Image Classification and Object Localization Algorithm Based On Support Vector Machines

Ömer Faruk Karakaya
Department of Computer Science and Engineering
Bilkent University
Ankara, Turkey
faruk.karakaya@ug.bilkent.edu.tr

Musab Erayman
Department of Computer Science and Engineering
Bilkent University
Ankara, Turkey
musab.erayman@ug.bilkent.edu.tr

Abstract*—Computer vision applications commonly do image classification, object detection and localization for image understanding. In order to do these, various machine learning algorithms are used recently by the increasing availability of image datasets and computation power. In order to enhance image understanding we decided to develop a method for image classification and object localization based on Support Vector Machines(SVM) at the high level and feature extraction based on Region-Based Convolutional Neural Networks(R-CNN) like pre-trained ResNet-50 network[1] at the low level. While this approach can be highly accurate and reliable for object recognition and localization on large train datasets, it may be not reliable on relatively small datasets. We apply the proposed method on a subset of ImageNet dataset[2] and obtained encouraging results. The results may be suggestioning further investigation into the benefit of using SVM and R-CNN in image classification and object localization.

Keywords*—computer vision, object recognition, image classification, object localization, R-CNN.

I. Introduction

SVMs are supervised learning models are used and developed for classification commonly. We used SVM model for image classification. We trained the classifier with the training image dataset and their extracted features so that our model would learn the image classes from each classes features so that we would accurately classify unknown images accurately.

There are many approaches for image feature extraction but since R-CNNs give promising results at feature extraction and used commonly, at the low level we used R-CNN like neural network for feature extraction. However, since R-CNN training is taking a lot of time we used pretrained network called ResNet-50.

For object localization we used Edge Boxes method rather than traditional sliding window approach. Among the top 50 edge boxes the most relevant box is selected by scoring candidate 50 boxes with respect to classification success.

We test our classification and localization on a subset of ImageNet dataset with the size of 100. Even though we do not have a train dataset that has more than 500 train image for SVM classifier, we got satisfying results.

In this paper, we propose our algorithm for image classification and object localization and provide outcomes of the method.

II. Background

We used a framework that is similar to the R-CNN model[3] proposed by Girschick et al.

III. Approach

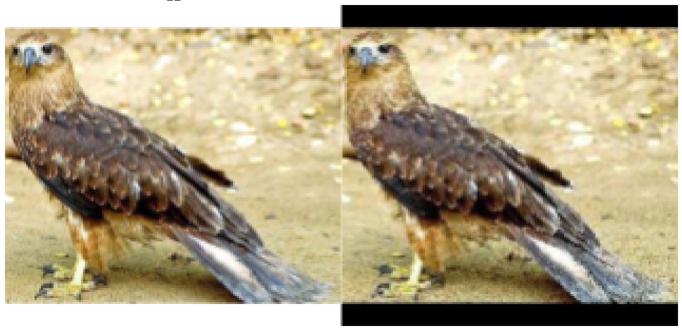
A. Preprocessing

1) Data normalization

In the training set each image had different sizes and color representation. In order to make the images proper input for ResNet-50 network firstly we applied data normalization. We converted each image into RGB color format and reshape the size of 224x224. Since we used PyTorch deep learning framework[3] we did following normalization for each image:

- Divide an image by 255 so that its values are in the range [0,1],
- Subtract 0.485, 0.456, 0.406 from red, green and blue channels, respectively,
- Divide red, green and blue channels by 0.229, 0.224, 0.225, respectively,

as described at [].


ImageSet-1 Before, after padding.

2) Feature extraction

In order to be able to classify images with SVM we extracted features of all training images by using precomputed ResNet-50 network model. We used PyTorch library to use pre-trained network model. After extracting features we normalize our features with l2 norm to make representations more robust.

B. Training the System

We used Support Vector Machines(SVM) from sklearn python library to train classification model. We experiment model with couple different parameters.
Parameter C is the cost of misclassification.
A large C gives you low bias and high variance which make sense because you penalize the cost of misclassification a lot. A small C gives you higher bias and lower variance. In our example number of samples from each group are different for example tiger has 45 images but cat has 35 images and others has 40 images. In this case with low C creates a model which is biased through tigers. Therefore we needed to use high C.
Parameter Kernel is a option to choose gaussian kernel for classification, available options are linear, rbf(Radial Basis Function), polynomial and sigmoid. In background search find out that , RBF kernels are widely used since they can represent a wide variety of classification boundaries from simple, almost-linear models to complex, highly non-linear models.
Parameter Gamma is the parameter of a Gaussian Kernel. Small gamma will give you low bias and high variance while a large gamma will give you higher bias and low variance.
Parameter decision_function_shape has options of ovo and ovr. Option ovo basicly mimics creating binary classifier for each class in multi class classification, therefore we used it.
Most important decision on choosing parameters is finding right balance between C and gamma parameters. We test the code many time with different parameters to tune the model. At the end we use parameters as C = 1000 and gamma = 0.05.

