ENSE 885AY

Application of Deep Learning in Computer Vision

Assignment A05

Face detection with a sliding window

Instructed by

Dr. Kin-Choong Yow

Student:

Marzieh Zamani Alavijeh

1. Introduction

1.1. Overview (Key points from the assignment description) [1]

Assignment Subject:
Face detection with a sliding window

Assignment objectives:

Face detection using sliding window and HOG features

Steps to local feature matching between two images (image1 & image 2):

1. Extracting features:
   - ⇨ get_positive_features()
   - ⇨ get_random_negative_features()
2. Mining hard negatives:
   - ⇨ mine_hard_negs()
3. Train a linear classifier:
   - ⇨ train_classifier()
4. Detect faces on the test set:
   - ⇨ run_detector()

2. Student Code

2.1. Extracting features using get_positive_features()

Algorithm of get_positive_features() [1]:

feats = get_positive_features(train_path_pos, feature_params)

1. Initialize feature matrix (feats)

for each image path in {image _paths}:

2. Load image as grayscale
3. Obtain a mirrored copy of image using cv2.flip()

4. Obtain HOG features of regular image and mirrored image using vlfeat.hog.hog() with below parameters
5. Flatten and save HOG features as individual rows in “feats” matrix

Free parameters and other choices:

- win_size = 36
- cell_size = 6
- Source images:
    - Normal images
    - Flipped images
    - Warpped images

2.2. Extracting features using get_random_negative_features ()

Algorithm of get_random_negative_features() [1]:

feats = get_random_negative_features(non_face_scn_path, feature_params, num_samples)

1. Initialize feature matrix (feats)
2. Define list of scale values (scale_list)

for each image path in {image_paths}:

3. Load image as grayscale

for each image path in {image_paths}:

4. Resize image according to current scale
5. Divide the scaled image to win_size by win_size patches

for each patch:

6. Obtain HOG features of the patch using vlfeat.hog.hog() with below parameters
7. Flatten and save HOG features as individual rows in “feats” matrix

Free parameters and other choices:

- win_size = 36
- cell_size = 6
- Source images: Scaled images

- num_samples = 10000 (default)

2.3. Mining hard negatives using mine_hard_negs()

Algorithm of mine_hard_negs() [1]:

feats = mine_hard_negs(non_face_scn_path, svm, feature_params)

1. Initialize feature matrix (feats)
2. Define list of scale values (scale_list)

for each image path in {image_paths}:

3. Load image as grayscale

for each image path in {image_paths}:

4. Resize image according to current scale
5. Divide the scaled image to win_size by win_size patches

for each patch:

6. Obtain HOG features of the patch using vlfeat.hog.hog() with following parameters:
    a. win_size = 36
    b. cell_size = 6
7. Predict the label of HOG features of patch

If patch label == 1

8. Flatten and save HOG features as individual rows in "feats" matrix

2.4. Train a linear classifier using train_classifier()

Algorithm of train_classifier() [1]:

svm = train_classifier(features_pos, features_neg, C)

1. Construct x_train by vertically stacking positive and negative features
2. Construct y_train by setting positive labels to 1 and negative labels to -1
3. Initialize classifier as LinearSVC(C)

4. Fit classifier to train data
5. Save classifier as svm

Free parameters and other choices:

- svm = LinearSVC(C = 0.01)

Free parameters and other choices:

- win_size = 36
- cell_size = 6
- Source images: Scaled images

2.5. Detect faces on the test set using run_detector()

Algorithm of run_detector() [1]:

bboxes, confidences, image_ids = run_detector(test_scn_path, svm, feature_params, verbose=False)

1. Initialize matrices for bboxes, confidences, image_ids
2. Initialize matrices for cur_bboxes, cur_confidences
3. Define list of scale values (scale_list)
4. Define number of top detections to feed to NMS (topk)
5. Define step size
6. Define decision threshold

for each image path in {image_paths}:

7. Load image as grayscale
8. Resize image according to current scale
9. Obtain HOG features of the scaled image using vlfeat.hog.hog()
10. Divide the HOG features matrix to template_size by template_size patches

for each patch:

11. Obtain the confidence value using trained svm()

if patch confidence > decision threshold

12. Append the confidence of accepted patch to "cur_bboxes"

13. Append the corresponding corners of accepted patch to “cur_ confidences”
14. Pass total confidence values and bounding boxes of each image to non_max_suppression_bbox() to remove duplicate detections

Free parameters and other choices:

- win_size = 36
- cell_size = 6
- Source images: Scaled images
- Scale values =
- topk = 50
- decision_thres = 0
- step_size = 1

3. Experiment Design

3.1. Experiment A ('cell_size' = 6) and B ('cell_size' = 4): SVM Classification Results vs. Dataset, Negative features, SVM lambda , and Detection Scale

Experiment A & B summary:

- **Train Dataset #1:**
  - Positive feature: HOG features extracted from original face dataset
  - Negative feature: HOG features extracted from original non-face dataset
  - Hard Negative feature: HOG features extracted from original non-face dataset
  - Test dataset: HOG features extracted from original test dataset
- **Train Dataset #2 (augmented):**
  - Positive feature: HOG features extracted from original face dataset + flipped images + warped images (@ 10, 20, and 30 deg.)
  - Negative feature: HOG features extracted from original non-face dataset + scaled images (@ scale = [0.8, 0.65])
  - Hard Negative feature: HOG features extracted from original non-face dataset + scaled images (@ scale = [0.8, 0.65])
- **Test Dataset:**
  - Test dataset: HOG features extracted from original test dataset
- **Varying parameters:**
  - Dataset #1 | #2

    - SVM lambda = [0.0001, 0.001, 0.01, 0.05, 0.1, 0.5, 1, 2.5]
    - Negative features: SVM_1 (Regular negative features) | SVM_2 (Hard negative features)
    - Detection scale: single-scale | multi-scale (scale = [0.8, 0.65, 0.5, 0.3, 0.25])
- **Total number of experiments:**
    - 2 (datasets) * 8 (SVM lambda) * 2 (neg. feats.) * 2 (detection scale) = 64
- **Results:**
    - HOG templates vs. dataset and SVM lambda
    - Training results vs. dataset and SVM lambda
    - Classification Results vs. Dataset, Negative features, SVM lambda , and Detection Scale

4. Results and Discussion

4.1. Tuned SVM Classification Results: 'cell_size' = 4, Augmented dataset, SVM lambda = 0.05, and Multi-scale detection

Table 1: Training results

Trained HOG template	Training results

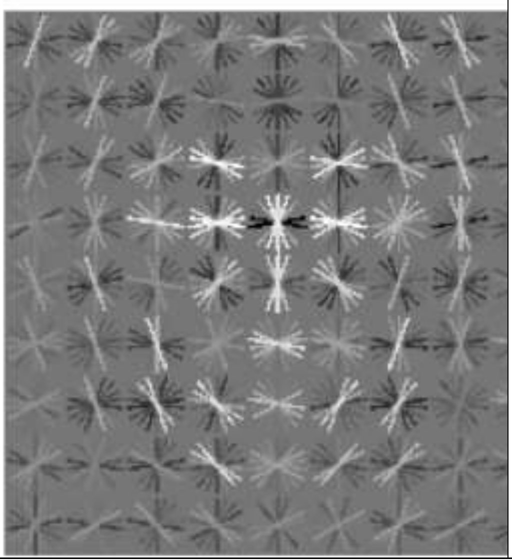
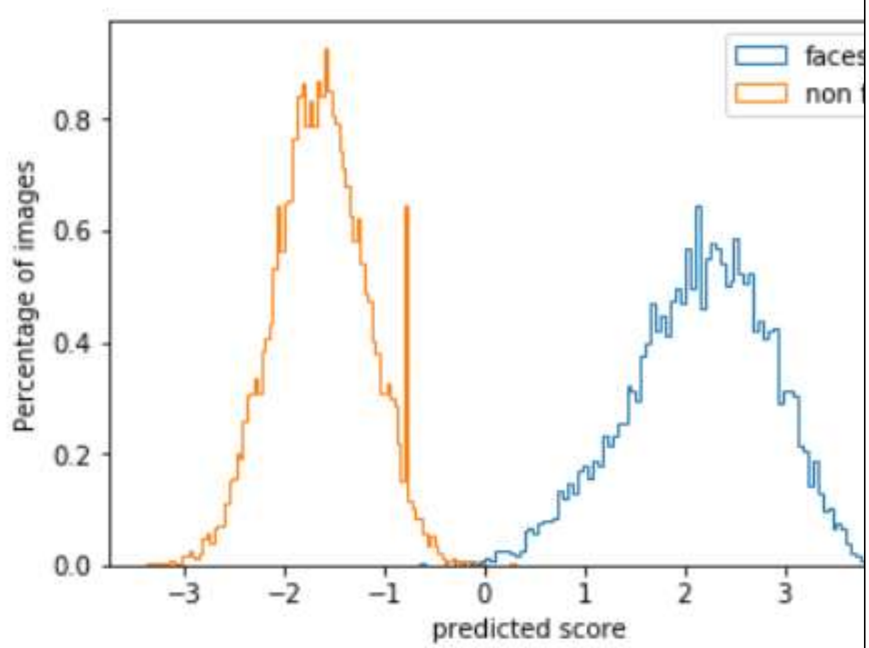
<table>
<tr>
<td>
</td>
<td>Accuracy = 99.659%
True Positive rate = 99.359%
False Positive rate = 0.140%
True Negative rate = 99.860%
False Negative rate = 0.641%

