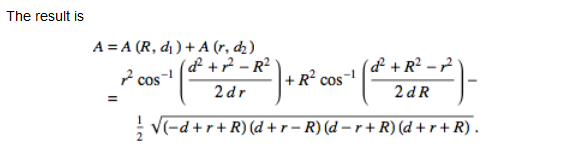
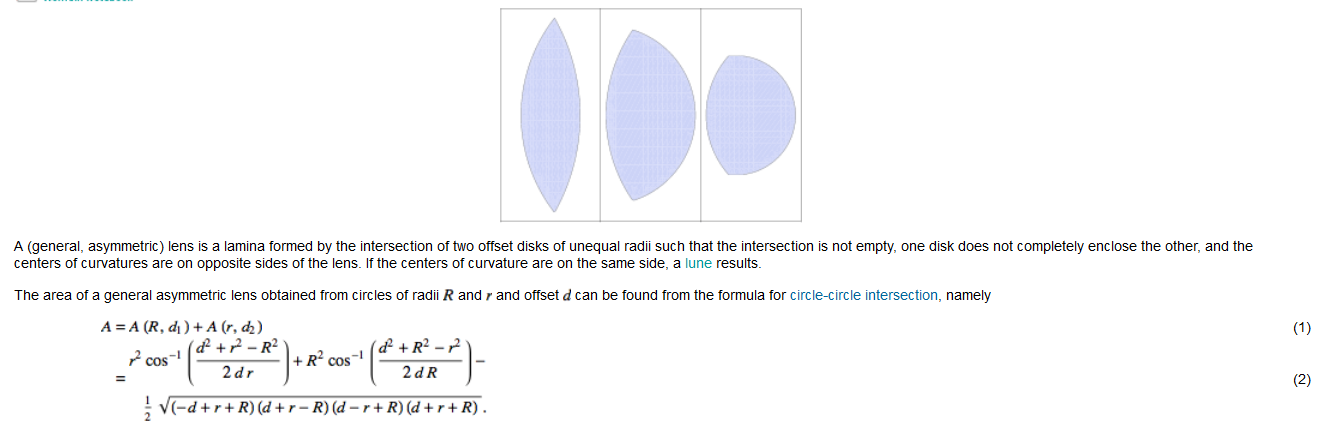
Documentation for Thesis work



<http://mathworld.wolfram.com/Circle-CircleIntersection.html>



<http://mathworld.wolfram.com/Lens.html>

<https://www.xarg.org/2016/07/calculate-the-intersection-area-of-two-circles/>

-> I have used this formula to calculate the area of the intersection of 2 circles.

-> Generalized IoU

<https://arxiv.org/abs/1902.09630>

-> Library for making shapes and manipulating them in the cartesian plane

<https://github.com/Toblerity/Shapely>

-> ellipse ellipse overlap

<https://stackoverflow.com/questions/48808941/overlap-area-of-2-ellipses-using-matplotlib>

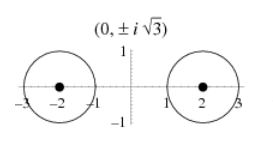
-> I used shapely to generate built-in functions. Then I used built-in methods to calculate the overlapping area and find the IoU.

For script *bounding box smaller circle.py*

* This script calculates the IoU of two overlapping circles by finding the overlapping area between the 2 circles.
* There are 2 special cases where the original formula does not provide any solution. These are solved by me using concepts from geometry

1. When the 2 circles do not overlap each other:

According to the theory provided by [wolfram alpha](http://mathworld.wolfram.com/Circle-CircleIntersection.html), they intersect on an imaginary point



Due to this, I used the following logic:

If distance between the 2 circles is greater than the sum of their radii, then it shows that the circle do not overlap at all

if distance > (radius\_circle1 + radius\_circle2):

1. When 1 circle lies inside the other circle:

This happens when the 2 circles are of unequal length. To solve this case, I use the following logic:

If the distance between the 2 circles + the smaller circle is greater than the radius of the large circle, then this means that the overlapping area would be equal to the area of the smaller circle.

if (distance + min(radius\_circle1, radius\_circle2)) < max(radius\_circle1, radius\_circle2):

For script *inscribed\_circle.py*

* This script inscribes a cirlce inside a square.
* This is useful for converting the datasets which are already annotated. Otherwise we would need to manually label each dataset again.
* We can also inscribe different shapes inside the square to compare against other shapes.

For script *ellipse ellipse overlap.py*

* This script calculates the overlapping area between 2 ellipses.