C. Testing the System

1) Extracting candidate windows

In order to obtain candidate windows from test images we used edge boxes method[4]. OpenCV has provides edge boxes implementation at it's ximgproc(Extended Image Processing) library which is not in default packages. We use minBoxArea parameter from createEdgeBoxes method to prune out small boxes which are generally irrelevant. For edge detection we use model supplied by OpenCV which is in same library with edge boxes. Here are the visualizations of edges and 30 most scored edge boxes.

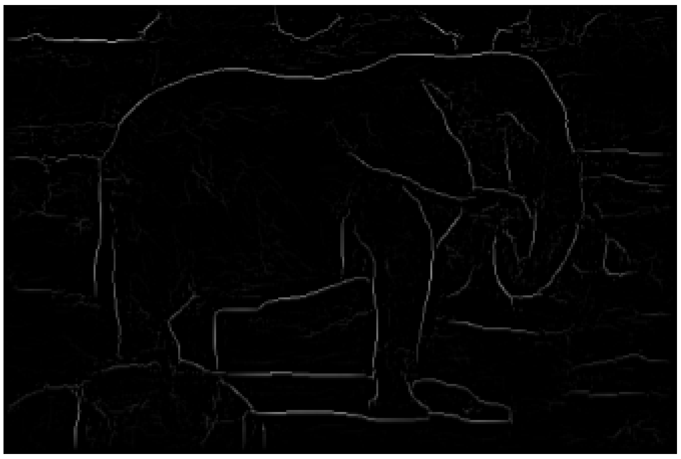


Figure-1 19.png edges

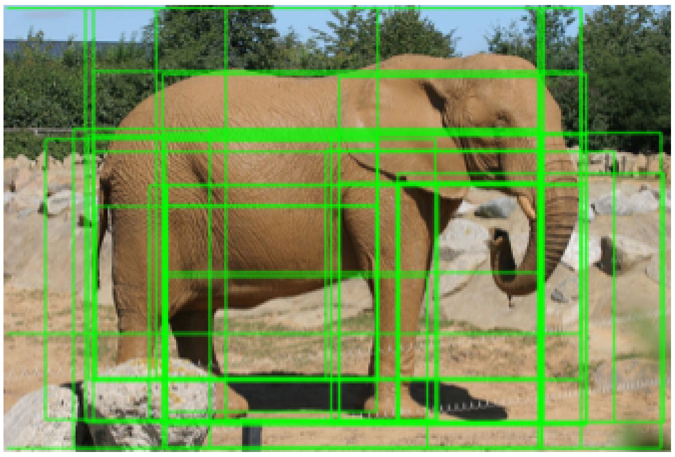


Figure-4 19.png edge boxes

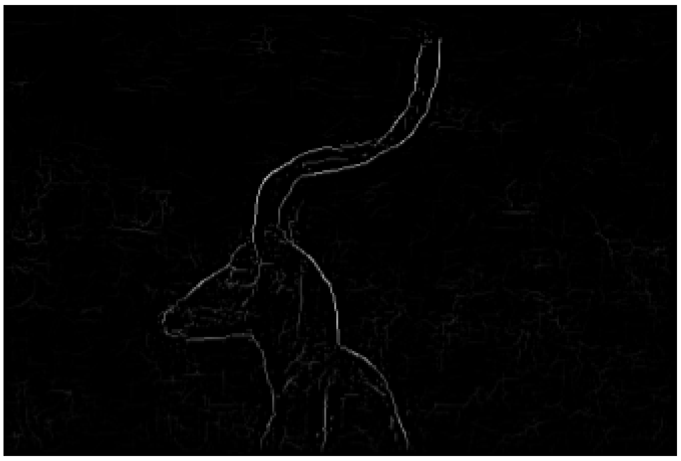


Figure-3 67.png edges

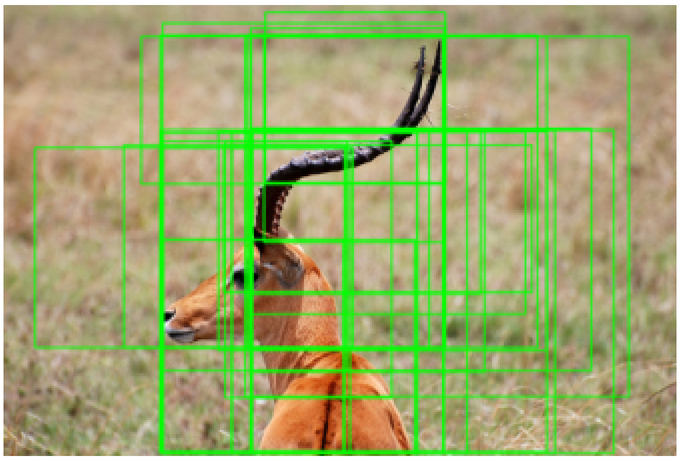


Figure-4 67.png edge boxes

2) Classification and localization

In order to classify images we used SVM model that we train before. For each candidate window that we obtain through edge boxes method, we apply same preprocessing procedure as train images in order to extract features of candidate windows from R-CNN model. Then