Project Report

CS484

1st Doruk Çakmakçı
Department of Computer Science
Bilkent University
Ankara, Turkey
doruk.cakmakci@ug.bilkent.edu.tr
21502293

2nd İrem Ural
Department of Computer Science
Bilkent University
Ankara, Turkey
yunusolez@gmail.com
21502278

3rd Yunus Ölez
Department of Computer Science
Bilkent University
Ankara, Turkey
iremural@live.com
21401539

***Abstract*—This report is contains the description of the approaches and methodologies in order to implement the project, the results of the implementation, various design decisions and discussion of the results and design decisions.**

***Index Terms*—object classification, object localization, feature extraction, SVM**

I. Introduction

This project is mainly composed of two steps: which are object classification, and object localization. In object classification, the question "what is this object in the image" is investigated, while in object recognition "where is the object" is examined. These procedures are applied on a small sample of images taken from ImageNet. 500 samples are separated into two, 400 of these images are used for training, which will be explained in Section 2.A, while 100 samples are used for testing. There are 10 object classes in the dataset: antelope, bison, cat, chimpanzee, dog, eagle, elephant, tiger, starfish, zebra.

In general, as an implementation procedure, firstly data normalization is applied to each training image in order to have the same format for all images, and in order to avoid some noises to affect the results. Later in order to classify those images, feature extraction method is used by ResNet-50 pretrained convolutional neural network. Secondly, 10 different SVM classifier is used by using One vs All strategy. As a last step, the candidate windows are extracted from test images by using object proposals, and the top 50 scoring object proposals are selected. For each 50 candidate windows, the procedure pipeline is applied and the results are examined in Section IV. These procedures are implemented by using Python and MATLAB.

II. Abbreviations and Acronyms

The list of commonly used acronyms for this report is below:

- SVM: Support Vector Machine
- OVO: One Versus One
- OVA: One Versus All

III. Procedure

The procedure used for this project can be divided to two phases: object localization and object recognition, which are applied on 500 images. As specified earlier, there are 400 train images and they contain only the objects itself.However, in test images, there is the interference of background and other objects. For instance, the test image of chimpanzee is composed of people as well as the chimpanzee itself. Besides, note that the test images contain only a single occurrence of the object of interest. Regarding these differences, the following steps are applied.

Before object classification and localization, the train and test images are subject to feature extraction. After features from the train images are extracted, an SVM per animal type is trained with one versus all approach. Since test set images contains natural scenes, they are preprocessed to extract 50 candidate windows which contains different the object proposals for those images. In the following stage, features are extracted from the candidate windows per image(total of 50 different feature vectors per test image). The extracted features are then classified using all of the SVMs. Overall, the test image is classified as the most frequently occured class for all of its candidate windows. The top scoring window is selected as the window, which is classified as same as the predicted class of the test image, with the maximum score obtained from the object proposal method. Lastly, these procedure steps are analyzed according to two performance evaluation metrics. Firstly, classification accuracy is computed by computing the confusion matrix, precision and recall for each object type and the percentages are presented, Secondly, the object localization results are evaluated by using bounding box overlap ratio above The operations mentioned will be discussed in detail.

A. Preprocessing Training Images

Firstly each image is converted to RGB to avoid some images to have alpha channel. In the dataset there are different sized images, by keeping the aspect ratio of the images same, each image is converted to 224x224 format. In order to achieve this step, by looking at the width, height difference, the padding width is calculated, in this case, it is equal to —width - height— / 2. Then each side of the image, or to the top and

the bottom of the image, zeros are added. As a deep learning framework, PyTorch is used during implementation. There is a specific scheme which requires further data normalization to use PyTorch. Hence, the following steps are used:

- The image content is divided by 255 to have values between [0, 1].
- Extract red, green and blue channels and subtract the respective values: 0.485, 0.456, 0.406.
- Lastly, divide red values by 0.299, green by 0.224 and blue values by 0.225.

B. Feature Extraction

For feature extraction ResNet50, a 50 layer residual, pretrained network, is used. In the past, deep convolutional neural network is used for image classification, though when people try to get better results with deeper networks, it becomes harder and harder to predict the parameters, and it started to reduce the performance. As a solution residual networks are used. In this case, it creates some shortcuts between layers, input of the layer is connected to the output of another layer. By using this network, the performance bottleneck is overcomed. In Python, the pretrained PyTorch model is used in the test mode. Each preprocessed image is given as an input to ResNet50 model and features of these images are extracted.

