Classifying Everyday Activity Through Label Propagation

With Sparse Training Data

by

Vaishnav Desai

A Thesis Presented in Partial Fulfillment
of the Requirements for the Degree
Master of Science

Approved June 2013 by the
Graduate Supervisory Committee:

Hari Sundaram, Co-Chair
Baoxin Li, Co-Chair
Pavan Turaga

ARIZONA STATE UNIVERSITY

August 2013

ABSTRACT

We solve the problem of activity verification in the context of sustainability. Activity verification is the process of proving the user assertions pertaining to a certain activity performed by the user. Our motivation lies in incentivizing the user for engaging in sustainable activities like taking public transport or recycling. Such incentivization schemes require the system to verify the claim made by the user. The system verifies these claims by analyzing the supporting evidence captured by the user while performing the activity. The proliferation of portable smart-phones in the past few years has provided us with a ubiquitous and relatively cheap platform, having multiple sensors like accelerometer, gyroscope, microphone etc. to capture this evidence data in-situ.

In this research, we investigate the supervised and semi-supervised learning techniques for activity verification. Both these techniques make use the data set constructed using the evidence submitted by the user. Supervised learning makes use of annotated evidence data to build a function to predict the class labels of the unlabeled data points. The evidence data captured can be either unimodal or multimodal in nature. We use the accelerometer data as evidence for transportation mode verification and image data as evidence for recycling verification. After training the system, we achieve maximum accuracy of 94% when classifying the transport mode and 81% when detecting recycle activity. In the case of recycle verification, we could improve the classification accuracy by asking the user for more evidence. We present some techniques to ask the user for the next best piece of evidence that maximizes the probability of classification. Using these techniques for detecting recycle activity, the accuracy increases to 93%.

The major disadvantage of using supervised models is that it requires extensive annotated training data, which expensive to collect. Due to the limited training data, we look at the graph based inductive semi-supervised learning methods to propagate the labels among the unlabeled samples. In the semi-supervised approach, we represent each instance in the data set as a node in the graph. Since it is a complete graph, edges interconnect these nodes, with each edge having some weight representing the similarity between the points. We propagate the labels in this graph, based on the proximity of the data points to the labeled nodes. We estimate the performance of these algorithms by measuring how close the probability distribution of

the data after label propagation is to the probability distribution of the ground truth data. Since labeling has a cost associated with it, in this thesis we propose two algorithms that help us in selecting minimum number of labeled points to propagate the labels accurately. Our proposed algorithm achieves a maximum of 73% increase in performance when compared to the baseline algorithm.

DEDICATION

To my family.

ACKNOWLEDGEMENTS

It has been an exciting and enriching journey towards my thesis. I am indebted to several people who have contributed directly and indirectly to my thesis.

I would like to express my deep gratitude to my thesis advisors, Dr. Hari Sundaram and Dr. Pavan Turaga, for their patient guidance, inspiring thoughts and useful critiques during my masters study. Their enthusiastic attitude towards research always inspired me throughout this journey. Their approach to problem formulation and presentation of the same is a quality, which I try to emulate and has totally transformed my attitude towards research. Besides my advisors, I would like to thank Dr. Baoxin Li for agreeing to be a part of my thesis committee and providing insightful comments on my thesis.

I would like to offer my special thanks to Kuldeep Kulkarni, for providing stimulating discussions and helpful hints whenever I was stuck on a problem.

Last but not the least, this thesis would not have been possible without the support of my entire family. I wish to thank my grandparents, my parents and my brother for their endless love and unconditional support. No words are enough to express my gratitude for all that they had to sacrifice and endure so that I may have this opportunity.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

Chapter 1

INTRODUCTION

In this research, we investigate the problem of Activity Verification and provide some approaches to reduce the cost for doing so. We define verification as the process of proving the user assertions pertaining to some activity performed by the user. Formally, given a piece of evidence ξ that was captured while the activity χ was being performed by the user υ, the task of activity verification is to determine if the activity χ was performed by the user υ. Here, the evidence ξ is provided to the system by the user and can be uni-modal or multi-modal in nature. This research is applicable in places where the user needs to be rewarded based on the activity performed. We draw our motivation from the idea of promoting a sustainable lifestyle by rewarding the user for performing certain sustainable activities like recycling, eating local food, using cold water, taking public transport etc.

We can achieve global sustainable development when every one of us tries to have a sustainable relationship with one's environment, by adopting a sustainable lifestyle. Sustainable lifestyle attempts to decrease an individual's use of natural resources by reducing his/her own consumption. We can measure the consumption of these natural resources by looking at the carbon footprint of an individual. Wiedmann et al [37] defines Carbon Footprint as the measure of the total amount of carbon dioxide emissions caused directly or indirectly by activity performed by a person. The increasing level of carbon dioxide in the atmosphere can lead to climatic change that will have a major impact on the world in the coming decades. Each one of us can reduce our carbon footprint by adopting alternative behaviors during a certain time intervals of the day. An average American causes around 140 pounds of carbon dioxide equivalent emissions a day. US government estimated [7] a 20% reduction in the emission to reduce the impact of climate change. We can adopt some of the alternate activities like taking public transport or carpooling, which can save up to 9.4 pounds of carbon dioxide per day, eating locally grown food which can save up to 1.5 pound per day, recycling which can save up to 2.15 pounds of carbon dioxide per week, avoiding beef which can save up to 4.26 pounds of carbon dioxide per day, air drying clothes which saves up to 2 pounds of carbon dioxide per day etc. As we can notice sustainable living not only takes into consideration the transportation mode used by the person, which has the largest carbon footprint but also takes into account the energy consumed by an individual and also the type of diet a person is on etc. All these activities aid in

sustainable as well as healthier lifestyle. With the ubiquitous presence of smart-phones having a large array of sensors like accelerometer, barometer, microphones etc., we can easily record the impact a person has on the environment. Most of the activities like recycling verification or verifying if the person has had beef are inherently hard problems due to the lack of proper supporting evidence, and we might need human intervention for successful verification of the activity.

In the present research, we tackle two of the above-discussed problems - transport mode verification and recycling verification. We first investigate the supervised learning approaches, which requires us to provide the system with a large amount of training data. In transport mode verification problem, we extract features from the evidence collected using the tri-axial accelerometer present in the smart-phone and in recycle verification problem, we extract features from the image data provided by the user as evidence. The evidence recorded by the accelerometer is usually suggestive of the activity performed by the user, and we can achieve high levels of accuracy during the verification process. On the other hand, in recycling verification problem we might need additional evidence form the same or different modality to increase the accuracy. We investigate the image modality to detect the recycle bin object for verification. We train three classifiers on the features exacted from the evidence - Naive Bayes [44], decision trees[38] and random forest[46]. We get the best case accuracy of 94% using random forest for detecting public transport and 81% accuracy using random forest for detecting recycle bin from the image. One of the major challenge faced while implementing supervised learning techniques is the lack of training data. To tackle this problem we look at some of the semi supervised learning approaches, where we assume the number of labeled instances to be extremely small when compared to the unlabeled instances. We propagate the labels from this small set of labeled data to other unlabeled points based on the nearest neighbor technique. Accurately labeling all the data-points requires proper selection of the labeled data-points from the given data set. We incur a cost every time we pick points from the unlabeled data set and label them by asking the oracle. We propose some techniques to choose the best set of labeled data-points from the set of all the unlabeled data-points to reduce the classification error and the incurred cost.

We organize the rest of the thesis into four Chapters. In Chapter 2, we investigate the supervised learning techniques to tackle the transportation mode and recycling verification problem. In Chapter 3, we look at some of the challenges faced while using these supervised

approaches on the recycling problem and suggest a few methods to overcome this by making the system ask for additional evidence. In Chapter 4, we investigate the semi-supervised learning approaches to solve the recycle verification problem. In the final Chapter, we conclude the work and explore the scope for future research in this area.

Chapter 2

SUPERVISED LEARNING APPROACHES

Machine learning algorithms are widely used for many applications in the field of activity recognition as they help us learn the distribution of the data to form the model, which we can use to predict unknown data attributes. The algorithms can fall either in supervised or unsupervised category. Unsupervised approaches do not need any labeled instances to form the model, and work on the principle of density estimation. Clustering is a popular unsupervised learning technique that we will be making use of in Chapter 4. Supervised learning techniques on the other hand require a set of labeled training data to learn the mapping function, which maps the data point to its label. With supervised learning approaches, there is always a risk of over-fitting the data, which we overcome by making use of cross-validation techniques. Cross-validation is a technique of partitioning the data in multiple sets and running classifiers on combination of these sets to provide us with a good estimate of the predictive error. We first look at the supervised models for transportation mode detection and recycling verification problem.

United States Environment Protection Agency (EPA) [7] provides us statistics which indicate that using public transport, carpooling or even walking can help us drastically reduce the amount of carbon dioxide (CO_2) emissions. According to calculations by EPA an individual can save up to 984 pounds of (CO_2) equivalent annually just by choosing a smart way to commute. We summarize the amount of savings in (CO_2) when we choose alternate mode of transport in Table 2.1 [7].

