Summer 8-19-2014

Using GIST Features to Constrain Search in Object Detection

Joanna Browne Solmon
Portland State University

Using GIST Features to Constrain Search in Object Detection

by

Joanna Browne Solmon

A thesis submitted in partial fulfillment of the
requirements for the degree of

Master of Science
in
Computer Science

Thesis Committee:
Melanie Mitchell, Chair
Feng Liu
Bart Massey

Portland State University
2014

Abstract

This thesis investigates the application of GIST features [13] to the problem of object detection in images. Object detection refers to locating instances of a given object category in an image. It is contrasted with object recognition, which simply decides whether an image contains an object, regardless of the object's location in the image.

In much of computer vision literature, object detection uses a "sliding window" approach to finding objects in an image. This requires moving various sizes of windows across an image and running a trained classifier on the visual features of each window. This brute force method can be time consuming.

I investigate whether global, easily computed GIST features can be used to classify the size and location of objects in the image to help reduce the number of windows searched before the object is found. Using K-means clustering and Support Vector Machines to classify GIST feature vectors, I find that object size and vertical location can be classified with 73-80% accuracy. These classifications can be used to constrain the search location and window sizes explored by object detection methods.

Contents

List of Tables

List of Figures

Chapter 1

Introduction

Petacat [11] is a project begun by Melanie Mitchell at Portland State University which seeks to build a system than can generate human-like interpretations of image content by identifying relationships between objects in an image. For example, the system would be able to identify an image as representing an abstract concept such as "dog walking" (e.g. Figure 1.1) rather than merely identifying that the image contains a dog and a human.

Figure 1.1: Sample images from the Petacat [11] dog walking image set. Figure best viewed in color.

My work takes on the problem of object detection in the context of "dog walking" images by seeking to improve the speed with which a dog or dog walker can be detected in an image. In this thesis I investigate methods for classifying the locations and sizes of such objects based on cheap-to-compute image features called GIST features [13]. The hypothesis of this thesis is that GIST features can be used to do an initial, coarse analysis of an image in order to constrain the locations and sizes of windows to be searched.

1.1 Object Detection

Object recognition is the problem of deciding whether an image contains an instance of a particular image category. For object recognition, the input image is generally sized to only contain the object being classified and a trained classifier is used to identify the object within the image. *Object detection*, in contrast, is the problem of finding the location of an object within an image. No assumptions are made about the size, orientation, or location of the object within the image.

Object detection typically requires searching an image for the object by repeatedly running a trained classifier over different subsections of the image. The most common approach to image detection uses the brute force "sliding window" method (Fig. 1.2). Sliding window, as the name suggests, means creating a grid of overlapping windows across the whole surface of the image and running the classifier in each window. Additionally, to account for possible size variations in the object, this must be repeated multiple times, either with different sized windows or by resizing the image. Even with a relatively cheap classifier, this approach to object detection becomes very expensive and wasteful.

In spite of its inefficiency, the sliding window approach remains very common in current object detection research [2, 4, 7, 18]. Many researchers use some variation of the "cascading" method introduced by Viola and Jones in 2001 [19]. With this approach, a series of simple classifiers are run consecutively in each window. The classification process is short-circuited so that the first time one of these classifiers fails to classify the window as containing the object no further classifiers are run on the window. Harzallah et al. [7] and Vedaldi et al. [18] both use support vector machines with faster linear kernels as an initial classifier and then move on to slower more complex kernels for windows which are classified as containing the object by the

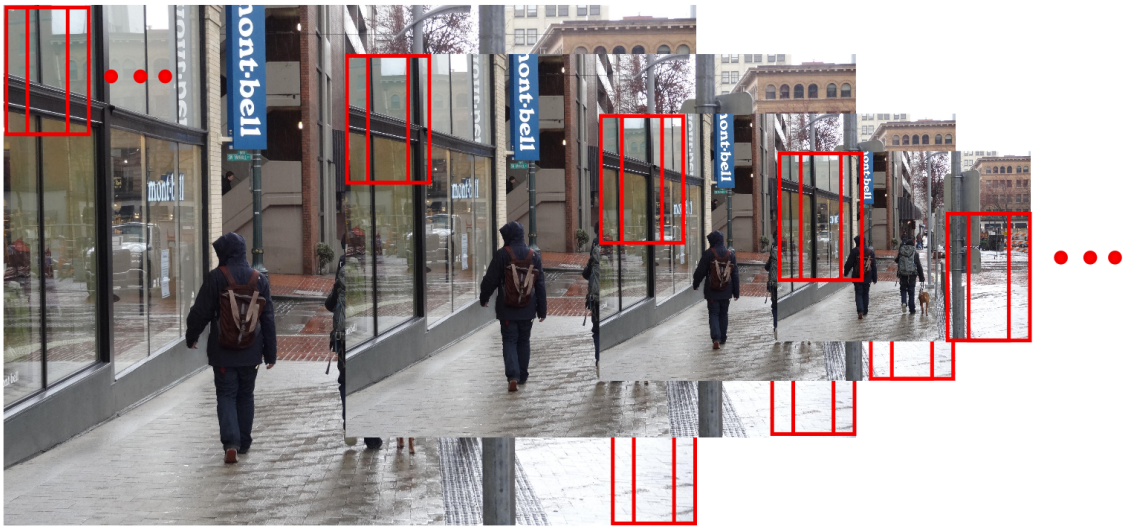


Figure 1.2: Illustration of the sliding window technique for object detection. An object detector is run in each overlapping window across the whole area of the image. The process is repeated on differently sized images to allow for different sized detection windows. Figure best viewed in color.

linear SVM. These cascading methods save time by not running heavy-duty classifiers on all windows, but they still require running some classifier on each window.

Several people have done work attempting to narrow down the number of windows which must be examined during object detection. Elazary and Itti [3] use saliency maps based on a simple set of features (intensity, color opponency, and edge detection at four orientations) and calculate the intensity of each feature for the objects they are detecting. Their method learns probability density functions (PDFs) for these saliency maps relative to different object classes. Using this information, they can generate a probability map of the likely locations of an object within a given image. Elazary and Itti's method is fast to compute and greatly decreases the number of windows examined before an object is found. However, while it performed very well on the ALOI data set [6] which contains identically-sized rotated copies of the exact object being searched for, its performance was less impressive when searching satellite images for images of previously-unseen houses. Additionally, this approach does not

indicate anything about the size of the object, only its probable location.

Alexe et al. [1] minimize the number of windows which must be searched by probabilistically calculating the location and size of the next search window based on the content of all windows searched previously. For example, if they are searching for a car and the current window examined contains sky, they would move their search location lower in the image because cars are likely to be beneath sky. This logic is implemented by learning a dictionary of the contents of randomly sized and placed windows relative to the ground-truth bounding box of the desired object. In a new test image, the location and size of the next window to be searched is determined by a vote of the dictionary entries that most closely match previously searched windows. Though Alexe et al. report good results in minimizing the number of windows searched, their learning process is very complex and cumbersome.

Several researchers have shown that object detection performance can be improved by using scene context within the image. Galleguillos et al. [5] first use segmentation to break images up into regions of sky, ground, and object of interest. They then use Bayesian models based on location of an object relative to its contextual background to refine object identification. For example, they could improve identification of boats within an image by identifying whether or not the object in question was surrounded by water.

Hoiem et al. [8, 9] also use Bayesian models and an initial object segmentation; however they use these as a starting point to infer the photographer's perspective and build a 3D representation of the image. They show [9] how perspectival information can be used to limit the locations and bounding boxes searched to find pedestrians in urban images. They show that this approach can significantly improve correct object detection, but as with Alexe et al. their approach is complex. It also requires significant human work to create elaborately labeled training data.

1.2 My Approach

My goal was to find a simple and cheap-to-calculate method for determining the likely size and location of dogs and dog walkers within an image, such as those shown in Figure 1.1. Being able to predict the approximate location and size of likely bounding boxes could eliminate a large subset of the windows which need to be searched for the object under the "sliding windows" model. I decided to investigate Oliva and Torralba's work with GIST features [13]. GIST features are low-level image features which can be used to classify images according to scene categories such as *coast*, *forest*, *highway*, or *city street*. My hypothesis was that if GIST features can distinguish between a distant landscape like a coast and a closer object like a city building, they would likely also be able to approximately predict the size of the humans and dogs within the image.

My hypothesis was thus that GIST features can be used to constrain the size, location, and orientation of objects within an image, thus shortening the task of image detection. If we can determine the likely size of the object, we can know which window size to try first and limit the search space to find the object. Similarly, if we know the object's likely location within the image, we can begin searching in an area more likely to yield a positive result. Knowing the object's orientation could be helpful in potentially choosing a more accurate classifier to identify the object.

To determine the usefulness of GIST features for image detection, I generated GIST features for each of the images in a data set collected by members of Melanie Mitchell's research group (described in Section 3.1). I then used these features to train two types of classifiers in order to predict size, location, and orientation of dogs and dog walkers. The two classification methods I investigated were K-means clustering and support vector machines (SVMs). I found that the GIST features did

not correlate with orientation or horizontal location but could be used to classify size and vertical location with 73-80% accuracy.

Chapter 2

GIST Features

2.1 Description of GIST Features

Oliva and Torralba introduced GIST features [13] in 2001 as a method for classifying images into scenic categories such as coast, forest, street, and highway. Categorization is based on what they term the image's "spatial envelope", meaning the shape formed by all solid surfaces within the scene. For example, on a city street the spatial envelope would be represented by the faces of buildings and the surface of the street, while in a forest the spatial envelope would be the ground and the shapes of the surrounding trees. Oliva and Torralba identify five distinguishing characteristics which can be combined to categorize a scene:

- naturalness: whether or not the scene is man-made
- openness: whether a scene is enclosed by objects or extends to infinity
- roughness: the fractal dimension or complexity
- expansion: how much depth is present in the image, particularly in the form of vanishing lines
- ruggedness: whether oblique features obscure the horizon line

Human volunteers ordered 7500 training images according to each the five characteristics to establish a continuum for each characteristic.

Images can be categorized as particular scene types according to where along the five continua they fall. For example, a highway receding toward the horizon would

score high on openness and expansion because it has open sky above the midpoint in the image and strong perspectival lines. An image of a city street taken facing directly at the front of a building would score low on both openness and expansion because there is no open space on the horizon and no perspectival lines. For natural images, a forest will score low on ruggedness and low on openness because there is a clear horizon line but no sky visible above the image. A coastal landscape will score low on ruggedness and high on openness because it has a clear horizon line and visible sky.

The process of extracting GIST features begins by converting images to greyscale and cropping them to be square. Then discrete Fourier transforms (DFTs) are calculated based on the intensity of the image:

$$I(f_x, f_y) = A(f_x, f_y)e^{j\phi(f_x, f_y)} \tag{2.1}$$

where f_x and f_y represent the spatial frequency of the image intensity in the x and y dimensions, the $A(f_x, f_y)$ term represents the amplitude spectrum of the image, and the $\phi(f_x, f_y)$ term represents the phase function. The phase function relates to local properties within the image while the amplitude represents information about overall image structure such as the orientation and smoothness of contours in the image. The squared magnitude of a Fourier transform, $|I(f_x, f_y)|^2 = A(f_x, f_y)^2$ produces an energy spectrum which represents the distribution of image intensity across different spatial frequencies. This energy spectrum for non-local image information is the basis for GIST features.