C. SVM Training

For this project, 10 different SVMs are trained using the extracted features of the training images. In the following stages the trained SVMs are used together to simulate a multi-class SVM. Each SVM is trained using One vs All approach. To make our implementation more intuitive, animal types are enumerated from 0 to 9 with the order: eagle, dog, cat, tiger, starfish, zebra, bison, antilope, monkey and elephant. Note that, before training the SVMs the training set is shuffled. Therefore train set accuracies can be slightly different from run to run.

1) Using Grid Search to optimize parameters: To obtain meaningful results in the following stages of the project, each SVM is optimized with respect to some parameters. These parameters are:

- C: This parameter is the regularization parameter of the SVM. It is used as the penalty coefficient for the slack variables of the objective function of SVM. When C is high, the model is highly punished for misclassifying a datapoint. In this case, the SVM is more prone to overfitting(i.e. fitting the noise to the model). Setting C to a high value generates a hard margin SVM since margin violation is highly penalized. When C is low, the classifier generated becomes a soft margin SVM since the margin violation penalty is low enough so that the classifier tend to misclassify noise so that the resulting classifier is more generalizable. Also note that soft margin SVM works well with non-linearly separable data. The high and low C values are dependent to the training data. The values of C used for optimization are: 0.01, 0.1, 1, 10, 100, 1000.
- Kernel Type: the type of the kernel used for the kernel trick of the SVM. Kernel trick helps to transform the feature space so that the non-linearly separable data become linearly separable. The values used for this project are: linear(no kernel), rbf, polynomial.
- γ: This parameter is used to handle the non-linearly separable data and to determine the shape of the decision boundary . This parameter is important for rbf kernel type. For this project, the gamma values used are: 10, 1, 0.1, 0.001, 0.0001, 0.00001.

With the specified parameters, Grid Search is used to determine the optimal set of parameters per SVM.

D. Extract Candidate Windows From Test Set

In order to extract desired objects from test images, edge boxes [2], [3] algorithm for MATLAB was used. Installing and running the project for the algorithm was rather challenging. The project was last updated for MATLAB R2013, and it gave errors while trying to compile the C code to mex files [4], [5]. After some research, 3 classes were modified to work with the latest version of MATLAB.

After setting up the environment, EdgeBoxesDemo.m and EdgeBoxes.m were examined. It was found that there were 4 different parameters to set to run the algorithm: alpha, beta, minScore and maxBoxes. The last two parameters affected the number of bounding-boxes calculated for an image. maxBoxes determined the maximum number of boxes, which was determined as 50 for this assignment as stated in the project description. minScore, determined a threshold for confidence score for each bounding-box candidate in an image. minScore was set to a low number, 0.01, to ensure that there will be at least 50 bounding-boxes for each image.

The other two parameters, alpha and beta, adjusted the localization of each candidate bounding-box. It was said that, increasing these parameters, in general, improve the outputs while considering intersection over union of bounding-boxes, and these parameters had a default value of 0.70. For alpha, this was true for the corresponding assignment. Therefore, alpha was set to a high value of 0.80. On the other hand increasing beta lowered the differentiation of bounding-boxes, which would give almost fully correct results for some images, but lots of wrong bounding-boxes for some others. The best value for this parameter was found with trials-and-errors and it is observed to be 0.50.

E. Test SVM Classifiers With Extracted Windows

In this section, previously trained SVM models with OVA approach were used to predict the classes of the test images. In the previous step, 50 candidate windows per image have been extracted. In order to predict the label of a test image, first all of the extracted windows belonging to the image have been predicted by all of the 10 classifiers simultaneously. Among all of the predictions of the candidate windows, the image's class has been determined as the most commonly occurring class. Also during the classification phase, the top candidate window is selected as the window with highest score computed with

edge boxes method and has the same predicted class as the image itself.

In order to examine the correctness of the classification, confusion matrices, precision and recall metrics were used. However, each metric is generated for two different evaluation types. First a 10x10 confusion matrix containing all of the classes, precision and recall values are generated. Then, using OVA approach, 10 different confusion matrices for the 10 classes were generated along with their precision and recall values.