	Taking public transport (Bus, Rail)	Car Pooling (with 3 other people)	Walking or Riding a bike
Annual Savings (given in pounds of carbon dioxide equivalent)	984 lb	738 lb	328 lb

Table 2.1: Summary of all the annual savings of an individual in pounds of Carbon Dioxide (CO_2) equivalent. We compute these values assuming the user uses these alternative transport means for 35 workweeks a year. We assume, for walking/biking activities, the user travels 1 mile/day, and for bus/car, the user travels about 3 miles/day with fuel efficiency of 22.4 miles/gallon.

In order to promote these alternative behaviors, we reward the user for performing them. Rewarding the user is a tricky task, as we need to verify their assertion. As discussed previously, the user supports his assertions by providing evidence to the system. Due to the widespread adoption of smart-phones, it is easier to collect this evidence in-situ. In our research, we assume the user records the accelerometer data on his/her smart-phone and sends it to the system for verification. Almost all the smart-phones have a tri-axial accelerometer, which records the acceleration along the breadth of the phone (X-axis), along the length of the phone (Y-axis) and along the axis perpendicular to the surface of the phone (Z-axis). We train the classifiers on the feature vectors extracted from this accelerometer data. We have observed that supervised techniques provide high accuracy for detecting the public transport. We discuss the implementation details in Section 2.3.

We also investigate recycling, which is another common activity that can help us lead a more sustainable life. Recycling as we can see from the Table 2.2 [7] could help us save up to 90 pounds of CO_2 emissions per year per person.

Action	Average number of pounds of CO_2 equivalent per person per year that could be saved.
Magazines	14.86 lb
Newspaper	89.76 lb
Glass	6.83 lb
Plastic	18.96 lb
Aluminum and Steel cans	86.05 lb

Table 2.2: Amount of Carbon Dioxide gas (CO_2) in pound, an individual can save per year by recycling different objects.

Verifying if the user has performed the recycling activity is an inherently hard problem, as we do not have any evidence that is suggestive of this activity. In order to verify recycling activity accurately, we need to ask the user to provide us with as much supporting evidence as possible. Some of the different pieces of evidence that the user could provide us are image, audio, video and GPS. Once the system for verification is operational, we can make use of the user's history of supporting evidence in order to verify the recycling activity. In this re-

search, we ask the user for the image evidence and try to identify the recycle bin in the image to prove the user assertion. We experiment with different features, some of which are affine and scale invariant to recognize the recycle bin in the image provided by the user. We discuss the implementation details in the Section 2.4.

We organize this chapter is into three subsections. In the first section, we discuss some of related work done on this topic followed by the problem definition. In the next couple of sections, we look at the two activities we are trying to tackle using the supervised methods; recycling verification and transport mode verification. Finally, we conclude the chapter by providing some improvements.

2.1 Related Work

Transport mode detection has been studied using both the body mounted sensors and the using the sensors present on the smart-phones. In the literature, human ambulatory motions were effectively classified using different sensors like pedometer, GPS, gyroscope and accelerometer. In the paper by Murat et al. 2005 [21], two waist mounted pedometers are used to detect the walking speed of the user. Even though the paper shows a good accuracy in classification, pedometers offer a limited amount of information and cannot distinguish between activities like bus or rail. Another popular sensor that is used for activity recognition is bi-axial and tri-axial accelerometer. Early work done by Randall et al. 2000 [23] discusses the use of single bi-axial accelerometers which are positioned in the front trouser pocket of the user to achieve accuracy of about 85%-90%. The paper detects only walking, sitting and running activities. Since then, researchers have used multiple accelerometers for activity recognition in order to classify a broader range of activities. In the work by Nicola et al. [4], sixteen different classes were considered by them which includes car driving, walking, running, climbing etc. The paper makes use of three body worn sensors to record the acceleration. The paper attains classification accuracy of 80%. Bao et al. 2004 [3], worked on using five body mounted bi-axial accelerometers to classify standing, running and walking activities. They achieved accuracy of 80% using leave-one-out validation with the Decision Tree classifier. Their paper also discusses the different recognition accuracies obtained when the positions of the body-mounted accelerometer changes. Most of the work done using the body-mounted sensors is impractical due to the obtrusive nature of these devices.

In the recent past, we have observed tremendous amount of progress towards making activity sensing as portable as possible. Akos et al. 2008 [8], tries to make use of portable

wrist devices containing accelerometers, gyroscope and magnetometer which connect to the mobile phones using wireless Bluetooth connection. They achieve an average recognition rate of 79.7%. Lester et al. [11] used a mobile sensing platform (MSP) to record the sensor data along with the GPS traces, to recognize if the user was in car, bus or was walking or riding the bike. This is one of the earliest work, which extends the scope of the number of activities. The major disadvantage of this technique is that the user need to place the MSP on the waist for it to work. Even though transmitting the data acquired by the portable sensing device to the mobile phone was less obtrusive, it still makes it hard to use for everyday sensing of activities in-situ. We next investigate the work done using the smart-phones.

Smart-phones have become ubiquitous and help in collecting data from the user unobtrusively while the user is performing the activity. Work by Anderson et al. [2] was one of the earliest to make use of the mobile phone alone to detect the transport mode using the behavior of GSM signal strength. They achieve about 82% accuracy using Hidden Markov Models in an urban environment. A similar study was performed by Sohn et al. [29] using the GSM signal, and they obtained an average recognition rate of 85%. These approaches cannot distinguish between more fine-grained activities taking a bus and a rail. In addition, the GSM readings are dependent on the cell network density hence it does not provide a reliable reading. Mun et al. [19] shows us that we could compensate for the large GSM cell sizes with the small cell size advantage obtained using WiFi. They attain overall accuracy of 88% on three activities - dwelling, walking and driving. A recent paper by Miluzzo et al. [17] discusses an application called CenceMe. ChanceMe captures the accelerometer data along with audio and GPS data to infer the state of the individual. Using this application, they achieve classification accuracy of 68% when detecting if the user is using a vehicle or not. A much comprehensive experimentation was performed by Reddy et al. [25] where they try to classify still, walk, run, bike and motor activities using the data obtained from the accelerometer and the GPS sensors. They get average classification accuracy of 94% using a two-stage classifier based on Decision Trees and Discrete Hidden Markov Model.

In this chapter, we also discuss some techniques to perform recycle verification. Our goal is to detect if the recycle bin is present in the image submitted to the system. Our assumption here is that an image with the recycle bin in it is indicative of the activity performed. The problem of object detection is a topic of great interest in the field of computer vision. We make use of color histograms to compare if the two images are equal. Color histograms provide the

spatial distribution of color in the image. The paper by Swain et al. [31], shows us the use of color histograms to identify objects from the images. Even though histograms help us in efficient image indexing, they are susceptible to changing camera viewpoints. Pass et al. [22] describes a much better way to compare two images using the Color Coherence Vectors (CCV). This technique measures the numbers of coherent pixels in the images. Vailaya et al. [35] makes use of histograms and Color Coherence Vectors to classify the natural and urban landscapes. They achieve overall accuracy of 94% using a weighted K-Nearest Neighbor (kNN) classifier using leave-one-out validation. One of the drawback of CCV's is that the color intensity of the image affects them, for example, the same object captured in daylight and nighttime will have different CCVs. A more common feature according to Munder et al. [20] that is extracted to account for differences in intensities at multiple scales is Haar Wavelet. In our experiment, we extract the top 50% of the wavelets obtained for an image.

In the following section, we formally define the problem and the different performance metrics, used to analyze the results obtained after classification. In Section 2.3, we describe the experiments performed by us for detecting the transport mode followed by the analysis of the results obtained. In Section 2.4, we present our approach to solving the recycle verification problem followed by the analysis of the results obtained. Finally, we conclude the chapter by giving some suggestions on improving the results and introduce the next chapter.

2.2 Problem Definition

In the previous section, we looked into the related work done in the area of transportation mode detection and object detection. In this section, we formally define the problem we are trying to solve and explain features and the classifiers used.

