**Deep understanding of YOLOv1 and Object detection system based on YOLOv1**

Binfang Ye (@by2034)
Simon Wang (@ssw8641)

**Problem Statement**

Drivers worldwide have been no stranger to tragic accidents of cars running into wild animals on the road. Not only do these accidents severely injure (if not kill) the animals, but also pose a great safety concern to the drivers. While Computer Vision Deep Learning has made significant progress in developing object detection for autonomous driving, additional features can also be implemented so that an object detection system can identify and alert drivers of any animals that appear in front of the vehicle.

**Dataset Description**

Our dataset of choice is Open Images Dataset V6 by Google LLC. This dataset offers a huge number of annotated images and we will focus on using its "Bounding Box" annotations. An annotation CSV file is also provided with useful information including "ImageID", "LabelName", "Confidence", "XMin", "XMax", "YMin", "YMax". During our training process, we will use such information to identify the object(s) within an image and their locations in terms of normalized XY coordinates.

**Description of model**

The YOLO model treats object detection like a regression problem. Every input image is firstly divided into S-by-S grid cells. Each grid will then predict B bounding boxes and these bounding boxes' corresponding x (bounding box center X coordinate), y (bounding box center Y coordinate), w (width of bounding box), h (height of bounding box) and c (confidence). The detection network contains 24 convolutional layers and 2 fully connected layers. The precision will be calculated by IOU(Intersection over Union). The model will eventually produce, for each image input, a tensor of shape $S \times S \times (B * 5 + num_{class})$

**Implementation**

We believe that YOLO (You Only Look Once) can address the problem we proposed. The training input is the annotated images that are collected from Google open image. The input images have features Xmin, Xmax, Ymin, Ymax, and c. We plan to change the features to x, y, w, h, and c (aforementioned). Therefore, the input will be images along with parameters x, y, w, h, and c. In convolution layers, we will alternately use $1 \times 1$convolutional layers(network in network design) to reduce the dimensions. The input will be trained in CNN and output $7 \times 7 \times (2 * 5 * num_{class})$ tensors that include 2 bounding boxes with 5 parameters differently and num classes. The given loss function from Joseph, et al paper will be used to calculate the model loss. We will implement it using Pytorch.

**Project Outcome & Challenges**

We will try to implement an animal-on-road detection feature. By the end of this project, we hope to have successfully reproduced the YOLOv1 object detection algorithm with Pytorch. Our primary goal is to reproduce the YOLOv1 algorithm and we will compare the result with the latest version (v5) if possible. We will use the collected test dataset to test the YOLOv1 model we reproduced. The outcome should show the bounding box that can detect model-trained animals, confidence, and the name of the animal.  After we test our model on the given image test dataset, we will try to test the model using real-word images or stream videos to check how the model performs. The main concern for this project is computational power. Our device will not train the model at a fast speed due to the lack of an excellent GPU and it is hard to get a good accuracy.

**Reference:**


https://storage.googleapis.com/openimages/web/factsfigures.html
Open Images Dataset V6
Google LLC

https://arxiv.org/pdf/1506.02640.pdf
You Only Look Once: Unified, Real-Time Object Detection
Joseph Redmon, Santosh Divvala, Ross Girshick , Ali Farhadi

https://github.com/abeardear/pytorch-YOLO-v1
Guide on YOLO V1 implementation