F. Object Localization Using The Top Candidate Windows

To determine our accuracy while localizing objects, the most promising bounding-box was compared with the correct bounding-box given in the dataset. By most promising we mean the bounding-box with the correct class classification and with the highest confidence score given from edge boxes algorithm.

To determine the most promising bounding-box, first, all of the 50 bounding-boxes and their x, y, width and height information, order by descending confidence score, for each image were sent to the classification algorithm. The classification algorithm then determined the most promising localization bounding box while classifying each cropped image with the given logic above and wrote every localization information to a txt file.

After the creation of this txt file, calculatelocalizationpercentage file can calculate the overlap ratio between the top candidate and correct localization windows. The algorithm, first extracts the corner locations of both the top candidate and correct localization windows for each image. Since, each of the corner locations of the intersection area have to be a corner of top candidate or correct localization window, at this step, the program has all of the information it needs[1]. It just decides which 4 corners to use, using simple min and max operations. After finding the corner points for the intersection area, it calculates the intersection area and the union area. Finally, if the division of these two areas are greater than 0.5, the program labels the candidate window as successful, else unsuccessful. The algorithm applies these above steps to the top candidate windows for each image.

IV. Results and Evaluation

A. Best Paramaters for SVM models

The best parameter setting per class is found via grid search as mentioned above. The comparison metric to select the best parameters is A sample version of the evaluation of the parameters for animal type eagle can be found in Appendix A.The best parameters can be more than 1 since there are many parameter settings that produce same results however the first setting is selected. The best parameters resulting from grid search per class is below:

- Eagle: 'C': 10, 'gamma': 10.0, 'kernel': 'linear'
- Dog: 'C': 0.01, 'gamma': 10.0, 'kernel': 'poly'
- Cat: 'C': 0.01, 'gamma': 10.0, 'kernel': 'poly'
- Tiger: 'C': 10, 'gamma': 10.0, 'kernel': 'rbf'
- Starfish: 'C': 0.01, 'gamma': 10.0, 'kernel': 'poly'
- Zebra: 'C': 0.01, 'gamma': 10.0, 'kernel': 'poly'
- Bison: 'C': 1, 'gamma': 1.0, 'kernel': 'rbf'
- Antilope: 'C': 1, 'gamma': 1.0, 'kernel': 'poly'
- Monkey: 'C': 1, 'gamma': 10.0, 'kernel': 'linear'
- Elephant: 'C': 1, 'gamma': 1.0, 'kernel': 'rbf'

B. Classification Results of Test Set

The test set accuracy of classification is 97%. Two instances of test set labeled bison are classified as eagle and one instance of tiger is classified as zebra. The performance metrics used for classification are: 10x10 confusion matrix(Table 1), recall versus animal type(Table 2) and precision versus animal type(Table 3). The performance metrics of the OVA version can be found by running "test.py". The results of the OVA version is too massive to put in this paper.

C. Candidate Windows and Localization result

After finding candidate windows and top candidate window is found, they are overlaid on the images to see the classification result and and the best windows per image. Detected candidate windows and top candidate window of selected sample images can be seen in figures: 1, 2, 3, 4, 5, 6, 7, 8, 9 and 10. Our localization accuracy is 50%. The results will be discussed in the discussion section.

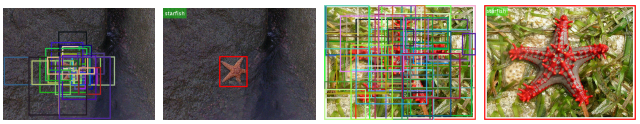

Fig. 1. Sample images classified as starfish correctly and their top candidate window

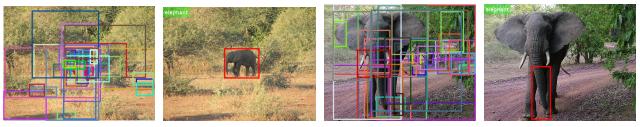

Fig. 2. A sample of images that are correctly classified as elephant. The localization of the image at left is better than the localization of the image at right.

Fig. 3. A sample of images that are correctly classified as zebra. The localization of the image at left is better than the localization of the image at right.