The energy spectrum can be decomposed into discrete spatial frequencies using the Karhunen-Loeve Transform (KLT) which yields a KL basis as follows:

$$A(f_x, f_y)^2 \cong \sum_{i=1}^{N_G} v_i \psi_i(f_x, f_y) \tag{2.2}$$

where

$$v_i = \int \int A(f_x, f_y)^2 \psi_i(f_x, f_y) df_x df_y \tag{2.3}$$

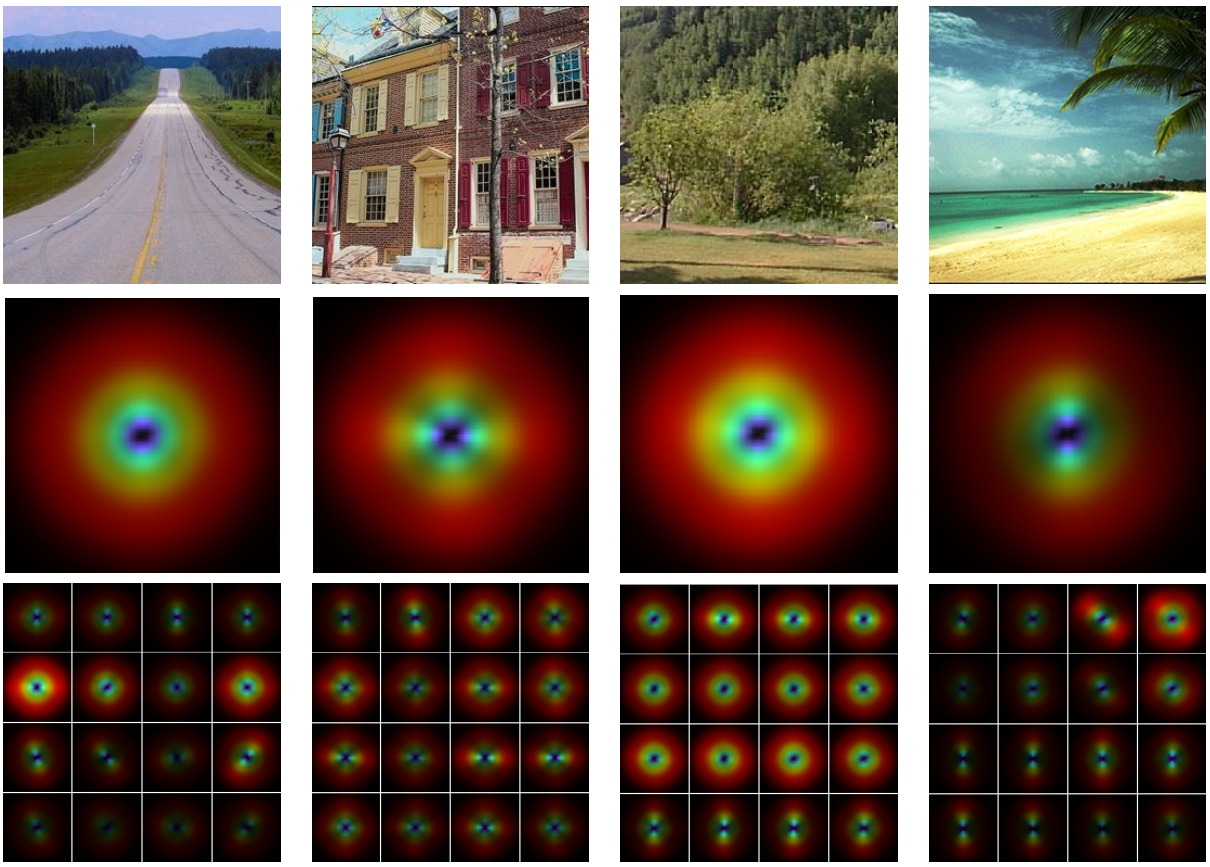
and $\psi_i(f_x, f_y)$ is the ith KL basis function. Once segments of the energy spectrum are isolated, they must be selected to determine which contribute to each feature being measured. s is defined as quantification of some scene feature from the energy spectrum and it can be defined as:

$$\hat{s} = \mathbf{v}^T \mathbf{d} \tag{2.4}$$

where $\mathbf{v}$ is a matrix containing all v_i and $\mathbf{d}$ is a column vector called the Discriminant Spatial Template (DST). The DST defines how much each v_i in $\mathbf{v}$ contributes to the final value of s and to the definition of the feature.

To learn $\mathbf{d}$ for each feature, Oliva and Torralba took 500 images at random from their image set and ordered them according to the property being measured. For instance, if training for openness, the images would be ordered from the images with the most openness to those with the least openness. Images were then each assigned some s value according to where they fell in the ordered set. Finally, values for $\mathbf{d}$ were determined which minimized the mean squared error in the calculation of s across all 500 images.

Localized energy spectra were also calculated similarly for 16 evenly divided sub-squares within each image. Figure 2.1 shows example landscape images alongside visualizations of their large-scale and local energy spectra. The full-image spectrum shows how each frequency is reflected across the whole image while the each local spectrum reflects frequencies over the associated subsection of the image. Oliva and Torralba show that combining both local and full-image energy spectra provides the better classification results than either on their own.



(a) Highway (b) City (c) Forest (d) Coast

Figure 2.1: Visualizations of the energy spectra generated by landscape images from Oliva and Torralba's [13] data set. The top row shows the original image. The middle row shows energy spectra generated over the whole image, and the bottom row shows the 16 localized energy spectra. Figure best viewed in color.

Oliva and Torralba classified their image features using K nearest neighbors. A new image was compared against labeled training images in a database. "Nearness" between two images was represented by taking the Euclidean distance between the scene attributes for openness, roughness, and either ruggedness for natural scenes or expansion for man-made scenes. In [13], Oliva and Torralba do not specify how many neighbors they compared against. Using K nearest neighbors for classification, Oliva and Torralba were able to achieve 88.7% accuracy when classifying images according to scene categories such as coast, forest, mountain, highway, and street.

2.2 Further Work with GIST Features

Torralba has applied GIST features to object detection tasks in several papers [14, 15, 16, 17]. Torralba and Oliva showed how GIST features can be used to estimate the mean depth of an image. [16] They use expectation maximization to create separate probability density functions (PDFs) for artificial and natural scenes and use these PDFs to predict the mean depth of an image based on the activation of different orientations of Gabor filters. The paper demonstrates how this depth estimation can be used to give an estimation of a person's head size in an image.

In a later paper [15], Torralba et al. demonstrate how GIST features can be used both for object presence detection (determining whether or not an image contains an object), and object localization (narrowing down the location of the object in the image). They assert that GIST features are useful only for predicting vertical location within an image and not horizontal location because only vertical location is constrained by the landscape-level details that make up GIST featues. They use a probabilistic graphical model trained on labeled training images to predict the probability that a given object will appear within a vertical band in the image. They give results for detecting cars and show improved detection rates when their localization

information is used, though a poor detector means that they still miss 70% of cars in the images.

I was not able to find examples in the literature of GIST features used for object detection outside of [14, 15, 16, 17].

2.3 GIST Features and Object Localization

In applying GIST features to the Petacat dog walking image set, I expected to find a correlation between the size and location of the dogs and dog walkers and scene characteristics picked up by GIST features. Images with a strong horizon line likely represent greater distance between the photographer and their subject and are likely to contain smaller humans and dogs, which are located higher up in the image. Similarly, in a photo looking straight at the front of a building with no visible sky, the photographer is likely closer to the subject and the image will contain larger humans and larger dogs positioned lower in the image.

Chapter 3

Methodology

3.1 Image Set

I used an image set of 484 photographs of people walking dogs (Figure 3.1) taken by members of Melanie Mitchell's research group for the Petacat [11] project. Each image contains a single dog and single dog-walker, though there are often multiple pedestrians within the image.

Figure 3.1: Sample images from the Petacat [11] dog walking image set. Yellow boxes represent labeled bounding boxes for the dog and dog walker. Figure best viewed in color.

Some images in the data set are vertically oriented and some are horizontally oriented. They also represented a variety of different aspect ratios and resolutions. Each image has been manually labeled by members of Mitchell's research group to provide bounding boxes enclosing the dog and dog walker. These bounding boxes are labeled with the categories "dog" and "dog-walker". These bounding boxes are the only ground truth that my classifiers use. For classification, I measured three values from each image for dog and dog walker bounding boxes: first, the ratio of the object height to the image height; second, the ratio of the object area to the image area; and third, the ratio between the vertical location of the object and the image height.

Two-thirds of the 484 images were used for training, yielding a training set size of 322 and a test set size of 162.

3.2 GIST Features

I used Matlab code provided by Oliva and Torralba [12] to generate GIST features for each image. Each set of GIST features is a vector of 256 features. The features are generated by measuring image activations at four orientations (vertical, horizontal, and the two diagonals), at the full image size and over 16 evenly-divided square subsections. Since our images are rectangular while the GIST code expects to work on square images, a maximal square area was taken from the center of each image for generating the GIST features. Figure 3.2 shows some sample Petacat dog walking images with visualizations of their GIST features.

The GIST features generated by Oliva and Torralba in the original GIST paper used 8 orientations [13] and in a later paper they broke the images up into 64 subsections to gauge image depth [16]. Using more feature-rich GIST vectors could potentially yield better results, but such vectors have 2048 features. With only 322 training images, there was not enough training data to allow for representative data

across all features if a larger feature vector is used.

I tested varying the number of orientations present in the GIST vectors between 1, 2, 4, and 8 and varying the number of subsections used between 4, 16, and 64. This yielded feature vectors ranging in size from 16 to 2048. I found that classification achieved the highest accuracy when using the 256-feature vectors with 4 orientations and 16 subsections. As stated above, I believe that more feature-rich GIST vectors could allow for even better results but this would require significantly more training data.

3.3 Classification Tasks

I performed the following six binary classification tasks on all images:

- **Area(Dog Walker) ∈ {Small, Large}**: Predicts whether the relative area of the dog walker's bounding box (i.e., dog walker bounding box area divided by image area) is less than or greater than the median training data value of 0.0773.

- **Height(Dog Walker) ∈ {Short, Tall}**: Predicts whether the relative height of the dog walker's bounding box (i.e., dog walker bounding box height divided by image height) is less than or greater than the median training data value of 0.5141.

- **Location(Dog Walker) ∈ {Low, High}**: Predicts whether the relative vertical location (i.e., dog walker's bottom Y coordinate, measured from the top of the image, divided by image height) is less than or greater than the median training data value of 0.761.

- **Area(Dog) ∈ {Small, Large}**: Predicts whether the relative area of the dog's

bounding box (i.e., dog bounding box area divided by image area) is less than or greater than the median training data value of 0.0215.

- **Height(Dog) ∈ {Short, Tall}**: Predicts whether the relative height of the dog's bounding box (i.e., dog bounding box height divided by image height) is less than or greater than the median training data value of 0.1648.

- **Location(Dog) ∈ {Low, High}**: Predicts whether the relative vertical location (i.e., dog bottom Y coordinate, measured from the top of the image, divided by image height) is less than or greater than the median training data value of 0.727.

Although we would expect height and area to be highly correlated, it is valuable to collect information on both because they can be combined to suggest dimensions for the search window.

The threshold for each classification task was determined by evenly dividing the training data so that one half would fall into the small class and one half would fall into the large class. For dog height, the median value is 0.1648, for dog walker height it is 0.5141, for dog area it is 0.0215, for dog walker area it is 0.0773, for dog vertical location it is 0.727 (as measured from the top of the image), and for dog walker vertical location it is 0.7610. See Figure 3.3 for visual examples of these values.

In the future it would be valuable to create more than two classes for each classification task but that would require a larger training data set.

3.3.1 Classification with K-means Clustering

K-means clustering is an unsupervised learning method used to cluster data into K clusters based on the distance between data points in the multi-dimensional space represented by their feature vectors. To begin, K points, called centroids, in the

data space are chosen at random. Each feature vector is then assigned to the nearest centroid. Once all data points are assigned, the centroids are relocated to be at the mean of their assigned feature vectors. Then the first step is repeated and each point is re-assigned to the nearest centroid. This process repeats until either some maximum iteration threshold is reached or there is no longer any movement of feature vectors between clusters. The output of the K-means algorithm is a set of K clusters, with each training example assigned to exactly one cluster.

When using K-means, I chose initial centroids at random from among the set of feature vectors and the remaining feature vectors were assigned to clusters based on minimum Euclidean distance. I performed 50 independent runs of K-means, each starting with randomly chosen centroids. I calculated the sum squared error (total Euclidean distance from each cluster element to its centroid) for each clustering and I report results only for the clustering with the smallest sum-squared error for classification. I used K values of 5, 10, and 15 clusters.

Each cluster was then assigned a class according to which class was in the majority among the training examples in that cluster. For example, if 75% of the data points in a particular cluster contained large dogs, that cluster would be classified as designating large dogs. To gauge the uniformity of classification within each cluster, I also calculated the cluster's entropy. Entropy for a cluster C_i is defined as

$$\text{entropy}(C_i) = -\sum_j p_{i,j} \log_2 p_{i,j}$$

where $p_{i,j}$ is the probability that a member of cluster i belongs to class j. Between 0 and 1 on the x-axis, $p_{i,j} \log_2 p_{i,j}$ forms a parabola with zeros at 0 and 1, meaning that clusters with both class probabilities closer to 0.5 will have higher entropies while clusters with one high-probability class and one low probability class will have low

entropies. Thus, cluster entropy gives a picture of how dominant the majority class is within the training data for a given cluster. Measuring cluster entropy allowed me to see how classification improved if I eliminated the clusters with the highest entropy.