We use supervised learning models to find out the correct class labels for the input accelerometer data, to verify transportation mode. We input the training data set to these models to produce a predictive function, which we use to label the unlabeled (test) data set. Let $(x_1, y_1), ..., (x_l, y_l)$ be the set of labeled training data points. Where l is the number of labeled points in the entire set. We want to learn the function $f : X \rightarrow Y$. Where $(x_1, x_2, .., x_l) \in X$ is the set of input points and the corresponding $(y_1, y_2, ..., y_l) \in Y$ is the output label assigned. Now using this learned function we compute the labels for the rest of the unlabeled points $(x_{l+1}, x_{l+2}, ..., x_{l+n}) \in X$. In our problem, we have four types of classes representing the modes of public transport. We attempt to solve the multi-class classification problem by mapping all the labeled data-points to this set of predefined class labels. In this research, we use

three supervised learning models:

1. Decision Trees

   We can define trees as predictive models that map the observations about a data item to its target label [38]. We can implement it as a tree data-structure where the leaves represent the class labels and the branches represent conjunction of features. Each node in the tree defines a constant value for one of the attributes of the data. When the classifier gets the input data, it starts the labeling process by comparing it with the root node. Based on the decision it can move to one of the branches of the tree. The tree is constructed recursively by selecting the values having maximum information gain. All the experiments performed in this research make use of J48 Decision Tree algorithm provided by the Weka software. We need to minimize the amount of noise/error in the data that is learnt (or over-fitting) during the training process. Over-fitting is always a problem when using the decision trees. We make use of the sub-tree raising strategy to prune the decision trees in order to avoid over-fitting. Figure 2.1 shows a sample J48 Decision tree generated using the software package.

```
feature5 <= 5226
|   feature7 <= 0
|   |   feature6 <= 95: NonRecycle (37.0/6.0)
|   |   feature6 > 95
|   |   |   feature12 <= 34: Recycle (7.0)
|   |   |   feature12 > 34: NonRecycle (82.0/28.0)
|   feature7 > 0
|   |   feature6 <= 0: NonRecycle (3.0/1.0)
|   |   feature6 > 0: Recycle (15.0)
feature5 > 5226
|   feature11 <= 4546: Recycle (33.0/1.0)
|   feature11 > 4546: NonRecycle (3.0/1.0)
```

Figure 2.1: Figure showing the pruned decision tree generated using the data mining software, Weka. We can notice the split condition at each internal node. Consider the example where the value of feature5<=5226 and feature7<=0 and feature6<=95 for a data point, then this point would be labelled NonRecycle.

2. Random Forest

   Random Forest is an ensemble learning approach where we construct multiple decision trees during the training phase of the classifier [46]. In Random forest technique each tree is completely grown and there is no pruning performed on them. During the testing phase, we assign the output label to a data point by running through all the trees generated in the training phase and choosing the mode of these output values.

3. Support Vector Machines

   Support Vector Machine constructs a set of hyper-planes that maximizes the separation between the different classes [47]. The decision function generated is a subset of the training samples called the support vectors. Support vectors are the data-points that lie closest to the decision surface and are the most difficult points to classify.

2.2.1 Performance Metrics

In this subsection, we look at some of ways in which we can measure the performance of the classifiers on the data set considered. We use the 10 cross validation technique to validate the data set. When a test set is provided to the classifier for labeling, we can measure the following quantities based on the output of the classifier - True Positives (T_P), False Positive (F_P), True Negatives (T_N) and False Negatives (F_N). True Positives (T_P) measure the number of correctly assigned labels, False Positives (F_P) measure the number of incorrectly assigned labels, True Negatives (T_N) measure the number of correctly rejected labels, and False Negatives (F_N) measure the number of incorrectly rejected labels. These four measures help us to determine the following metrics [45] to measure the performance of a particular model.

1. Accuracy : Accuracy measures the portion of true class labels in the entire data set after classification. We can represent this using the following equation:

$$\frac{T_P + T_N}{T_N + T_P + F_P + F_N} \tag{2.1}$$

2. Precision : Precision measures the portion of true positives among all the positives labels obtained after classification. We can represent this using the following equation:

$$\frac{T_P}{T_P + F_P} \tag{2.2}$$

3. Recall : Recall measures the probability that a point is assigned relevant label after classification. We can represent this using the following equation:

$$\frac{T_P}{T_P + F_N} \tag{2.3}$$

In the section, we described the features and classifiers that we will be making use of in all the experiments we have carried out. We have also presented different feature performance metrics that we will be using to measure the performance of the classifiers on the data set. In the next section, we will propose some techniques to tackle the transportation mode verification problem using supervised learning.

2.3 Transportation Mode Detection

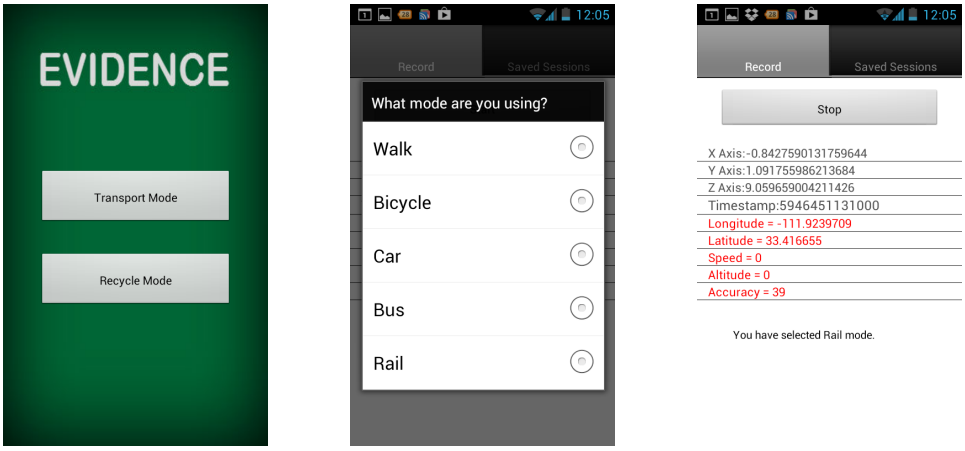
Transportation mode detection is the well-studied area of supervised learning. One of the popular way to solve this problem is using the accelerometer data obtained from the smart phone sensor. Accelerometer sensor on a smart phone provides us with the measurements along the three axis of the phone. Where X-axis provides acceleration along the breadth of the phone, Y-axis provides acceleration along the length of the phone, and Z-axis provides acceleration perpendicular to the surface of the phone.

We collected data for our experiment using an android application as shown in Figure 2.2. We ask the user to choose from four different classes - Rail, Bus, Bike and Walk and have collected about 50 hours of data for each class.

We use different classifiers like Naive Bayes, Random Forest and Decision Tree on the extracted feature vectors. As described in the performance evaluation section, we compare the classifiers based on the recall and precision, which provide us with the number of retrieved points and the number of relevant points respectively from the set of classified points. We could combine these two measures into a single metric called $F_{measure}$, which is the harmonic mean [39] of precision and recall shown in equation (2.4). We use $F_{measure}$ to determine the best performing classifier on the four class labels that we have considered.

2.3.1 Results and Analysis

The data we collected for the experiment has about 2000 data points in each class. We use 10 fold cross validation to access the performance of the predictive model constructed on the independent data set. In this validation approach, we split the data set into ten sets and use



(a) Home Screen (b) Options Tab (c) Status Tab

Figure 2.2: The data collection application used by the user to record the evidence. Sub-figure (a) shows us the home screen of the application where the user selects the type of activity he will be performing. Once he selects the transportation mode option, the application presents a list of transport modes that he could select, as shown in sub-figure (b). This application records the acceleration, GPS, speed and altitude values as shown in the sub-figure(c). The application stops recording when the user hits the stop button shown in sub-figure(c). After recording, the application sends the log files to the server for analysis.

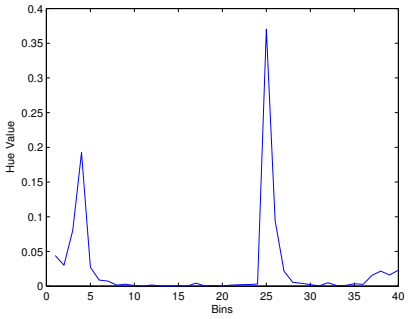
nine for training and one for testing. We repeated this ten times. Markatou et al. [15] shows us that 10-fold validation gives the least amount of variance. Table 2.3 shows the $F_{measure}$ for classifiers and the different activities.

$$F_{measure} = 2 * \frac{Precision * Recall}{Precision + Recall} \tag{2.4}$$

	Walk	Bus	Rail	Bike
Naive Bayes	0.82	0.46	0.77	0.49
Decision Trees	0.88	0.88	0.95	0.94
Random Forest	**0.91**	**0.90**	**0.96**	**0.94**

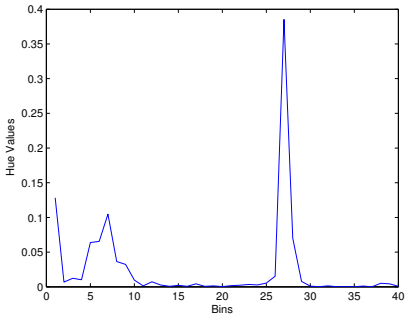
Table 2.3: Performance of the classifiers on different classes given by $F_{measure}$. Random Forest classifier gives the best accuracy for all the four classes considered. The average accuracy achieved is 92.75%. Random Forest is an ensemble classification technique where we construct of individual trees in the forest by selecting the subsets of randomly chosen predictor variables. Hence the Random Forest classifier is able achieve a better accuracy by examining the behavior of these predictor variables.[30]

(a) Image 1



(b) Hue histogram for image 1

(c) Image 2



(d) Hue histogram for image 2

Figure 2.3: This image shows the color histogram that is obtained for each of the images shown in sub-figure(a and c). Histogram provides us the color distribution in the image. We can notice the similarity of histograms when the images considered are similar.