TABLE I
10X10 CONFUSION MATRIX

True Label	eagle	dog	cat	tiger	starfish	zebra	bison	antilope	monkey	elephant	accuracy
eagle	10	0	0	0	0	0	0	0	0	0	100%
dog	0	10	0	0	0	0	0	0	0	0	100%
cat	0	0	10	0	0	0	0	0	0	0	100%
tiger	0	0	9	0	1	0	0	0	0	0	90%
starfish	0	0	0	10	0	0	0	0	0	0	100%
zebra	0	0	0	0	0	10	0	0	0	0	100%
bison	2	0	0	0	0	0	8	0	0	0	80%
antilope	0	0	0	0	0	0	0	10	0	0	100%
monkey	0	0	0	0	0	0	0	0	10	0	100%
elephant	0	0	0	0	0	0	0	0	0	10	100%

TABLE II
RECALL VS ANIMAL TYPE FOR 10X10 CONFUSION MATRIX

Animal Type	eagle	dog	cat	tiger	starfish	zebra	bison	antilope	monkey	elephant
Recall	1.0	1.0	1.0	0.9	1.0	1.0	0.8	1.0	1.0	1.0

TABLE III
PRECISION VS ANIMAL TYPE FOR 10X10 CONFUSION MATRIX

Animal Type	eagle	dog	cat	tiger	starfish	zebra	bison	antilope	monkey	elephant
Precision	0.833	1.0	1.0	1.0	1.0	0.909	1.0	1.0	1.0	1.0

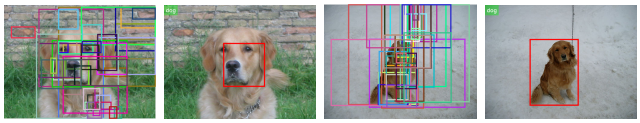

Fig. 4. A sample of images that are correctly classified as dog. The localization of the image at right is better than the localization of the image at left.

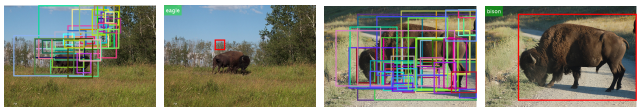

Fig. 5. A sample of images of bison. The left one is classified as eagle and the top candidate window does not contain any part of bison. The right one is classified correctly as bison

V. DISCUSSION

Though these steps seems simple, it is time consuming to get results. When data normalization and feature extraction is applied to all images, it is time consuming to get the

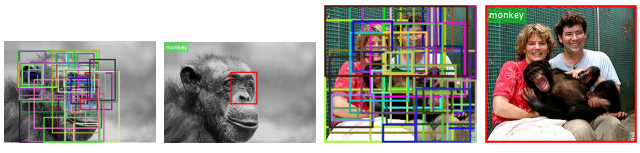

Fig. 6. A sample of images that are classified correctly as monkey. Althought there were better images in terms of localization, these two were worth to mention

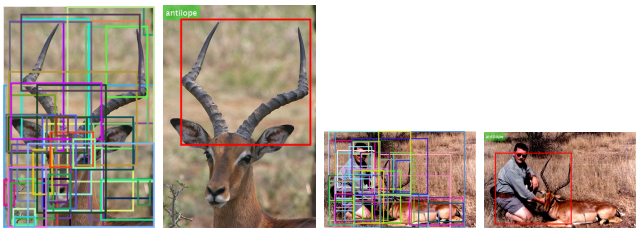

Fig. 7. A sample of images, with different localization accuracies, which are correctly classified as antilope.

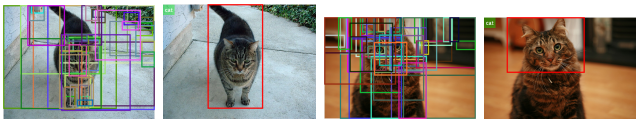

Fig. 8. A sample of images, with different localization accuracies, which are correctly classified as cat.