we obtain scores of candidate windows for each class. Among all 30 windows we find the window that has highest score in any of classes. Then we assign the class that get highest score as a class of this image. Also the edges of this window defines localization of object in this window. Here is an example of localization and classification result, detailed example is supplied on results chapter.

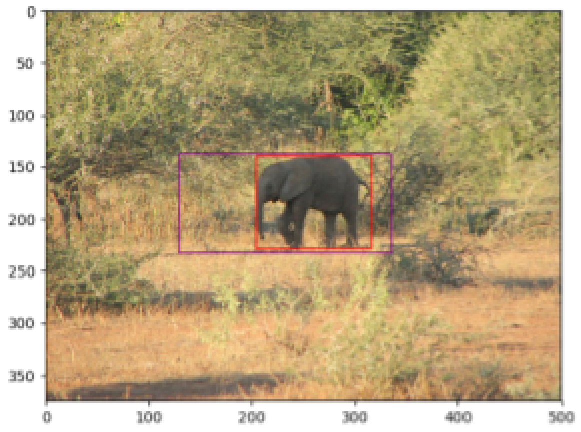


Figure-5 11.png classified truly as an elephant. Purple denotes our localization result and red denotes truth

IV. Results

In this section we provide our classification and localization result. Our correctly classification and localization metric for a test image is correct classification and having more than 0.5 bounding box overlap.

A. Test on Seastar Class

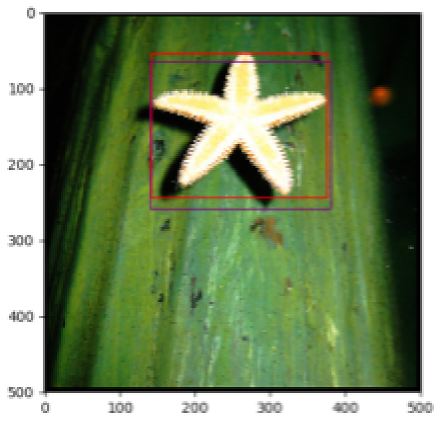


Figure-6 0.png classified truly as an sea-star. Purple denotes our localization result and red denotes truth

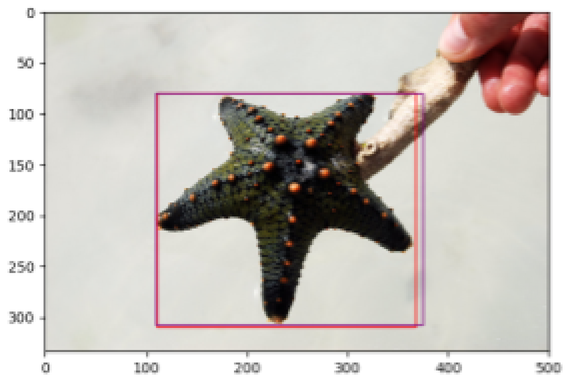


Figure-7 1.png classified truly as an sea-star. Purple denotes our localization result and red denotes truth