Table 2.3 shows us the $F_{measure}$ obtained after using the above listed classifiers. We notice from the above table that Random Forest classifiers give us the best classification accuracy. Random Forest classifiers perform well when the individual tree classifiers perform well, and each decision tree is a good amount of dependence on each other [5]. We can verify our claim by examining the $F_{measure}$ obtained for the second best classifier - Decision Tree.

In this section, we analyzed the transportation mode verification problem using various supervised learning models. In the next section, we will investigate the second activity we had considered: recycling verification. We will formally define the problem and describe the different features that we will be using. By the end of the next section, we will analyze the results obtained using the supervised learning techniques.

2.4 Recycle Bin Verification

In this section, we investigate the recycling verification problem. As we have discussed in the introduction, the user provides some evidence to prove his assertion that he has recycled something. The evidence is captured by the user using the smart-phone application. In our experiments, we work with the image evidence. In order to prove the assertion, we detect the recycle bin object from the image. The system can reward the user only if it is confident enough. We investigate the different supervised models and compare their performance with respect to each of the feature extracted from the images submitted by users. We first discuss the features extracted from the evidence image.

1. Wavelet co-efficient [1]

   Given a signal (or an image), its transform is another representation of the signal without altering its information content. Fourier transform (FT) is the most commonly used transform. Fourier transform represents a signal in the space-time domain by a linear combination of sinusoidal waves of different frequencies. However, FT does not give us the details of a particular frequency at a particular location. To overcome this, we propose wavelet transform wherein we analyze different frequencies at different resolutions. Wavelets are localized waves with finite energy concentrated in time and space. We obtain the basis function used in the transformation by scaling and translating a mother wavelet. Some of the common mother wavelets used are Haar, Daubechies, Coiflet, Morlet, etc. The choice of the mother wavelet used depends on the kind of application.

   $$W_c = B_w * I \tag{2.5}$$

   Consider the above equation, where W_c is a (N x 1) vector containing wavelet co-efficient, B_w is (N x N) matrix representing the wavelet and I is a (N x 1) vector representing the image. Each row of B_w is a scaled and translated version of the mother wavelet used. Invariably natural images have concise representations in wavelet domain. We use these concise representations as the feature vectors by throwing away the lower 50% of coefficients.

2. Hue histogram

   Histogram as described in [32] provides us the color distribution in the given image. This might be a useful feature as we can compare two images based on the number of pixels of a particular color. We divide the histogram into some constant number of bins; in our case, we consider 40 bins. Using the values present in these bins, we generate the histogram to represent the image. In Figure 2.3 we show how the histogram generated for two similar images. Given the advantages like rotation and scale in-variance, this might sometime be a misleading feature i.e. two different images having similar number of colored pixels can give the same histogram.

3. Color Coherence Vectors

   The coherence measure classifies pixels as either coherent or incoherent[22]. Coherent pixels are a part of some sizeable contiguous region, while incoherent pixels are not. A color coherence vector represents this classification for each color in the image. CCV's prevent coherent pixels in one image from matching incoherent pixels in another [22]. This allows us to make fine distinctions, which are not possible using color histograms. Using this we could eliminate false positive images which have the same amount of pixels of a particular color.

2.4.1 Results and Analysis

We carried out the experiment on a data-set containing 35 positive and 35 negative images. We label the positive image as "Recycle" since all these images have a recycle bin object with a visible logo. On the other hand, we label all the images that do not have a recycle bin in it as "NonRecycle". We construct the image data set using the evidence submitted by the user under different illumination conditions, and we chose the negative images such that they have an object that looks similar to the recycle bin object.

We make use of the feature vectors described in the previous section with three different classifiers - Random Forest, Decision Trees and Naive Bayes Classifier. We measure the performance of a classifier by examining the $F_{measure}$, which is calculated by using the precision and recall values as described in Equation (2.4).

	Random Forest	Decision Tree	Bayes Classifier
Wavelets (F1 Score)	**0.67**	0.67	0.55
HUE Histogram (F1 Score)	**0.74**	0.60	0.67
CCV (F1 Score)	**0.81**	0.62	0.70

Table 2.4: Performance of the classifiers on different feature vectors on recycle bin class given by $F_{measure}$. Random forest classifier gives us the best performance among all the classifiers considered. We can also notice that Color Coherence Vectors(CCVs) give the best accuracy.

The Table 2.4 shows us the $F_{measure}$ obtained for the class Recycle. We can observe that the random forest classifier performs the best and provides us with very good $F_{measure}$. The reason we are performing better with the random forest classifier because individual decision trees have high classification accuracy and seem to have a good amount of dependence on each other.

	Random Forest	Decision Tree	Bayes Classifier
Wavelets (F1-Score)	**0.71**	0.71	0.67
HUE Histogram (F1 Score)	**0.73**	0.65	0.53
CCV (F1 Score)	**0.82**	0.67	0.72

Table 2.5: Performance of the classifiers on different feature vectors on non-recycle bin class given by $F_{measure}$. Random forest classifier gives us the best performance among all the classifiers considered. We can also notice that Color Coherence Vectors(CCVs) give the best accuracy.

The Table 2.5 shows the $F_{measure}$ obtained for the class label Non_recycle. It is quite similar to the one we got for Recycle class; the best performing classifier is Random Forest.

2.5 Conclusion

In this section, we tried to solve two problems - transportation mode and recycle verification using the supervised learning techniques. Supervised learning models require training before they can predictions about the data. We train the classifier with the data obtained from the tri-axial accelerometer on the smart-phone. This data contains the acceleration in each of the

three axis of the phone. In our research, we look at four modes of transport - Bike, Walk, Rail and Bus. We achieve a very good average classification accuracy of 93%.

The second activity we investigate is recycling verification. Recycle verification is an inherently hard problem to solve, as we do not have any data that is representative of the activity performed. We can capture the data by taking advantage of various modalities like audio, video, image, GPS, accelerometer etc. provided by the smart-phones. In this chapter, we assume that the user submits the image data as the evidence. We extract the features like histogram, color coherence vectors and wavelets to identify the recycle bin object from the image. We assume that the image taken by the user with the recycle bin can be a good indicator of the activity performed. We train the classifiers on the previously mentioned feature vectors to achieve an average classification rate of 75%.

There is advantage and disadvantage of using pure supervised models for classification. Once we train the system, we can achieve significantly better classification accuracy. On the other hand, we will need a large amount of annotated data to train the classifier. We will address these issues in the following chapters.

Chapter 3

RECYCLING DETECTION - VIEW POINT CORRELATION

In this chapter, we will investigate the problem of recycling verification in detail. In the previous chapter, we assumed that the evidence provided by the user to the system is an image containing the recycle bin object. Having made this assumption, we detect the recycle bin in the image by using the supervised learning techniques on features extracted from this image data. We achieve an average accuracy of around 71% for detecting the recycle bin. In this chapter, we will look at improving the recognition accuracy to an acceptable level where the system is confident that the user has performed the activity. We also investigate some of the more recent and advanced object matching techniques for accurately detecting the recycle bin object in the image. We will show that using a single piece of evidence for this activity does not give us the required confidence; hence, we ask the user for additional supporting evidence. Asking for additional piece of evidence from the user can be tricky. We ideally would want an evidence that maximizes the recognition rate and reduces the amount of work done by the user. As the amount of work done by the user increases the cost incurred by the system also increases. Hence, our goal in this chapter is to get the next best piece of evidence, which maximizes the accuracy and reduces the cost incurred by the system.

The user can capture the image evidence from any viewpoint and under any illumination condition. We detect the recycle bin object from the image by using the recycle logo shown in Figure 3.1 (b). In the following sections, we describe the algorithm we used to detect the logo on the recycle bin. Now given that we detect recycle logo, we can show that we achieve different recognition rates when we capture the recycle bin object from different viewpoints. Consider an example, where the user submits two evidences, the first one is captured from an angle where the recycle logo is clearly visible and, the second piece of evidence, which is captured from the side view point where the recycle logo is tilted. We show that the first evidence submitted by the user has a better recognition rate when compared to the second evidence. Although the first evidence has better recognition rate, the system is not confident enough to prove the assertion. Hence, we ask for additional evidence from the user. Since we know that, the front viewpoint has better recognition rate we need to direct the user to this viewpoint. The previously mentioned technique serves two purpose, one it makes sure that we achieve maximum accuracy with a minimum number of evidence and we reduce the cost incurred.

Using recycle logo for detecting the recycle bin simplifies the process and provides greater confidence to the system. We now have to detect if the logo is present on the recycle bin object. We use Affine Scale Invariant Feature Transform (ASIFT) algorithm for object detection. ASIFT algorithm as the name suggests is invariant to tilts, rotation and scaling of the object in the images. The image evidence provided by the user can fall in one of these viewpoint-categories: font, close-up, side, top and far. We define the above-mentioned viewpoints with respect to the recycle logo on the object. We constructed a data set with 100 images in each category. We then train the classifiers (Support Vector Machines (SVM), Decision Trees and the Random Forests) on each of these viewpoint data sets to measure the classification accuracy. After classifying them into one of these categories, we decide if we require additional evidence. If required we choose the least correlated viewpoints, which minimized the error probability as these data points are independent.