</td>
</tr>
</table>

Table 2: Precision-Recall Curve

Precision-Recall Curve	Matching fig. to fig. 6 in Viola-Jones

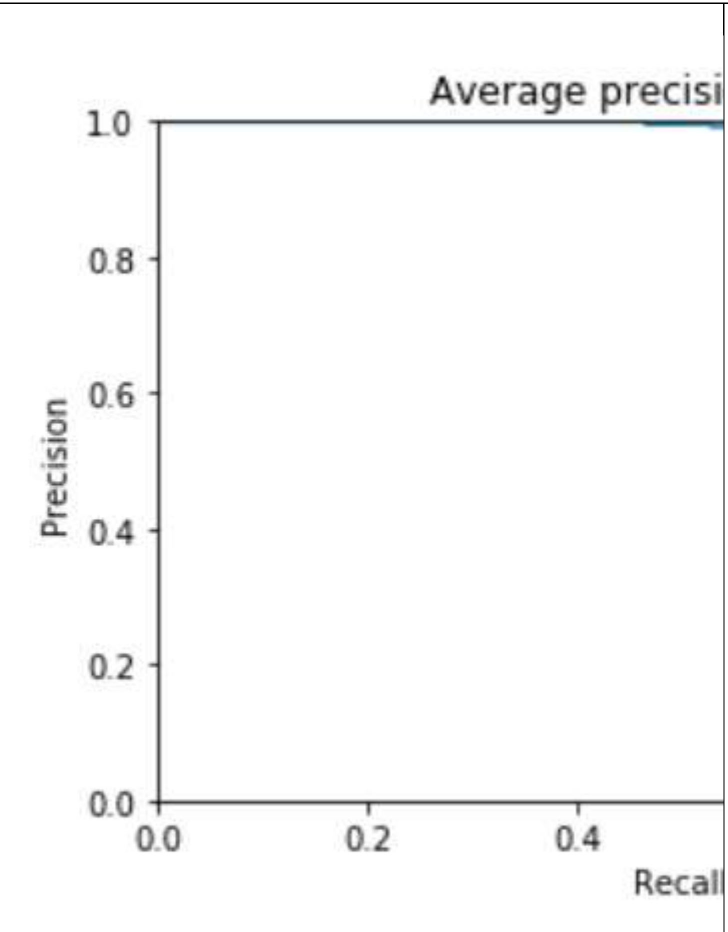
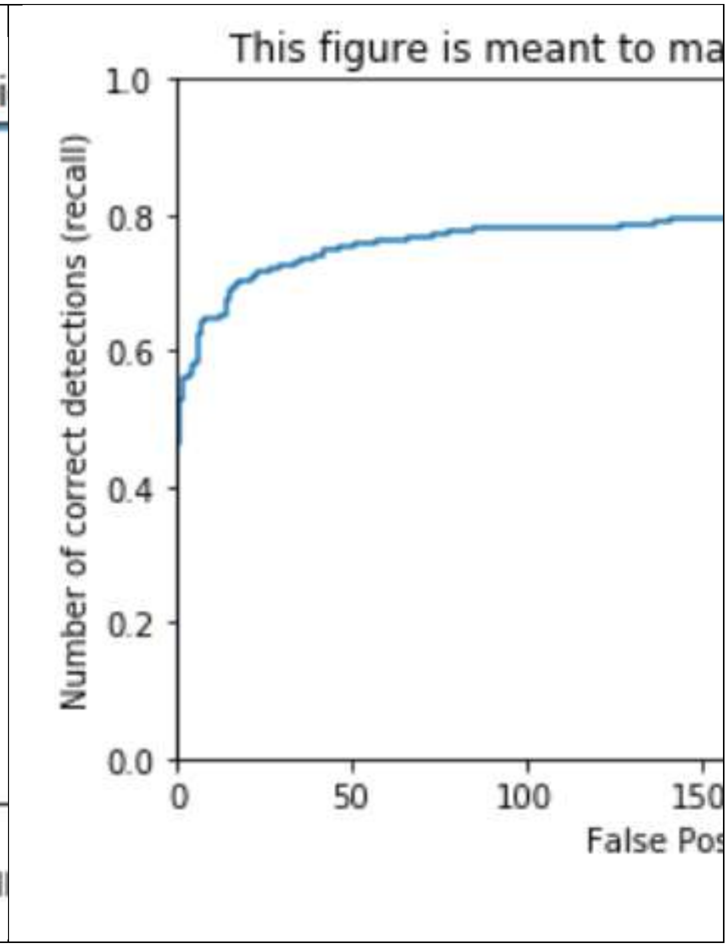
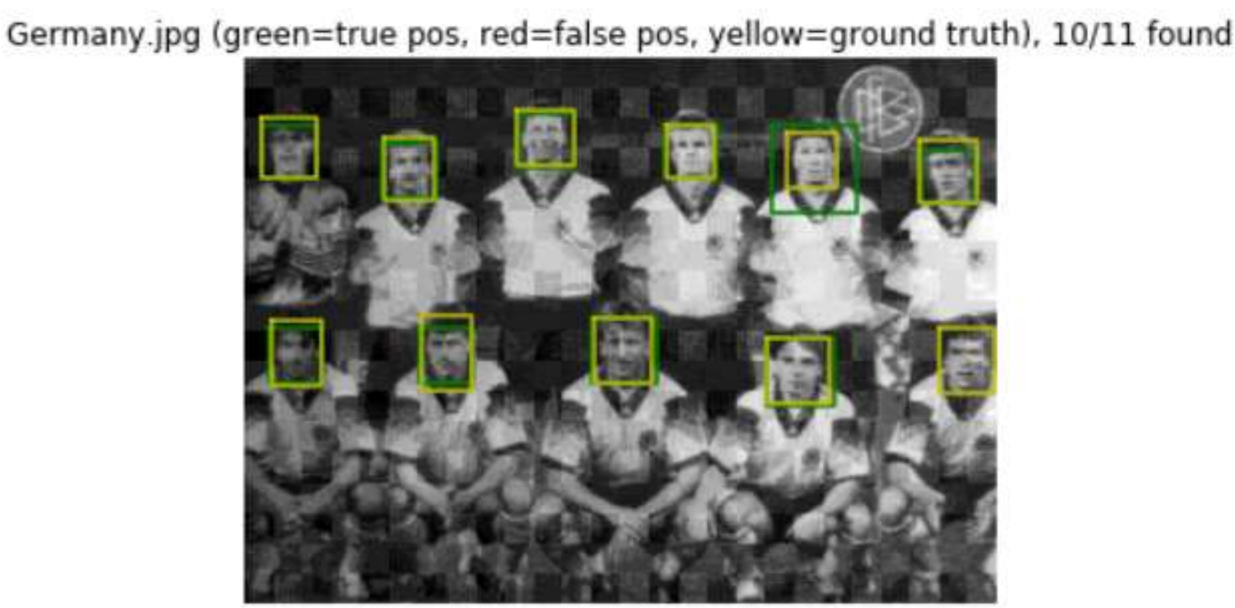
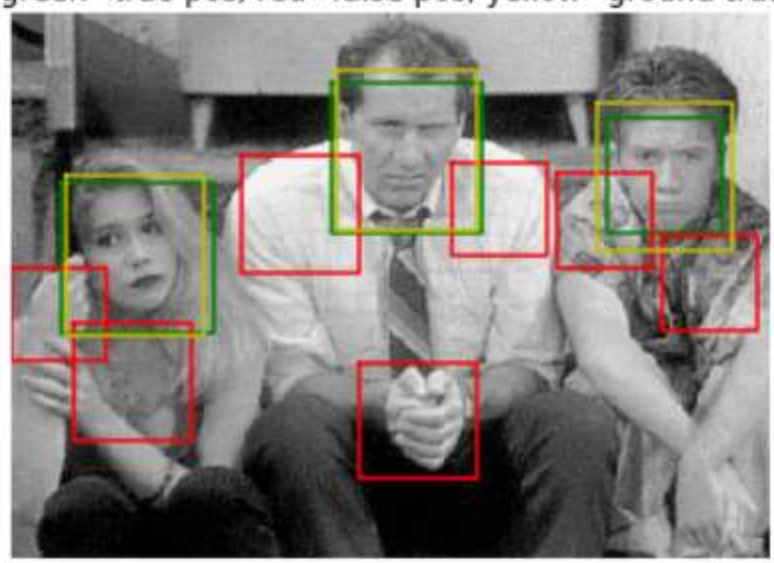
	

