluate what is more important between speed and the mAP. Speed is easily understandable however mAP isn't.

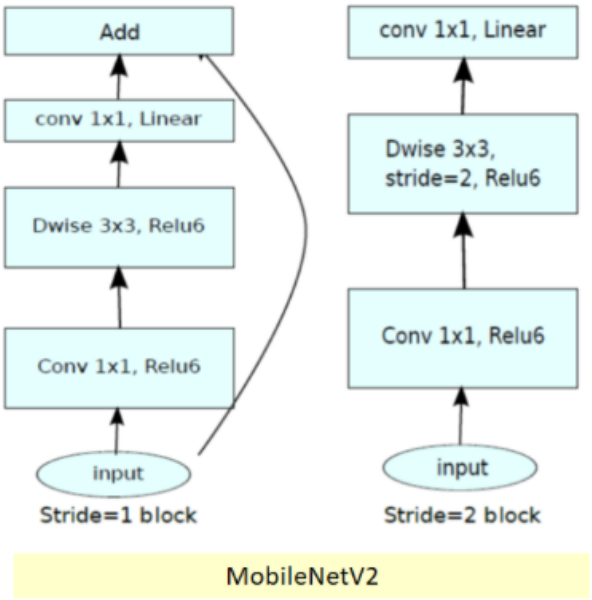
The mAP helps us understand how well our model is doing what we are asking it to do. As we are working with object detection model, we will have boxes of detected objects. To quantify how well a model is doing we can use the intersection over



union concept (IoU). It designates the intersection between the ground-truth bounding box to the detected box. The objective is to have the higher score as it can interpreted as a more accurate model.

Nonetheless, we also have to keep in mind the limitation of the computing process. Some models have a map of 30 but they take 89 ms to run and I am only coding on a Macbook Pro. The tradeoff was having both a fast model and an accurate model that is why I chose the SSD Mobile Net V2 FPN Lite which is balanced between a speed of 22 s and a mAP of 22.2. These two metrics have been extracted from an object detection named COCO. It is a dataset made public thanks to Microsoft to help data scientists enthusiasts and machine learning engineers create computer vision projects. COCO stands for Common Object in Context. There are over 200 000

images representing over 250 000 people for instance. This dataset is massive and



therefore the metrics extracted from this dataset can be generalized.

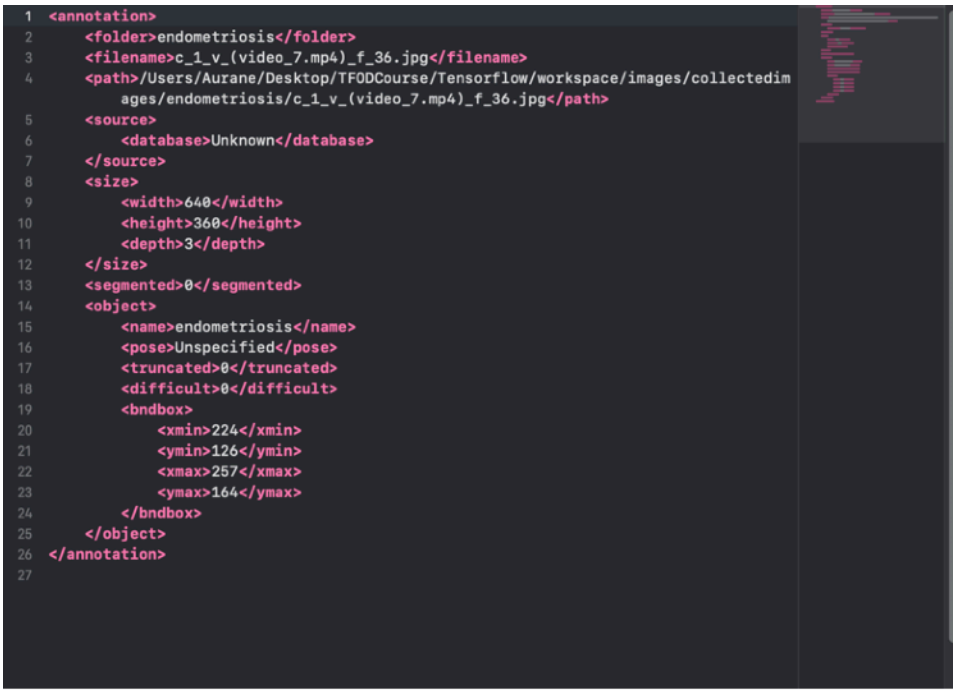
In the SSD Mobile Net V2 FPN Lite model contrary to mobile net v1, we have a bottleneck residual block. This method decreases the number of parameters in order to make the residual block thinner.

The architecture of the model is based on the ResNet which is an answer to solve the gradient issue where going deeper and deeper will lead to a decrease in performance. The architecture's idea is to avoid going deeper by jumping or skipping layers which is supposed to avoid decreasing the performance.

Besides, this model has three convolution layers. A convolution designates the process of intertwining and changing an object into something new. For instance in object detection algorithms, this process will do the preprocessing. It will use data augmentation which can be making an image darker or clearer, blurring them, cropping them, zooming in and/or out. And in the case of this model it will also do the post processing which will bring back the images at their initial step. To

summarize it helps identify the best strategies in data preprocessing to prepare the data in the best way possible before training the model.

After labelling my images, I installed the model in a Python script and incorporated the data. The hardest task with the model was the installation. Firstly, I had to create a virtual environment to avoid conflicts in my libraries. Creating a virtual environment is like creating a new room in a house. It is still in the system but it is hermetic to the rest and allows you to import all the libraries you need without worrying on whether or not it will have a conflict with a library you previously