Every time the system asks the user for additional evidence, we incur some cost. Our goal is to minimize this cost incurred by making the decision with the minimum number of additional evidence from the user. In order to achieve this, we aim to provide directions to the user to capture the most confident piece of evidence. In the case of recycling verification, if the user provides the system with an image from the viewpoint which is least confident we would want to direct the user to the most confident view point like - font viewpoint. We do this by choosing the least correlated viewpoint-classifier combination.

We organize the rest of the chapter in the following way. In Section 3.1, we look at some of the related work done in the past in this area. In Section 3.2 we formally define the problem we are working on and the performance metrics we use to compare the results. In section 3.3, we explain our proposed approach in detail. We then analyze the results of this section. Finally, we conclude by providing the summary for the chapter and provide some alternative techniques that we can use for tackling this problem.

3.1 Related Work

In recycling verification problem, we consider the images as the primary source of evidence provided by the user. Our goal is to detect the recycle bin in the image provided by the user. In the previous chapter, we used three features to help us classify the recycle images. The first feature we considered was HUE histogram, which can capture the distribution of color in the image. Tico et al. [32] shows us that HUE histograms are invariant to rotation and translation. The disadvantage of using this as a feature vector is that we cannot extract any

spatial information [22] from the image, for example, an image of a blue recycle bin might be similar to an image taken outdoors with blue sky in it as both the images have approximately same number of blue pixels. The second feature we considered was Color Coherence Vectors that stores the number of coherent and incoherent pixels for each color as described by Pass et al. [22]. When the images are compared the coherent regions are compared, which enables us to eliminate many false positives which might be obtained using the histogram features. The final feature we looked in was wavelets, which helps us, account for intensities when the scale of the object is varied.

Though we have detected the object, we need to make sure that the detected object was a recycle bin. As described in the introduction of to this chapter, we make use of the universal recycle logo as a template and try to find positive matches in the images provided by the user. Work done by Lowe et al. [12] discusses a scale invariant approach to extract features vectors from the image, which are translation, scale and rotation invariant. We use this technique to extract the SIFT key points from the image and compare them to the already extracted key points obtained from template image. When comparing the template image to the user submitted image we look at number of key-points matched. These key-points represent rotation and scale invariant descriptors calculated by noting the most stable points obtained for the image at different resolutions. This gives us an estimate of the matching strength between the two images. The computer vision community has extensively made use of SIFT algorithm to perform object recognition, for example, Wang et al. [36] provided a technique to recognize hand postures, Luo et al. [13] makes use of SIFT features for face recognition and Bicego et al uses SIFT based features to perform face authentication.

Despite the advantages provided by SIFT, it is susceptible to extreme transition tilts. We look at another approach suggested by Yu et al. [48] where they define a fully affine invariant SIFT (ASIFT) which can detect transition tilts up 32 as compared to 2 provided by SIFT. The transition tilt represents the compression factor for transforming one image to another. ASIFT provides us better features for solving our problem because the user at any angle can take the recycle bin image. In our experiment, we use the feature vector as the logarithm of NFA (Number of False Alarms), computed by ASIFT algorithm.

We organize the rest of the chapter is in the following way. Section 3.2 provides a formal problem definition and defines the performance metrics used to analyses the results. Section 3.3 provides the results and analyses them. We conclude the chapter with the summary of the

work done in this chapter and suggest new ways to tackle the current problem.

3.2 Problem Definition

In this section, we will state in detail the problem we are trying to solve. We will also explain the feature vectors that we will be using in our experiments along with the classifiers. In the final part of this section, we look at some of the performance metrics that we use to analyze the results.

The transportation mode verification problem is widely studied in the literature. As we have seen in the previous chapter, we obtain good probability of classification using just the accelerometer data from the smart-phone sensor. On the other hand, recycling activity verification is a little tricky as we can use different modalities to get the best classification probability. In recycle verification problem, we primarily work with the image data to determine if the user had performed the recycling activity. We try to analyze the image and determine if the object of interest (recycle bin) is present. We do this by matching the recycle logo template shown in Figure 3.1(b) with the input image. The matching strength depends up on the camera angle with respect to the axis, which is orthogonal to the recycle bin logo present on the object. Our main goal in this chapter is to analyze the recognition rates obtained by using different classifier-viewpoint combinations and then direct the user to the best viewpoint if the accuracy of classification using the already provided evidence is not high enough. We use the additional evidence in achieving higher accuracy for classifying the activity.

3.2.1 Features

In this subsection, we explain the features used by us in our experiments. We use a 17 dimensional feature vectors, where the first 16 are extracted from the Color Coherence Vectors(CCV) generated and one feature is taken from the Affine-Scale Invariant Feature Transforms (ASIFT) [18] algorithm by extracting Number of False Alarms (NFA). We have described CCVs in the previous chapter, and they provide us the spatial distribution of colors in the image.

ASIFT algorithm gives us the number of key-points detected in the image and then we compare it with the key points extracted from the template image provided to get the number of matching key-points. Most of the object detection techniques like the one described above focus on finding and comparing the key-points. However, in our case the images submitted by the user can be quite similar to each other and might vary in camera viewpoint. That is the number of matching key-points from both the image and the template might be similar. In order

to solve this problem, we can use the measure of the number of false alarms (NFA) detected.

3.2.2 Performance Metric

We measure the performance of the supervised learning models by measuring the prediction accuracy. In the first set of experiments, we compute the accuracy of the classifier on the five data sets. Each of these datasets: Far, Top, Front, Front-CloseUp and Side have 100 images each. We use the leave-one-out cross validation technique to calculate the accuracy. Leave one out validation is a technique to estimate the performance of the constructed model. In this technique, we train the classifier on the (N-1) data-points and test it on the remaining data point. N represents the total number of data points in the class. In our experiments, we compare the performance of three classifiers on the constructed data sets. We make use of the popular data mining tool Weka [10]. Weka provides us with the implementation for Decision Trees and Random Forest. We also measure the performance of Support Vector Machines (SVM), whose implementation is provided by LibSVM [6] package, which can be used along with Weka. We measure the performance of the classifier using the classification accuracy obtained. In the second set of experiments, we work on the problem of asking for additional piece of evidence. For performing these experiments, we rearrange the training data into three data sets: Close, Near and Far. We now compute the correlation between each classifier-viewpoint pair to select the next least correlated viewpoint. We use the correlation to choose the next most independent sample to reduce the classification error. We can define correlation as follows:

$$\rho = \frac{\sigma_{C_1,C_2}}{\sigma_{C_1}\sigma_{C_1}} \tag{3.1}$$

where ρ gives the correlation and σ_{C_i} is the standard deviation of the values obtained by using the classifier C with a viewpoint i. Now that we have seen the features and the performance metrics, we present the techniques that we have used to solve the problem.

3.3 Proposed Approach

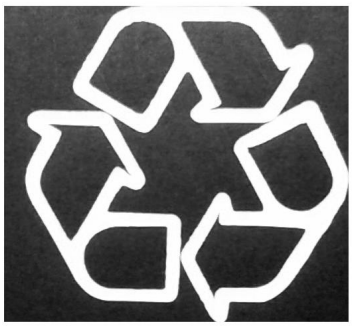
In this section, we will discuss the procedure followed to achieve the higher recognition rates and lower the cost incurred by the system. Let us consider the case of recycling verification problem where the user submits an image of a recycle bin as an evidence. In our experiments, we assume that all the recycle bins have the universal recycle bin logo as shown in Figure 3.1. In the first set of experiments, we try to estimate the classification error for different viewpoints. In order to do this, we constructed five data sets, each containing images from a particular

viewpoint. We consider the following viewpoints while constructing the data set - Front, Front-CloseUp, Side, Far and Top. Each of these viewpoints represents the camera angle from the axis, which is orthogonal to the surface of the recycle logo present on the recycle bin object. For each of these sets, we used a common set of negative images, which do not have the recycle bin in them. We now compute the accuracy of recognition for each of the data sets using the above-defined classifiers. Figure 3.4 shows the accuracy obtained for each of the classifiers along with the average classification accuracy.