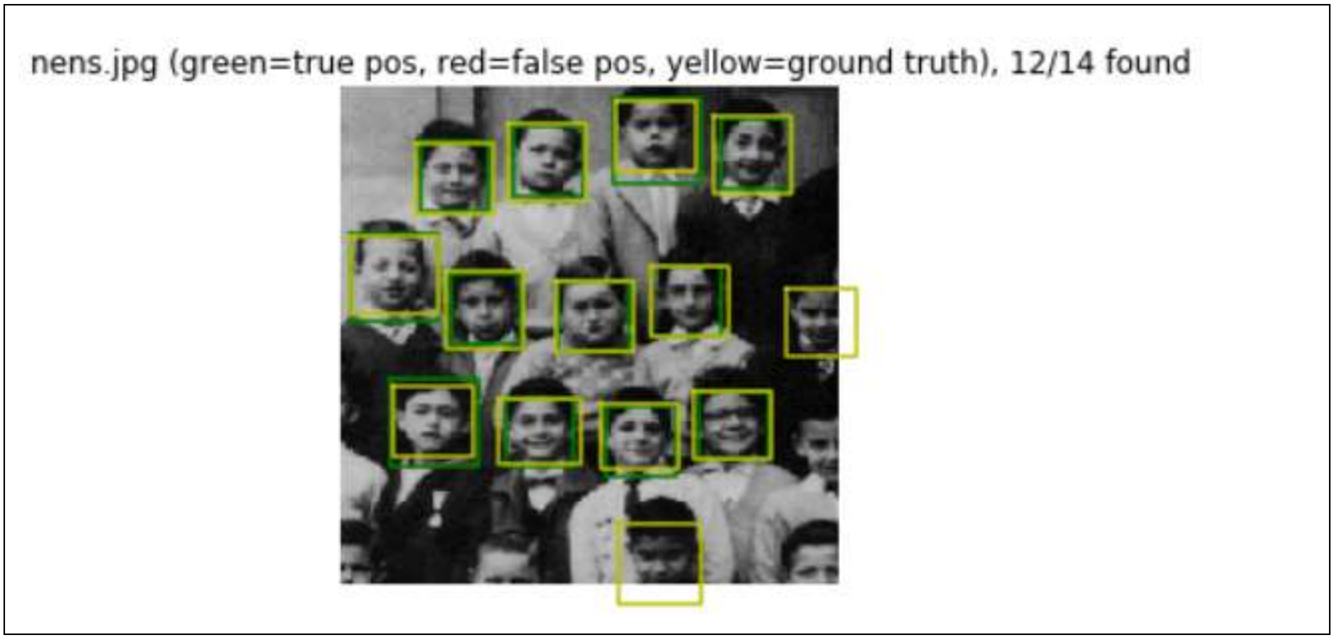
Table 3: Detection visualization

Detection Visualization


lawoman.jpg (green=true pos, red=false pos, yellow=ground truth), 4/4 found

married.jpg (green=true pos, red=false pos, yellow=ground truth), 3/3 found

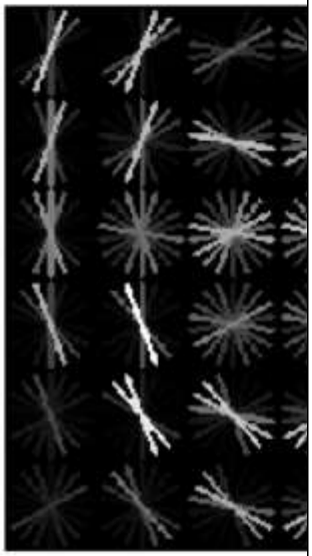
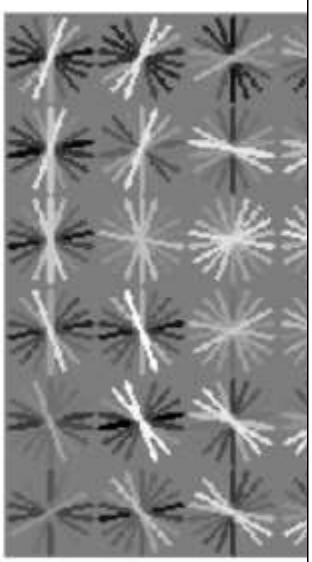
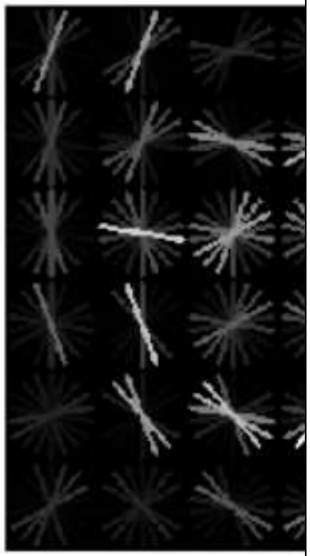
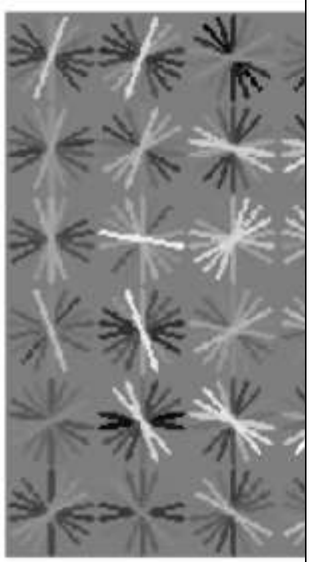
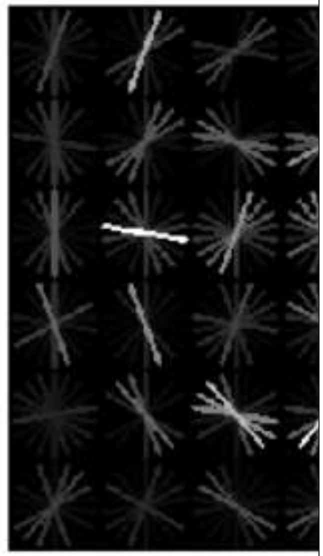
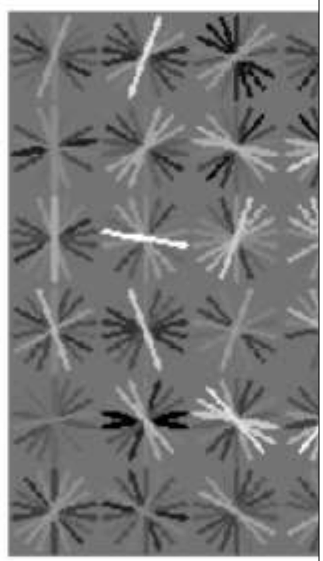
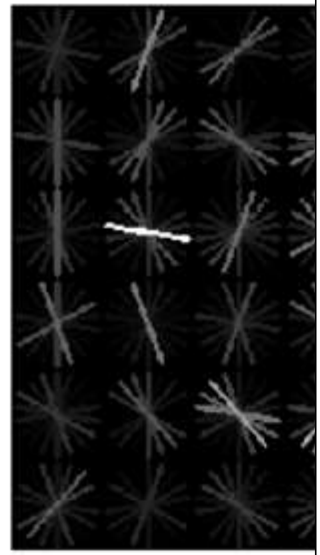
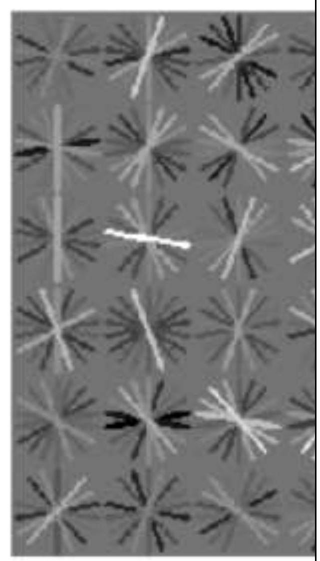




4.2. Experiment A: SVM ('cell_size' = 6) Classification Results vs. Dataset, Negative features, SVM lambda , and Detection Scale

Table 4: SVM_1 (Reg. negative features, 'cell_size' = 6) | HOG Template vs. SVM lambda & Dataset

	Trained HOG template			
Dataset	**SVM_1 C = 0.0001**	**SVM_1 C = 0.01**	**SVM_1 C = 0.1**	**SVM_1 C = 1**

Dataset #1	 	 	 	 

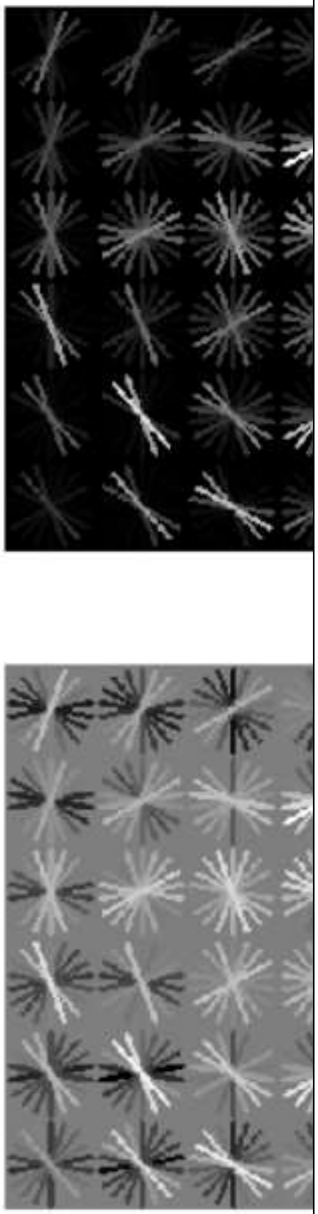
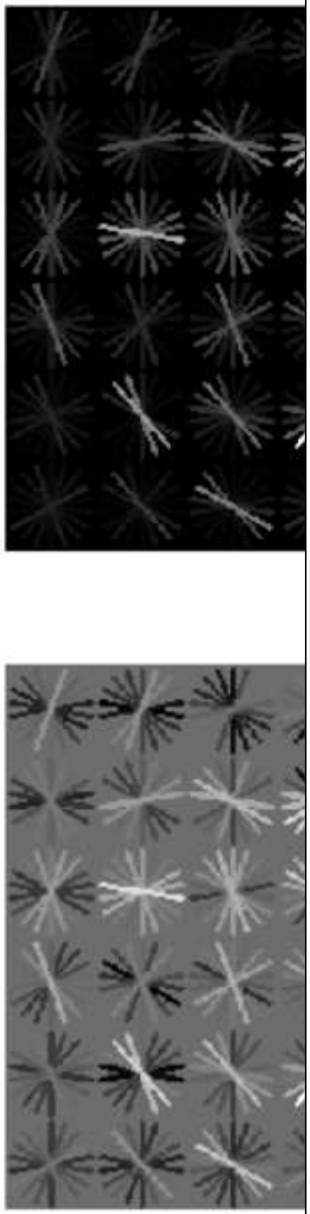
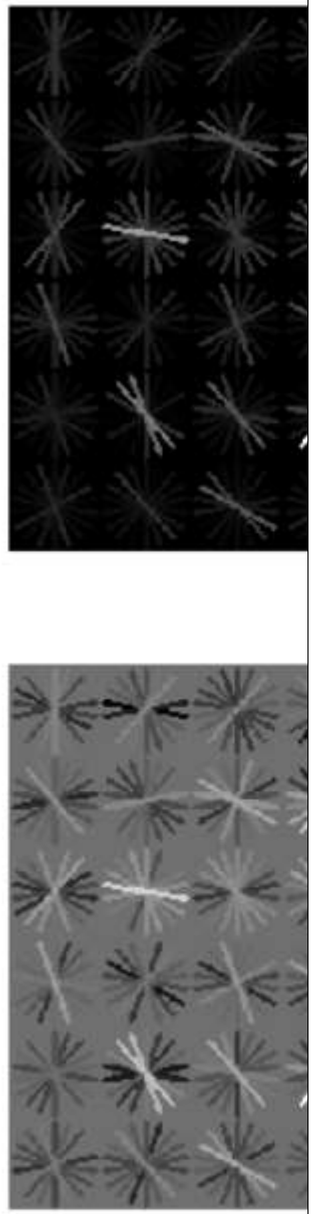
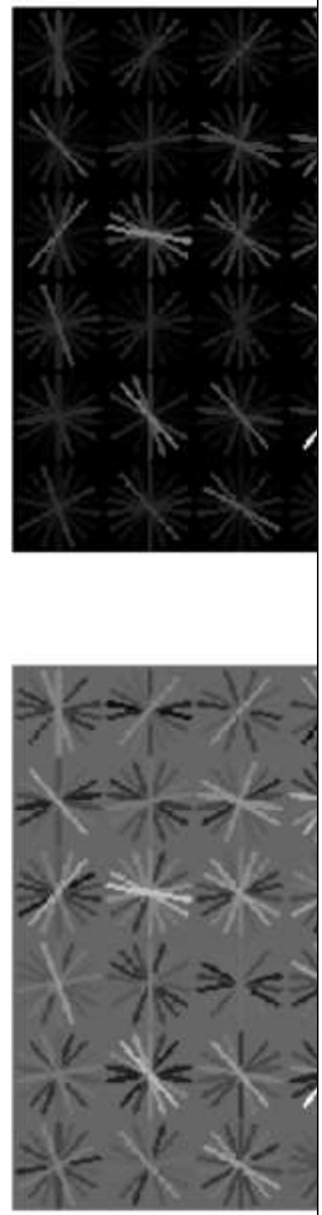
Dat aset #2				

Table 5: SVM_1 (Reg. negative features, 'cell_size' = 6) | Training results vs. SVM lambda & Dataset

	Training results			
Dat aset	SVM_1 C = 0.0001	SVM_1 C = 0.01	SVM_1 C = 0.1	SVM_1 C = 1
Dat aset	Accuracy = 98.211%	Accuracy = 99.659%	Accuracy = 99.940%	Accuracy = 100.000%