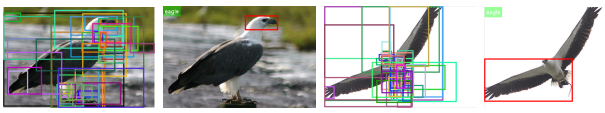

Fig. 9. A sample of two eagle images. Both are classified correctly as eagle images however, the right one has a localization which contains only the head of the eagle. The left one contains the body and the left wing of the eagle but not the right wing(this turned out to be a common localization mistake compared to other images in the test sample)

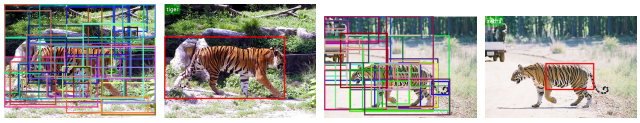

Fig. 10. A sample of two tiger images. The right one is falsely classified as zebra. On the other hand, the left one is classified correctly.

classifier results, it almost takes an hour. As mentioned in the procedure section SVM has 3 parameters: kernel, C and γ. The Grid search showed that there are different combinations that produce the optimal accuracy setting. The first one producing the optimal output is selected as the optimal parameter setting. However, the it is expected that the decision boundary to be non-linear and polynomial and rbf kernels to produce better results than linear kernel. Using the same reasoning, higher gamma values produced better results than lower gamma values. We couldn't find a reasoning to support a statement about regularization parameter values. From our results we couldn't infer a clear value of the threshold value of C which makes SVM change methodologies between hard margin and soft margin. In order to test whether the trained model is biased or not according to the image ordering, we want to shuffle the images. Hence the extracted features for all images are shuffled before using it in classification. Though when the results are obtained, the classification accuracy is diminished to 89%, 11 images are classified wrongly as a result of the SVM classifier. Mostly, it classified elephant wrongly, 4 images were classified as eagle.For the shuffled version, some images were harder to classify. To illustrate, the second image in the test was hard, instead of starfish, the classifier was identifying it as an eagle. However it makes sense, it differs from the general view of the starfish, as the color feature is mostly black, it resembles to eagle. The feathers of the eagle creates a different view in normalized data set and by using shape based features, such as lines and curves etc. the features for this image should be differentiated from eagle in the classifier. Image 46 has a bison, which is a photograph from its back. Therefore the color and the shape of the bison resembles a lot to the elephant.

Using trial and error methodology, we have found out that if the training set is not shuffled, the multi-class classifier produces better outputs(97% accuracy). Hence, we decided not to shuffle the training dataset. The confusion matrix in I shows that classifier only misclassified two instances of bison as eagle and one instance of tiger as zebra. As shown in 10, with the help of the lighting of the background, the texture of the tiger appears similar to black and white. Even our eyes can find a resemblance between the texture in the image and the texture of a typical zebra. It is reasonable for classifier to misclassify this case. Also, when the localization output of the misclassified tiger image is examined, it is seen that the top candidate window appears to contain most of the texture on the tiger that seems to be black and white due to light reflection. On the other hand, our classifier misclassifies two bison instances in like manner. In their localization outputs, no part of the bison occurs and the intersection of top candidate window and the bounding box given at the provided txt file is 0%. In both cases, the top candidate window encloses a grassy area with high texture change. We concluded that at the grayscale conversion and normalization stages of the pipeline,the images both the feathers of the eagle and the high gradient background produce similar images. Therefore, resnet50 may have generated similar features for the both images. The misclassification mentioned can be seen in 5. The true classification of the other images does not show that our project works perfect. On the other hand we thought there may be a problem(maybe overfitting but then the classifier would not be able to do well with the test dataset). Then we analyzed and concluded that the classifier works as intended. Nevertheless when the localization results are analyzed, we found out that the ratio of the images that has overlap ratio between the provided bounding box and the calculated top candidate window, over 50% were only the 50% of all the images. We analyzed the results and cross checked the similar counterparts of the object of interest present in the localizations.

For the images classified as starfish, we found out that the most of the top candidate windows(9 out of 10) contained the body of the starfish(the merging point of the arms of the starfish). Also in most of the cases, the dimensions of the starfish is captured correctly however, in some cases we saw that the arms of the starfish is divided from their ends. We concluded that identifying starfish was in fact easier than other animal types. Also, in most of the test images, the color of the starfish was significantly different than the background. This fact helped to classify the find starfish in the images more easily. But, in image 9 there are some rocks with a similar color to the starfish and the computed top window also included those.