Each image in the test set is assigned a classification matching that of the centroid with the nearest Euclidean distance. The entropy values of the clusters clearly showed that some clusters were likely to be more accurate classifiers than others. To account for this, I tested two methods: (1) classifying test data using data matching all clusters and (2) classifying test data matching only clusters which fall below a certain entropy threshold. The second method yields better classification results at the expense of classifying a smaller portion of the test data.

3.3.2 Classification with Support Vector Machines

Support vector machines (SVMs) are a family of supervised learning methods that learn from labeled data by drawing a hyperplane to separate the data into two classes according to the labeling. The most basic form of SVM uses a linear kernel and simply finds the optimal hyperplane for separating the two classes of data. An SVM which uses a more complex kernel, such as the Radial Basis Function (RBF) kernel, will transform the input feature vectors into a different dimensional space to allow for a more complex division of data. RBF kernels take longer to train and classify but in some cases they can provide more accurate classification for data which is not linearly separable.

I trained SVMs to to perform the six binary classification tasks listed in Section 3.3. Again, each training and test image is represented by 256 GIST features. SVMs, regardless of which kernel is used, have a penalty parameter, C which determines how strictly the dividing hyperplane should be drawn. For example, it may be desirable to draw the hyperplane so that no training data winds up on the wrong side

of the line, or it may be desirable to draw the hyperplane to maximize its distance to the most data points even at the cost of some mis-classification. Similarly, the RBF kernel has an additional γ parameter which controls how closely classifications bind to specific data points or how large of an area around a given data point will be classified to match that point. Constructing an optimal SVM to classify a given data set requires varying both C and γ to determine which values yield the best results.

In training SVMs to classify the dog-walking images, I followed the steps recommended by Hsu et al. in "A Practical Guide to Support Vector Classification" [10]. For the linear SVM, I classified 322 instances of test data using C values of 2^{-5}, 2^{-3}, 2^{-1}, 1, 2, 2^3, 2^5, and 2^6. For the SVM with the RBF kernel, I varied C as I did for the linear kernel and also varied γ using the same parameters. For each SVM created using these parameters, I randomly chose 322 instances of the data for training and used the rest for test data. Ideally, I would have had a separate set of validation data to use when tuning the SVM parameters in order to guard against over-fitting. Unfortunately, my limited data set did not allow for this.

Ground Truth Counts in Train and Test Data				
Task	Small in Train	Large in Train	Small in Test	Large in Test
Area(Dog)	161	161	76	86
Height(Dog)	161	161	76	86
Location(Dog)	161	161	75	87
Area(DW)	161	161	76	86
Height(DW)	161	161	76	86
Location(DW)	161	161	52	109

Table 3.1: This table shows the number of instances in the training and test data which had ground truth classifications as small/short/low in image vs large/tall/high in image. Note that the training data was evenly split between classifications for all tasks.

The training and test sets used for the results included in this paper matched those used for K-means classification. There were 322 training images and 162 test

images. The training set was evenly divided between each classification so that there were 161 of each class represented across all tasks. Table 3.1 shows the counts for the different ground truth classifications across the train and test data. Although the training data was evenly split between the classes, the test data had more instances of the larger classes.

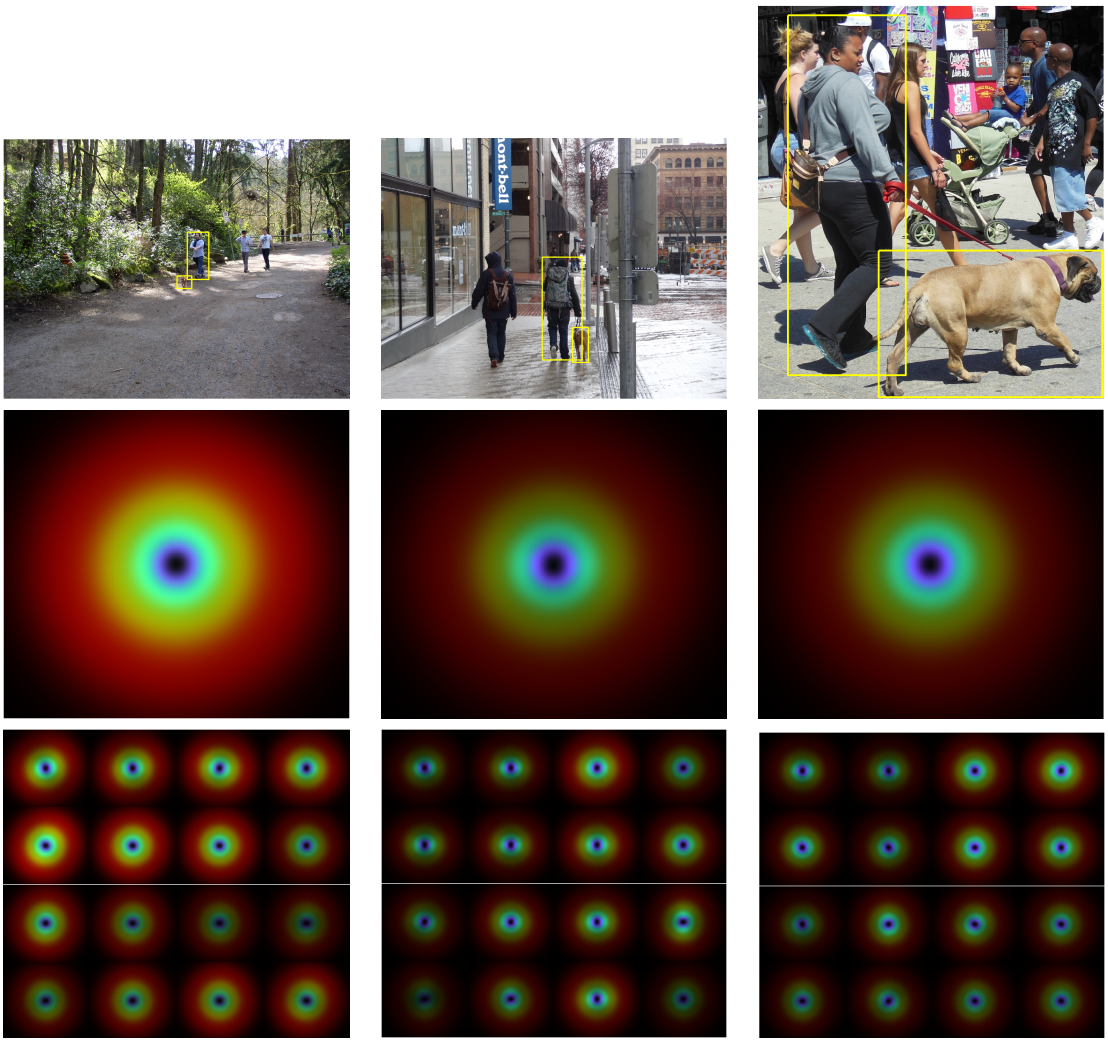


Figure 3.2: Images from the Petacat [11] image set are in the top row. The middle row displays full-scale GIST features calculated over the whole image. The bottom row shows localized GIST features taken in 16 subsections over the images. Figure best viewed in color.

(a) Dog median height, area, vertical (b) Dog walker median height, area, vertical

Figure 3.3: Illustrations of the median vertical location for the bottom of the object (blue), median height (yellow), and median area (red) for dogs and dog walkers. Images are classified as containing a dog or dog walker which falls above or below the illustrated sizes. Figure best viewed in color.

Chapter 4

Results

4.1 Results of Classification with K-means Clustering

When creating the K-means clusters, I first analyzed the entropy of each cluster as an indicator of its likely usefulness in classification. Figures 4.1, 4.2, and 4.3 show graphs of cluster entropy and percent of training data in each cluster for the 5, 10, and 15 cluster K-means results for dog walker areas (similar graphs for dog and dog walker height and vertical location can be found in Appendix A). As described in Section 3.3.1, the entropy of a cluster is indicative of how strongly one class dominates the training data in the cluster, with low entropies indicating a stronger domination by the labeling class. In the graphs, clusters which were classified as containing images with small dog walkers are colored red while clusters classified as containing images with large dog walkers are colored blue. The graphs also show the fraction of the training data which was mapped to each cluster. Clusters with smaller fractions of training data may contain so few data points that their classifications can switch with the addition of only one or two more data points.

On the 5-cluster graph, entropy correlates to cluster size with the smallest cluster also having the smallest entropy and larger clusters having higher entropies. This suggests that the larger clusters are not dominated by a single class and thus are probably less reliable as classifiers. As the number of clusters increases, this correlation falls away, suggesting that the larger cluster counts are giving us more reliable classifications. Additionally, we can see that at each cluster size, more clusters are consistently classified as small rather than large even though the training data is

evenly split between small and large instances. In the 15-cluster graph, 10 clusters are classified as small while only 5 are classified as large. This suggests that the small instances tend to spread out more evenly among the clusters and thus the classifier will be more successful on the more tightly-clustered large instances. The confusion matrix for the 15-cluster classification of test data (Table 4.1) bears this out, showing that of the large instances, 73 were correctly classified as large while only 7 were incorrectly classified as small. For small instances, 50 were correctly classified as small but 32 were classified as large. This yields 91% accuracy classifying large instances but only 61% accuracy classifying small instances.

Confusion Matrix - 15 Clusters Dog Walker Area / Image Area		
	Small	Large
Small	50	7
Large	32	73
Accuracy	**76%**	

Table 4.1: Confusion matrix for 15 cluster classification of Area(Dog Walker). The larger sized dog walkers are more likely to be correctly classified than the smaller.

The accuracy for classification on the different tasks by K-means is shown in the first three columns of Table 4.2. There is a clear improvement for classification between the 5 and 15 cluster K-means. K-means gives 70-75% accuracy on height and area for both dogs and dog walkers but performs more poorly on vertical location, giving a maximum of 67% accuracy. Unlike SVMs, K-means is an unsupervised learning algorithm, meaning that clusters are created based only on the contents of the feature vectors and not their class labels. Thus the assignment of data to clusters does not change when a different component of the image is being classified and dog size is classified using the same cluster configuration that is used to classify dog walker area. Given this, the poorer performance on vertical location suggests that vertical location is not as strongly correlated with height and area as height and area are to

each other.

Accuracy per Classification Method					
	5 Clusters	10 Clusters	15 Clusters	Linear SVM	RBF SVM
Area(Dog)	66	70	71	69	**73**
Area(DW)	68	68	76	74	**77**
Height(Dog)	71	69	72	75	**78**
Height(DW)	67	66	73	67	**76**
Location(Dog)	67	63	60	**80**	79
Location(DW)	60	65	67	77	**79**

Table 4.2: Percent accuracy on test data for classification with K-means using 5, 10, and 15 clusters and SVMs using linear and RBF kernels. Best accuracy results are in boldface. SVMs with RBF kernels returns the best classification results, except for Location(Dog) where it is outperformed by the SVM with the linear kernel.

Seeking to improve on the accuracy of K-means classification, I also looked at how classification could improve if I only classified test data which matched low-entropy clusters. Figure 4.4 shows how accuracy on the Area(Dog Walker) task changes as high entropy clusters are eliminated from classification (similar graphs for height and vertical location are in Appendix B). To create this figure I first evaluated classification accuracy using classifications from all clusters, regardless of entropy. These are the values which classify 100% of the test data on the far right of the graph. I then calculated classification accuracy and percentage of test data classified if test data which matched the highest entropy cluster was ignored during classification. I continued eliminating the highest entropy cluster and re-calculating classification accuracy until only the lowest entropy cluster remained. The furthest point to the left of the graph represents classification based only on test data which maps to the lowest entropy cluster. This process was repeated for 5, 10, and 15 cluster K-means clusterings.

As we would expect, classification accuracy improves as high-entropy clusters are removed from consideration, achieving close to 90% accuracy on 40% of test data

for 15 clusters. This shows that we can achieve higher accuracy classification with a subset of our test data. This more refined accuracy could prove valuable if K-means classification probabilities are being combined with other image salience methods as K-means classification results can serve as a more precise indicator of how heavily the K-means classification should be weighted. For example, if an instance is classified as small by a cluster which has 65% accuracy, it would have less influence in the final model than if it were classified as small by a cluster with 90% accuracy.