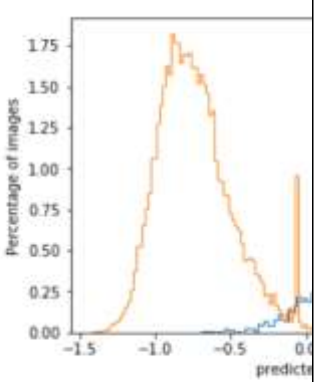
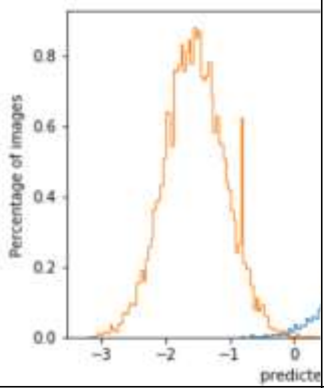
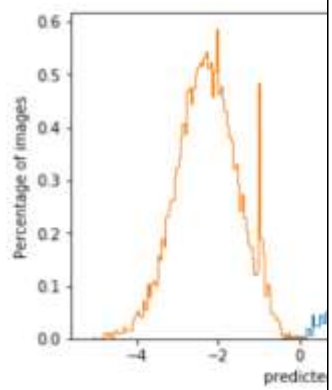
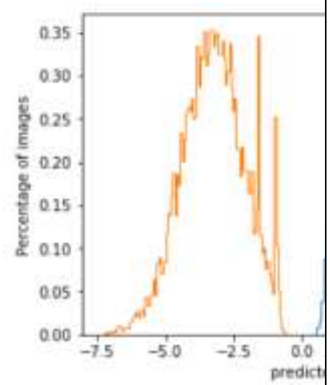
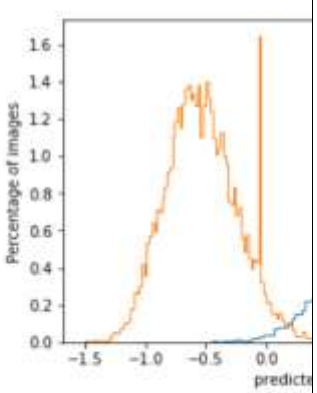
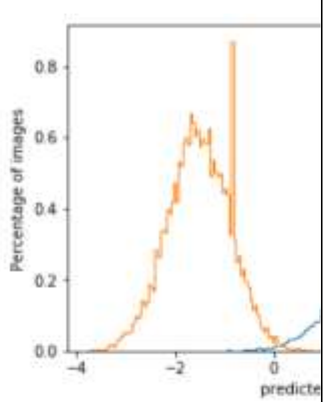
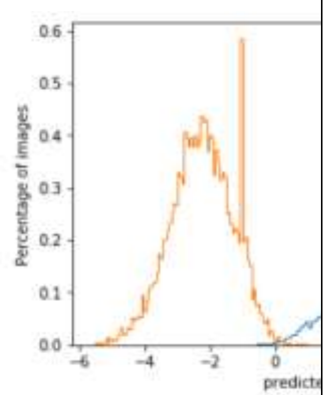
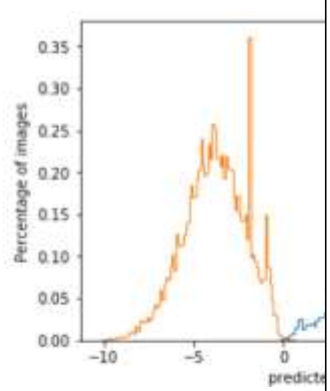
#1	**True Positive rate = 96.157%** **False Positive rate = 0.410%** **True Negative rate = 99.590%** **False Negative rate = 3.843%** 	**True Positive rate = 99.359%** **False Positive rate = 0.140%** **True Negative rate = 99.860%** **False Negative rate = 0.641%** 	**True Positive rate = 99.881%** **False Positive rate = 0.020%** **True Negative rate = 99.980%** **False Negative rate = 0.119%** 	**True Positive rate = 100.000%** **False Positive rate = 0.000%** **True Negative rate = 100.000%** **False Negative rate = 0.000%** 
Dat aset #2	**Accuracy = 98.712%** **True Positive rate = 99.455%** **False Positive rate = 3.780%** **True Negative rate = 96.220%** **False Negative rate = 0.545%** 	**Accuracy = 99.550%** **True Positive rate = 99.657%** **False Positive rate = 0.810%** **True Negative rate = 99.190%** **False Negative rate = 0.343%** 	**Accuracy = 99.777%** **True Positive rate = 99.833%** **False Positive rate = 0.410%** **True Negative rate = 99.590%** **False Negative rate = 0.167%** 	**Accuracy = 99.970%** **True Positive rate = 99.982%** **False Positive rate = 0.070%** **True Negative rate = 99.930%** **False Negative rate = 0.018%** 

Table 6: SVM_1 (Reg. negative features, 'cell_size' = 6) | Classification AP vs. SVM lambda & Dataset & Detection Scale

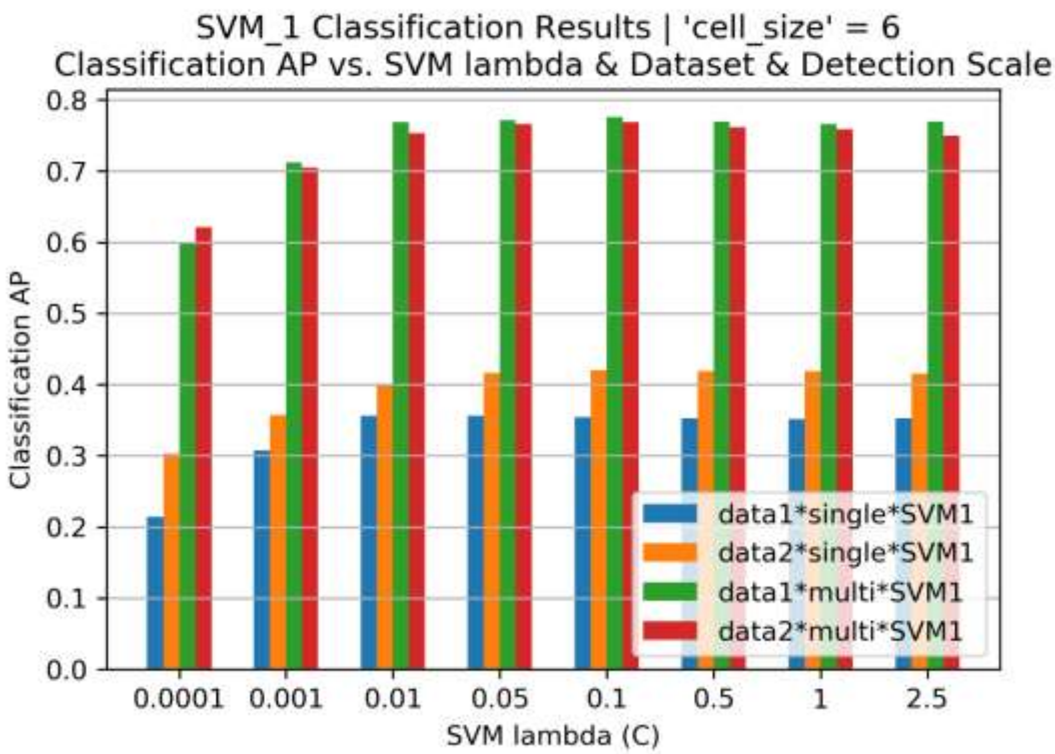
SVM_1 (Reg. negative features, 'cell_size' = 6) | Classification AP vs. SVM lambda & Dataset & Detection Scale


Table 7: SVM_2 (Hard negative features, 'cell_size' = 6) | Classification AP vs. SVM lambda & Dataset & Detection Scale

SVM_2 (Hard negative features, 'cell_size' = 6) | Classification AP vs. SVM lambda & Dataset & Detection Scale

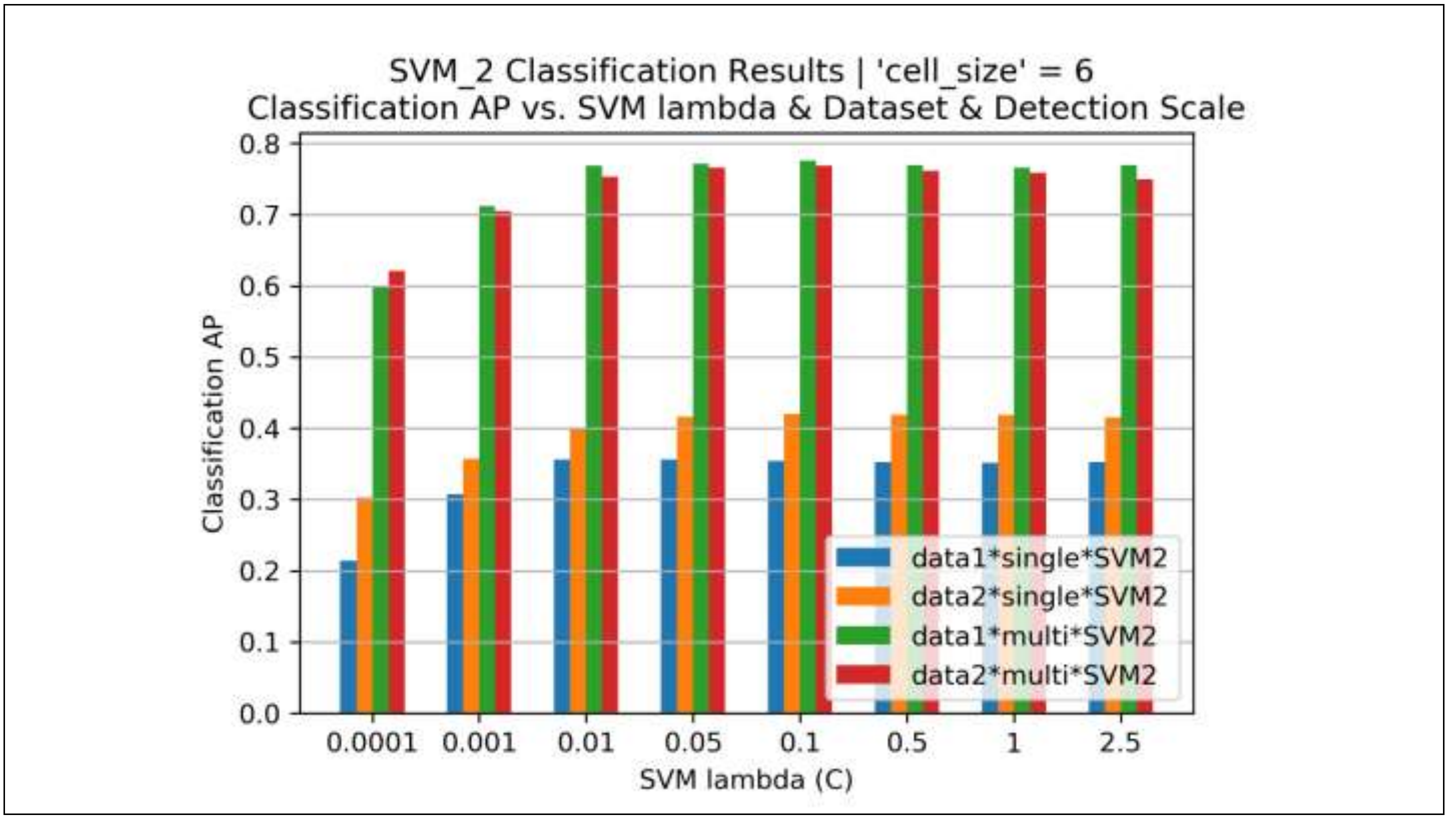


Table 8: SVM_1 & SVM_2 (Reg. & hard negative features, 'cell_size' = 6) | Classification AP vs. SVM lambda & Dataset @ Single-scale Detection