**Final Version for Thesis:**

**Abstract:**

This research explores the use of non-square bounding box for object detection. Specifically, we want to find the most interesting Region of Interest (ROI) by optimizing the Intersection over Union (IoU). This evaluation metric is mostly used to find the accuracy of an object detection network.

**Introduction:**

Object detection refers to the process of identifying certain objects in a given image or video. This allows us to develop a number of applications which leverage this technique and train systems based on it. This technique is based on supervised learning, which means that we need to supply the system with training data in order for it to learn from. This usually includes pictures which have a highlighted area which we want to predict. If we have an picture with 1 cat and 1 dog, then we need to properly label the cat and dog with bounding boxes. Bounding box basically represents the area which is of interest to us. This usually includes a rectangular box enclosing the relevant object. The network learns the regions of interest using these rectangular bounding boxes. It then performs prediction based on it. The metric used by the network to learn these object is usually called Intersection over union (IoU). It gives the ratio of the overlapping area between the actual bounding box and the predicted bounding box of the neural network. Since a rectangle encloses the most area, therefore it is the most commonly used shape to annotate the images. However ,there are certain cases in daily life where the object of interest is not necessarily rectangular. Wheels of a car, Round traffic signs, food items are some prevalent examples in daily life. Enclosing them inside a rectangular bounding box is not the most optimal solution since it also introduces noise from the background.

**Problem Statement:**

In order to solve the above mentioned problem, this research investigates the potential use of non-square bounding box e.g Ellipse, diamond, etc. There are two benefits for using this approach:

1. Since most objects are not rectangular, therefore using non-square bounding boxes will help us get more accurate representation when performing object detection
2. Since the area enclosed by the circle is less than the area enclosed by a rectangle, therefore the IoU will get decreased.

**Challenges:**

Some of the challenges that are present at the moment are listed below:

1. Converting the existing datasets’s annotated bounding box to the geometrical shape of choice
2. Making the existing networks learn non-square bounding boxes as well as generating them.

**Outcome:**

The outcome of this research is an object detection network capable of generating non-rectangular bounding boxes with an improved IoU as compared to the state-of-the-art networks. This work can be integrated into the tensorflow object detection API.

Introvduction:

**Abstract:**

* This research explores the use of non-square bounding box for object detection. Specifically, we want to find the most interesting Region of Interest (ROI) by optimizing the Intersection over Union (IoU). This evaluation metric is mostly used to find the accuracy of an object detection network.
* If this works, this can be a step towards semi object segmentation. This means that we can also have some element of object segmentation, depending upon the nature of the bounding box.
* This work can be integrated into object detection.
* Future work could include more complex convex bounding boxes which completely encapsulate the relevant object in an image.
* Current methods use square bounding boxes to label the ground truth of an object in an image.
* Some objects in real life are not rectangular. Therefore using rectangular bound boxes to label is not the most optimal approach.
* Some objects are based on different geometrical structures. Example: Wheels of a car, round traffic signs, human face, etc.
* Therefore it makes sense to encode them using the geometrical shape that best describes(encapsulate) them.
* Can be included in the Tensorflow object detection API.
* The idea is that since the circle has smaller area as compared to a square, therefore, the IoU will be decreased.

**Definitions**

Object detection:

It deals with the detection of semantic objects belonging to a certain category (humans, buildings, cars, etc.) in digital images and videos [1]. There are numerous real-world applications for this domain such as Video surveillance, car detection, etc.

CNN:

They are a special type of deep neural network which is most commonly used for visual imagery analysis. They perform convolution operation on image pixels to extract important features from images. Activation functions are used to classify either the image or the objects present in the image.

IoU:

It is also called Jaccard index. It is an accuracy metric used to measure the accuracy of an object detector [2]. The following illustration shows how IoU works:



Figure 1: Graphical illustration of IoU [2]

[1] https://en.wikipedia.org/wiki/Object\_detection

[2] https://www.pyimagesearch.com/2016/11/07/intersection-over-union-iou-for-object-detection/

**Methodology:**

The current work establishes a proof-of-concept for the research topic. The fundamental functions are implemented which would later be applied to the object detection network to achieve the objective. The work done so far encompasses the following domains:

1. Circle-circle overlap IoU:
   1. The research started with the implementation of the circle-circle overlap and calculating the IoU of them. Consider two circles with the following naming conventions:

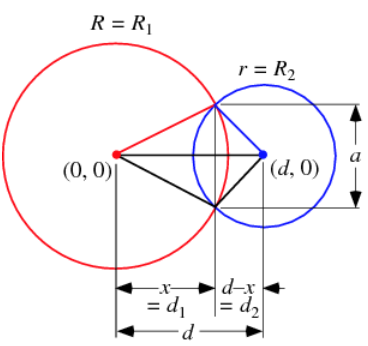
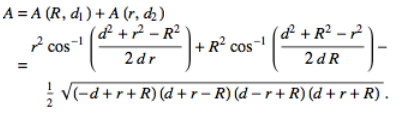
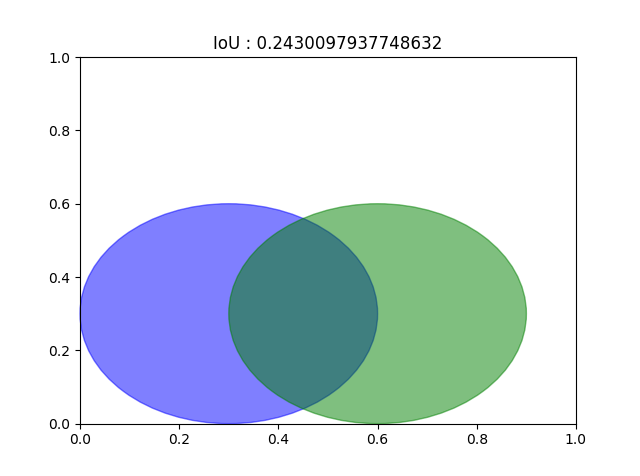


Figure 1: Attributes defined with appropriate variables as mentioned in [2]

* 1. Using the above representation, the following formula is used to calculate the overlapping area, which in turn is used to calculate the IoU:



* 1. This is the most simple geometrical shape from which the IoU can be calculated easily. The library used matplotlib [1] to generate the circles. The following formula The following figure illustrates this concept:



* 1. There are two special cases in the circle-circle overlap in which the original formula does not provide any solution. These are mentioned below:
     1. **When circles do not overlap each other:**

If the circles do not overlap each other at all, then they intersect on an imaginary point

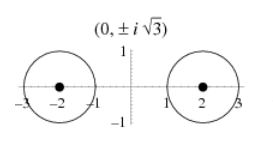
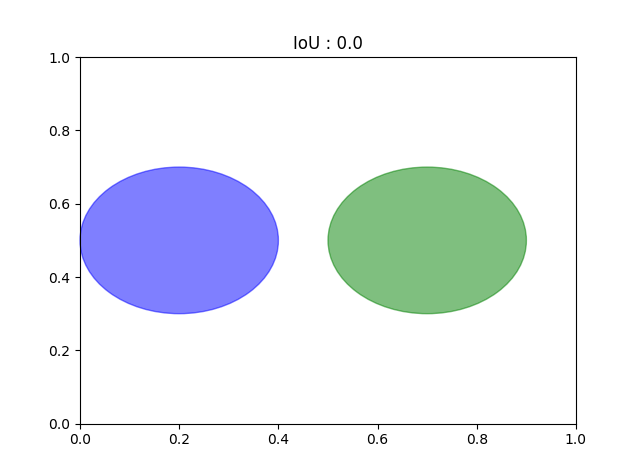


Fig 2: Circles intersecting on an imaginary point [2]

In order to overcome this problem, we make the assumption that if the distance between the two circles is greater than the sum of their radii, then it shows that the circle do not overlap at all. This is shown in the formula below

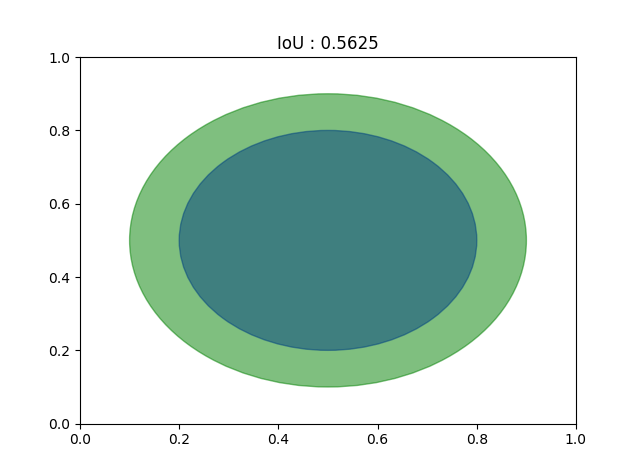
if distance > (radius\_circle1 + radius\_circle2):