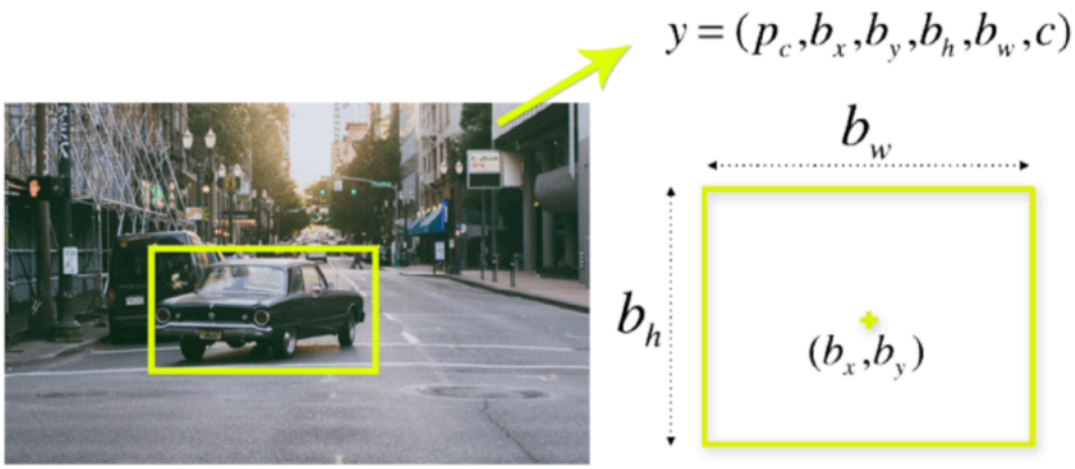


```
<annotation>
    <folder>endometriosis</folder>
    <filename>c_1_v_(video_7.mp4)_f_36.jpg</filename>
    <path>/Users/Aurane/Desktop/TFODCourse/Tensorflow/workspace/images/collectedim
        ages/endometriosis/c_1_v_(video_7.mp4)_f_36.jpg</path>
    <source>
        <database>Unknown</database>
    </source>
    <size>
        <width>640</width>
        <height>360</height>
        <depth>3</depth>
    </size>
    <segmented>0</segmented>
    <object>
        <name>endometriosis</name>
        <pose>Unspecified</pose>
        <truncated>0</truncated>
        <difficult>0</difficult>
        <bndbox>
            <xmin>224</xmin>
            <ymin>126</ymin>
            <xmax>257</xmax>
            <ymax>164</ymax>
        </bndbox>
    </object>
</annotation>
```

installed. Secondly, I had to create a label map. It helps the algorithm identify the class name. For instance, please find below what the annotation looks like on the picture above. The label map links the label with integer values and is a required step for Tensorflow Object Detection API. You can see below my label map:

2. Create Label Map

```
Entrée [51]: labels = [{'name':'endometriosis', 'id':1}]
```



Thirdly, I had to create TF records. It is a specific file that uses binary format. In order to reduce the training time, I converted my images file in TF files.

The last steps were setting up the model, I used the default configuration and launched the training on a python script via anaconda. After two days of training, the model still hadn't completed its training and my computer was highly heated. Therefore I had to stop the training and change the model as this was one taking too long to run and couldn't be stopped.

2.3 A model with You Only Look Once

One of the common challenge regarding object detection algorithm is the speed necessary to detect objects in real time. As the objective of this thesis is to work on a model that could be used during a surgical intervention and indicates the endometriosis lesion to the medical team, speed in the processing of images is of the utmost importance. The YOLO algorithm is one of the best model in regard to that condition.

As for the the SSD Mobile Net V2 FPN Lite, we are working with boxes. It means that we won't have to use another method to label the laparoscopy images. The original part in the YOLO algorithm is that, as its names says, it will look at the image as a whole once. Throughout the neural networks it will predict the notorious boxes and the class probabilities in one pass. To summarize, the objective is to identify but also locates endometriosis lesions.

Model works as follows:

- Each images will be divided into smaller squares and YOLO can predict no more than 5 objects per small squares.
- Pc stands for the probability of having an object in the box
- Bx and By stands for the center of the bounding box
- Bw is width
- Bh is the height
- The c stands for the class which in our case is endometriosis

In case the pc is low, the boxes will be removed.

The YOLO model I have used came from the darknet which is an open source neural network framework. Similar to Tensorflow it has a lot of tools, various architectures and models for computer vision. Neural networks are a suite of algorithms that mimic the way human brain process information. Its functioning is throughout thousands and sometimes even million of interconnected neurons. For instance, in computer vision, a model will never be able to be trained on every possible examples of a class. Therefore the neural networks will adapt at different scenarios of a class by identifying a pattern. A dog has a certain size, paws and a tail. The neural networks can then apply this pattern to new data.

For this model the installation was pretty straightforward, I opened a Google Colab notebook and chose a GPU configuration to avoid the limitation of my own machine.

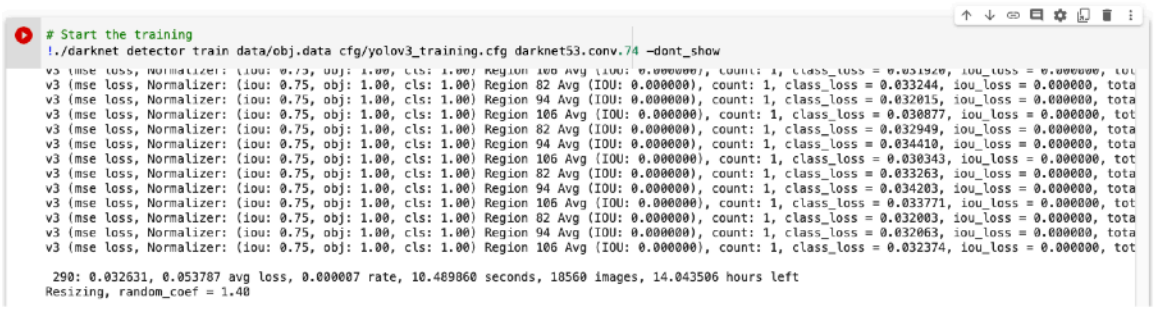
Then I compiled darknet by using the GPU. Darknet was written in C++ and I wanted to work with Python. Compiling is a process that translates a code into another language.

For YOLO, I didn't needed to create a label map, designing the class by endometriosis was enough.

Regarding the pre-processing of images, the annotation from the library labelimg were supported by the model. Therefore I did not have to use another methodology for labelling the GLENDA dataset. The only step required was to zip both the images and the annotations file to be read by YOLO.

The images and annotations were then converted to numbers and coordinates to help the model transform the image in a series of little squares.

Finally, I launched the training of the model. The noticeable difference with the SSD Mobile Net V2 FPN Lite was the indication of the required training time. For 20 images of endometriosis and a GPU of 11441MiB, the training time indicated was 24 hours. It was still less than the tensorflow model, however since I was on Colab and not Colab pro, the training stopped multiple times around five hours. The last training went until the 209th iterations with an average loss of 0,053787.



2.4 Summary

The first model with Tensorflow was less efficient than the YOLO algorithm in training time. Unfortunately we cannot compare the two models on another metrics. For the YOLO algorithm, being under an average loss of 0,6 is a success as it is the common threshold. Furthermore, the region average IoU is 0 which means that model made 0 overlaps. It is expected as the dataset is particularly small, the re is a tendency to overfit.

```
Region 82 Avg (IOU: 0.000000), count: 1, class_loss = 0.032310, iou_loss = 0.000000, tota
```

2.5 Discussion

The result of the YOLO algorithm seems promising despite the small size of the dataset we still achieved a good average loss. Increasing the computing power will allow us to increase the size of the dataset. Nevertheless one constraint cannot be avoided. The model takes a few hours before being relevant and the objective is to use this model during a laparoscopy to help a surgeon detect and remove (if needed) endometriosis lesions. For the moment, the model isn't fast enough for it. However in a healthcare context, we cannot trade accuracy for speed. The model will need to have both.