SVM_1 & SVM_2 (Reg. & hard negative features, 'cell_size' = 6) | Classification AP vs. SVM lambda & Dataset @ Single-scale Detection

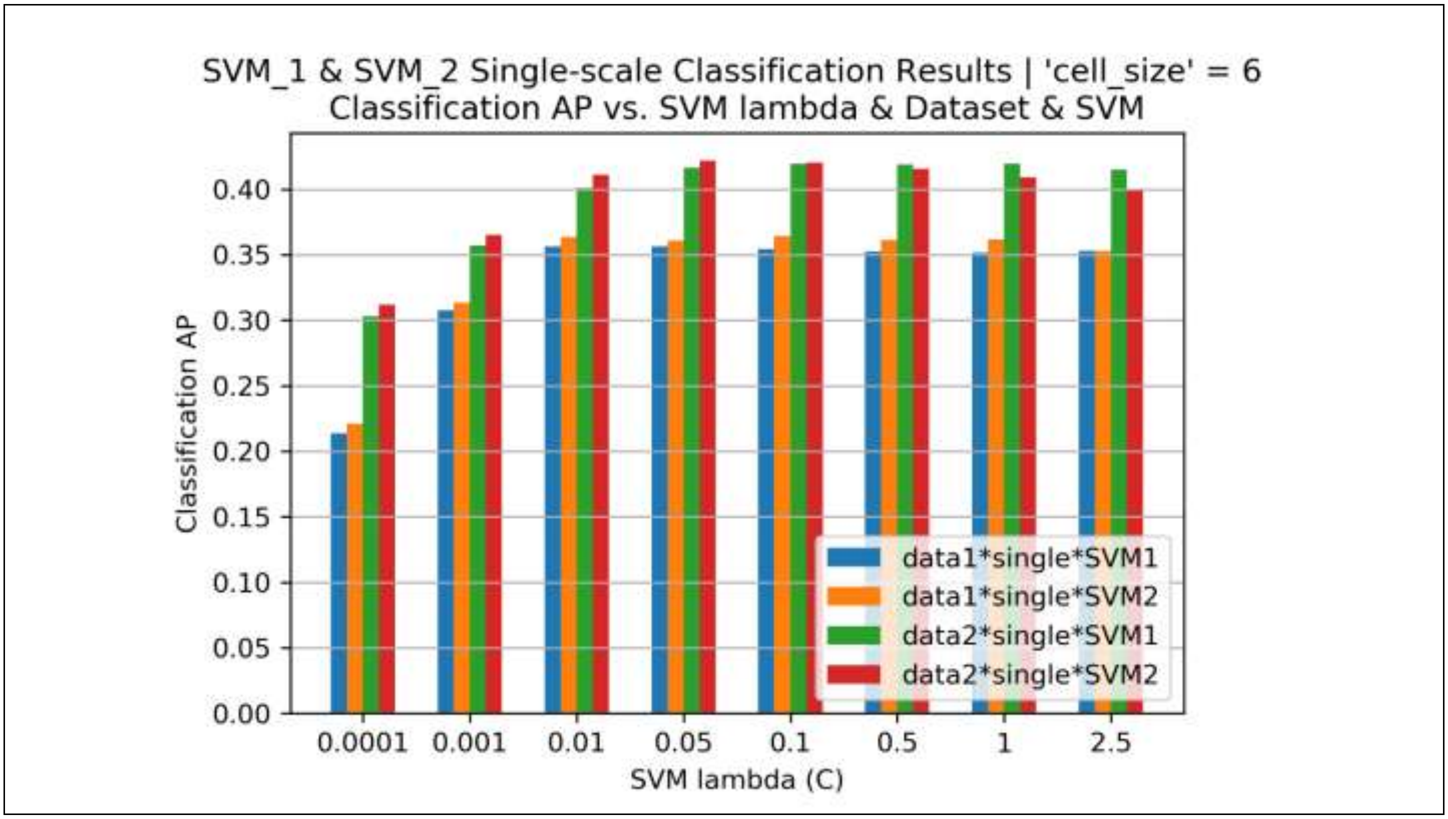
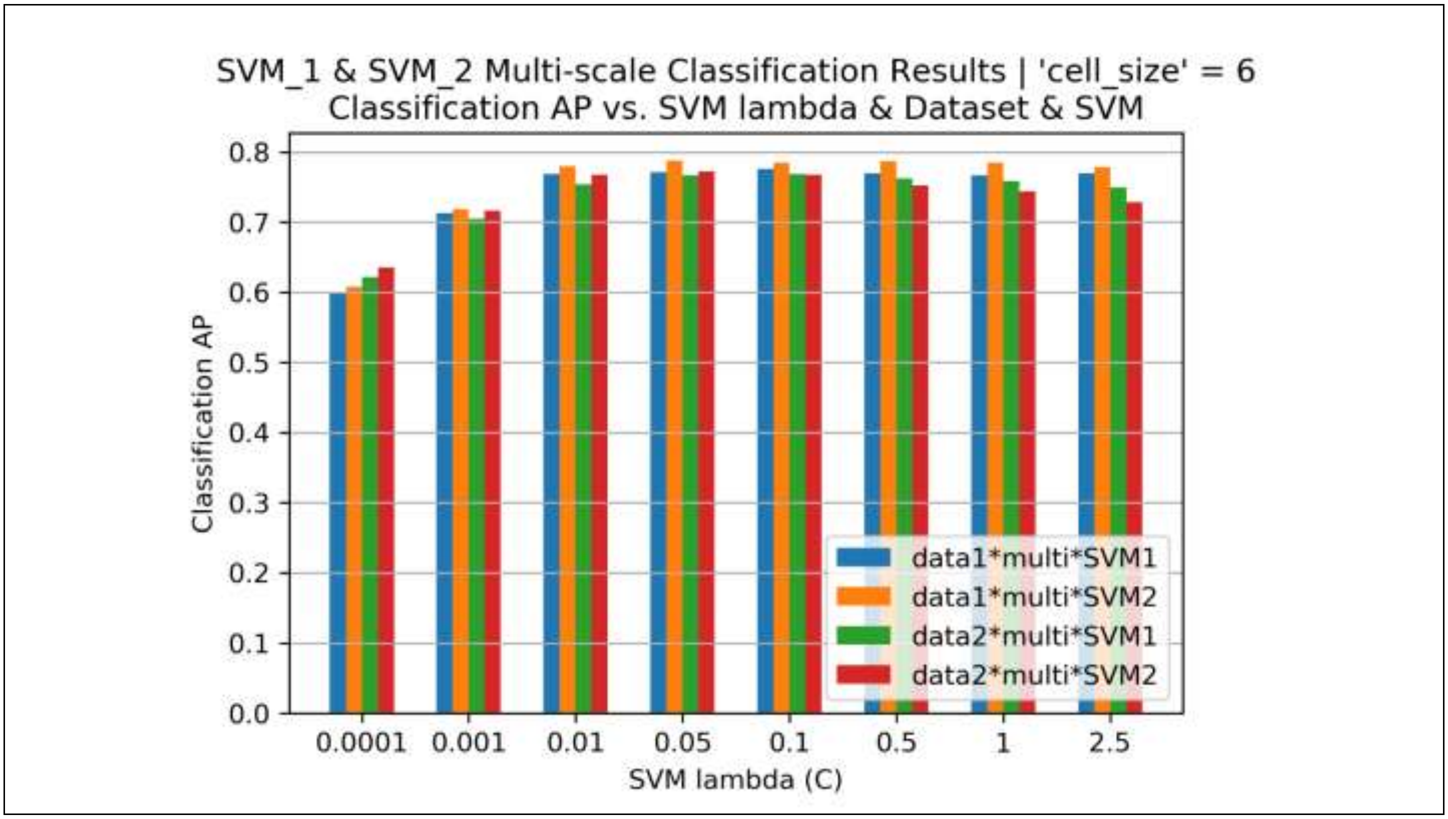


Table 9: SVM_1 & SVM_2 (Reg. & hard negative features, 'cell_size' = 6) | Classification AP vs. SVM lambda & Dataset @ Multi-scale Detection