4.2 Results of Classification with Support Vector Machines

Classification with support vector machines improves on the classification accuracy of K-means (see last two columns of Table 4.2). The linear kernel gives similar to slightly worse results for height and area, but improves significantly over K-means for vertical location. The RBF kernel gave the best results on all tasks except Location(Dog), performing 2-4% better than both the linear kernel and 15 cluster K-means. The task on which the SVM with RBF kernel has the worst accuracy is Area(Dog) with 73%. This may be due to the large variance in dog size across different breeds. Additionally, dog shape changes much more significantly than human shape depending on the dog's orientation, moving from a horizontal rectangle when a dog is facing left/right to a vertical rectangle when it is facing front/back. I would expect that these two factors would combine to make the classification of dog area a more difficult task than that for humans.

ROC curves (see Figure 4.5 and Appendix C) show the RBF slightly outperforming the linear kernel and that both kernels maintain AUCs between 0.74 and 0.86.

SVMs offer a slight improvement in classification accuracy over K-means on the tasks of predicting bounding box area and height, but their primary benefit is that they allow for very good classification of object vertical location relative to K-means.

This is likely because SVMs are supervised learners and the SVMs classifying vertical location have been trained to do exactly that. The K-means classifiers rely only on existing associations within the data and it appears that height and area correlate to these associations far better than vertical location does.

4.3 Discussion

Classification by SVMs with RBF kernels consistently outperformed K-means with cluster sizes between 5 and 15 and SVMs using linear kernels. SVMs with linear kernels performed similarly to K-means with 15 clusters. Although the SVMs gave better results, K-means can provide more detailed data about how accurately we can classify a particular instance based on the accuracy of the cluster to which the instance is mapped. This more detailed gauge of classification accuracy would allow for more accurate weighting of GIST-sourced location and size information when combining with other object detection methods.

The end goal of this work is improved image pre-processing for object detection in the Petacat project. Petacat currently uses salience models based on those of Elazary and Itti [3]. We have models for both *general salience*, locations within the image that are likely to contain some sort of object, and *object-specific salience*, locations likely to contain a specific object. The salience models compute a salience map based on combining many local features to give each pixel in the image a probability of being a part of the sought-after object. We have object-salience models for both dogs and dog walkers which give us the probability that the object is located at a given pixel space. However, while these salience maps provide good location data, they do not provide any information about the likely size of the object's bounding box.

My results can augment our salience model by suggesting the size of dog and dog walker bounding boxes to use when searching high probability locations in the salience

map. Additionally, location information provided by GIST features can augment that provided by the salience model. For example, pixels which fall within the suggested bounding box size and vertical coordinate suggested by the GIST features could receive a boost in the salience map and be prioritized for earlier search.

If we combine probabilities from GIST features with those from the salience map, it will be advantageous to get more precise probabilities for the GIST classifications. If GIST features have classified an image as having a dog low in the image with 65% probability, this should have a smaller effect on the salience map than if the probability was 90%. Here the granularity of probability provided by K-means would prove useful because it allows us to gauge how strongly to weight classification based on GIST features in our salience map based on the classification accuracy its K-means cluster.

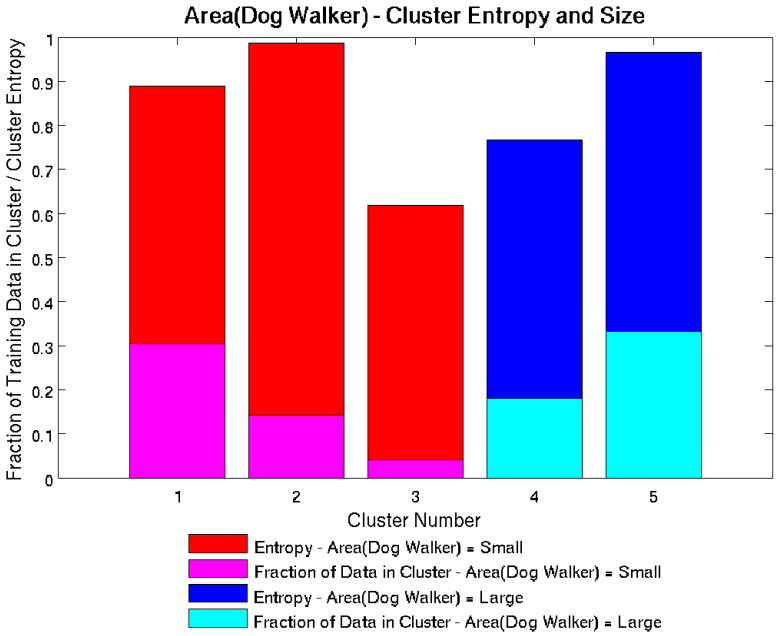


Figure 4.1: Entropy and fraction of data used for K-means with 5 clusters for Area(Dog Walker). Figure best viewed in color.

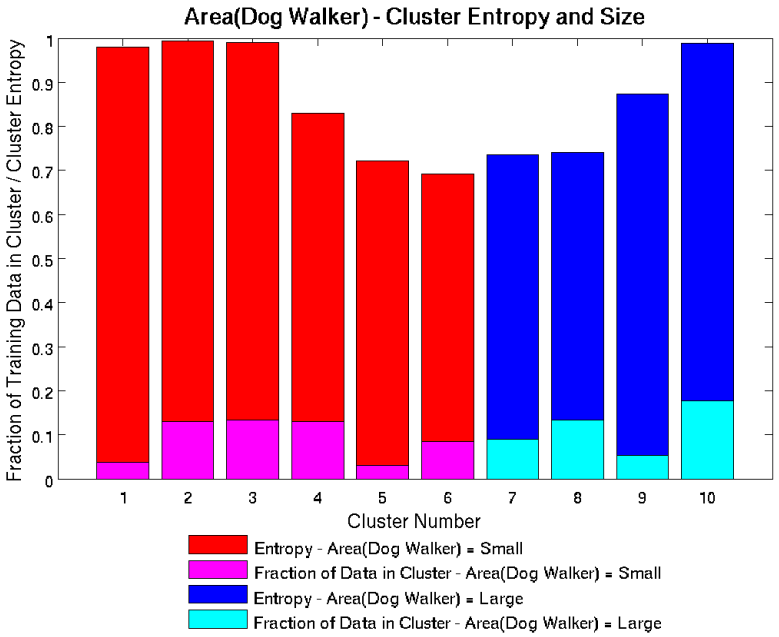


Figure 4.2: Entropy and fraction of data used for K-means with 10 clusters for Area(Dog Walker). Figure best viewed in color.

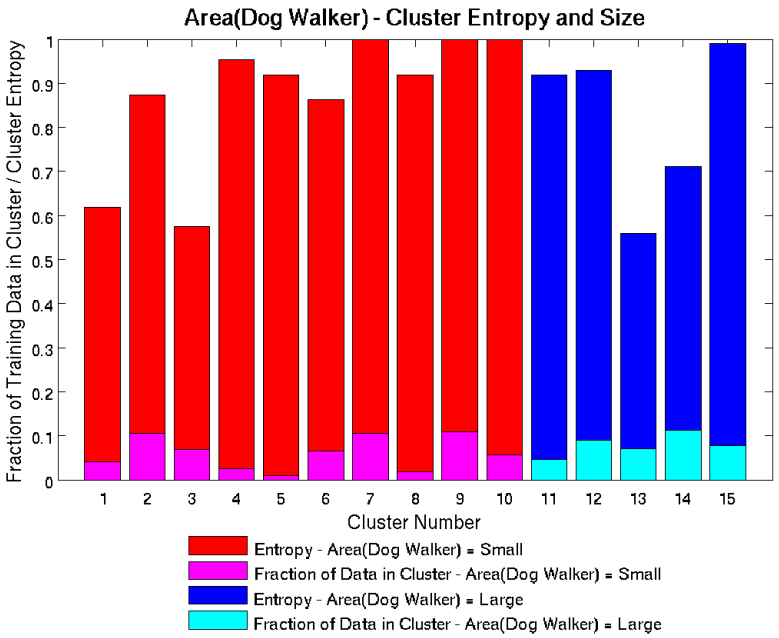


Figure 4.3: Entropy and fraction of data used for K-means with 15 clusters for Area(Dog Walker). Figure best viewed in color.

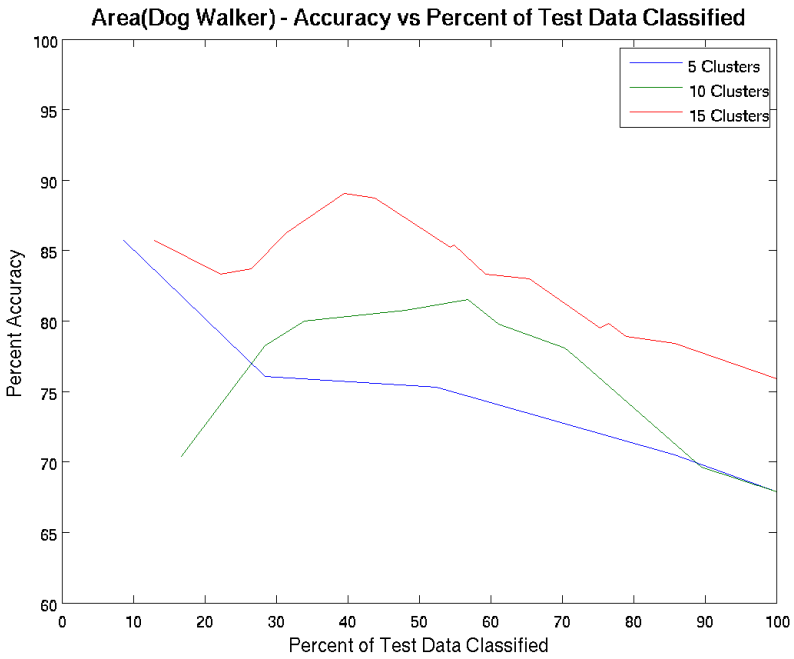


Figure 4.4: Accuracy of classification based on K-means clusters vs the percent of the test data classified. Moving toward the left of the graph, high entropy clusters are eliminated from consideration and the percent of data classified diminishes as the accuracy increases. Figure best viewed in color.

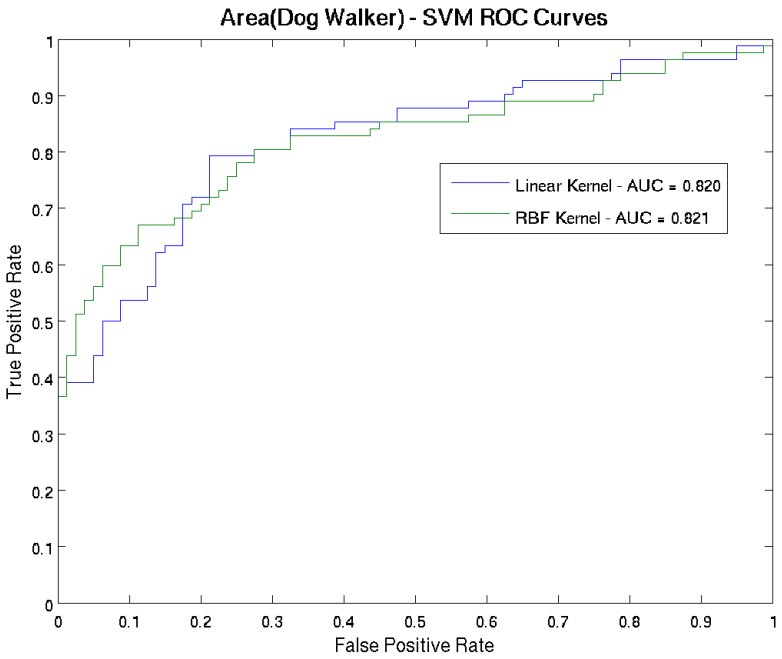


Figure 4.5: ROC curve classifying images based on the area of the dog walker's bounding box for SVM classifiers using linear and RBF kernels. A "positive" classification means classifying the image as having a small dog walker. Figure best viewed in color.

Chapter 5

Conclusions and Future Work

5.1 Conclusions

I hypothesized that GIST features could be used to classify images based on object size and location. Knowing the likely size and location of an object will limit the number of windows searched during object detection by making early detection more likely. Using both K-means and SVM classifiers, I classified images according to object size and location. I used classifiers to distinguish images having large or small dogs and dog walkers, short or tall dogs and dog walkers, and dogs and dog walkers positioned high or low in the image. I found that the best methods in my investigation can use GIST features to predict the size and location of dogs and dog walkers within the images with 73-80% accuracy.

These results are not outstanding but can still be useful for constraining the windows to be searched in the process of object detection. They have the potential to augment existing salience maps in the Petacat project by suggesting bounding box size.

There are several possible areas for future work, in particular obtaining a larger set of training data, which may lead to improved classification accuracy and allow for classification into more than two size and location classes.