As next steps, it could be interesting to test the faster model with the dataset and see if we actually endure a significant loss of accuracy.

Besides another challenge will be to label the rest of the images with boxes in order to widen the range of available models. It will take a considerable amount of time as it has to be done as tight as possible, the boxes must only surround the lesions.

Conclusion

Endometriosis is a disease that can be traced to 4000 years ago at least. Its existence has made millions of women suffer and we still have no other definitive treatment than a hysterectomy (complete removal of the uterus) to offer.

Endometriosis concerns over 1 women out of 10 however the diagnosis takes in average 10 years. Throughout history, women haven't been heard for their ‘suffocation of the womb’ and have in the contrary been ostracized and considered as possessed by the devil or suffering from a mental illness because how can period be source of so much pain. Bias towards women pain in healthcare isn’t resolved yet.

One of the areas where research can help women suffering from endometriosis is in the detection of the illness. As we have seen, the standard is a laparoscopic exploration however even for trained doctors some lesions are hardly different from healthy tissue but they cause significant pain to their patients. Computer vision can help not during the surgery but after the surgery. Surgeries nowadays are recorded for legal purposes however it is an incredible ressource that could be used as an input for the YOLO model. The model could help the medical team in collecting all the instances where it detected endometriosis.

The fight against endometriosis isn't over and far from won. Let’s hope that more research will be funded in this field to help alleviate the pain of millions of women.

Bibliography

1. Lim, Mc, et K. Pfaendler. « Type and Risk of Cancer Related to Endometriosis: Ovarian Cancer and Beyond ». BJOG: An International Journal of Obstetrics & Gynaecology, vol. 125, nº 1, 2018, p. 73‑73. Wiley Online Library, https://doi.org/10.1111/1471-0528.14831.

4. Nnoaham, Kelechi E., et al. « Impact of Endometriosis on Quality of Life and Work Productivity: A Multicenter Study across Ten Countries ». Fertility and Sterility, vol. 96, nº 2, août 2011, p. 366-373.e8. PubMed, https://doi.org/10.1016/j.fertnstert.2011.05.090.

5. Nezhat, Camran, et al. « Endometriosis: Ancient Disease, Ancient Treatments ». Fertility and Sterility, vol. 98, nº 6 Suppl, décembre 2012, p. S1-62. PubMed, https://doi.org/10.1016/j.fertnstert.2012.08.001.

Benagiano, Giuseppe, et al. « The History of Endometriosis ». Gynecologic and Obstetric Investigation, vol. 78, nº 1, 2014, p. 1‑9. PubMed, https://doi.org/10.1159/000358919.

6. Berker, Bulent, et Murat Seval. « Problems with the Diagnosis of Endometriosis ». Women's Health, vol. 11, nº 5, septembre 2015, p. 597‑601. SAGE Journals, https://doi.org/10.2217/whe.15.44.

7. Leibetseder, Andreas, et al. « GLENDA: Gynecologic Laparoscopy Endometriosis Dataset ». MultiMedia Modeling, édité par Yong Man Ro et al., Springer International Publishing, 2020, p. 439‑50. Springer Link, https://doi.org/10.1007/978-3-030-37734-2_36.

8. « Revised American Society for Reproductive Medicine Classification of Endometriosis: 1996 ». Fertility and Sterility, vol. 67, nº 5, mai 1997, p. 817‑21. PubMed, https://doi.org/10.1016/s0015-0282(97)81391-x.

9. Lee, Soo-Young, et al. « Classification of Endometriosis ». Yeungnam University Journal of Medicine, vol. 38, nº 1, janvier 2021, p. 10‑18. PubMed, https://doi.org/10.12701/yujm.2020.00444.

Webographie

2 https://www.forbes.com/sites/alicebroster/2020/08/27/why-it-takes-so-long-to-be-diagnosed-with-endometriosis-according-to-a-expert/?sh=fd954e569674

3 https://www.cnbc.com/2016/05/19/this-neglected-disease-is-a-hidden-drain-on-womens-success.html

https://tensorflow-object-detection-api-tutorial.readthedocs.io/en/latest/

https://github.com/pjreddie/darknet

Script

Tensorflow 2 model

1. Importation

Entrée [1]:
```
!pip install opencv-python
```

```
Requirement already satisfied: opencv-python in ./tfod/lib/python3.8/site-packages (4.5.4.58)
Requirement already satisfied: numpy>=1.17.3 in ./tfod/lib/python3.8/site-packages (from opencv-python) (1.21.4)
```

Entrée [2]:
```
import cv2
import uuid
import os
import time
```

2. Folder for images

Entrée [3]:
```
labels = ['endometriosis']
number_imgs = 20
```

Entrée [4]:
```
type(labels)
```

Out[4]:
```
list
```

Entrée [5]:
```
IMAGES_PATH = os.path.join('Tensorflow', 'workspace', 'images', 'collectedimages')
```

Entrée [6]:
```
print(IMAGES_PATH)
```

```
Tensorflow/workspace/images/collectedimages
```

Entr

3. Image Labelling

Entrée [18]:
```
!pip install --upgrade pyqt5 lxml
```

```
Requirement already satisfied: pyqt5 in ./tfod/lib/python3.8/site-packages (5.15.6)
Requirement already satisfied: lxml in ./tfod/lib/python3.8/site-packages (4.6.4)
Requirement already satisfied: PyQt5-sip<13,>=12.8 in ./tfod/lib/python3.8/site-packages (from pyqt5) (12.9.0)
Requirement already satisfied: PyQt5-Qt5>=5.15.2 in ./tfod/lib/python3.8/site-packages (from pyqt5) (5.15.2)
```

Entrée [19]:
```
pip list
```

```
Package           Version
----------------- -------
appnope           0.1.2
asgiref           3.4.1
backcall          0.2.0
debugpy           1.5.1
decorator         5.1.0
Django            3.2.9
entrypoints       0.3
ipykernel         6.5.0
ipython           7.29.0
jedi              0.18.0
jupyter-client    7.0.6
jupyter-core      4.9.1
lxml              4.6.4
matplotlib-inline 0.1.3
nest-asyncio      1.5.1
numpy             1.21.4
opencv-python     4.5.4.58
parso             0.8.2
pexpect           4.8.0
pickleshare       0.7.5
pip               21.3.1
prompt-toolkit    3.0.22
ptyprocess        0.7.0
Pygments          2.10.0
PyQt5             5.15.6
PyQt5-Qt5         5.15.2
PyQt5-sip         12.9.0
python-dateutil   2.8.2
pytz              2021.3
pyzmq             22.3.0
setuptools        47.1.0
six               1.16.0
sqlparse          0.4.2
tornado           6.1
traitlets         5.1.1
wcwidth           0.2.5
Note: you may need to restart the kernel to use updated packages.
```

