Computer Vision Project

Maayan Shalom	300312378
Tzabar Nasich	201586864

We used SSD (single shot detection) for our project and Inception v2 network (by google) as the base network for feature extraction.
We implemented the model using TensorFlow and its Object Detection API.

Inception

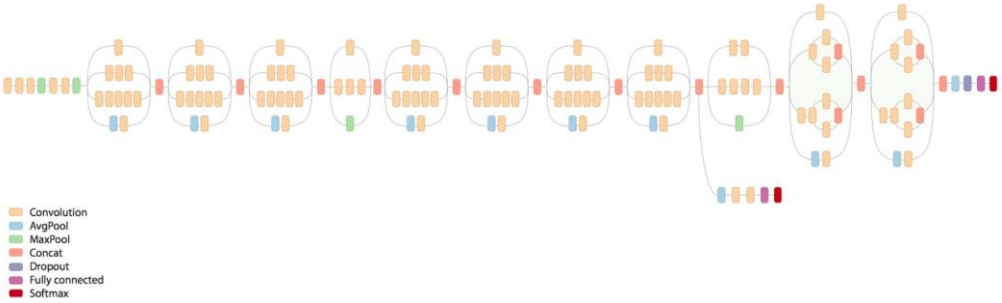


Figure 1: Inception architecture

A very deep neural network for image classification that we use as a base network in the SSD network for feature extraction.

SSD

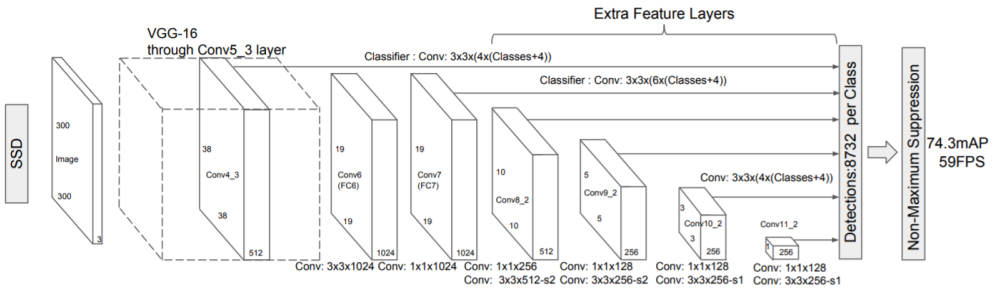
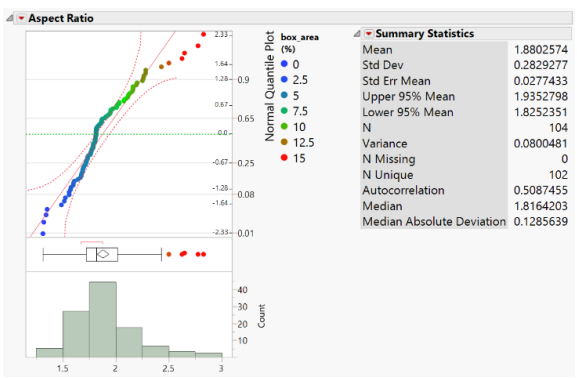


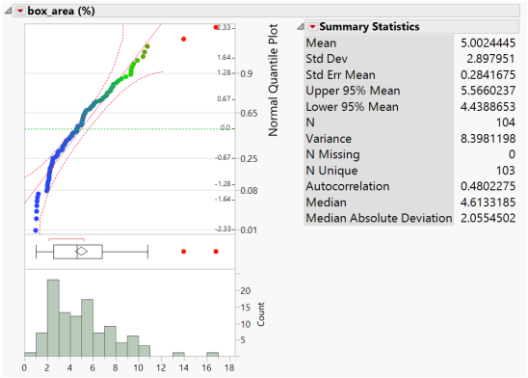
Figure 2: SSD architecture

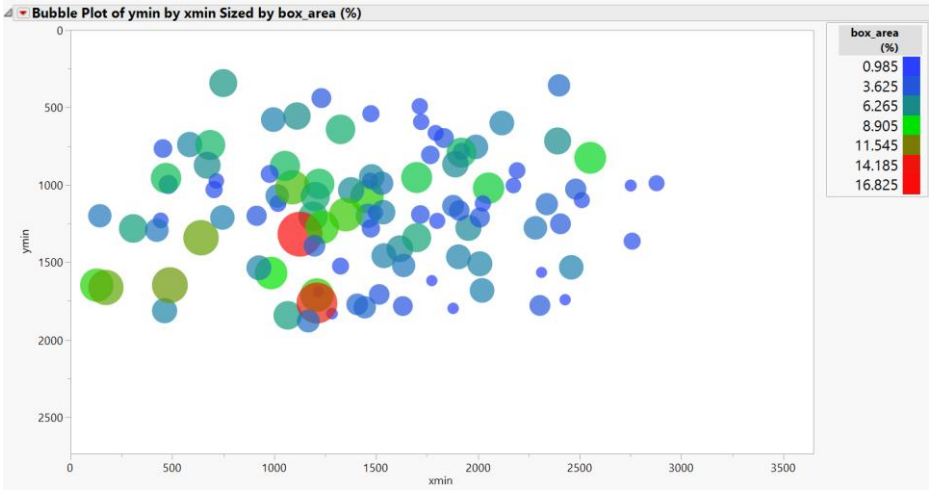
A network used for object detection (classification and localization). On top of the base network (Inception v2) there are 6 more layers, each is smaller than the previous one, which are used to solve the object classification and the regression of the bounding box.
We set the classification layer to detect maximum 6 classes and that no object can be located twice in an image.
We used Cross-Entropy function for the classification loss and L1 norm for the box regression.