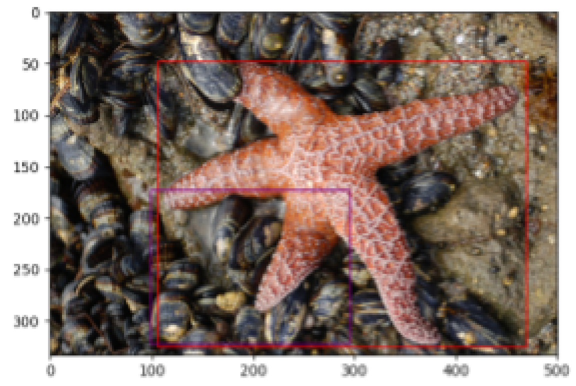


Figure-8 7.png classified false as a cat. Purple denotes our localization result and red denotes truth

In image above there is a noise in background unlike Figure-6, 7. Hence window and class calculated wrong.

B. Test on Elephant Class

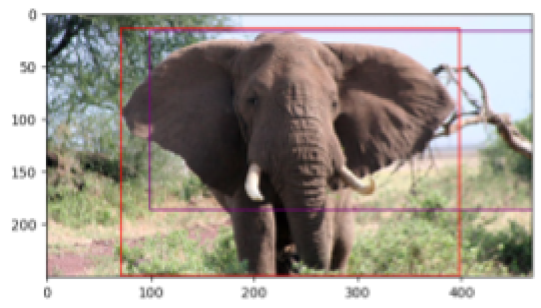


Figure-9 7.png classified true as an elephant. Purple denotes our localization result and red denotes truth.

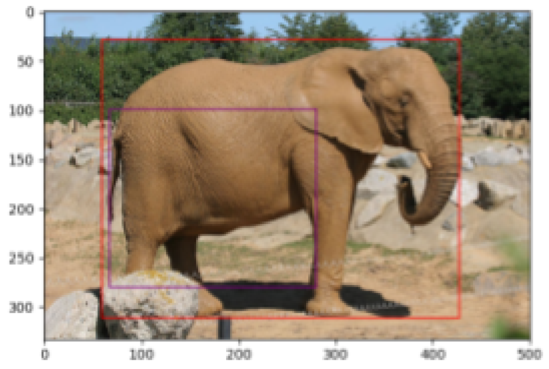


Figure-10 19.png classified false as an buffalo. Purple denotes our localization result and red denotes truth.

C. Test on Zebra Class

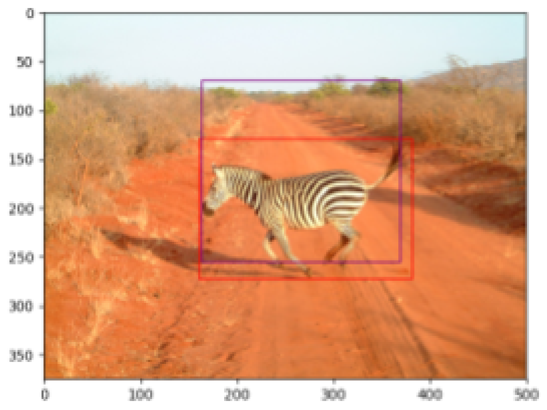


Figure-11 27.png classified true as a zebra. Purple denotes our localization result and red denotes truth

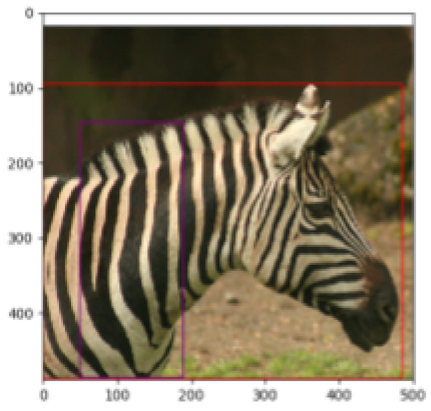


Figure-12 25.png classified false as a tiger. Purple denotes our localization result and red denotes truth

D. Test on Dog Class

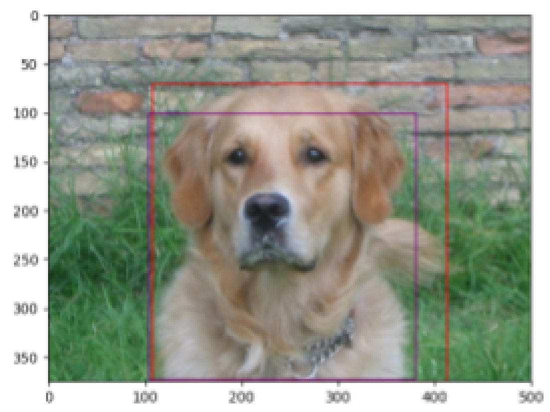


Figure-13 32.png classified true as a dog. Purple denotes our localization result and red denotes truth.