* + 1. **When a circle is inscribed inside another circle:**

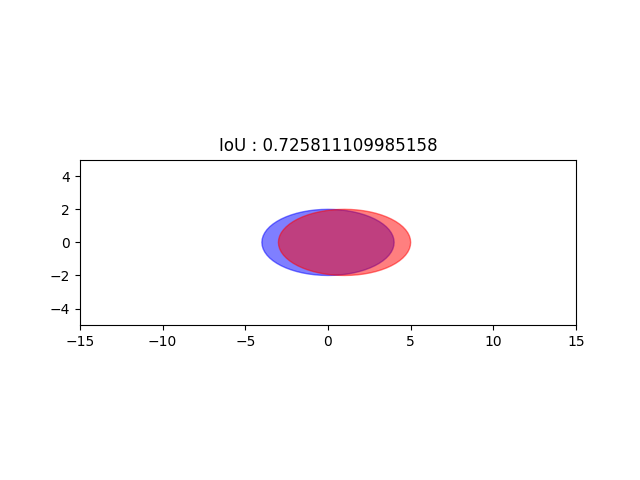
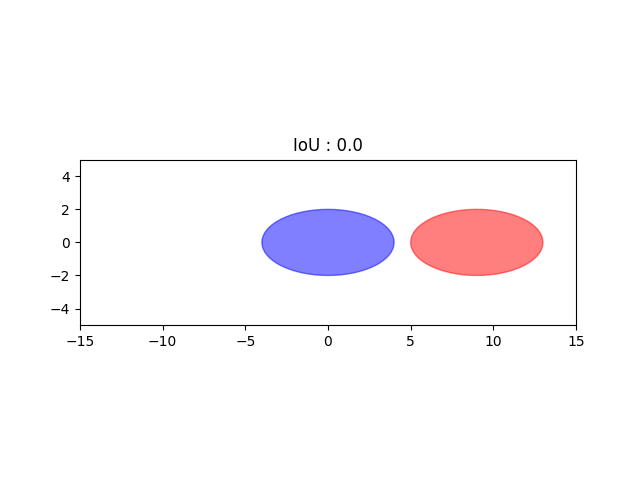
This case arises when the two circles have unequal radii and one circle lies inside the other circle. In order to solve this special case we make the assumption that if the sum of the distance between the two circle and the radius of the smaller circle is greater than the radius of the bigger circle, this means that the overlapping area is equal to the area of the smaller circle. This is shown in the formula below:

if (distance + min(radius\_circle1, radius\_circle2)) < max(radius\_circle1, radius\_circle2):



**References:**

1. <https://matplotlib.org/>
2. <http://mathworld.wolfram.com/Circle-CircleIntersection.html>
3. Ellipse-ellipse overlap IoU:
   1. In order to find the IoU of two ellipses, we made a proof-of-concept study which uses the Shapely library [1] to generate the ellipses to calculate the IoU between them. This library also allows the generation of different geometrical shapes, which can later be used in the research.
   2. The following figures illustrates the concept.

1. From the above figures, it can be deduced that using ellipse as a bounding box can certainly work since the IoU can be found.
2. Inscribed ellipse:
   1. Since the dataset used has rectangular bounding boxes as the ground truth, therefore it becomes necessary to convert these to non-rectangular in order to feed the network and train on them. Our work initially starts with ellipse as the candidate for non-rectangular bounding boxes. Since the datasets that are currently used to train deep neural networks have rectangular bounding boxes, therefore there is need for a dataset which suits our purpose.
   2. Therefore, the approach taken in this research is to convert the existing ground truth annotations to ellipse so that we avoid the tedious task of annotating images manually.
   3. To achieve this objective, the initial work generates an inscribed ellipse inside a rectangle. The following figures illustrate this concept:

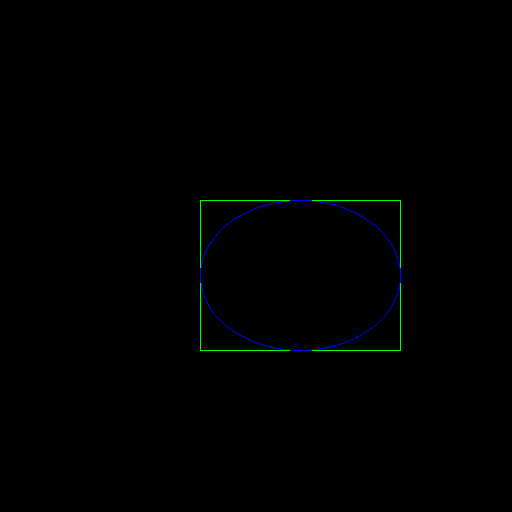
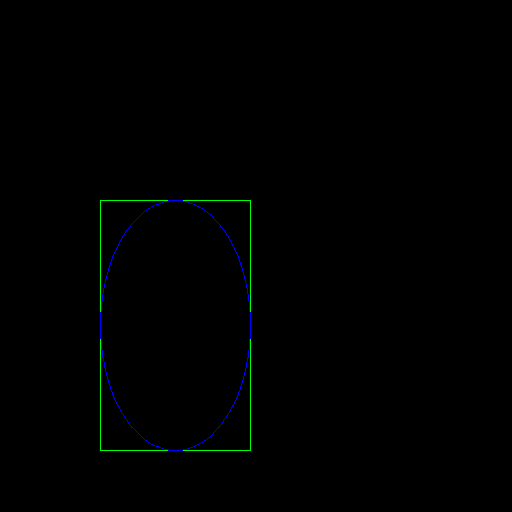