5.2 Future Work

5.2.1 Classification Using K-Nearest Neighbors

The original GIST paper [13] uses K-nearest neighbors to classify instances. K-nearest neighbors is very cheap to compute and could be useful if it gives results comparable to those returned from the support vector machines. It also has the potential to allow for more granular classification. For example, if the K nearest neighbors of an instance all fall within a small window of dog walker height, we may be able to classify the dog walker height within a narrow range of these dog walker heights. Using K-nearest neighbors could allow us to classify the data into three, four, or five different size and location ranges rather than two.

5.2.2 Larger Data Set

While Oliva and Torralba [13] had a data set of 8100 images, 6000 of which were used for training, my data set was limited to 484 images. Having only 484 images in the data set and 322 in the training set meant that I was limited in the number of GIST features I could generate for classification. If I used too many features, I could not be sure that the current dog-walking data set would give a range of values across each element in the feature vector, making the set of training data too disparate to accurately map new instances in the test data. Although an argument could be made that my feature set size of 256 should be too large to allow the 322 training image to give good coverage over the range of features, I found that I achieved the best classification accuracy with 256 feature vectors. Given that my results improved as I added more GIST features until I reached 256, it is likely I could get better results if I were able to use more than 4 filter orientations or more than 16 subsections for

each image.

With more data, I could also use more than 15 K-means clusters. My current data shows improved classification as more clusters are used, but with this limited data set, using more clusters would spread the data too thinly. Additional clusters would likely allow for more accurate classification. Additional clusters could also allow for more granular classification, allowing classification by size into more than two groups. Classifying images into three or four categories rather than two for each task would further constrain the window sizes and locations that need to be searched.

5.2.3 Feature Selection

Without needing a larger image set, it may be possible to use more GIST features by applying some form of feature selection. This would eliminate from the GIST feature vector any features which are not relevant to classifying objects according to size and location. Feature selection could allow me to gain both better classification with the current feature set by eliminating noise in the feature set. It may also allow me to use a larger initial set of GIST features, perhaps 512 rather than 256, which would then be pared down to a smaller size.

5.2.4 Depth from GIST Features

As mentioned in section 2.3, Oliva and Torralba have a follow-up paper [16] demonstrating how GIST features can be used to estimate the mean depth of an image. They use this depth information to predict the likely size of a human figure's head within an image. Using depth in conjunction with location would allow us to make a very good guess at a bounding box size for objects.

It could be valuable to try to reproduce this depth estimation work and apply it

to predicting the size of dogs and dog walkers in the dog-walking image set.

Bibliography

[1] Alexe, B., Heess, N., Teh, Y. W., and Ferrari, V. Searching for objects driven by context. In *Advances in Neural Information Processing Systems 25*, P. Bartlett, F. Pereira, C. Burges, L. Bottou, and K. Weinberger, Eds. 2012, pp. 890–898.

[2] Desai, C., Ramanan, D., and Fowlkes, C. C. Discriminative models for multi-class object layout. *International Journal of Computer Vision 95*, 1 (2011), 1.

[3] Elazary, L., and Itti, L. A bayesian model for efficient visual search and recognition. *Vision Research 50*, 14 (2010), 1338 – 1352. Visual Search and Selective Attention.

[4] Felzenszwalb, P. F., Girshick, R. B., McAllester, D. A., and Ramanan, D. Object detection with discriminatively trained part-based models. *IEEE Transactions on Pattern Analysis and Machine Intelligence 32*, 9 (2010), 1627–1645.

[5] Galleguillos, C., Rabinovich, A., and Belongie, S. Object categorization using co-occurrence, location and appearance. In *Computer Vision and Pattern Recognition* (2008).

[6] Geusebroek, J.-M., Burghouts, G. J., and Smeulders, A. W. M. The amsterdam library of object images. *International Journal of Computer Vision 61*, 1 (Jan. 2005), 103–112.

[7] Harzallah, H., Jurie, F., and Schmid, C. Combining efficient object localization and image classification. In *International Conference on Computer Vision* (September 2009).

[8] Hoiem, D., Efros, A. A., and Hebert, M. Geometric context from a single image. In *International Conference of Computer Vision (ICCV)* (October 2005), vol. 1, IEEE, pp. 654 – 661.

[9] Hoiem, D., Efros, A. A., and Hebert, M. Putting objects in perspective. In *Proc. IEEE Computer Vision and Pattern Recognition (CVPR)* (June 2006), vol. 2, pp. 2137 – 2144.

[10] Hsu, C.-W., Chang, C.-C., and Lin, C.-J. A practical guide to support vector classification. http://www.csie.ntu.edu.tw/∼cjlin/papers/guide/guide.pdf.

[11] Mitchell, M. A scalable architecture for image interpretation: The petacat project. http://web.cecs.pdx.edu/∼mm/Petacat.html.

[12] Oliva, A., and Torralba, A. Modeling the shape of the scene: A holistic representation of the spatial envelope. http://people.csail.mit.edu/torralba/code/spatialenvelope/.

[13] Oliva, A., and Torralba, A. Modeling the shape of the scene: A holistic representation of the spatial envelope. *International Journal of Computer Vision 42*, 3 (May 2001), 145–175.

[14] Torralba, A. Contextual priming for object detection. *International Journal of Computer Vision 53*, 2 (July 2003), 169–191.

[15] Torralba, A., Murphy, K. P., and Freeman, W. T. Using the forest to see the trees: Exploiting context for visual object detection and localization. *Commun. ACM 53*, 3 (2010), 107–114.

[16] Torralba, A., and Oliva, A. Depth estimation from image structure. *IEEE Transactions on Pattern Analysis and Machine Intelligence 24*, 9 (2002), 1226–1238.

[17] Torralba, A., and Sinha, P. Statistical context priming for object detection. *International Conference on Computer Vision 01* (2001), 763.

[18] Vedaldi, A., Gulshan, V., Varma, M., and Zisserman, A. Multiple kernels for object detection. In *International Conference on Computer Vision* (2009), pp. 606–613.

[19] Viola, P. A., and Jones, M. J. Rapid object detection using a boosted cascade of simple features. In *Computer Vision and Pattern Recognition* (2001), vol. 1, pp. 511–518.

Appendix A

K-means Entropy and Cluster Size

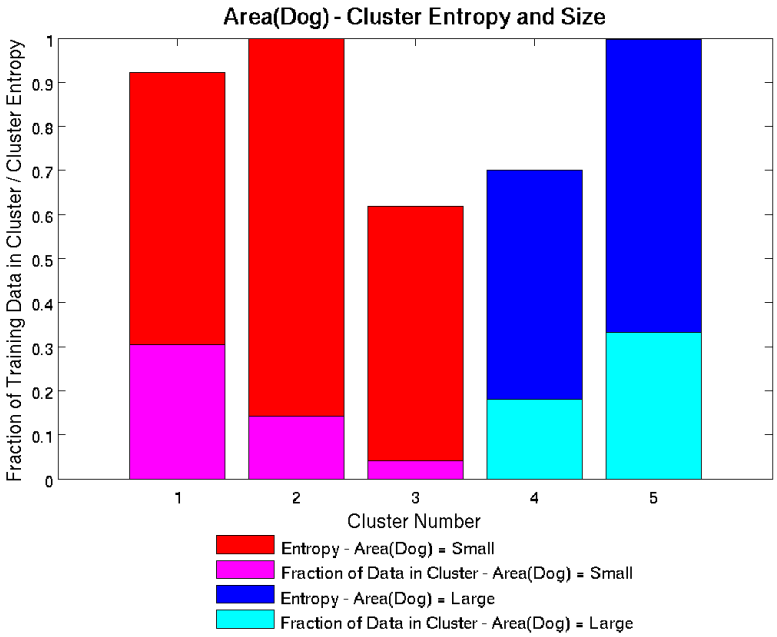


Figure A.1: Entropy and fraction of data used for K-means with 5 clusters for Area(Dog). Figure best viewed in color.

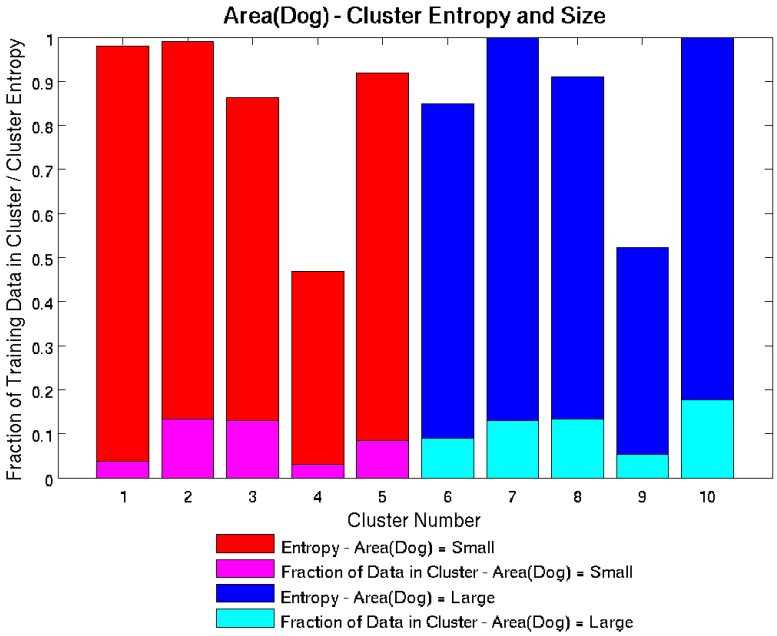


Figure A.2: Entropy and fraction of data used for K-means with 10 clusters for Area(Dog). Figure best viewed in color.

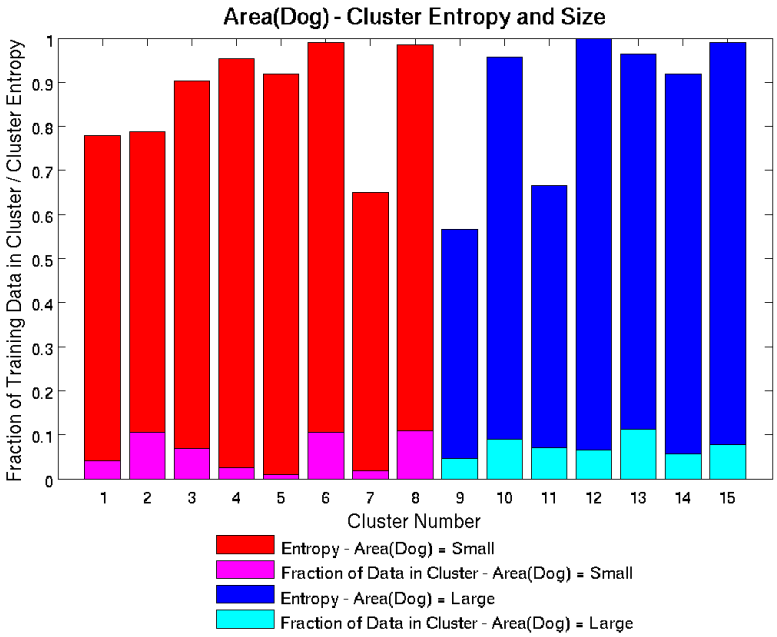


Figure A.3: Entropy and fraction of data used for K-means with 15 clusters for Area(Dog). Figure best viewed in color.

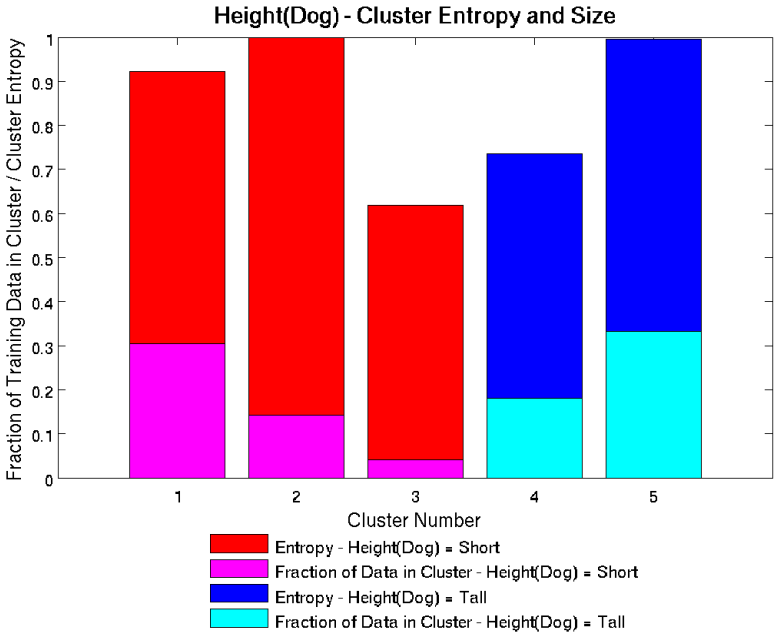


Figure A.4: Entropy and fraction of data used for K-means with 5 clusters for Height(Dog). Figure best viewed in color.