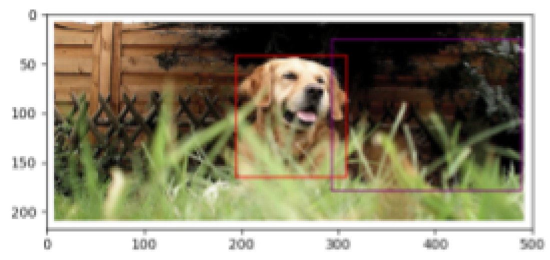


Figure-14 33.png classified false as a buffalo. Purple denotes our localization result and red denotes truth.

In image above also there is a significant noise in background. Hence window and class calculated wrong.

E. Test on Buffalo Class

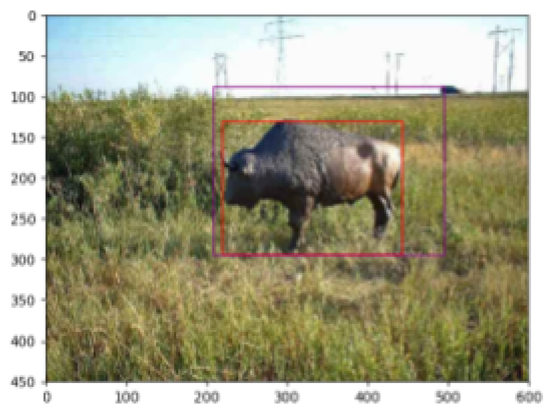


Figure-15 47.png classified true as a buffalo. Purple denotes our localization result and red denotes truth.

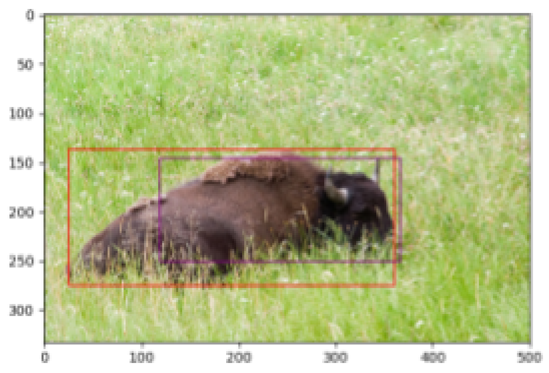


Figure-16 49.png false as a chimpanzee. Purple denotes our localization result and red denotes truth.

In image above only upper part of buffalo is shown. Although window found in good ratio image classified wrong because of occlusion.

F. Test on Chimpanzee Class

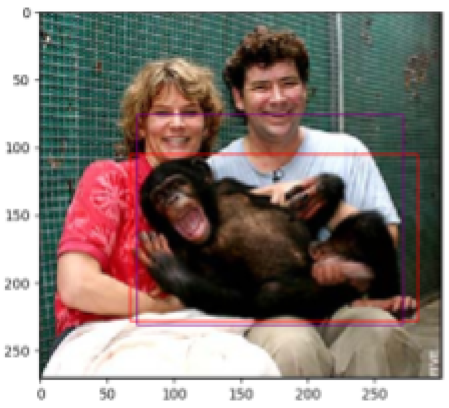


Figure-17 55.png true as a chimpanzee. Purple denotes our localization result and red denotes truth.

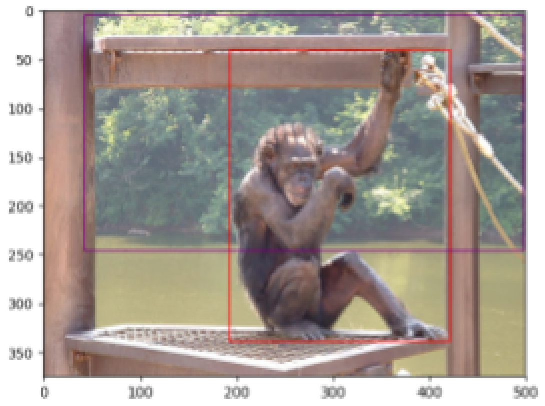


Figure-18 51.png false as an elephant. Purple denotes our localization result and red denotes truth.