Entrée [10]:
```
pip install Django
```

```
Requirement already satisfied: Django in ./tfod/lib/python3.8/site-packages (3.2.9)
Requirement already satisfied: sqlparse>=0.2.2 in ./tfod/lib/python3.8/site-packages (from Django) (0.4.2)
Requirement already satisfied: asgiref<4,>=3.3.2 in ./tfod/lib/python3.8/site-packages (from Django) (3.4.1)
Requirement already satisfied: pytz in ./tfod/lib/python3.8/site-packages (from Django) (2021.3)
Note: you may need to restart the kernel to use updated packages.
```

Entrée [20]:
```
LABELIMG_PATH = os.path.join('Tensorflow', 'labelimg')
```

Entrée [21]:
```
if not os.path.exists(LABELIMG_PATH):
    !mkdir {LABELIMG_PATH}
    !git clone https://github.com/tzutalin/labelImg {LABELIMG_PATH}
```

Entrée [22]:
```
if os.name == 'posix':
    !cd {LABELIMG_PATH} && make qt5py3
```

```
pyrcc5 -o libs/resources.py resources.qrc
```

Entrée [ ]:
```
!cd {LABELIMG_PATH} && python labelImg.py
```

Entrée [9]:
```
VERIFICATION_SCRIPT = os.path.join(paths['APIMODEL_PATH'], 'research', 'object_detection', 'builders', 'model_builde
# Verify Installation
!python {VERIFICATION_SCRIPT}
```

```
[       OK ] ModelBuilderTF2Test.test_invalid_model_config_proto
[ RUN      ] ModelBuilderTF2Test.test_invalid_second_stage_batch_size
INFO:tensorflow:time(__main__.ModelBuilderTF2Test.test_invalid_second_stage_batch_size): 0.0s
I1106 17:58:19.024005 4453682688 test_util.py:2188] time(__main__.ModelBuilderTF2Test.test_invalid_second_stage_b
atch_size): 0.0s
[       OK ] ModelBuilderTF2Test.test_invalid_second_stage_batch_size
[ RUN      ] ModelBuilderTF2Test.test_session
[  SKIPPED ] ModelBuilderTF2Test.test_session
[ RUN      ] ModelBuilderTF2Test.test_unknown_faster_rcnn_feature_extractor
INFO:tensorflow:time(__main__.ModelBuilderTF2Test.test_unknown_faster_rcnn_feature_extractor): 0.0s
I1106 17:58:19.025223 4453682688 test_util.py:2188] time(__main__.ModelBuilderTF2Test.test_unknown_faster_rcnn_fe
ature_extractor): 0.0s
[       OK ] ModelBuilderTF2Test.test_unknown_faster_rcnn_feature_extractor
[ RUN      ] ModelBuilderTF2Test.test_unknown_meta_architecture
INFO:tensorflow:time(__main__.ModelBuilderTF2Test.test_unknown_meta_architecture): 0.0s
I1106 17:58:19.025530 4453682688 test_util.py:2188] time(__main__.ModelBuilderTF2Test.test_unknown_meta_architect
ure): 0.0s
[       OK ] ModelBuilderTF2Test.test_unknown_meta_architecture
[ RUN      ] ModelBuilderTF2Test.test_unknown_ssd_feature_extractor
```

Entrée [15]:
```
!pip install tensorflow --upgrade
```

...

Entrée [13]:
```
!pip uninstall protobuf matplotlib -y
!pip install protobuf matplotlib==3.2
```

...

Entrée [10]:
```
import object_detection
```

Entrée [18]:
```
!pip list
```

```
tensorflow                  2.5.0rc0
tensorflow-addons           0.12.1
tensorflow-datasets         4.2.0
tensorflow-hub              0.11.0
tensorflow-model-optimization 0.5.0
termcolor                   1.1.0
tf-estimator-nightly        2.5.0.dev2021032501
tf-models-official          2.4.0
tf-slim                     1.1.0
tornado                     6.1
traitlets                   5.0.5
typing-extensions           3.7.4.3
urllib3                     1.26.4
wcwidth                     0.2.5
Werkzeug                    1.0.1
wget                        3.2
wheel                       0.36.2
wrapt                       1.12.1
zipp                        3.4.1
```

Install the model

Entrée [ ]:
```
```

Entrée [7]:
```
if not os.path.exists(os.path.join(paths['APIMODEL_PATH'], 'research', 'object_detection')):
    !git clone https://github.com/tensorflow/models {paths['APIMODEL_PATH']}
```

```
Cloning into 'Tensorflow/models'...
remote: Enumerating objects: 66323, done.
remote: Counting objects: 100% (140/140), done.
remote: Compressing objects: 100% (86/86), done.
remote: Total 66323 (delta 80), reused 110 (delta 52), pack-reused 66183
Receiving objects: 100% (66323/66323), 575.86 MiB | 21.98 MiB/s, done.
Resolving deltas: 100% (46448/46448), done.
```

Entrée [8]:
```
# Install Tensorflow Object Detection
if os.name=='posix':
    !apt-get install protobuf-compiler
    !cd Tensorflow/models/research && protoc object_detection/protos/*.proto --python_out=. && cp object_detection/p

if os.name=='nt':
    url="https://github.com/protocolbuffers/protobuf/releases/download/v3.15.6/protoc-3.15.6-win64.zip"
    wget.download(url)
    !move protoc-3.15.6-win64.zip {paths['PROTOC_PATH']}
    !cd {paths['PROTOC_PATH']} && tar -xf protoc-3.15.6-win64.zip
    os.environ['PATH'] += os.pathsep + os.path.abspath(os.path.join(paths['PROTOC_PATH'], 'bin'))
    !cd Tensorflow/models/research && protoc object_detection/protos/*.proto --python_out=. && copy object_detection
    !cd Tensorflow/models/research/slim && pip install -e .
```

```
Collecting contextlib2
  Downloading contextlib2-21.6.0-py2.py3-none-any.whl (13 kB)
Collecting tf-slim
  Downloading tf_slim-1.1.0-py2.py3-none-any.whl (352 kB)
     |████████████████████████████████| 352 kB 47.3 MB/s
Requirement already satisfied: six in /Users/Aurane/Desktop/TFODCourse/tfod/lib/python3.8/site-packages (from obj
ect-detection==0.1) (1.16.0)
Collecting pycocotools
  Downloading pycocotools-2.0.2.tar.gz (23 kB)
  Preparing metadata (setup.py) ... done
Collecting lvis
  Downloading lvis-0.5.3-py3-none-any.whl (14 kB)
Collecting scipy
  Downloading scipy-1.7.2-cp38-cp38-macosx_10_9_x86_64.whl (33.0 MB)
     |████████████████████████████████| 33.0 MB 256 kB/s
Collecting pandas
  Downloading pandas-1.3.4-cp38-cp38-macosx_10_9_x86_64.whl (11.4 MB)
     |████████████████████████████████| 11.4 MB 9.2 MB/s
Collecting tf-models-official>=2.5.1
```