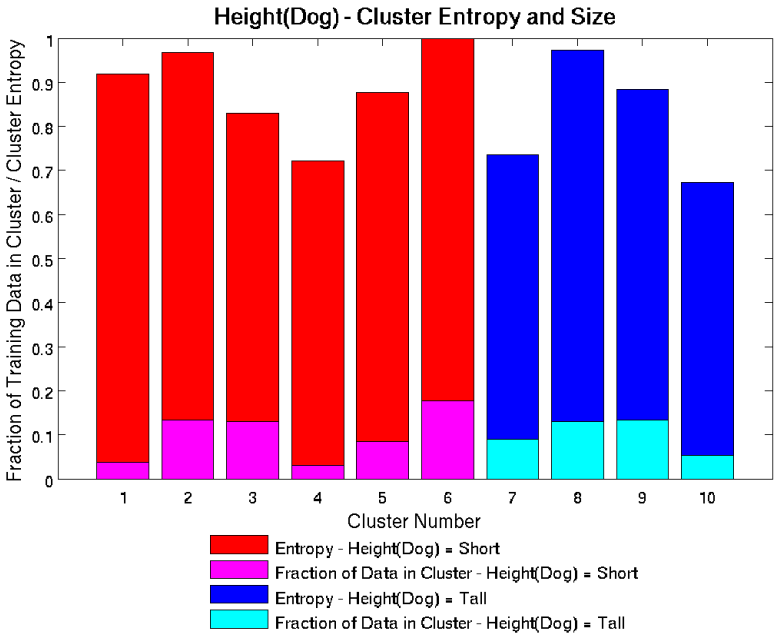


Figure A.5: Entropy and fraction of data used for K-means with 10 clusters for Height(Dog). Figure best viewed in color.

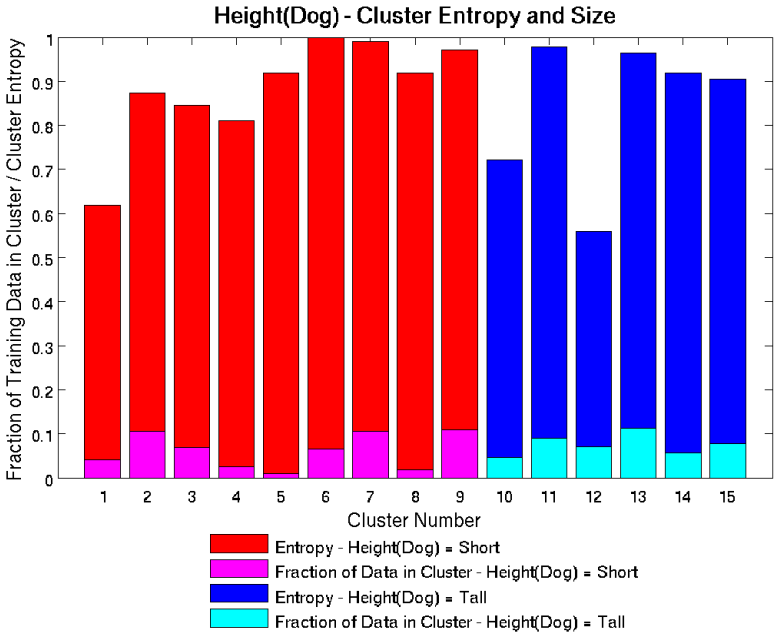


Figure A.6: Entropy and fraction of data used for K-means with 15 clusters for Height(Dog). Figure best viewed in color.

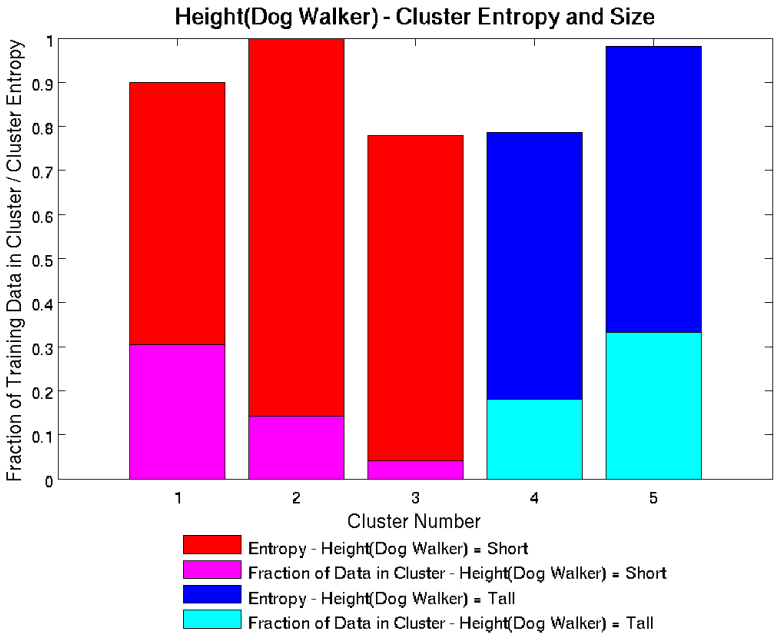


Figure A.7: Entropy and fraction of data used for K-means with 5 clusters for Height(Dog Walker). Figure best viewed in color.

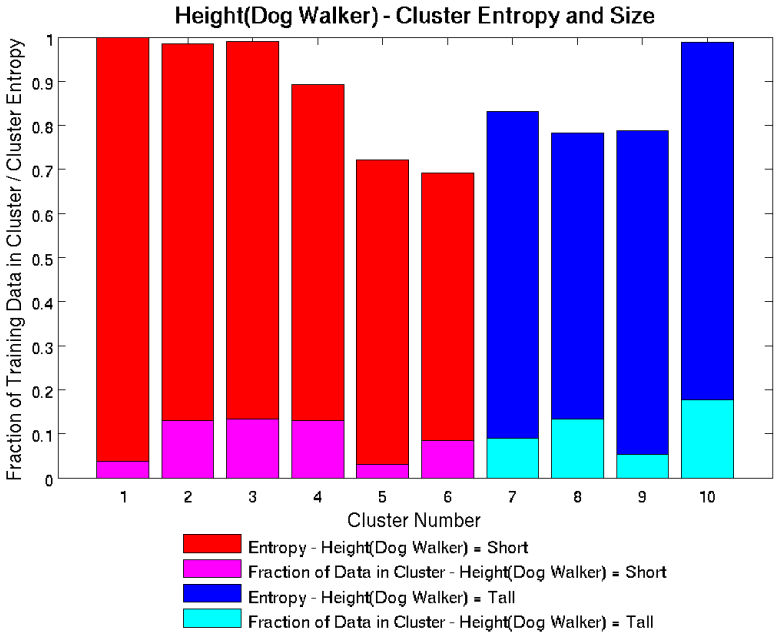


Figure A.8: Entropy and fraction of data used for K-means with 10 clusters for Height(Dog Walker). Figure best viewed in color.

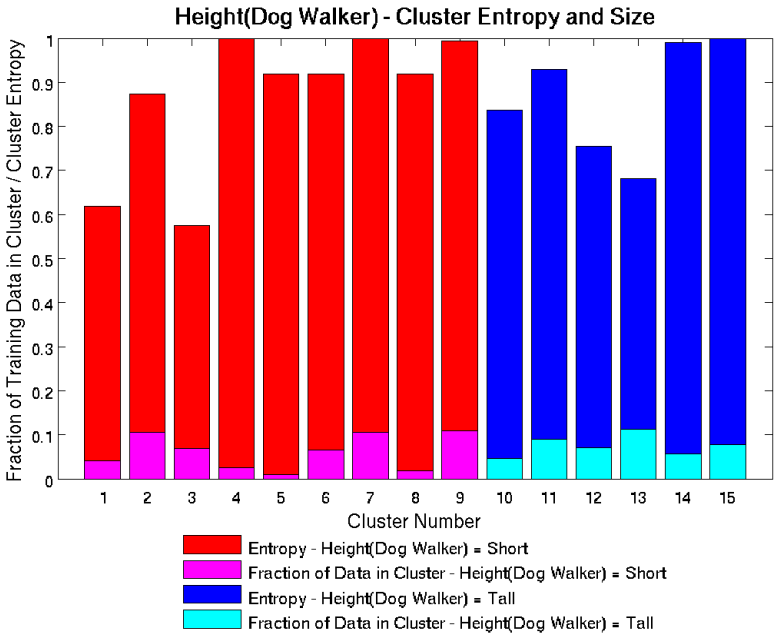


Figure A.9: Entropy and fraction of data used for K-means with 15 clusters for Height(Dog Walker). Figure best viewed in color.

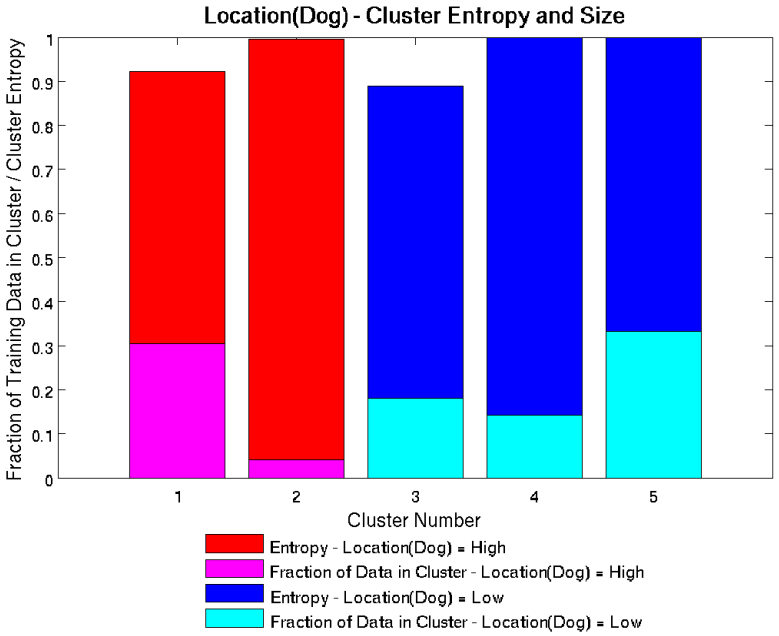


Figure A.10: Entropy and fraction of data used for K-means with 5 clusters for Location(Dog). Figure best viewed in color.

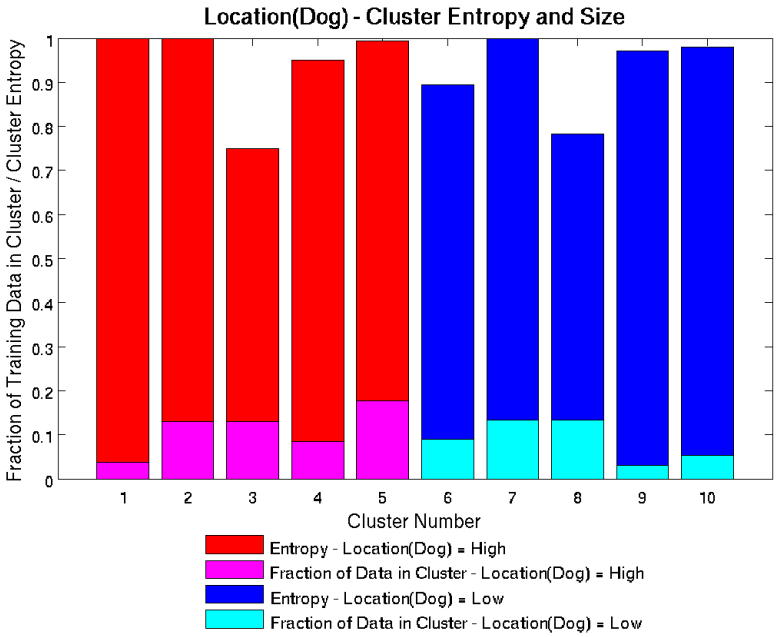


Figure A.11: Entropy and fraction of data used for K-means with 10 clusters for Location(Dog). Figure best viewed in color.