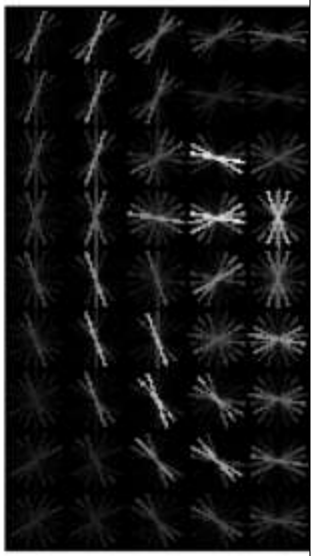
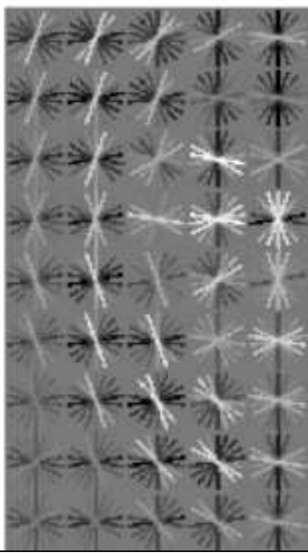
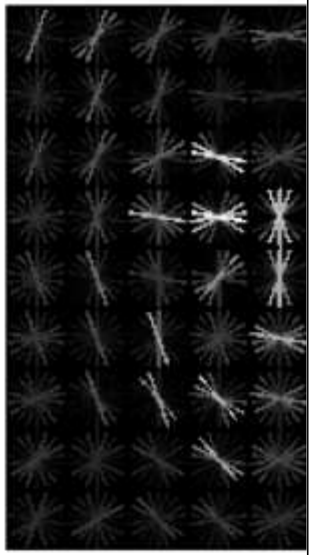
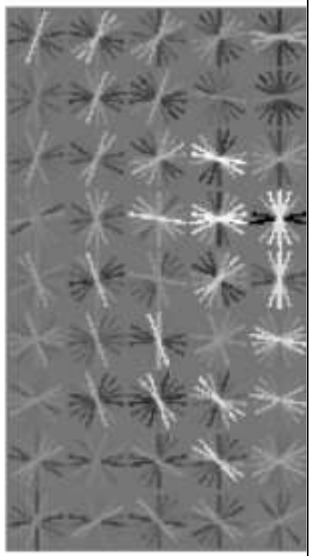
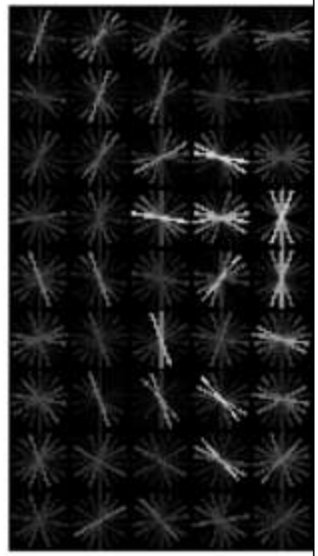
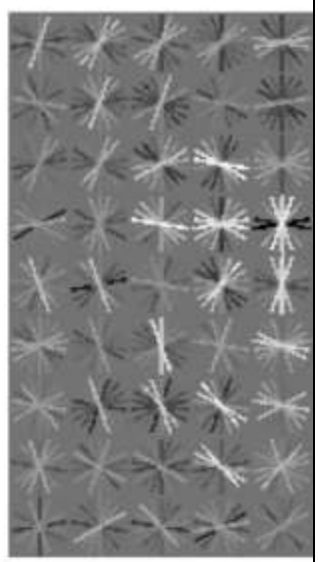
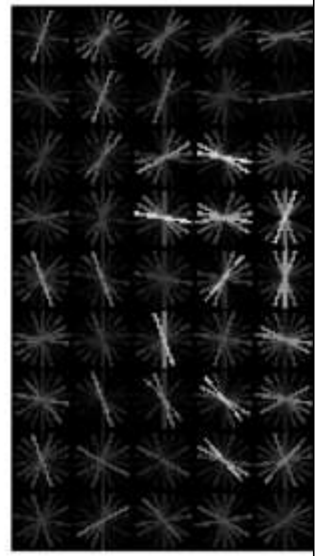
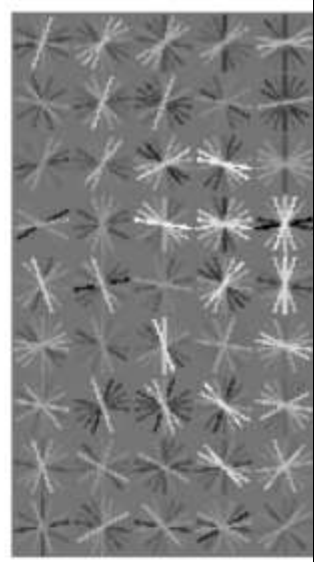
SVM_1 & SVM_2 (Reg. & hard negative features, 'cell_size' = 6) | Classification AP vs. SVM lambda & Dataset @ Multi-scale Detection



4.3. Experiment B: SVM ('cell_size' = 4) Classification Results vs. Dataset, Negative features, SVM lambda , and Detection Scale

Table 10: SVM_1 (Reg. negative features, 'cell_size' = 4) | HOG Template vs. SVM lambda & Dataset

	Trained HOG template			
Dataset	**SVM_1 C = 0.0001**	**SVM_1 C = 0.01**	**SVM_1 C = 0.1**	**SVM_1 C = 1**

Dataset #1	 	 	 	 

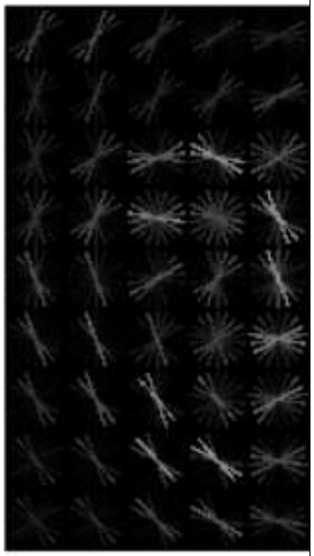
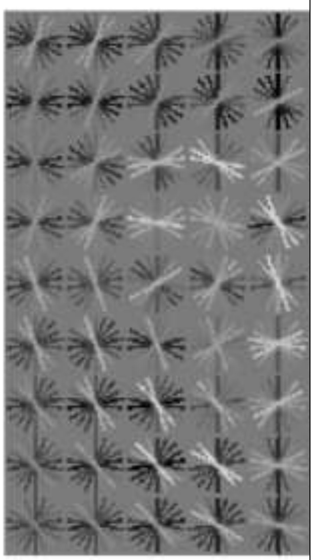
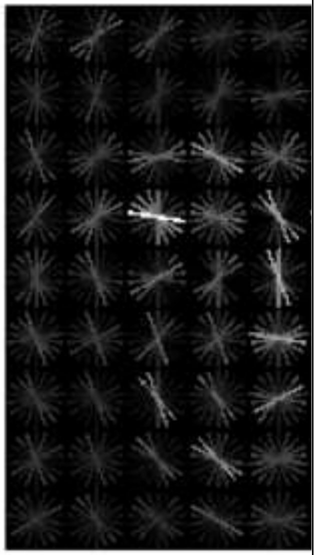
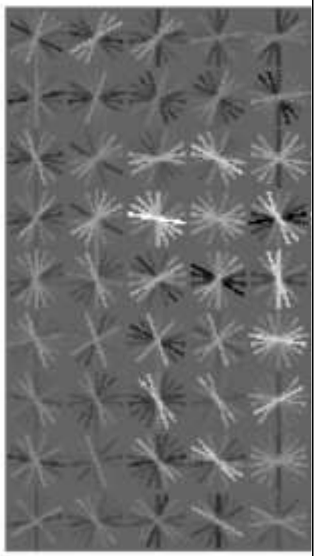
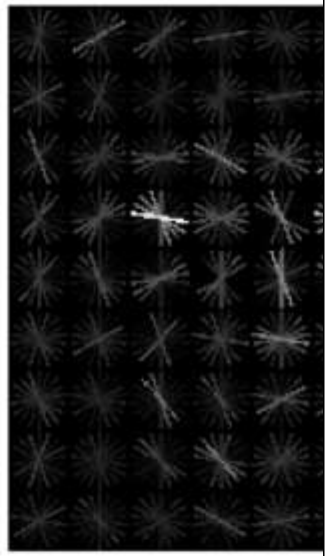
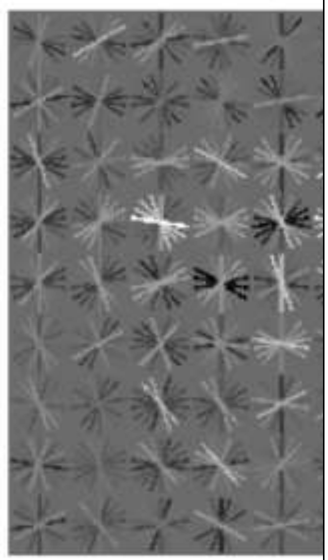
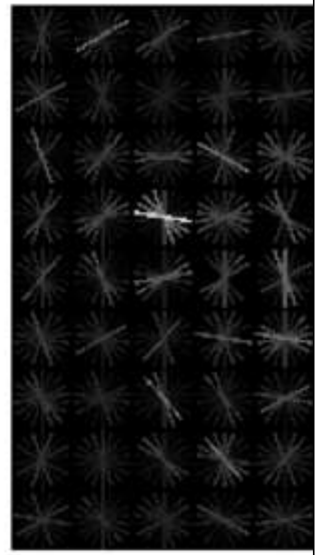
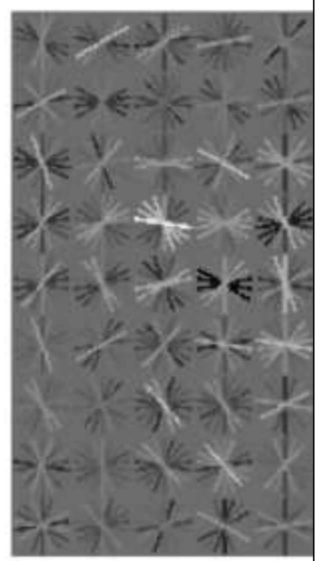
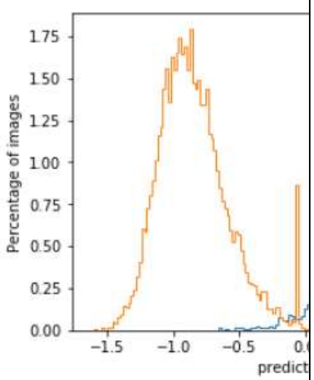
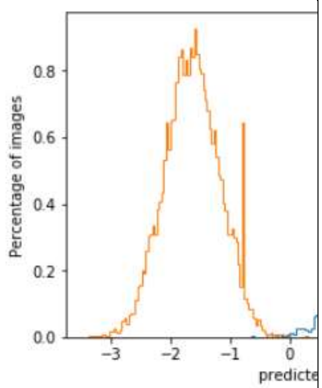
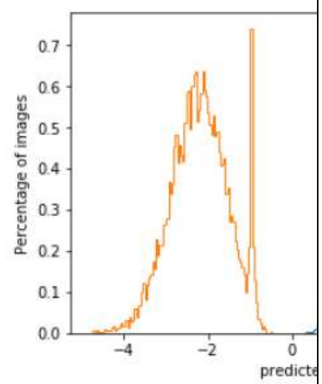
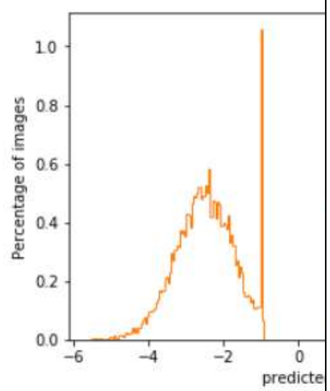
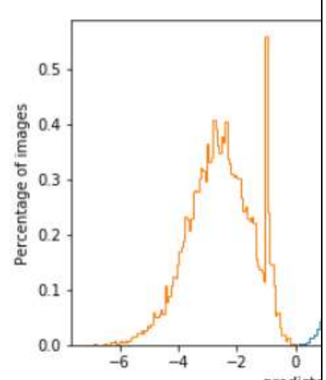
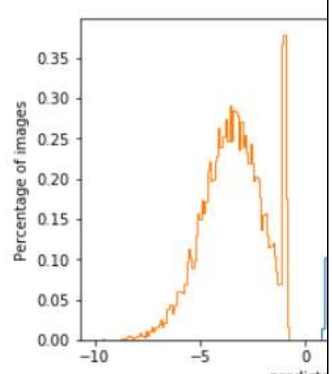
Dat aset #2	 	 	 	 