Entrée [12]:
```
if os.name =='posix':
    !wget {PRETRAINED_MODEL_URL}
    !mv {PRETRAINED_MODEL_NAME+'.tar.gz'} {paths['PRETRAINED_MODEL_PATH']}
    !cd {paths['PRETRAINED_MODEL_PATH']} && tar -zxvf {PRETRAINED_MODEL_NAME+'.tar.gz'}
```

```
--2021-11-06 19:13:44--  http://download.tensorflow.org/models/object_detection/tf2/20200711/ssd_mobilenet_v2_fpnli
te_320x320_coco17_tpu-8.tar.gz
Résolution de download.tensorflow.org (download.tensorflow.org)… 2a00:1450:4007:80c::2010, 172.217.18.208
Connexion à download.tensorflow.org (download.tensorflow.org)|2a00:1450:4007:80c::2010|:80… connecté.
requête HTTP transmise, en attente de la réponse… 200 OK
Taille : 20515344 (20M) [application/x-tar]
Sauvegarde en : « ssd_mobilenet_v2_fpnlite_320x320_coco17_tpu-8.tar.gz »

ssd_mobilenet_v2_fp 100%[===================>]  19,56M  30,1MB/s    ds 0,6s

2021-11-06 19:13:46 (30,1 MB/s) – « ssd_mobilenet_v2_fpnlite_320x320_coco17_tpu-8.tar.gz » sauvegardé [20515344/205
15344]

x ssd_mobilenet_v2_fpnlite_320x320_coco17_tpu-8/
x ssd_mobilenet_v2_fpnlite_320x320_coco17_tpu-8/checkpoint/
x ssd_mobilenet_v2_fpnlite_320x320_coco17_tpu-8/checkpoint/ckpt-0.data-00000-of-00001
x ssd_mobilenet_v2_fpnlite_320x320_coco17_tpu-8/checkpoint/checkpoint
x ssd_mobilenet_v2_fpnlite_320x320_coco17_tpu-8/checkpoint/ckpt-0.index
x ssd_mobilenet_v2_fpnlite_320x320_coco17_tpu-8/pipeline.config
x ssd_mobilenet_v2_fpnlite_320x320_coco17_tpu-8/saved_model/
x ssd_mobilenet_v2_fpnlite_320x320_coco17_tpu-8/saved_model/saved_model.pb
x ssd_mobilenet_v2_fpnlite_320x320_coco17_tpu-8/saved_model/variables/
x ssd_mobilenet_v2_fpnlite_320x320_coco17_tpu-8/saved_model/variables/variables.data-00000-of-00001
x ssd_mobilenet_v2_fpnlite_320x320_coco17_tpu-8/saved_model/variables/variables.index
```

2. Create Label Map

Entrée [51]:
```
labels = [{'name':'endometriosis', 'id':1}]

with open(files['LABELMAP'], 'w') as f:
    for label in labels:
        f.write('item { \n')
        f.write('\tname:\'{}\'\n'.format(label['name']))
        f.write('\tid:{}\n'.format(label['id']))
        f.write('}\n')
```

3. Create TF records

Entrée [14]:
```
if not os.path.exists(files['TF_RECORD_SCRIPT']):
    !git clone https://github.com/nicknochnack/GenerateTFRecord {paths['SCRIPTS_PATH']}
```

```
Cloning into 'Tensorflow/scripts'...
remote: Enumerating objects: 3, done.
remote: Counting objects: 100% (3/3), done.
remote: Compressing objects: 100% (2/2), done.
remote: Total 3 (delta 0), reused 1 (delta 0), pack-reused 0
Receiving objects: 100% (3/3), done.
```

Entrée [15]:
```
!python {files['TF_RECORD_SCRIPT']} -x {os.path.join(paths['IMAGE_PATH'], 'train')} -l {files['LABELMAP']} -o {os.pa
!python {files['TF_RECORD_SCRIPT']} -x {os.path.join(paths['IMAGE_PATH'], 'test')} -l {files['LABELMAP']} -o {os.pat
```

```
Successfully created the TFRecord file: Tensorflow/workspace/annotations/train.record
Successfully created the TFRecord file: Tensorflow/workspace/annotations/test.record
```

4. Copy Model Config to Training Folder

Entrée [18]:
```
!cp {os.path.join(paths['PRETRAINED_MODEL_PATH'], PRETRAINED_MODEL_NAME, 'pipeline.config')} {os.path.join(paths['CH
```

5. Update Config For Transfer Learning

Entrée [19]:
```
import tensorflow as tf
from object_detection.utils import config_util
from object_detection.protos import pipeline_pb2
from google.protobuf import text_format
```

Entrée [20]:
```
config = config_util.get_configs_from_pipeline_file(files['PIPELINE_CONFIG'])
```

Entrée [21]:
```
config
```

```
    }
  }
  post_processing {
    batch_non_max_suppression {
      score_threshold: 9.99999993922529e-09
      iou_threshold: 0.600000023841858
      max_detections_per_class: 100
      max_total_detections: 100
      use_static_shapes: false
    }
    score_converter: SIGMOID
  }
  normalize_loss_by_num_matches: true
  loss {
    localization_loss {
      weighted_smooth_l1 {
      }
    }
    classification_loss {
```

Entrée [22]:
```
pipeline_config = pipeline_pb2.TrainEvalPipelineConfig()
with tf.io.gfile.GFile(files['PIPELINE_CONFIG'], "r") as f:
    proto_str = f.read()
    text_format.Merge(proto_str, pipeline_config)
```

Entrée [23]:
```
pipeline_config.model.ssd.num_classes = len(labels)
pipeline_config.train_config.batch_size = 4
pipeline_config.train_config.fine_tune_checkpoint = os.path.join(paths['PRETRAINED_MODEL_PATH'], PRETRAINED_MODEL_NA
pipeline_config.train_config.fine_tune_checkpoint_type = "detection"
pipeline_config.train_input_reader.label_map_path= files['LABELMAP']
pipeline_config.train_input_reader.tf_record_input_reader.input_path[:] = [os.path.join(paths['ANNOTATION_PATH'], 't
pipeline_config.eval_input_reader[0].label_map_path = files['LABELMAP']
pipeline_config.eval_input_reader[0].tf_record_input_reader.input_path[:] = [os.path.join(paths['ANNOTATION_PATH'],
```

Entrée [24]:
```
config_text = text_format.MessageToString(pipeline_config)
with tf.io.gfile.GFile(files['PIPELINE_CONFIG'], "wb") as f:
    f.write(config_text)
```