Fig 1: Illustration of inscribed ellipse inside a rectangle

* 1. In order to obtain these inscribed ellipses, the following methodology is used.

We need the origin points, the minor and major axis coordinates to generate the ellipses. The assume that the rectangle uses the following notation to denote the axis points:

(x1, y1)

(x2, y2)

To find the major axis, we use the following formula:

(x1-x2)/2

Similarly for the minor axis:

(y1-y2)/2

To find the origin points, we provide the offset the above generated points:

x\_origin = x1 + (x1-x2)/2

y\_origin = y1 + (y1-y2)/2

References:

[1] https://shapely.readthedocs.io/en/stable/manual.html

**Related work**

Selective search:

Introduced the concept of region proposals in which multiple regions on an image are proposed to get a superficial view of the image. This saves computation as the network does not have to perform an exhaustive search on the whole image.

RCNN:

The authors used selective search method to find the candidate regions and classify them using a Convolutional Neural Network. It also predicts four coordinates for the bounding box in order to perform object localization.

Fast R-CNN

RCNN had drawbacks in that it was slow and very high training times. Therefore in order to optimize speed of the network, researchers came up with the idea of passing the image first from a CNN without generating any regions beforehand. The CNN outputs a feature map, which shows the area of interest. Using these maps, region of proposals are generated and concatenated with other regions to generate an NxN image indicating region of interest (RoI). By using this technique, Fast R-CNN achieved significant speedups as compared to simple RCNN

Faster R-CNN

The above algorithms used selective search for region proposals. The difference from Fast R-CNN is that a separate network learns to perform region proposals. These proposals are reshaped using the RoI pooling layer.

YOLO:

All of the algorithms described previously used region proposals to perform object localization. However in YOLO, we divide the whole image in an NxN grid and generate m bounding boxes in them. The network then predicts the probability value for each of these boxes.

References for related work:

@article{uijlings2013selective,

title={Selective search for object recognition},

author={Uijlings, Jasper RR and Van De Sande, Koen EA and Gevers, Theo and Smeulders, Arnold WM},

journal={International journal of computer vision},

volume={104},

number={2},

pages={154--171},

year={2013},

publisher={Springer}

}

@inproceedings{girshick2015fast,

title={Fast r-cnn},

author={Girshick, Ross},

booktitle={Proceedings of the IEEE international conference on computer vision},

pages={1440--1448},

year={2015}

}

@inproceedings{ren2015faster,

title={Faster r-cnn: Towards real-time object detection with region proposal networks},

author={Ren, Shaoqing and He, Kaiming and Girshick, Ross and Sun, Jian},

booktitle={Advances in neural information processing systems},

pages={91--99},

year={2015}

}

@inproceedings{redmon2016you,

title={You only look once: Unified, real-time object detection},

author={Redmon, Joseph and Divvala, Santosh and Girshick, Ross and Farhadi, Ali},

booktitle={Proceedings of the IEEE conference on computer vision and pattern recognition},

pages={779--788},

year={2016}

}

@inproceedings{girshick2014rich,

title={Rich feature hierarchies for accurate object detection and semantic segmentation},

author={Girshick, Ross and Donahue, Jeff and Darrell, Trevor and Malik, Jitendra},

booktitle={Proceedings of the IEEE conference on computer vision and pattern recognition},

pages={580--587},

year={2014}

}

**Dataset:**

The dataset that will be used for the research is the Pascal VOC. This dataset provides a

@article{everingham2010pascal,

title={The pascal visual object classes (voc) challenge},

author={Everingham, Mark and Van Gool, Luc and Williams, Christopher KI and Winn, John and Zisserman, Andrew},

journal={International journal of computer vision},

volume={88},

number={2},

## pages={303--338},

year={2010},

publisher={Springer}

}

0

1000

800

400

200

Kosten in GE/ZE

10

2000

1800

1600

1400

1200

5

3

25

2

15