Table 11: SVM_1 (Reg. negative features, 'cell_size' = 4) | Training results vs. SVM lambda & Dataset

	Training results			
Dat aset	SVM_1 C = 0.0001	SVM_1 C = 0.01	SVM_1 C = 0.1	SVM_1 C = 1
Dat aset	Accuracy = 99.091%	Accuracy = 99.940%	Accuracy = 100.000%	Accuracy = 100.000%

#1	**True Positive rate = 97.929%** **False Positive rate = 0.130%** **True Negative rate = 99.870%** **False Negative rate = 2.071%** 	**True Positive rate = 99.866%** **False Positive rate = 0.010%** **True Negative rate = 99.990%** **False Negative rate = 0.134%** 	**True Positive rate = 100.000%** **False Positive rate = 0.000%** **True Negative rate = 100.000%** **False Negative rate = 0.000%** 	**True Positive rate = 100.000%** **False Positive rate = 0.000%** **True Negative rate = 100.000%** **False Negative rate = 0.000%** 
Dataset #2	**Accuracy = 99.174%** **True Positive rate = 99.514%** **False Positive rate = 1.970%** **True Negative rate = 98.030%** **False Negative rate = 0.486%**	**Accuracy = 99.844%** **True Positive rate = 99.872%** **False Positive rate = 0.250%** **True Negative rate = 99.750%** **False Negative rate = 0.128%**	**Accuracy = 99.998%** **True Positive rate = 99.997%** **False Positive rate = 0.000%** **True Negative rate = 100.000%** **False Negative rate = 0.003%** 	**Accuracy = 100.000%** **True Positive rate = 100.000%** **False Positive rate = 0.000%** **True Negative rate = 100.000%** **False Negative rate = 0.000%** 

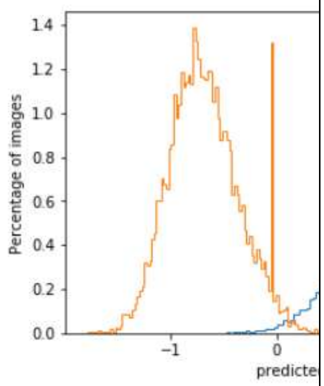
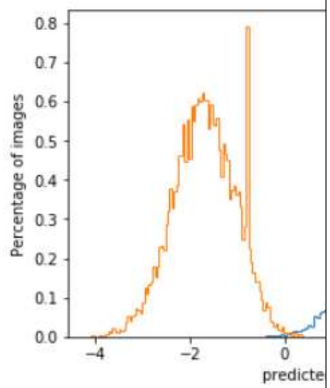
				

Table 12: SVM_1 (Reg. negative features, 'cell_size' = 4) | Classification AP vs. SVM lambda & Dataset & Detection Scale

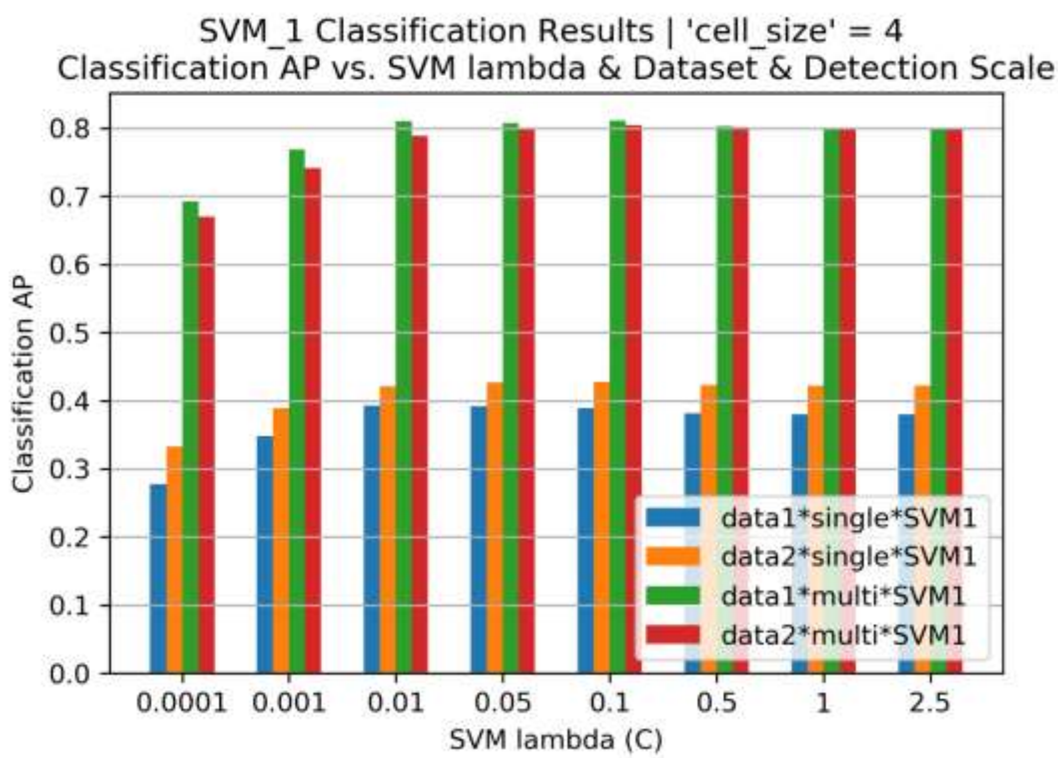
SVM_1 (Reg. negative features, 'cell_size' = 4) | Classification AP vs. SVM lambda & Dataset & Detection Scale


Table 13: SVM_2 (Hard negative features, 'cell_size' = 4) | Classification AP vs. SVM lambda & Dataset & Detection Scale

SVM_2 (Hard negative features, 'cell_size' = 4) | Classification AP vs. SVM lambda & Dataset & Detection Scale

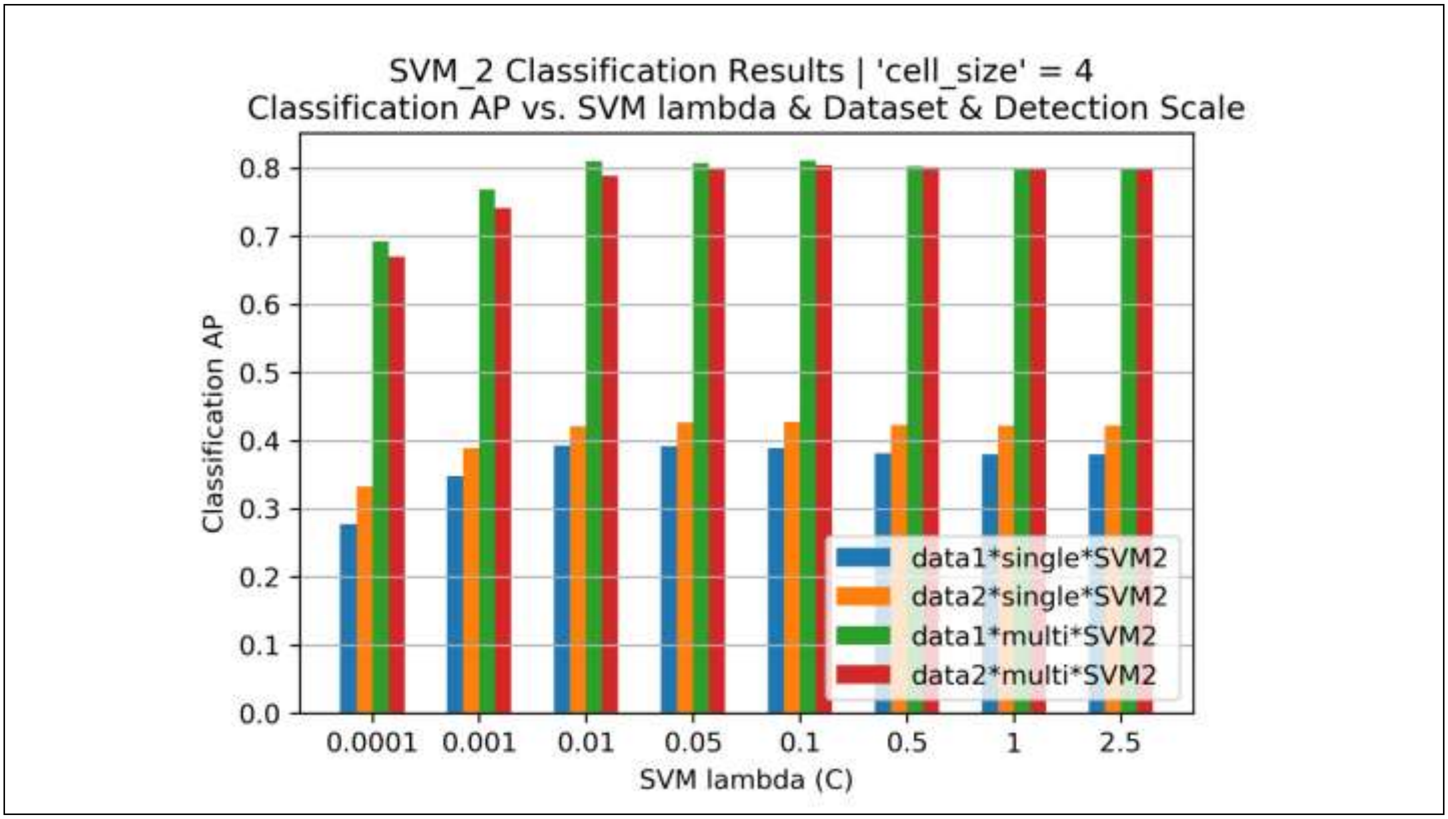


Table 14: SVM_1 & SVM_2 (Reg. & hard negative features, 'cell_size' = 4) | Classification AP vs. SVM lambda & Dataset @ Single-scale Detection

SVM_1 & SVM_2 (Reg. & hard negative features, 'cell_size' = 4) | Classification AP vs. SVM lambda & Dataset @ Single-scale Detection