In image above there a many other object that creates edges so window could not created well.

G. Test on Deer Class

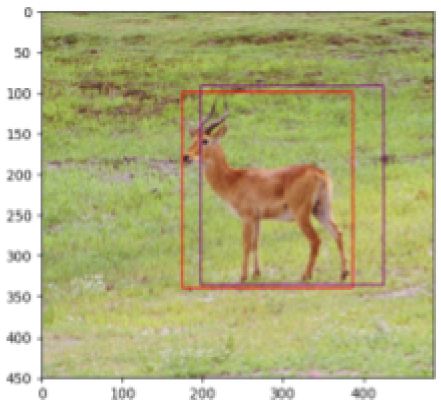


Figure-19 66.png true as a deer. Purple denotes our localization result and red denotes truth.

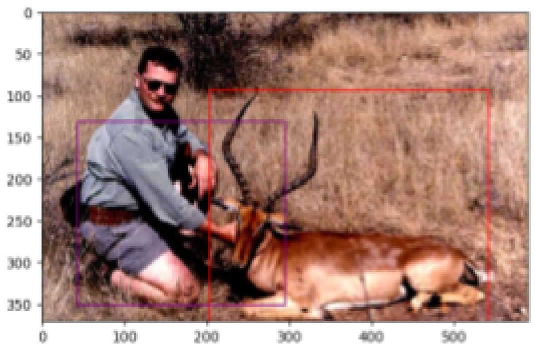


Figure-20 64.png false as a dog. Purple denotes our localization result and red denotes truth.

In image above edge detector find man instead of deer.

H. Test on Cat Class

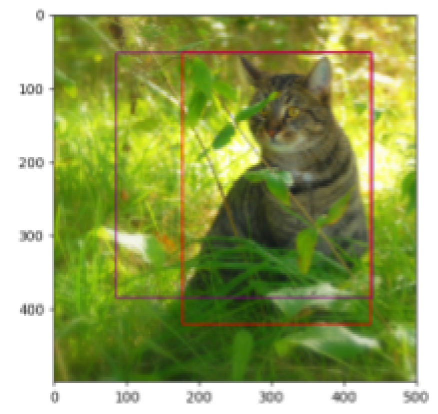


Figure-21 79.png true as a cat. Purple denotes our localization result and red denotes truth.

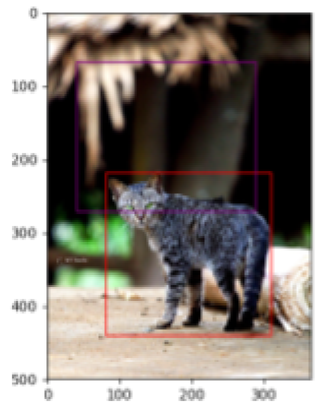


Figure-22 77.png false as a sea-star. Purple denotes our localization result and red denotes truth.

I. Test on Bird Class

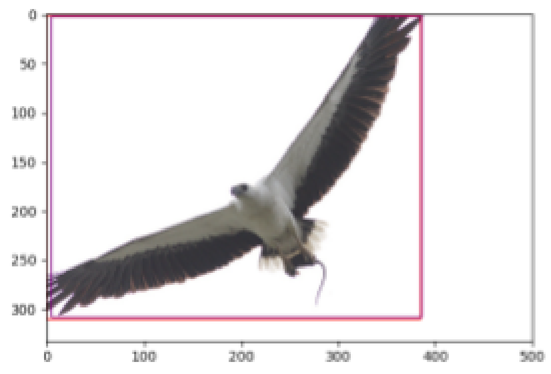


Figure-23 85.png true as a bird. Purple denotes our localization result and red denotes truth.

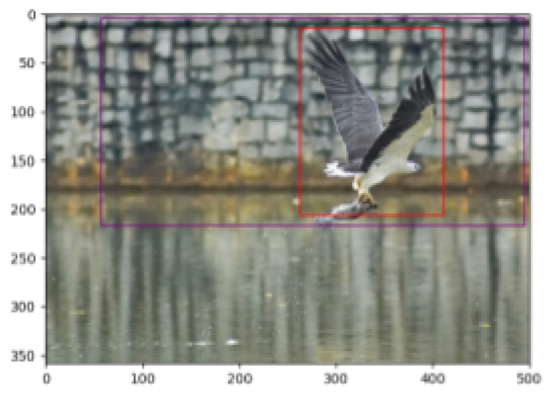


Figure-24 82.png false as a zebra. Purple denotes our localization result and red denotes truth.