6. Train the model

Entrée [25]:
```
TRAINING_SCRIPT = os.path.join(paths['APIMODEL_PATH'], 'research', 'object_detection', 'model_main_tf2.py')
```

Entrée [48]:
```
command = "python {} --model_dir={} --pipeline_config_path={} --num_train_steps=1000".format(TRAINING_SCRIPT, paths[
```

Entrée [27]:
```
print(command)
```

```
python Tensorflow/models/research/object_detection/model_main_tf2.py --model_dir=Tensorflow/workspace/models/my_ssd
_mobnet --pipeline_config_path=Tensorflow/workspace/models/my_ssd_mobnet/pipeline.config --num_train_steps=2000
```

Entrée [45]:
```
import os
os.environ["CUDA_DEVICE_ORDER"]="PCI_BUS_ID"
os.environ["CUDA_VISIBLE_DEVICES"]="0"
```

Entrée [49]:
```
!{command}
```

7. Evaluate the Model

Entrée [50]:
```
command = "python {} --model_dir={} --pipeline_config_path={} --checkpoint_dir={}".format(TRAINING_SCRIPT, paths['CH
```

Entrée [21]:
```
print(command)
```

```
python Tensorflow\models\research\object_detection\model_main_tf2.py --model_dir=Tensorflow\workspace\models\my_ssd
_mobnet --pipeline_config_path=Tensorflow\workspace\models\my_ssd_mobnet\pipeline.config --checkpoint_dir=Tensorflo
w\workspace\models\my_ssd_mobnet
```

Entrée [22]:
```
!{command}
```

```
^C
```

Yolo model

```
!nvidia-smi
```

```
[ ] from google.colab import drive
    drive.mount('/content/gdrive')
    !ln -s /content/gdrive/My\ Drive/ /mydrive
    !ls /mydrive
```

```
[ ] !git clone https://github.com/AlexeyAB/darknet
```

```
Cloning into 'darknet'...
remote: Enumerating objects: 15339, done.
remote: Total 15339 (delta 0), reused 0 (delta 0), pack-reused 15339
Receiving objects: 100% (15339/15339), 13.99 MiB | 18.35 MiB/s, done.
Resolving deltas: 100% (10314/10314), done.
```

```
!wget https://pjreddie.com/media/files/darknet53.conv.74
```

```
--2021-11-07 20:07:15--  https://pjreddie.com/media/files/darknet53.conv.74
Resolving pjreddie.com (pjreddie.com)... 128.208.4.108
Connecting to pjreddie.com (pjreddie.com)|128.208.4.108|:443... connected.
HTTP request sent, awaiting response... 200 OK
Length: 162482580 (155M) [application/octet-stream]
Saving to: 'darknet53.conv.74'

darknet53.conv.74   100%[===================>] 154.96M  55.9MB/s    in 2.8s

2021-11-07 20:07:18 (55.9 MB/s) - 'darknet53.conv.74' saved [162482580/162482580]
```

```
!unzip /mydrive/yolov3/image.zip -d data/obj
```

```
txt_file_paths = glob.glob(r"data/obj/*.txt")
for i, file_path in enumerate(txt_file_paths):

    with open(file_path, "r") as f_o:
        lines = f_o.readlines()

        text_converted = []
        for line in lines:
            print(line)
            numbers = re.findall("[0-9.]+", line)
            print(numbers)
            if numbers:


                text = "{} {} {} {} {}".format(0, numbers[1], numbers[2], numbers[3], numbers[4])
                text_converted.append(text)
                print(i, file_path)
                print(text)

        with open(file_path, 'w') as fp:
            for item in text_converted:
                fp.writelines("%s\n" % item)
```

```
images_list = glob.glob("data/obj/*.jpg")
print(images_list)
```

2) Compile Darknet

```
%cd darknet
!sed -i 's/OPENCV=0/OPENCV=1/' Makefile
!sed -i 's/GPU=0/GPU=1/' Makefile
!sed -i 's/CUDNN=0/CUDNN=1/' Makefile
!make
```

```
                                                                                                    %4ld
./src/softmax_layer.c: In function ‘forward_contrastive_layer’:
./src/softmax_layer.c:244:27: warning: variable ‘max_truth’ set but not used [-Wunused-but-set-variable]
                    float max_truth = 0;
                          ^~~~~~~~~
./src/softmax_layer.c:423:71: warning: format ‘%d’ expects argument of type ‘int’, but argument 2 has type ‘size_t {aka const long unsigned int}’ [-W
             printf(" Error: too large number of bboxes: contr_size = %d > max_contr_size  = %d \n", contr_size, max_contr_size);
                                                                      ~^
                                                                      %ld
gcc -Iinclude/ -I3rdparty/stb/include -DOPENCV `pkg-config --cflags opencv4 2> /dev/null || pkg-config --cflags opencv` -DGPU -I/usr/local/cuda/inclu
./src/data.c: In function ‘load_data_detection’:
./src/data.c:1297:24: warning: unused variable ‘x’ [-Wunused-variable]
                int k, x, y;
                       ^
./src/data.c:1090:43: warning: variable ‘r_scale’ set but not used [-Wunused-but-set-variable]
     float r1 = 0, r2 = 0, r3 = 0, r4 = 0, r_scale = 0;
                                           ^~~~~~~
gcc -Iinclude/ -I3rdparty/stb/include -DOPENCV `pkg-config --cflags opencv4 2> /dev/null || pkg-config --cflags opencv` -DGPU -I/usr/local/cuda/inclu
gcc -Iinclude/ -I3rdparty/stb/include -DOPENCV `pkg-config --cflags opencv4 2> /dev/null || pkg-config --cflags opencv` -DGPU -I/usr/local/cuda/inclu
./src/network.c: In function ‘train_network_waitkey’:
./src/network.c:435:13: warning: unused variable ‘ema_period’ [-Wunused-variable]
         int ema_period = (net.max_batches - ema_start_point - 1000) * (1.0 - net.ema_alpha);
             ^~~~~~~~~~
./src/network.c: In function ‘resize_network’:
./src/network.c:660:42: warning: passing argument 1 of ‘cudaHostAlloc’ from incompatible pointer type [-Wincompatible-pointer-types]
         if (cudaSuccess == cudaHostAlloc(&net->input_pinned_cpu, size * sizeof(float), cudaHostRegisterMapped))
                                          ^
In file included from /usr/local/cuda/include/cuda_runtime.h:96:0,
                 from include/darknet.h:41,
                 from ./src/network.c:1:
/usr/local/cuda/include/cuda_runtime_api.h:4811:39: note: expected ‘void **’ but argument is of type ‘float **’
 extern __host__ cudaError_t CUDARTAPI cudaHostAlloc(void **pHost, size_t size, unsigned int flags);
                                       ^~~~~~~~~~~~
gcc -Iinclude/ -I3rdparty/stb/include -DOPENCV `pkg-config --cflags opencv4 2> /dev/null || pkg-config --cflags opencv` -DGPU -I/usr/local/cuda/inclu
./src/connected_layer.c: In function ‘forward_connected_layer_gpu’:
./src/connected_layer.c:346:11: warning: unused variable ‘one’ [-Wunused-variable]
     float one = 1;    // alpha[0], beta[0]
           ^~~
./src/connected_layer.c:344:13: warning: unused variable ‘c’ [-Wunused-variable]
     float * c = l.output_gpu;
             ^
./src/connected_layer.c:343:13: warning: unused variable ‘b’ [-Wunused-variable]
     float * b = l.weights_gpu;
             ^
./src/connected_layer.c:342:13: warning: unused variable ‘a’ [-Wunused-variable]
     float * a = state.input;
             ^
./src/connected_layer.c:341:9: warning: unused variable ‘n’ [-Wunused-variable]
     int n = l.outputs;
         ^
./src/connected_layer.c:340:9: warning: unused variable ‘k’ [-Wunused-variable]
     int k = l.inputs;
         ^
./src/connected_layer.c:339:9: warning: unused variable ‘m’ [-Wunused-variable]
     int m = l.batch;
         ^
gcc -Iinclude/ -I3rdparty/stb/include -DOPENCV `pkg-config --cflags opencv4 2> /dev/null || pkg-config --cflags opencv` -DGPU -I/usr/local/cuda/inclu
gcc -Iinclude/ -I3rdparty/stb/include -DOPENCV `pkg-config --cflags opencv4 2> /dev/null || pkg-config --cflags opencv` -DGPU -I/usr/local/cuda/inclu
./src/parser.c: In function ‘parse_network_cfg_custom’:
```

```
[ ] file = open("data/train.txt", "w")
    file.write("\n".join(images_list))
    file.close()
```