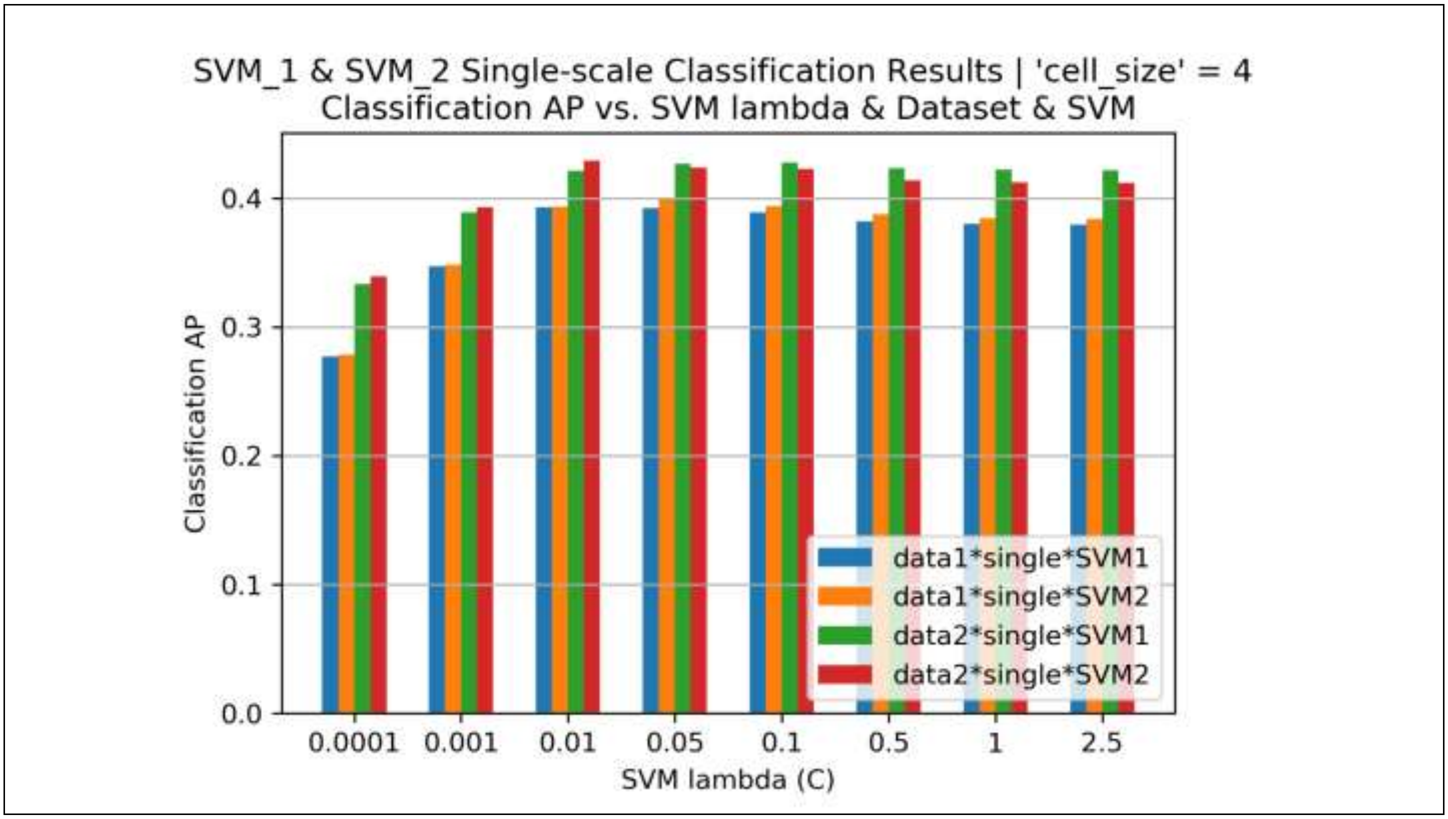
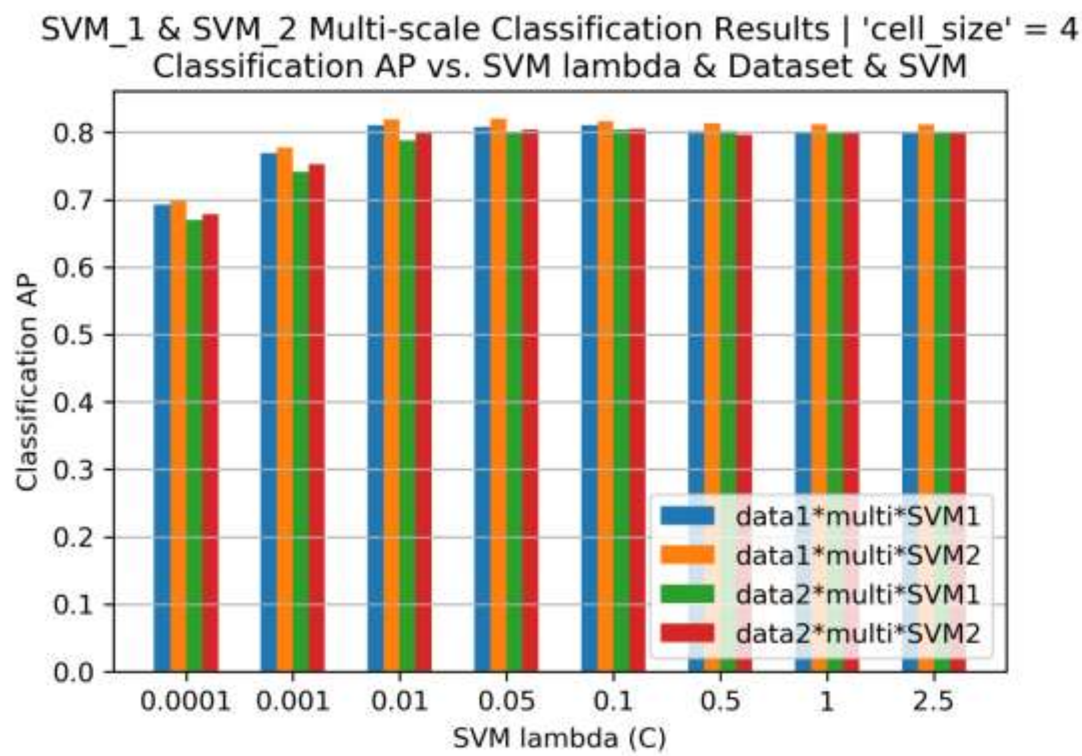


Table 15: SVM_1 & SVM_2 (Reg. & hard negative features, 'cell_size' = 4) | Classification AP vs. SVM lambda & Dataset @ Multi-scale Detection

SVM_1 & SVM_2 (Reg. & hard negative features, 'cell_size' = 4) | Classification AP vs. SVM lambda & Dataset @ Multi-scale Detection



Remarks on the classification results for experiments A and B

Note:

- Experiments A and B share the same design. They also resulted is similar trend. Therefore, they are both discussed here.

Experiment A ('cell_size' = 6) and B ('cell_size' = 4) summary (rewritten here for convenience of reader):

- **Train Dataset #1:**
  - Positive feature: HOG features extracted from original face dataset
  - Negative feature: HOG features extracted from original non-face dataset
  - Hard Negative feature: HOG features extracted from original non-face dataset
  - Test dataset: HOG features extracted from original test dataset
- **Train Dataset #2 (augmented):**
  - Positive feature: HOG features extracted from original face dataset + flipped images + warped images (@ 10, 20, and 30 deg.)
  - Negative feature: HOG features extracted from original non-face dataset + scaled images (@ scale = [0.8, 0.65])

    - Hard Negative feature: HOG features extracted from original non-face dataset + scaled images (@ scale = [0.8, 0.65])
- **Test Dataset:**
    - Test dataset: HOG features extracted from original test dataset
- **Varying parameters:**
    - Dataset #1 | #2
    - SVM lambda = [0.0001, 0.001, 0.01, 0.05, 0.1, 0.5, 1, 2.5]
    - Negative features: SVM_1 (Regular negative features) | SVM_2 (Hard negative features)
    - Detection scale: single-scale | multi-scale (scale = [0.8, 0.65, 0.5, 0.3, 0.25])
- **Total number of experiments:**
    - 2 (datasets) * 8 (SVM lambda) * 2 (neg. feats.) * 2 (detection scale) = 64
- **Results:**
    - HOG templates vs. dataset and SVM lambda
    - Training results vs. dataset and SVM lambda
    - Classification Results vs. Dataset, Negative features, SVM lambda , and Detection Scale

AP vs. Dataset:

- For both SVM_1 (reg. negatives) & SVM_2 (hard negatives):
    - At single-scale detection, augmented dataset #2 is resulting higher accuracy than non-augmented dataset #1.
    - However, at multi-scale detection, for 7 (out of 8) SVM lambdas, augmented dataset #2 is resulting in slightly lower accuracy than non-augmented dataset #1.

AP vs. SVM lambda:

- The AP increases from C = 0.0001 to C = 0.01. However, it is almost constant for higher values of C. It seems that it has reached the highest possible values.

AP vs. negative features (SVM_1: regular / SVM_2: hard):

- At both single-scale & multi-scale detection:
    - For the first 5 SVM lambdas, SVM_2 results slightly higher accuracy than SVM_1 meaning that hard mining is helping the accuracy.
    - However, for higher lambda values, SVM_2 is weaker for dataset #2.

AP vs. detection scale:

- This parameter is the most influential.

- The average single-scale AP is 35%, while the average multi-scale AP is 75%.

Overall conclusion from cross-validation results:

- Augmented dataset is most influential at single-scale detection.
- SVM reaches its best performance at C = 0.01 & C = 0.05.
- Hard mining negative features has only slightly positive effect on AP.
- Detection scale is the most influential parameter that increases AP from 35% to 75%.
- The difference between experiment A and B is that the AP values are slightly higher for experiment B ('cell_size' = 4).
- The highest obtained AP for experiment A is slightly below 80% while experiment B resulted in slightly higher than 80% AP for various C values (C > 0.001).

Extra Works

Following functions and code were done outside student_code.py predefined functions and proj4.ipynb:

- Augmenting the provided training data using following ways:
    - Horizontally flipping images
    - Warping images (@ 10, 20, and 30 deg.)
- Extracting negative examples in multi-scale (scale = [0.8, 0.65])
- Experimenting SVM lambda values
- Combining all above with detection scale in experiment A ('cell_size' = 6) and B ('cell_size' = 4)
- Overall number of experiments:
    - 2 (experiments) * 2 (datasets) * 8 (SVM lambda) * 2 (neg. feats.) * 2 (detection scale) = 128

References

[1] Assignment 05 description by Dr. Kin-Choong Yow

[2] Szeliski, R. (2010). Computer vision: algorithms and applications. Springer Science & Business Media.