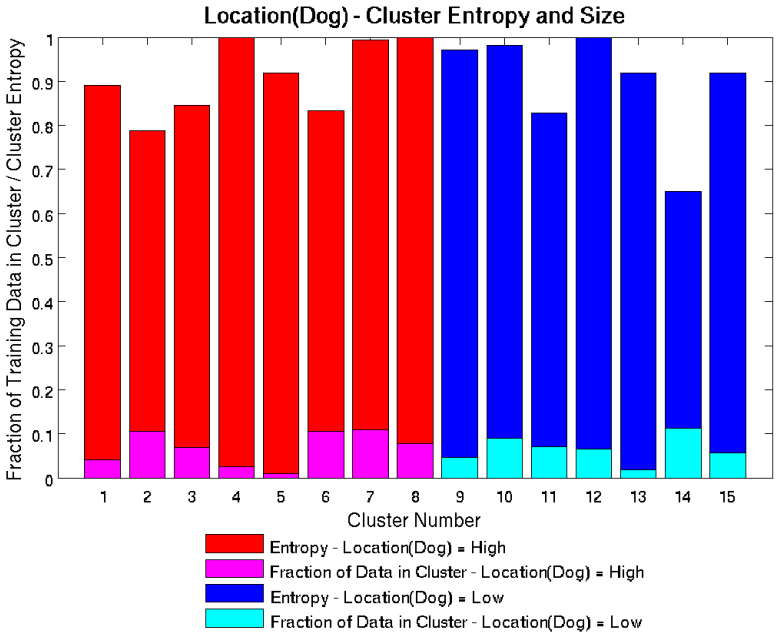


Figure A.12: Entropy and fraction of data used for K-means with 15 clusters for Location(Dog). Figure best viewed in color.

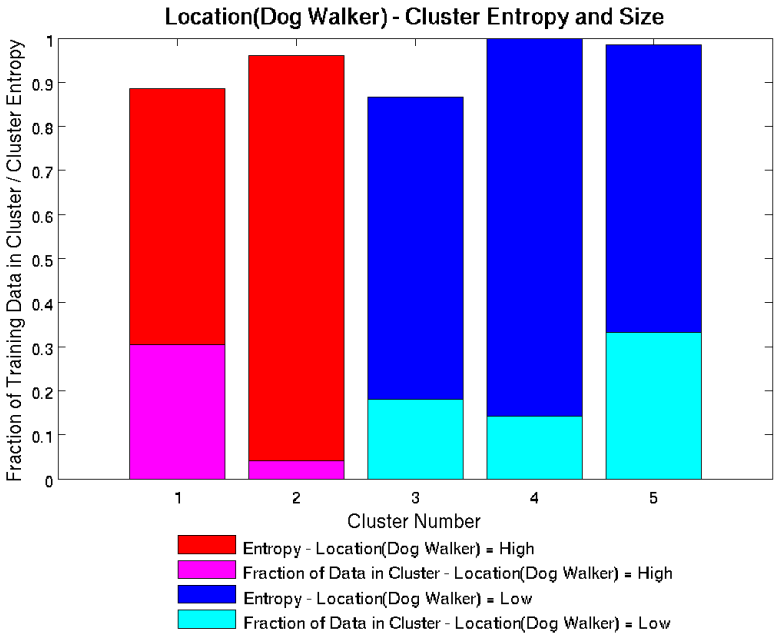


Figure A.13: Entropy and fraction of data used for K-means with 5 clusters for Location(Dog Walker). Figure best viewed in color.

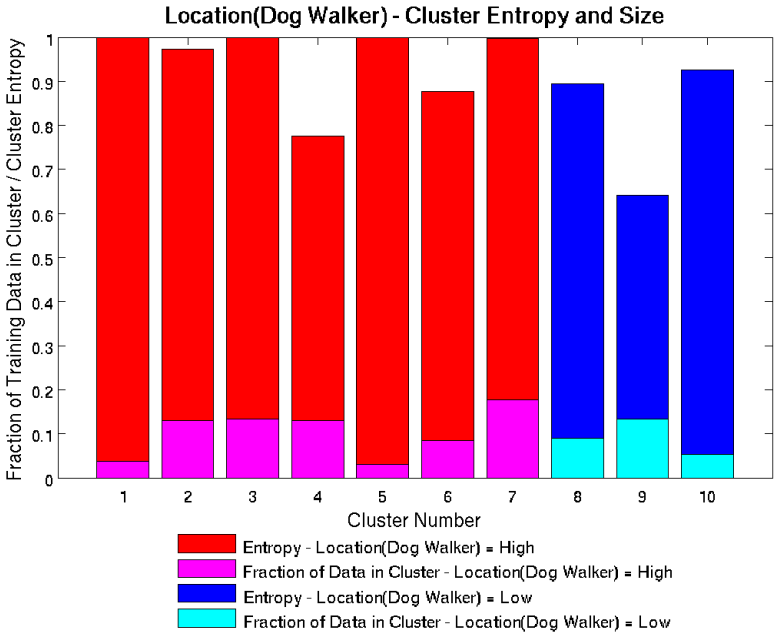


Figure A.14: Entropy and fraction of data used for K-means with 10 clusters for Location(Dog Walker). Figure best viewed in color.

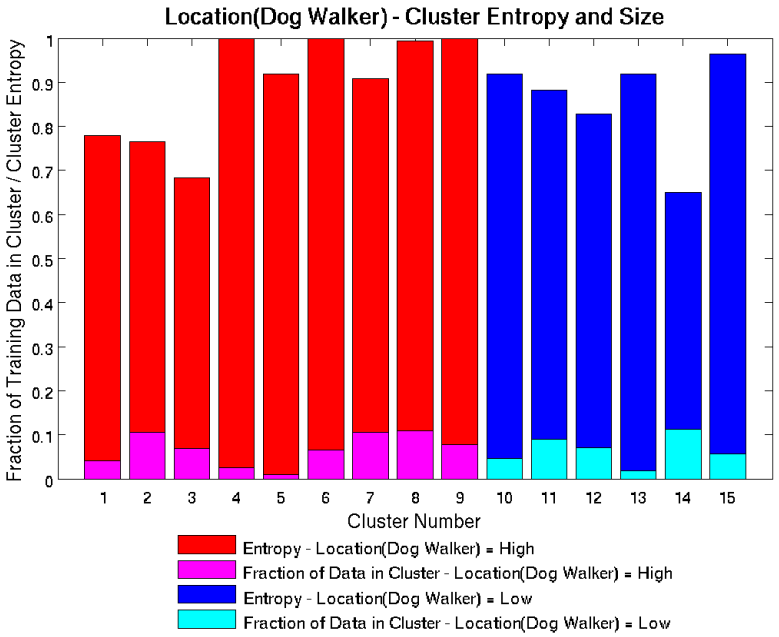


Figure A.15: Entropy and fraction of data used for K-means with 15 clusters for Location(Dog Walker). Figure best viewed in color.

Appendix B

K-means Accuracy vs Percent Test Data Classified

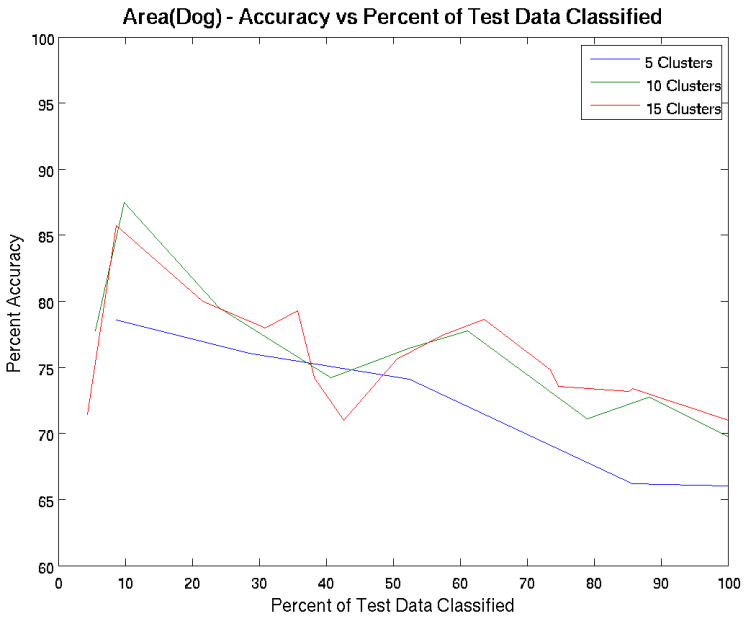


Figure B.1: Accuracy of classification based on K-means clusters vs the percent of the test data classified. Moving toward the left of the graph, high entropy clusters are eliminated from consideration and the percent of data classified diminishes as the accuracy increases. Figure best viewed in color.

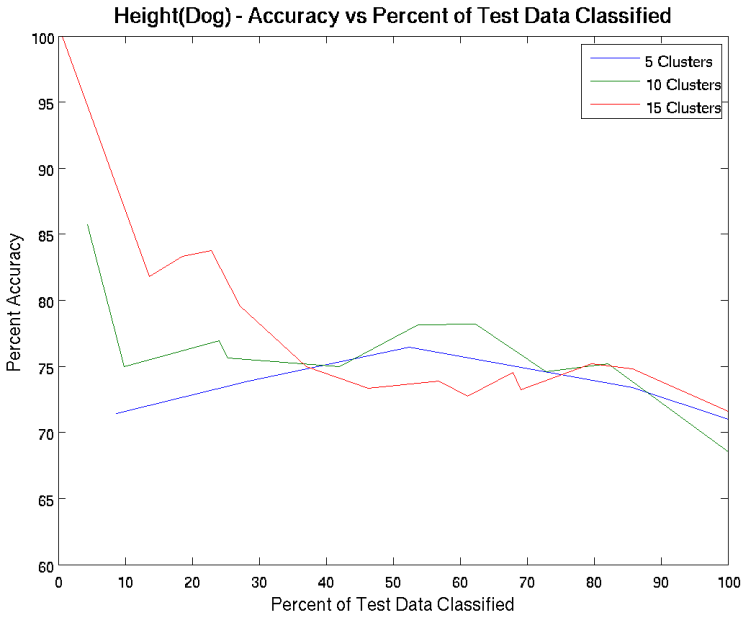


Figure B.2: Accuracy of classification based on K-means clusters vs the percent of the test data classified. Moving toward the left of the graph, high entropy clusters are eliminated from consideration and the percent of data classified diminishes as the accuracy increases. Figure best viewed in color.

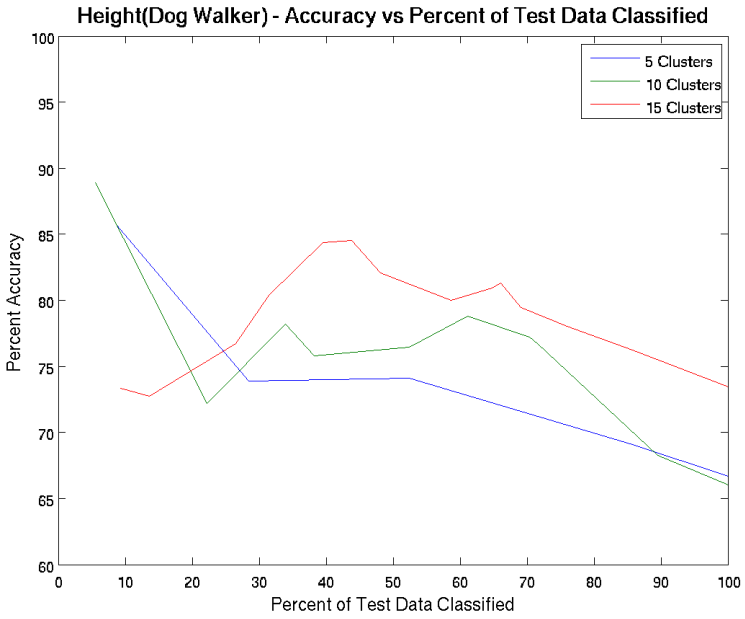


Figure B.3: Accuracy of classification based on K-means clusters vs the percent of the test data classified. Moving toward the left of the graph, high entropy clusters are eliminated from consideration and the percent of data classified diminishes as the accuracy increases. Figure best viewed in color.