We can notice from the Figure 3.4 that the average classification rate for each of the viewpoint is around 65%-73%, which is quite low as the system is not confident enough. In our next set of experiments, we try to combine these data sets to see if we can get a better accuracy. This scenario is close to what we discussed in the introduction of this chapter where we ask for additional evidence from the user if the already provided evidence does not provide enough confidence to the system. Figure 3.2 and Figure 3.3 show us that the accuracy obtained with Front and Front-CloseUp data-sets is higher than the other three data-sets due the fact that there are a higher number of key-point matches with the template image. Due to this reason, we combine images from these two data sets with images from the remaining data sets to record the classification accuracy that we obtain. Figure 3.5 and Figure 3.6 shows the classification accuracy that we obtain when we combine the Front-CloseUp dataset and Front data set respectively with the rest of the data sets. Figure 3.5 shows the results that we obtain when we combine the Front-CloseUp viewpoint with side, top and far viewpoints to achieve average classification accuracy of 90.6% using Random Forest classifier. Figure 3.6 shows the results that we obtain when we combine the Front viewpoint with side, top and far viewpoints and obtain average classification accuracy of 93.4% using Random Forest classifier.

As seen from the above results, we achieve a 21% increase in accuracy when we combine the two least correlated data sets. Our goal is to direct the user to the next viewpoint which helps us achieve maximum classification accuracy. To solve this we propose a correlation based approach where we direct the user to the next least correlated viewpoint. In order to implement this we reduced the number of data sets to three: Near, Far and Close so that we can provide better directions to the user. We compute the correlation between each of the viewpoint-classifier combination. Table 3.1 shows these values. In the next section, we will analyze these results.

(a) Images from each of the viewpoints in the data set, shown clockwise direction a) Front-CloseUp view b) Front view c) Side view d) Top view e) Far view. Each of these viewpoints represents the camera angle from the axis, which is orthogonal to the surface of the recycle logo present on the recycle bin object.



(b) The universal recycling logo template used for detecting the recycle bin from the image evidence submitted by the user.

Figure 3.1: Sub-figure (a) shows the recycle bin captured different viewpoints and sub-figure (b) shows the recycle template used for detecting recycle bin.

3.4 Results

We ran the data sets on three classifiers - Support Vector Machines (SVM), Random Forest (RF), Decision Tree (DT) and Random Forests (RT). Figure 3.4 shows the results that we obtain using the Leave-one out validation. We can make a couple of observations based on the results obtained - 1) The accuracy is higher when the images are captured from a certain viewpoint like front or top and decreases as we go away from the image. 2) Certain classifiers like Random Forest (RF) always does better than the average. The reason for better performance of the random forest is similar to the one described in the previous chapter. When the system directs the user for additional pieces of evidence to increase the confidence, the system expects that the new sample is independent of the first sample taken. For example if a second image is required for recycling verification problem, we would like it to be independent of the first image to reduce the overall error rate. Mathematically we can say that if V_1 and V_2 are two view points for the two pieces of evidence respectively and C_i be some classifier then the independence of viewpoints holds true $P(V_1.V_2|C_i) = P(V_1|C_i).P(V_2|C_i)$. To investigate this further, we try to measure the viewpoint level correlation to direct the user to the least correlated viewpoint to the present viewpoint. As seen from Figure 3.4 some classifiers work well for certain viewpoints; hence we also need to find the best possible classifier for each of the viewpoint V_1 and V_2. Table 3.1 shows the correlation table.

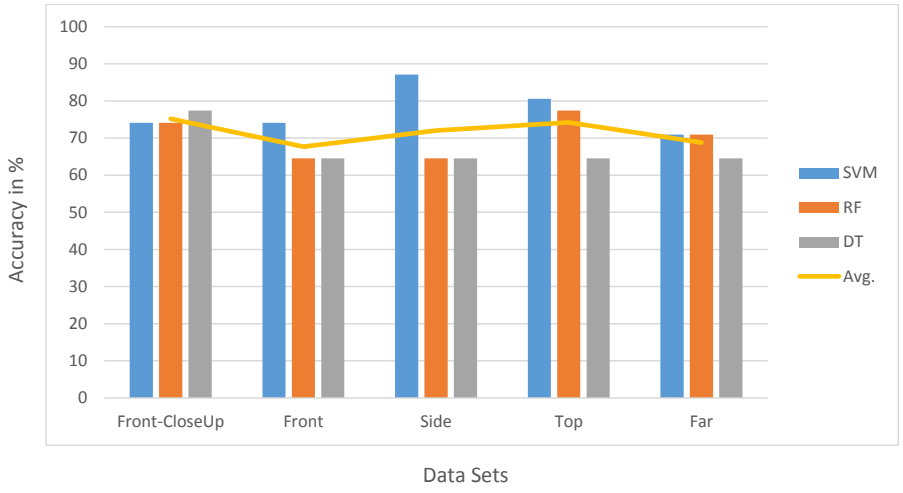


Figure 3.2: Performance of three classifiers when we consider the Color Coherence Vectors (CCV) as our feature vector. The average line in the graph represents the average accuracy for each data set. As we can notice from the graph, SVM provides us the best accuracy. In this case since we have just considered the CCVs, we can notice that FrontCloseUp, Side and Top provide a much better accuracy then the rest due to the larger amount of coherent blue pixels present in these data sets.

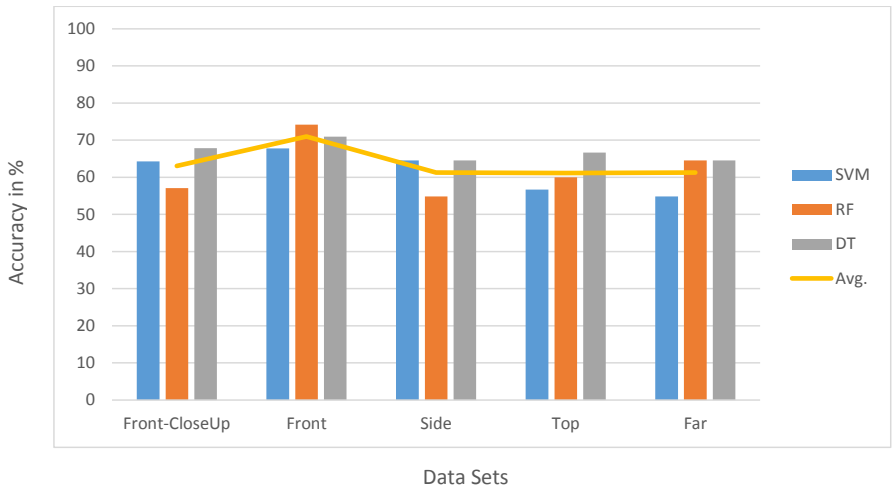


Figure 3.3: Performance of three classifiers when we consider the log of Number of False Alarms (log(NFA)) extracted using the Affine Invariant Feature Transforms technique as our feature vectors. The average line in the graph represents the average accuracy for each data set. We notice that the Random Forest classifier performs the best on all the data sets and provides the maximum accuracy on the ones that have the recycle logo clearly visible i.e. Front and Side give better accuracy than the Far data set.

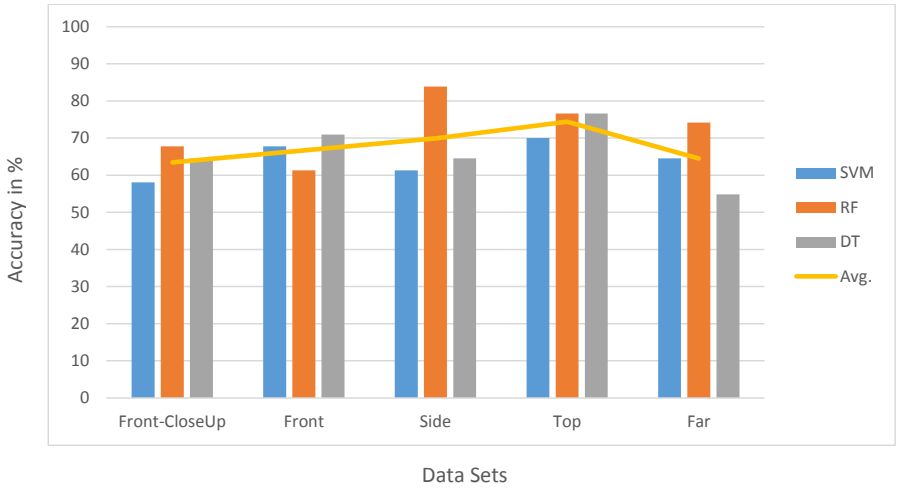


Figure 3.4: Performance of three classifiers when we consider both the Color Coherence Vectors and log of Number of False Alarms (log(NFA)) as our feature vector. We get an average accuracy of around 70% over all the data sets and can notice that Random Forest is the best performing classifier.