6) Start the training

```
[ ] # Start the training
    !./darknet detector train data/obj.data cfg/yolov3_training.cfg darknet53.conv.74 -dont_show
```

```
Can't open label file. (This can be normal only if you use MSCOCO): data/obj/c_4_v_(video_48.mp4)_f_402.txt
Can't open label file. (This can be normal only if you use MSCOCO): data/obj/c_3_v_(video_27.mp4)_f_163.txt
v3 (mse loss, Normalizer: (iou: 0.75, obj: 1.00, cls: 1.00) Region 106 Avg (IOU: 0.000000), count: 1, class_loss = 0.032531, iou_loss = 0.000000, tot
v3 (mse loss, Normalizer: (iou: 0.75, obj: 1.00, cls: 1.00) Region 82 Avg (IOU: 0.000000), count: 1, class_loss = 0.034261, iou_loss = 0.000000, tota
v3 (mse loss, Normalizer: (iou: 0.75, obj: 1.00, cls: 1.00) Region 94 Avg (IOU: 0.000000), count: 1, class_loss = 0.035687, iou_loss = 0.000000, tota
v3 (mse loss, Normalizer: (iou: 0.75, obj: 1.00, cls: 1.00) Region 106 Avg (IOU: 0.000000), count: 1, class_loss = 0.034079, iou_loss = 0.000000, tot
v3 (mse loss, Normalizer: (iou: 0.75, obj: 1.00, cls: 1.00) Region 82 Avg (IOU: 0.000000), count: 1, class_loss = 0.032517, iou_loss = 0.000000, tota
v3 (mse loss, Normalizer: (iou: 0.75, obj: 1.00, cls: 1.00) Region 94 Avg (IOU: 0.000000), count: 1, class_loss = 0.034307, iou_loss = 0.000000, tota
v3 (mse loss, Normalizer: (iou: 0.75, obj: 1.00, cls: 1.00) Region 106 Avg (IOU: 0.000000), count: 1, class_loss = 0.035892, iou_loss = 0.000000, tot
v3 (mse loss, Normalizer: (iou: 0.75, obj: 1.00, cls: 1.00) Region 82 Avg (IOU: 0.000000), count: 1, class_loss = 0.034241, iou_loss = 0.000000, tota
v3 (mse loss, Normalizer: (iou: 0.75, obj: 1.00, cls: 1.00) Region 94 Avg (IOU: 0.000000), count: 1, class_loss = 0.034284, iou_loss = 0.000000, tota
v3 (mse loss, Normalizer: (iou: 0.75, obj: 1.00, cls: 1.00) Region 106 Avg (IOU: 0.000000), count: 1, class_loss = 0.034054, iou_loss = 0.000000, tot
v3 (mse loss, Normalizer: (iou: 0.75, obj: 1.00, cls: 1.00) Region 82 Avg (IOU: 0.000000), count: 1, class_loss = 0.033051, iou_loss = 0.000000, tota
v3 (mse loss, Normalizer: (iou: 0.75, obj: 1.00, cls: 1.00) Region 94 Avg (IOU: 0.000000), count: 1, class_loss = 0.034212, iou_loss = 0.000000, tota
v3 (mse loss, Normalizer: (iou: 0.75, obj: 1.00, cls: 1.00) Region 106 Avg (IOU: 0.000000), count: 1, class_loss = 0.034000, iou_loss = 0.000000, tot
v3 (mse loss, Normalizer: (iou: 0.75, obj: 1.00, cls: 1.00) Region 82 Avg (IOU: 0.000000), count: 1, class_loss = 0.032961, iou_loss = 0.000000, tota
v3 (mse loss, Normalizer: (iou: 0.75, obj: 1.00, cls: 1.00) Region 94 Avg (IOU: 0.000000), count: 1, class_loss = 0.035365, iou_loss = 0.000000, tota
v3 (mse loss, Normalizer: (iou: 0.75, obj: 1.00, cls: 1.00) Region 106 Avg (IOU: 0.000000), count: 1, class_loss = 0.033725, iou_loss = 0.000000, tot
v3 (mse loss, Normalizer: (iou: 0.75, obj: 1.00, cls: 1.00) Region 82 Avg (IOU: 0.000000), count: 1, class_loss = 0.034502, iou_loss = 0.000000, tota
v3 (mse loss, Normalizer: (iou: 0.75, obj: 1.00, cls: 1.00) Region 94 Avg (IOU: 0.000000), count: 1, class_loss = 0.034324, iou_loss = 0.000000, tota
v3 (mse loss, Normalizer: (iou: 0.75, obj: 1.00, cls: 1.00) Region 106 Avg (IOU: 0.000000), count: 1, class_loss = 0.035905, iou_loss = 0.000000, tot
v3 (mse loss, Normalizer: (iou: 0.75, obj: 1.00, cls: 1.00) Region 82 Avg (IOU: 0.000000), count: 1, class_loss = 0.033758, iou_loss = 0.000000, tota
v3 (mse loss, Normalizer: (iou: 0.75, obj: 1.00, cls: 1.00) Region 94 Avg (IOU: 0.000000), count: 1, class_loss = 0.034185, iou_loss = 0.000000, tota
v3 (mse loss, Normalizer: (iou: 0.75, obj: 1.00, cls: 1.00) Region 106 Avg (IOU: 0.000000), count: 1, class_loss = 0.033791, iou_loss = 0.000000, tot
v3 (mse loss, Normalizer: (iou: 0.75, obj: 1.00, cls: 1.00) Region 82 Avg (IOU: 0.000000), count: 1, class_loss = 0.033840, iou_loss = 0.000000, tota
v3 (mse loss, Normalizer: (iou: 0.75, obj: 1.00, cls: 1.00) Region 94 Avg (IOU: 0.000000), count: 1, class_loss = 0.035096, iou_loss = 0.000000, tota
v3 (mse loss, Normalizer: (iou: 0.75, obj: 1.00, cls: 1.00) Region 106 Avg (IOU: 0.000000), count: 1, class_loss = 0.035902, iou_loss = 0.000000, tot