J. Test on Tiger Class

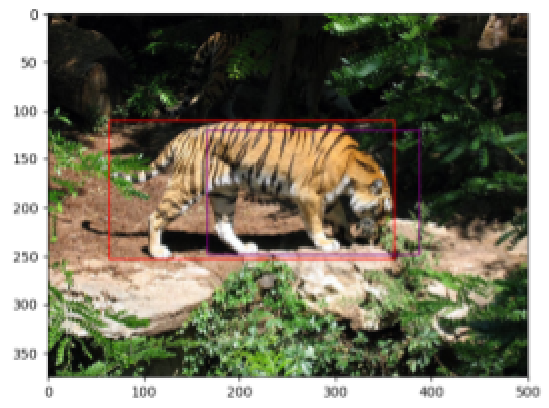


Figure-25 98.png true as a tiger. Purple denotes our localization result and red denotes truth.

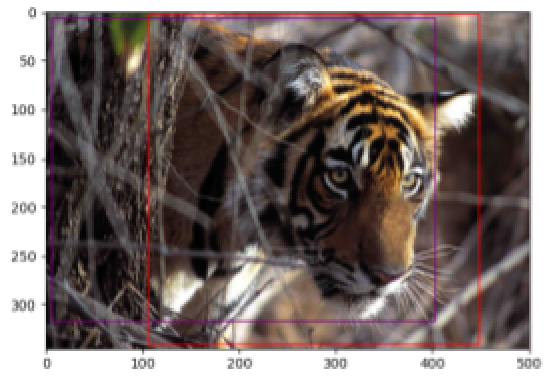


Figure-26 96.png false as a cat. Purple denotes our localization result and red denotes truth.

In image above only head part of tiger is shown and it is pretty similar with head of cats. Since our training data contains more frontal cat faces, this image might be evaluated as cat for that reason.

K. Comparing Results

In this section results from different classes are compared and discussed why they are accurate or not.

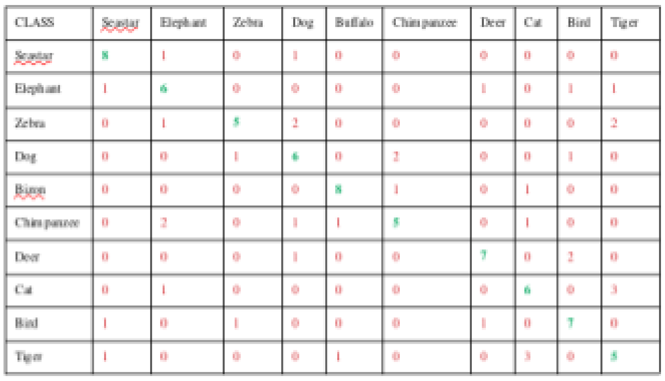
The overall classification accuracy is satisfying even though it was <0.7 because of the fact that our train dataset was not large enough to train a great classifier. Also, the similarity between animals can be one effect, too.

The overall localization accuracy is quite close to the classification accuracy as we expected. The reason behind this may be the fact that we classify the image by classifying the top scored bounding box. So, it is expected that classification accuracy close and similar to the localization accuracy.

Seastar class is the most successful class with respect to the other classes. The reason maybe the variety of training seastar images and the fact that seastars have more unique shape with respect to the other animals.

In tiger class 30% of the images were classified as cat and in cat class 30% of the images were classified as tiger. It can be because tigers and cats look alike to each other. So, the false classifications were not surprising even though correctly localized.

L. Confusion Matrix, Precision and Recall



CLASS	Seastar	Elephant	Zebra	Dog	Buffalo	Chimpanzee	Deer	Cat	Bird	Tiger
Seastar	8	1	0	1	0	0	0	0	0	0
Elephant	1	6	0	0	0	0	1	0	1	1
Zebra	0	1	5	2	0	0	0	0	0	2
Dog	0	0	1	6	0	2	0	0	1	0
Bison	0	0	0	0	8	1	0	1	0	0
Chimpanzee	0	2	0	1	1	5	0	1	0	0
Deer	0	0	0	1	0	0	7	0	2	0
Cat	0	1	0	0	0	0	0	6	0	3
Bird	1	0	1	0	0	0	1	0	7	0
Tiger	1	0	0	0	1	0	0	3	0	5

Figure 27 - Confusion Matrix for Classification

Confusion matrix represents for each class number of classification of respective class. For a confusion matrix M, M[i,j] is the number of images that are belonged to class i and classified class j. Therefore, diagonal gives the correct classifications. Sum of diagonal gives the total number of correct classification.