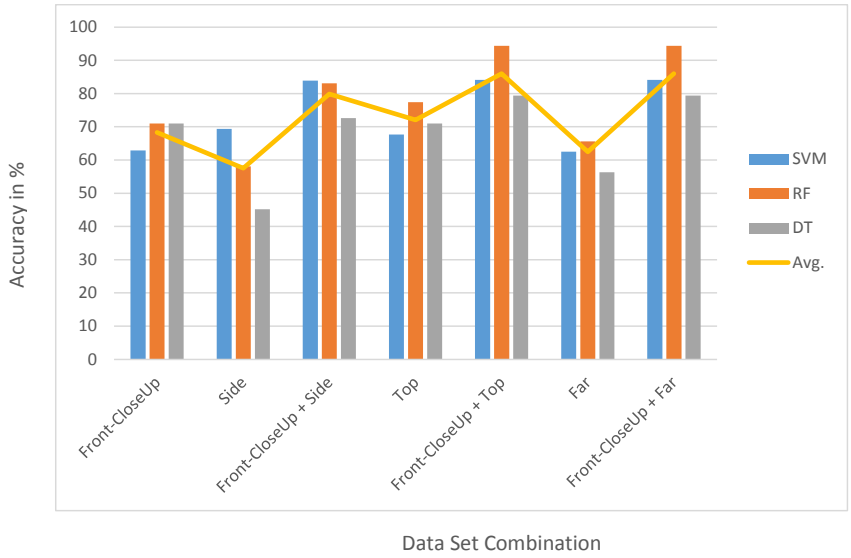


Figure 3.5: Performance of three classifiers when we combine the Front-CloseUp viewpoint data set with other low performing viewpoint data sets. Here we use both the CCV and log(NFA) as the feature vectors. As we can see from the above results the accuracy increases significantly, when we combine two low performing data sets. We achieve an 13.0% increase in accuracy when we combine Front-CloseUp data set with Side dataset, 32.3% when we combine Front-CloseUp data set with Top data set and 32% with we combine Front-CloseUp data set with Far data set.

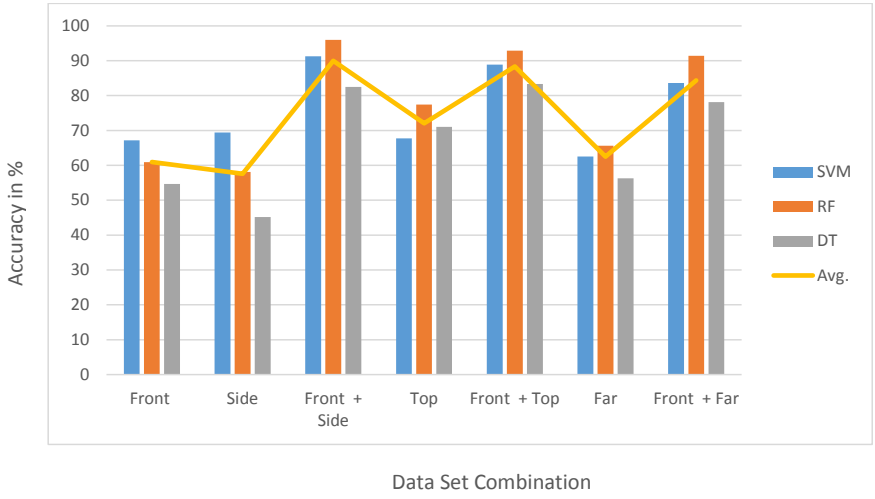


Figure 3.6: Performance of three classifiers when we combine the Front viewpoint data set with other low performing viewpoint data sets. We use both the CCV and log(NFA) as the feature vectors. We achieve an 41.0% increase in accuracy when we combine Front data set with Side dataset, 35.0% when we combine Front data set with Top data set and 33% with we combine Front data set with Far data set.

One of the problems faced by this approach is to determine the viewpoint of the first image. In the experiments, we assume that the user starts with the close viewpoint and then based on the confidence of the classifier we determine if he needs to provide the additional evidence. In the previous two chapters, we have made use of supervised learning techniques where we had to train the system using labeled data. In the following chapter, we will look at some ways to take the problem of having large amount of annotated data to train the system.

	Close_SVM	Close_RF	Close_DT	Far_SVM	Far_RF	Far_DT	Near_SVM	Near_RF	Near_DT
Close_SVM	1.000	0.315	0.284	-0.021	0.140	0.173	0.217	0.037	0.306
Close_RF	0.315	1.000	0.556	0.109	0.068	0.031	0.284	0.236	0.556
Close_DT	0.284	0.556	1.000	0.166	0.032	-0.054	0.333	0.337	0.951
Far_SVM	-0.021	0.109	0.166	1.000	0.073	0.037	0.172	0.221	0.211
Far_RF	0.140	0.068	0.032	0.073	1.000	0.537	0.128	0.019	0.076
Far_DT	0.173	0.031	-0.054	0.037	0.537	1.000	0.049	0.005	-0.010
Near_SVM	0.217	0.284	0.333	0.172	0.128	0.049	1.000	0.193	0.357
Near_RF	0.037	0.236	0.337	0.221	0.019	0.005	0.193	1.000	0.383
Near_DT	0.306	0.556	0.951	0.211	0.076	-0.010	0.357	0.383	1.000

Table 3.1: This table shows the correlation between different viewpoints and classifier combination. We make the following observations from the table: the least correlated pair of viewpoint combinations are far-close and near-close. Using the result obtained we guide the user to the next best viewpoint which maximizes the accuracy. For example, if the user takes the image of the recycle bin from a distance then we might ask the user to come closer to the recycle bin and then take another evidence image.

3.5 Conclusion

In the previous chapter, we discussed about training the classifiers using the color coherence vectors, hue histogram and wavelets features extracted from the image. Using this approach, we got an average classification accuracy of 75% with Random Forest for the recycle bin class. In this chapter, our main goal was to increase the confidence of the system by asking for additional supporting evidence. Every time we ask the system for evidence, the system incurs some cost, as the user has to put extra effort on his side to collect more evidence. If the cost is too high, then the user might lose interest in the final reward offered by the system. Hence, we have to make sure that we ask the best possible evidence from the user, which maximizes the confidence of the system. We can achieve this by directing the user to the location where he can collect this best piece of evidence. In all these experiments, we assume that the user is collecting data in-situ using his/her smart phone.

In this chapter, we extend the number of feature vectors to include the features extracted from Affine Invariant Feature Transform (ASIFT) algorithm. We consider this technique because in our experiments, we try to match the universal recycle logo to the recycle bin object extracted from the image evidence submitted by the user. If we can find the recycle logo in the evidence then we have a high probability of the object being a recycle bin. We first start by constructing five data sets, each having set of images from a different viewpoint. We construct this data set because the user from any viewpoint can capture the image and we need to choose the best viewpoint in which the logo is clearly visible. We use both the color coherence vectors and ASIFT features to train the classifier. Using this approach, we get average classification accuracy of 80% across all the data sets using the random forest classifier.

As discussed in the previous paragraphs, we also explored the scenario where we ask the user for additional evidence to verify the assertion. We expect to increase the classification accuracy each time we ask for additional evidence from the user, and we would like to achieve the desired accuracy with the minimum amount of evidence. We start by exploiting the correlation between the different viewpoints. When we have to ask for a new piece of evidence, we direct the user to the least correlated viewpoint. We select the least correlated viewpoint to ensure that the evidence provided by the user is independent of the previous piece of evidence. Using this technique of combining the viewpoints, we get an average classification rate of 93%, which is a significant improvement over the single viewpoint classifier.

In the future, we can do better by training classifiers, which in addition to taking the features as inputs, also takes in the confidences of the previous classifier. In the next chapter, we will investigate a different approach to activity verification using the semi-supervised learning technique.

Chapter 4

SEMI-SUPERVISED LEARNING APPROACHES

Labeled data is essential for building good supervised predictive models. In the previous chapters, we investigated the supervised learning techniques and applied them to the transportation mode verification and the recycle verification. To achieve high accuracy using these techniques, we need to train the system. Obtaining the labeled data for training is extremely expensive and difficult process considering the activities we are trying to verify. For instance, if we need to train the system for verifying the recycling activity then we need to have labeled training data set, which covers all possible modalities under every situation. Asking humans to label the data is expensive, as the user needs to spend time on analyzing the piece of evidence and providing his opinions. Although it is expensive to ask for human input, it is difficult to avoid it completely. We could develop smart approaches to minimize the cost incurred while asking for human intervention and maximize the accuracy.

Labeling of the samples by human networks is a time consuming process and does not scale up properly. As the amount of labeled data is small compared to the amount of unlabeled data, we will have to investigate the semi-supervised learning techniques. Mathematically, we can use the semi-supervised learning techniques to learn the function $f(c) : X \leftarrow C_1...C_k$, where X is the set of labeled and unlabeled input samples in D dimensional space i.e. $X = x_1...x_n \in R^D$ and C_i represents the labels for some of the input samples $\{x_1...x_k\}$ [49]. Semi-supervised learning aims at propagating the label of the known data point to its nearest neighbor until we label all the input points. As we use the density of the data to propagate the label, we measure the proximity between the labeled and unlabeled using the Euclidean distance measure.