For the images classified as elephant, almost always the top candidate windows contained all of the elephant. In some cases the found localization was nearly the minimum bounding box. Mostly, the dimensions of the elephant were captured successfully(with some noisy counterparts of course). For image 16, the top candidate window contained only the trunk of the elephant even though the some of the candidate windows contained most of the elephant. We couldn't find a good rationale behind this selection. Our conclusion is that according to the features, trunk of elephant was a good indicator. Compared to other animal types, it can be inferred from our results that elephant was rather easy to detect.

For the images classified as zebra, when we thought of the misclassified tiger image, the high gradient in horizontal direction and black and white colors had influence for the classifier to predict zebra. However, for the image 21, the top candidate window only contains the legs of the zebra, we couldn't find a meaningful explanation. Also, for zebra case we can state that even if only the face of the zebra is present in the image, the pipeline could detect it(Image 22). But in some images, particularly image 25, the top candidate window did not contain the face of the zebra. Therefore, there

are some features extracted, which possibly are no perceivable by human eye or we could not perceive it. Compared to other animals, zebra has a unique coloring, except a tiger under sunlight. Classifying zebra was not harder than other animal types.

For the images classified as dog, we can say that the nose and tail of a dog is an important feature for classification. For some of the images, the top candidate window contained the neighboring pixels of the nose of a dog. Also, other than the images mentioned, the dimensions of the dog in the image is captured successfully. Nevertheless, for image 33, the top candidate window contained nearly all of the original image. We think this noisy output is due to the interference of the grass in the foreground and the color of the background. We think that classifying dog was as easy as starfish and elephant.

For the images classified as bison, we think that the horns of the bison create an important contrast compared to its fur. This feature may be important for classifying an image as bison. Top candidate windows for this animal types, most of the time, captures the dimensions of the bison correctly. In the two bison images that are identified as eagle, the background clutter was effective. We think that comparing bison is hard because of its dark color however its white horns, and its different shaped nose make improve the classification rate.

For the images identified as monkey, monkey has an explicit nose and ears like humans. Therefore in 6, it can be seen that humans are also identified as monkeys. The facial representation of the monkeys, a nose between two eyes, is a good separator for the monkeys. Also from another point of view, the some of the monkeys have a high contrast in their faces, between their skin and fur. This may help to classify the monkeys. We think that monkeys are easy to identify because of their facial difference compared with the animals. If a classification is made among humans and monkeys, it could not be easy.

For the images classified as antilope, we think that the large horns is a perfect indicator. In fact in image 60, only the horns of the antilope are selected as the top candidate window. For antilopes without horns, their long facial structure and the contrast on their face is a good feature for classification. For image 64, the top candidate window also includes the man at the left of the antilope, we couldn't resolve why this is the case. For image 68, the top candidate window only contains the legs of the antilope. We think that the base classifier for zebra also identified this candidate window as zebra but the vote of the majority of the candidate windows turned out to be antilope. We think that since there are lots of interesting features of an antilope, it is easy to classify an antilope as antilope.

For the images classified as cat, we think that the results captured the cat as the ears, the fur with high gradient, eyes and nose are the good identificator features. The top candidate windows capture the dimensions of the cat well. We think that due to the features mentioned, cat, as an animal type is easy to identify.

For the images classified as eagle, the spreaded wings, the beak, the eyes, the inharmonious directions of the feathers are good identificators. In image 82, the top window candidate contains most of the background. We think that this fact is due to the color of the background. Since the images in the training set are restricted to some extent, the top candidate window for image 89 only contains the white region on the eagle. We think that identifying eagle is easy due to the features mentioned. However, we see that some background clutter in the bison images are identified as eagle, as mentioned above. This is a consequence of the gradient of the feathers or the object proposals.

For the images identified as tiger, we think that the orange and black stripes, the white regions on the face of the tigers and the white region in the stomach region of tigers are good identificators. However, for this test set, the lighting of the environment can lead to a tiger being classified as a zebra. We think that identifying tiger is not easy for this project dur to it's similarities with zebra.

When the animal types and their classification rates are concerned, we see that bison, tiger, eagle and zebra are harder than the other types. Bison's colors are dark therefore its hard to find feature. Tiger and eagle can be mismatched with each other under certain lighting conditions. Eagle has a high gradient components which leads to a misclassification with background clutter. However we can say that the background is important for classifying an animal. It is hard for an animal to be classified correctly when it is present in an environment with similar textures and color histograms. Camouflage is hard to detect.