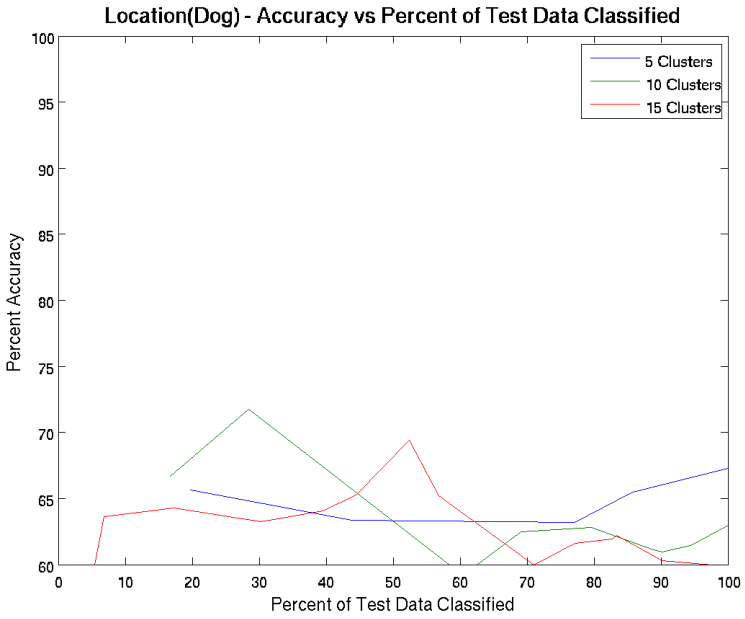


Figure B.4: Accuracy of classification based on K-means clusters vs the percent of the test data classified. Moving toward the left of the graph, high entropy clusters are eliminated from consideration and the percent of data classified diminishes as the accuracy increases. Figure best viewed in color.

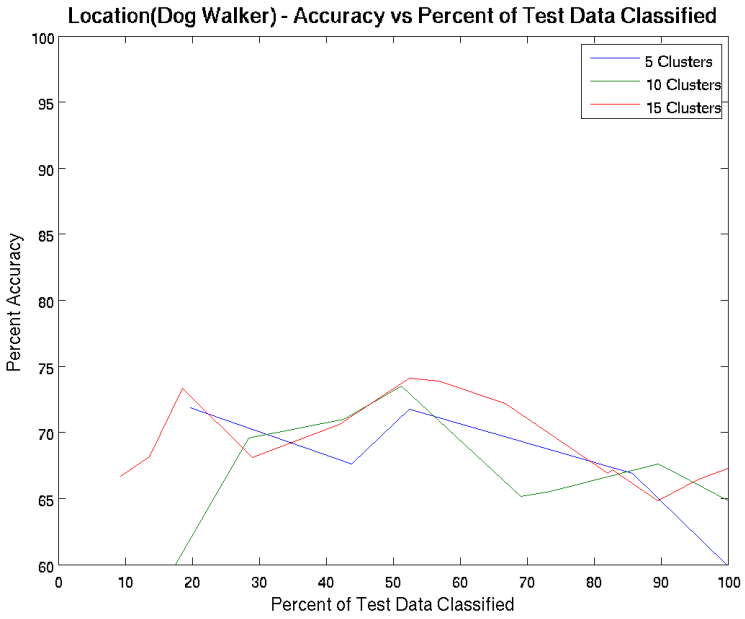


Figure B.5: Accuracy of classification based on K-means clusters vs the percent of the test data classified. Moving toward the left of the graph, high entropy clusters are eliminated from consideration and the percent of data classified diminishes as the accuracy increases. Figure best viewed in color.

Appendix C

SVM ROC Curves

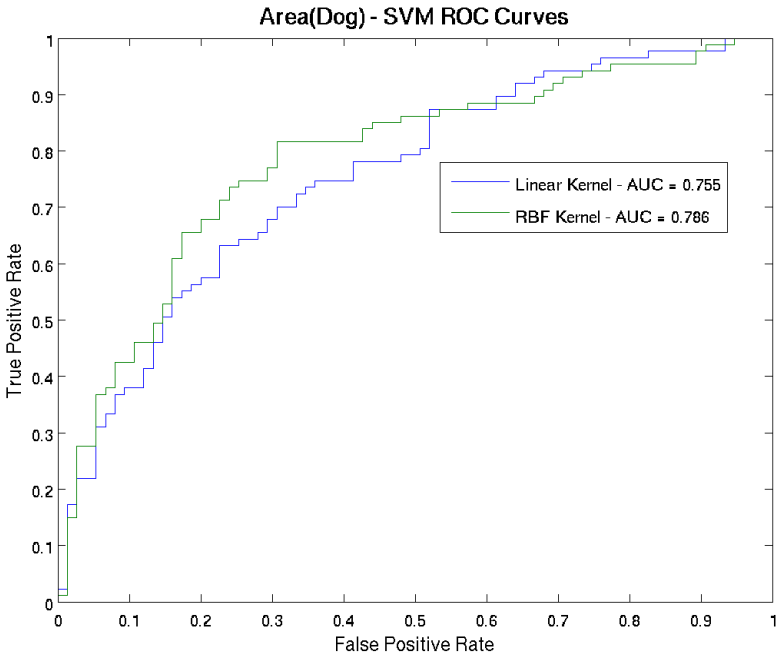


Figure C.1: ROC curve classifying images based on the area of the dog walker's bounding box for SVM classifiers using linear and RBF kernels. A "positive" classification means classifying the image as having a small dog area. Figure best viewed in color.

:

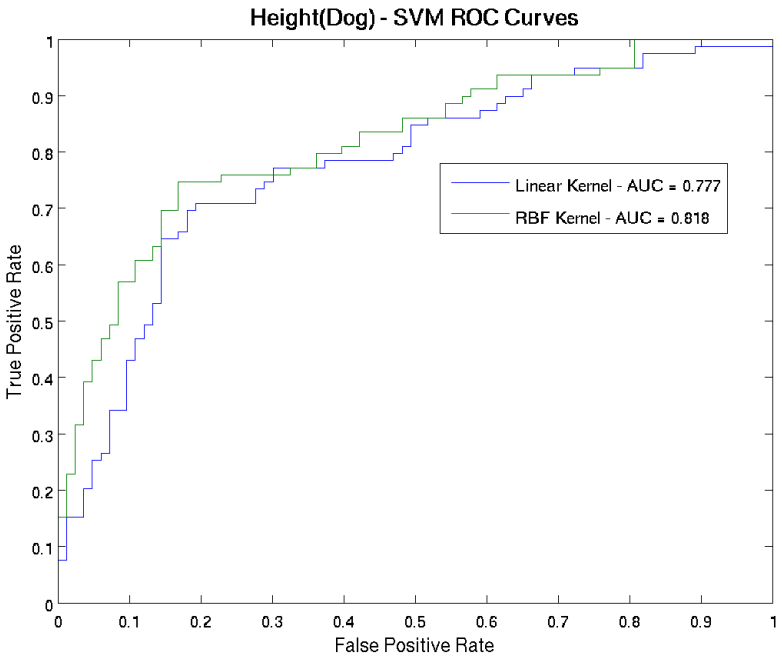


Figure C.2: ROC curve classifying images based on the area of the dog walker's bounding box for SVM classifiers using linear and RBF kernels. A "positive" classification means classifying the image as having a short dog height. Figure best viewed in color.

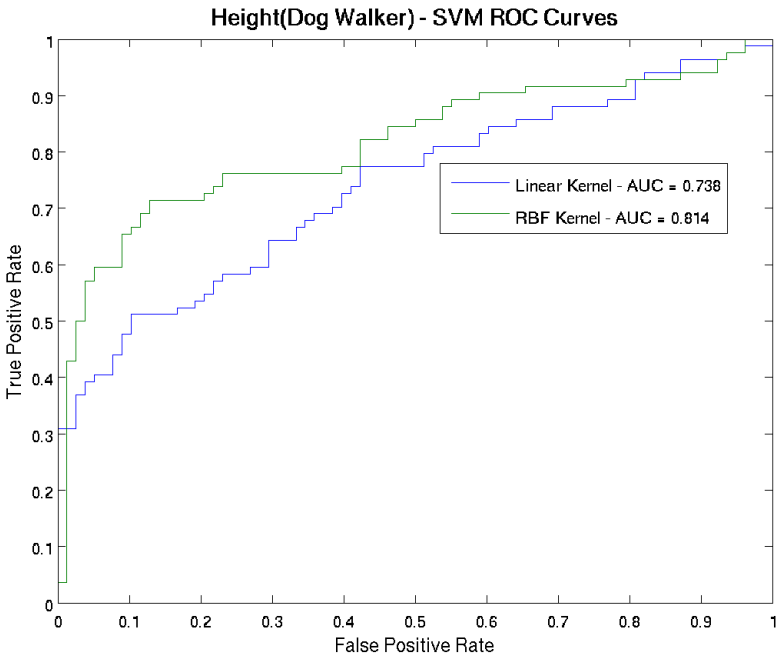


Figure C.3: ROC curve classifying images based on the area of the dog walker's bounding box for SVM classifiers using linear and RBF kernels. A "positive" classification means classifying the image as having a short dog walker height. Figure best viewed in color.

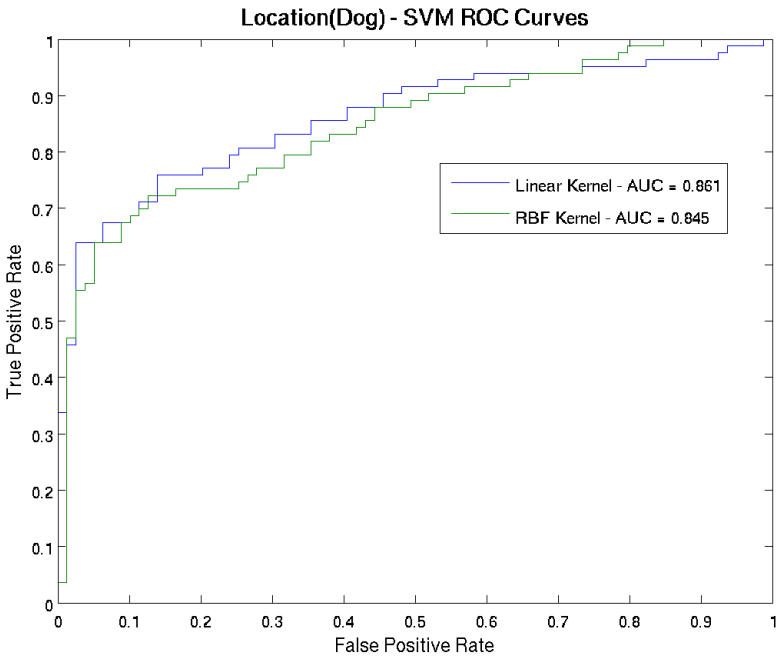


Figure C.4: ROC curve classifying images based on the area of the dog walker's bounding box for SVM classifiers using linear and RBF kernels. A "positive" classification means classifying the image as having a dog higher in the image. Figure best viewed in color.

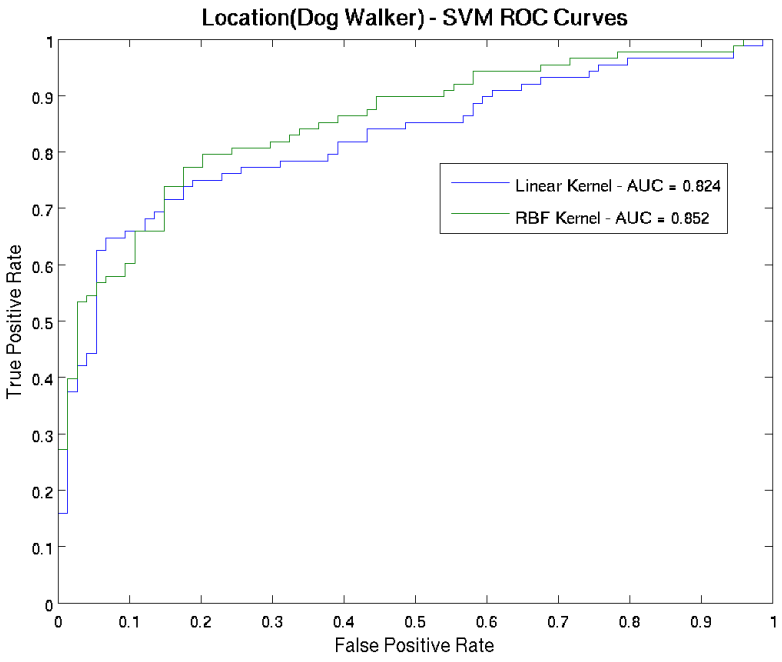


Figure C.5: ROC curve classifying images based on the area of the dog walker's bounding box for SVM classifiers using linear and RBF kernels. A "positive" classification means classifying the image as having a dog walker. Figure best viewed in color.