v3 (mse loss, Normalizer: (iou: 0.75, obj: 1.00, cls: 1.00) Region 82 Avg (IOU: 0.000000), count: 1, class_loss = 0.033947, iou_loss = 0.000000, tota
v3 (mse loss, Normalizer: (iou: 0.75, obj: 1.00, cls: 1.00) Region 94 Avg (IOU: 0.000000), count: 1, class_loss = 0.034928, iou_loss = 0.000000, tota
v3 (mse loss, Normalizer: (iou: 0.75, obj: 1.00, cls: 1.00) Region 106 Avg (IOU: 0.000000), count: 1, class_loss = 0.033418, iou_loss = 0.000000, tot
v3 (mse loss, Normalizer: (iou: 0.75, obj: 1.00, cls: 1.00) Region 82 Avg (IOU: 0.000000), count: 1, class_loss = 0.033944, iou_loss = 0.000000, tota
v3 (mse loss, Normalizer: (iou: 0.75, obj: 1.00, cls: 1.00) Region 94 Avg (IOU: 0.000000), count: 1, class_loss = 0.034771, iou_loss = 0.000000, tota
v3 (mse loss, Normalizer: (iou: 0.75, obj: 1.00, cls: 1.00) Region 106 Avg (IOU: 0.000000), count: 1, class_loss = 0.031058, iou_loss = 0.000000, tot
v3 (mse loss, Normalizer: (iou: 0.75, obj: 1.00, cls: 1.00) Region 82 Avg (IOU: 0.000000), count: 1, class_loss = 0.034141, iou_loss = 0.000000, tota
v3 (mse loss, Normalizer: (iou: 0.75, obj: 1.00, cls: 1.00) Region 94 Avg (IOU: 0.000000), count: 1, class_loss = 0.035000, iou_loss = 0.000000, tota
v3 (mse loss, Normalizer: (iou: 0.75, obj: 1.00, cls: 1.00) Region 106 Avg (IOU: 0.000000), count: 1, class_loss = 0.032614, iou_loss = 0.000000, tot
v3 (mse loss, Normalizer: (iou: 0.75, obj: 1.00, cls: 1.00) Region 82 Avg (IOU: 0.000000), count: 1, class_loss = 0.032806, iou_loss = 0.000000, tota
v3 (mse loss, Normalizer: (iou: 0.75, obj: 1.00, cls: 1.00) Region 94 Avg (IOU: 0.000000), count: 1, class_loss = 0.034259, iou_loss = 0.000000, tota
v3 (mse loss, Normalizer: (iou: 0.75, obj: 1.00, cls: 1.00) Region 106 Avg (IOU: 0.000000), count: 1, class_loss = 0.035034, iou_loss = 0.000000, tot
v3 (mse loss, Normalizer: (iou: 0.75, obj: 1.00, cls: 1.00) Region 82 Avg (IOU: 0.000000), count: 1, class_loss = 0.032880, iou_loss = 0.000000, tota
v3 (mse loss, Normalizer: (iou: 0.75, obj: 1.00, cls: 1.00) Region 94 Avg (IOU: 0.000000), count: 1, class_loss = 0.033798, iou_loss = 0.000000, tota
v3 (mse loss, Normalizer: (iou: 0.75, obj: 1.00, cls: 1.00) Region 106 Avg (IOU: 0.000000), count: 1, class_loss = 0.031097, iou_loss = 0.000000, tot
v3 (mse loss, Normalizer: (iou: 0.75, obj: 1.00, cls: 1.00) Region 82 Avg (IOU: 0.000000), count: 1, class_loss = 0.033029, iou_loss = 0.000000, tota
v3 (mse loss, Normalizer: (iou: 0.75, obj: 1.00, cls: 1.00) Region 94 Avg (IOU: 0.000000), count: 1, class_loss = 0.034725, iou_loss = 0.000000, tota
v3 (mse loss, Normalizer: (iou: 0.75, obj: 1.00, cls: 1.00) Region 106 Avg (IOU: 0.000000), count: 1, class_loss = 0.032430, iou_loss = 0.000000, tot
v3 (mse loss, Normalizer: (iou: 0.75, obj: 1.00, cls: 1.00) Region 82 Avg (IOU: 0.000000), count: 1, class_loss = 0.034564, iou_loss = 0.000000, tota
v3 (mse loss, Normalizer: (iou: 0.75, obj: 1.00, cls: 1.00) Region 94 Avg (IOU: 0.000000), count: 1, class_loss = 0.034916, iou_loss = 0.000000, tota
v3 (mse loss, Normalizer: (iou: 0.75, obj: 1.00, cls: 1.00) Region 106 Avg (IOU: 0.000000), count: 1, class_loss = 0.033618, iou_loss = 0.000000, tot

 289: 0.033985, 0.056138 avg loss, 0.000007 rate, 10.494356 seconds, 18496 images, 14.076086 hours left
Loaded: 0.000076 seconds
Can't open label file. (This can be normal only if you use MSCOCO): data/obj/c_3_v_(video_25.mp4)_f_249.txt
Can't open label file. (This can be normal only if you use MSCOCO): data/obj/c_3_v_(video_25.mp4)_f_249.txt
Can't open label file. (This can be normal only if you use MSCOCO): data/obj/c_3_v_(video_29.mp4)_f_287.txt
Can't open label file. (This can be normal only if you use MSCOCO): data/obj/c_3_v_(video_25.mp4)_f_249.txt
Can't open label file. (This can be normal only if you use MSCOCO): data/obj/c_3_v_(video_27.mp4)_f_501.txt
Can't open label file. (This can be normal only if you use MSCOCO): data/obj/c_5_v_(video_62.mp4)_f_681.txt
Can't open label file. (This can be normal only if you use MSCOCO): data/obj/c_3_v_(video_29.mp4)_f_480.txt
```