To improve this project, we can of course train the classifiers with massive training datasets to improve their generalizability. Also training with massive datasets can also lead to a change in the C, γ and kernel type parameters since the decision boundary can be made more clear compared to its current version. From another perspective, object proposals can be improved, candidate window detection can be tried with other parameter values to check if we can get a better candidate windows. The maximum candidate window count can be increased. However, this increase can also lead to a deterioration of the localization performance. There is always a counterpart of the project to be improved.

As a note to SVM implementation, sklearn is used to generate the confusion matrix. Also, some code snippets from sklearn's documentation is used for SVM and Grid Search Implementation [1], [6]. They are also cited in the source code.

APPENDIX A

Note that the output is shortened by discarding some parameter settings.

Tuning hyper-parameters for class 1

Best parameters set found on development set:

'C': 10, 'gamma': 10.0, 'kernel': 'linear'

Grid scores on development set:

0.899 (+/-0.001) for 'C': 0.01, 'gamma': 10.0, 'kernel': 'rbf'
0.899 (+/-0.001) for 'C': 0.01, 'gamma': 10.0, 'kernel': 'linear'
0.992 (+/-0.012) for 'C': 0.1, 'gamma': 10.0, 'kernel': 'poly'
0.899 (+/-0.001) for 'C': 0.1, 'gamma': 1.0, 'kernel': 'rbf'
.....
0.899 (+/-0.001) for 'C': 1, 'gamma': 0.001, 'kernel': 'poly'
0.899 (+/-0.001) for 'C': 1, 'gamma': 0.0001, 'kernel': 'rbf'
0.990 (+/-0.010) for 'C': 1, 'gamma': 0.0001, 'kernel': 'linear'
0.899 (+/-0.001) for 'C': 1, 'gamma': 0.0001, 'kernel': 'poly'
0.920 (+/-0.030) for 'C': 10, 'gamma': 10.0, 'kernel': 'rbf'
0.995 (+/-0.012) for 'C': 10, 'gamma': 10.0, 'kernel': 'linear'
.....
0.899 (+/-0.001) for 'C': 10, 'gamma': 0.0001, 'kernel': 'poly'
0.920 (+/-0.030) for 'C': 100, 'gamma': 10.0, 'kernel': 'rbf'
0.995 (+/-0.012) for 'C': 100, 'gamma': 10.0, 'kernel': 'linear'
.....
0.899 (+/-0.001) for 'C': 1000, 'gamma': 0.0001, 'kernel': 'poly'

APPENDIX B

- Doruk: Mostly worked on training and testing SVMs. Helped to preprocess the images. Discussed SVMs, overall results and helped to wrote the overall discussion. Also, formatted the report to IEEE format.
- Irem: Mostly worked on preprocessing images and latter she worked on bounding box extraction using the edgeBoxes method and calculating the bounding box overlap ratio.
- Yunus: Worked on setting the environment for and implementation of the edgeBoxes algorithm and calculation of the bounding-box overlap ratio.He I also generated sample images for these two algorithms and discussed them overall the report.

REFERENCES

[1] Confusion matrix 1.4. Support Vector Machines scikit-learn 0.19.2 documentation. [Online]. Available: https://scikit-learn.org/stable/auto_examples/model_selection/plot_confusion_matrix.html. [Accessed: 10-Jan-2019].

[2] GitHub. (2019). pdollar/edges. [online] Available at: https://github.com/pdollar/edges [Accessed 10 Jan. 2019].

[3] GitHub. (2019). pdollar/toolbox. [online] Available at: https://github.com/pdollar/toolbox [Accessed 10 Jan. 2019].

[4] GitHub. (2019). Compile Mex Code returns error · Issue #21 · pdollar/edges. [online] Available at: https://github.com/pdollar/edges/issues/21 [Accessed 10 Jan. 2019].

[5] Pdollar.github.io. (2019). [online] Available at: https://pdollar.github.io/files/papers/ZitnickDollarECCV14edgeBoxes.pdf [Accessed 10 Jan. 2019].

[6] sklearn.grid_search.GridSearchCV, 1.4. Support Vector Machines - scikit-learn 0.19.2 documentation. [Online]. Available: https://scikit-learn.org/0.16/modules/generated/sklearn.grid_search.GridSearchCV.html. [Accessed: 10-Jan-2019].