Total number of correct classification : 63.
Total number of image in the test set: 100.
Classification accuracy: 0.63

Precision and recall are calculated generally using true positives, true negatives, false positives and false negatives in binary classification. So, we calculate precision and recall for each classes. For the most of the classes their precision and recall values are close. For the zebra class recall value is less than its precision. It means classifier is sensitive on zebra classifying. For starfish class, classifier gives importance on finding starfish as it can be seen from Figure 29.

Figure 28 shows the overlap ratios of bounding boxes for each class. Since our success threshold is %50, we can say that all classes are successful at finding bounding boxes except for zebra and chimpanzee classes.

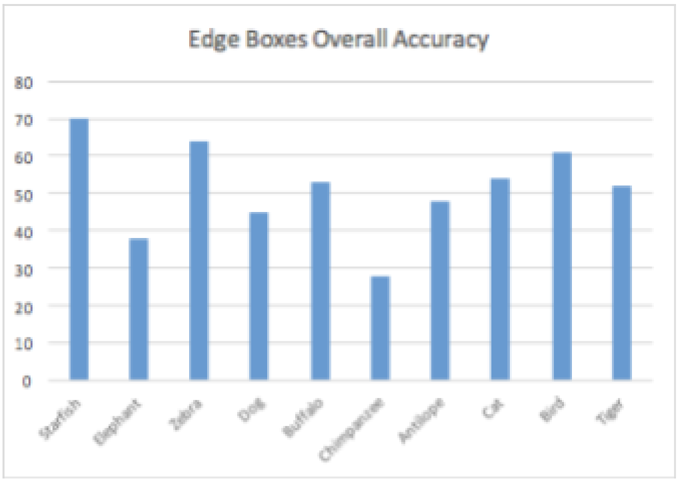


Figure-28 Overall Accuracy of Edge Boxes

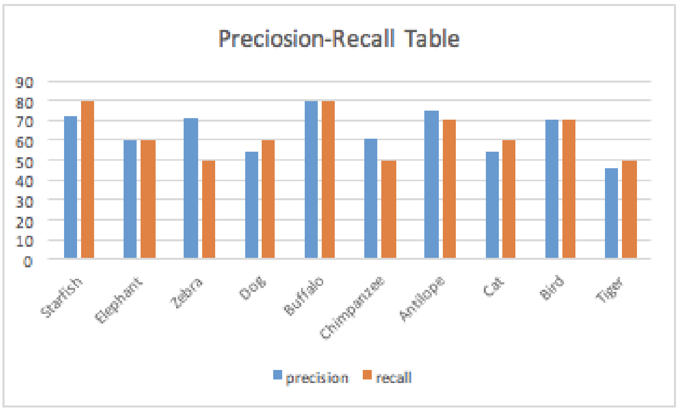


Figure-29 Precision and Recall for Classification

Contributions

Ömer Faruk Karakaya, setted up the working environment and determined python requirements. His main contribution was training the SVM model, optimize classifier and extracting candidate windows. He also helped to preprocessing stage. He contributed to the project report. **Musab Erayman**, prepared IEEE format report template. His main contribution was implementing preprocess functions and evaluation functions. He made statistical calculations and test the system and training. He also helped to optimize parameters of SVM. He contributed to the project report.

References

[1] K. He, X. Zhang, S. Ren, J. Sun, “Deep Residual Learning for Image Recognition,” IEEE Conference on Computer Vision and Pattern Recognition, 770-778, 2016.

[2] “ImageNet,” *ImageNet Large Scale Visual Recognition Competition (ILSVRC)*. [Online]. Available: http://www.image-net.org/. [Accessed: 10-Jan-2019].

[3] “PyTorch,” *PyTorch*. [Online]. Available: https://pytorch.org/. [Accessed: 10-Jan-2019].

[4] C. L. Zitnick, P. Dollar, “Edge boxes: Locating object proposals from edges,” European Conference on Computer Vision (ECCV), pp. 391–405, 2014.