Object Analysis

Analyzing the objects features help us with tweaking the network to increase accuracy. We focused on the aspect ratio of the objects (fig. 3 [a]) which defined as width/height, the object area compare to the image and the location of the boxes.







[a] [b] [c]

Figure 3: Analyzing the aspect ratio [a] and box area [b] helped setting the prior boxes to fit our data needs and plotting the object location in the training set [c] help us generating relevant augmented images.

Training

- Image Augmentation

We used image augmentation to extend our training set and to add noisy sample to gain better generalization of the network's classifiers.
We generated augmented images by using 2D affine transformation, crop and resize, dropout boxes, brightness and contrast modification and addition of noise and blur.
After the augmentation process we end up with 600 images.

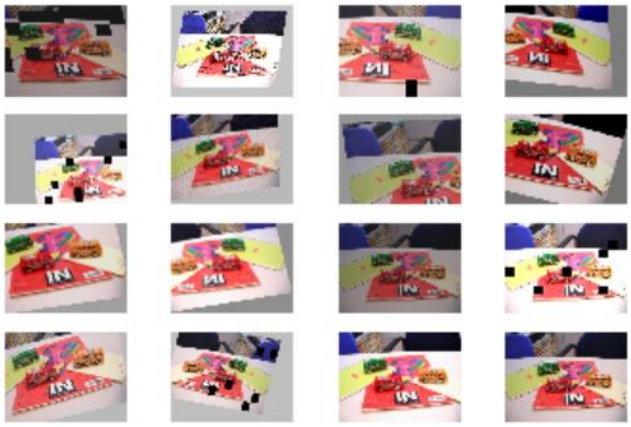


Figure 4: 16 samples of one augmented image. Originally we generated 100 more images.

- RMSprop Optimizer

An optimizer that utilizes the magnitude of recent gradients to normalize the gradients. Keeping a moving average over the root mean squared (hence RMS) gradients, by which we divide the current gradient.

$$E[g^2]_t = 0.9E[g^2]_{t-1} + 0.1g_t^2$$

$$\theta_{t+1} = \theta_t - \frac{\eta}{\sqrt{E[g^2]_t + \epsilon}} g_t$$

Figure 5: RMSprop equations. While g is the gradient of the loss function with respect to the weights.

Evaluation

- K-Fold Cross Validation

We randomly split the images into two sets: training set (90%) and evaluation set (10%). we repeated this process 4 times when in each time we selected different images for the evaluation set. By comparing the results of the different evaluation sets we could measure how well the model is generalizing while we maximize the training set.

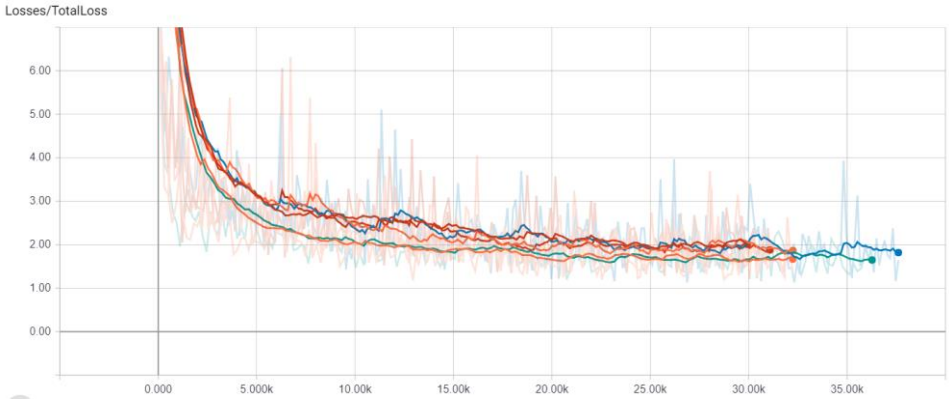


Figure 6: The k-fold train loss converge roughly to the same value

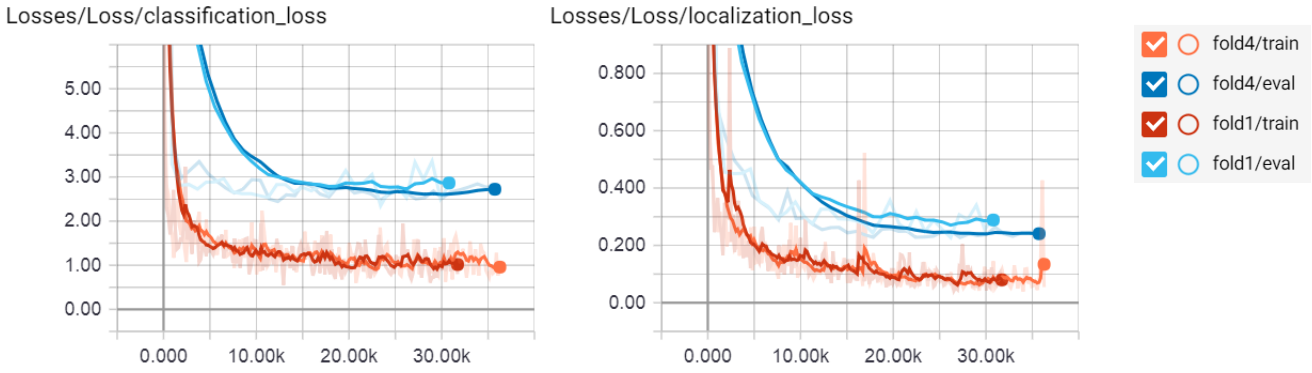


Figure 7: 2 out of 4 folds that were tested. It can be seen that in both cases the train and the evaluation loss functions are similar. Also, there is no sign of over\under fitting.