In this chapter, we investigate the graph based inductive semi-supervised approach that we introduced in the previous paragraph and apply it to our problem. We first discuss the label propagation technique, which we use for labeling the unlabeled data points. In this approach, each instance in the data set represents a node in the graph, and we connect these nodes with edges, and each edge has some weight representing the similarity between the points. We investigate techniques to assign proper weights to these edges using a weight parameter σ. Determining the right σ is tricky; hence we use the entropy based approach proposed by Zhu et al. to select the best value. After calculating the weights, we find out the transition

probability. Transition probability is the probability of moving from one data point to another in the graph for propagating the label. We estimate the performance of these algorithms by measuring how close the probability distribution of the data after label propagation is to the probability distribution of the ground truth data. Since labeling has a cost associated with it, we propose algorithms that help us in selecting minimum number of labeled points to propagate the labels accurately. In the following sections, we describe a couple of techniques for selecting the next set of seed points from the data set to achieve the desired classification accuracy.

We organize this chapter as follows; we discuss the related work in Section 4.1 and in Section 4.2, we formally define the problem along with the performance metrics used in our experiments. In Section 4.3, we look at the initial seed selection algorithm. We use this seed selection algorithm for all the experiments performed in this chapter. In the following sections, we investigate the techniques to select the next set of labeled points from the data set by asking the oracle. In Section 4.5 and 4.6, we propose a couple of approaches to choose the next set of points and analyze the performance with respect to the baseline algorithm described in the Section 4.4. We conclude this chapter by summarizing the main findings of this chapter and provide a few improvements to the proposed techniques.

4.1 Related Work

Semi-supervised approaches have a tremendous amount of application especially in the smart-phone era, where producing the data is very cheap but annotating them requires time and money. Hence, we need smart techniques to determine the characteristics of the data without providing extensive training to the system. Some of the early work was done by Zhu et al. [50] where they provide a proximity based approach to propagate the label. The paper discusses few techniques on finding the proximity of the data-points and compares them to the results obtained using the k-nearest neighbor technique (k-NN) techniques used in supervised learning. Our work uses a slight modification of the label propagation technique described in that paper. In a follow up paper by Zhu et al. [51], they discuss methods for incorporating the supervised learning approach by using the classifier predictions and the class priors. Specifically they change the class distributions to match the class priors. They test their approach on the problem of hand writing recognition where they achieve maximum accuracy of about 97% compared to the 94% achieved using the nearest neighbor classifier. These techniques are also used by Goldberg et al. [9] for inferring the ratings of the documents based on its predictive sentiment.

There are different techniques in the area of semi-supervised learning; Zhu et al.[49]

discusses some of them. One of the simplest technique is self-training used by Maeireizo et al. [14] for predicting the emotions in the spoken dialogue. In this technique, we train the classifier iteratively, i.e. first, the classifier predicts the labels of the most confident unlabeled point and then we add these points to the set of labelled points to predict the labels for the remaining data points. More recently Rosenberg et al. [26]made use of this technique for the purpose of object detection in an image. They use a concept of weekly-labeled training data, which provides a likelihood of the object in the image. Another related technique is co-training where we capture the sub-features from the data set and train two classifiers on them; each classifier trains the other classifier with the most confident unlabeled data-points. We add these points to the labeled data set after prediction and then we repeat the process.

There is a related field of Semi-Supervised Learning (SSL) called Active learning (AL). Active learning essentially tries to achieve the same goal as the semi-supervised learning where we start with a small number of labeled data points and try to label the rest of points in the data set. In both of these approaches, we assume that the number of labeled points (l) is extremely small compared to the number of unlabeled (u) data points, i.e. $l \ll u$. Active Learning (AL) as defined by Settles et al. [27], works by interactively querying the user or oracle for the unlabeled data-point's ground truth label. One of the key differences between the SSL and AL is that in SSL, we start with the labeled data points and then select the predicted output for the next iteration where as in AL we ask the oracle for label. In our approach, we try to work with the semi-supervised approach along with active learning approach. Active learning techniques have been used in the literature for image classification[33], text classification [28, 34] etc.

In this section, we looked into the related work in the area of semi-supervised learning and active learning. We also describes some of the differences between the areas of active learning and semi-supervised learning and how the recent literature uses it. In the following section, we formally define the problem and discuss proposed technique for seed selection. We also define the performance metrics used to analyses the results and talk about the convergence criteria for the algorithms proposed by us.

4.2 Problem Definition

In this section, we formally define the problem that we are going to solve using the semi-supervised learning techniques. Semi-supervised learning techniques attempts to make use of both labeled as well as unlabeled data for learning about the data point labels. For all the experiments performed in this chapter, we have used the feature vectors extracted from the ac-

celerometer, recorded using the smart-phone while performing activities like running, jumping, standing, climbing up the stairs and descending down the stairs.

We now formally define the problem. Assume there are N data points under consideration and each data point had D dimensions i.e. $X = (x_1, x_2, .., x_N) \in R^D$. Let $C = (c_1, c_2, .., c_m)$ be the total number of classes in the data set and let matrix Y which give the probability of each point x_i belonging to class $c \; \forall c \in C$. We can represent the NxD matrix input data matrix and the NxC label matrix as:

$$\text{X} = \begin{bmatrix} \gamma_{11} & \cdots & \gamma_{1N} \\ \vdots & \ddots & \vdots \\ \gamma_{D1} & \cdots & \gamma_{DN} \end{bmatrix} \text{ and Y} = \begin{bmatrix} \varphi_{11} & \cdots & \varphi_{1C} \\ \vdots & \ddots & \vdots \\ \varphi_{N1} & \cdots & \varphi_{NC} \end{bmatrix}$$

where each column of the matrix X is a data point in the set represented by $\vec{x_i} = [\gamma_{1i}, \gamma_{2i}, .., \gamma_{Di}]$ and each row of matrix Y is a label vector $\vec{y_i} = [\varphi_{i1}, \varphi_{i2}, .., \varphi_{iC}]$ representing the probabilities of the points belonging to each of the class. The input data set X is made up of points whose label is known (X_L) and unknown (X_U) where $X \ni \{X_L, X_U\}$.The labeled data points X_L are represented as $\{(x_i, y_i)\} \forall x \in X, y \in Y \; and \; i \in \{1, 2, .., L\}$and the unlabeled data points X_U are represented as $\{(x_i)\} \forall x \in X, y \in Y \; and \; i \in \{L+1, L+2, .., N\}$ where $\mid X_L \mid \ll \mid X_U \mid$. Let us consider an example, if the number of classes(C) is two and the data point x_1, belongs to class 1 with probability 0.7 then we can represent the row vector as $y_1 = \{0.7, 0.3\}$. We use the algorithm presented in [50] to propagate the label through the data set. We have seen that each point in the data set has the label vector y_i defined. Although, the labeled points in the data set have y_i already set, since they belong to one of the classes, but the unlabeled data set does not have the output class probabilities assigned. Hence we initialize the probabilities of these unlabeled data points to be the normalized sum of Euclidean distances to the each of the labeled data point. Algorithm 1 shows the idea.

Here norm is the Euclidean norm of the D dimensional vectors. We can represent the L2 Euclidean norm as:

$$\parallel x_i \parallel := \left(\sum_{i=1}^{D} \gamma_{i1} \right)^{\frac{1}{2}} \tag{4.1}$$

We now proceed with the label propagation technique and investigate the convergence

Algorithm 1: Procedure to assigning initial probabilities to unlabeled data points. We are using this procedure because we do not know the labels for rest of the unlabeled data set. In this algorithm, we assign the probability with which it belongs to each class in the data set. It is set to be the normalized sum of Euclidian distance to each of the labeled data point in the class.

Input: $(x_1, x_2, .., x_L) \in X_L$ - the set of labelled points, $(x_{L+1}, x_{L+2}, .., x_{L+U}) \in X_U$ - the set of unlabelled points and $X_L \cup X_U \in X$

Output: $(y_1, y_2, .., y_{L+U}) \in Y$ - Probability matrix

. **for** $i = L + 1\ to\ L + U$ **do**
  for $j = 1\ to\ L$ **do**
    $Dist(j, c)\ =\ norm(\vec{x_i} - \vec{x_j});\ \ \forall \vec{x_i} \in X_L \wedge \vec{x_j} \in X_U$
  end
  for $k = 1\ to\ |C|$ **do**
    $\xi_k = e^{-\sum Dist(:, c_k)}$;
  end
  for $m = 1\ to\ |C|$ **do**
    $\varphi_{im} = \frac{\xi_m}{\sum_{l=1}^{|C|} \xi_l}$;
  end
  $y_i \leftarrow [\varphi_{i1}, \varphi_{i2}, .., \varphi_{iC}]$;
end

criteria. Once we have all the points initialized with some probability values, we use a slightly modified version of the algorithm mentioned in [50] for label propagation. We modify the convergence criteria to be the difference between the successive Kullback Leibler (KL) Divergences, less than some small constant threshold ϵ. In our experiment we consider the value of ϵ to be 0.0001. KL Divergence gives the distance between any two probability density functions. We use Kernel density estimation (KDE) technique, which is a non-parametric way to estimate the probability density function of a random variable [41]. Let us consider $(y_1.y_2, .., y_n)$ to be the label vector for class c where $c \in C$ taken from distribution with unknown density f. We define the KDE as:

$$\hat{f}_h(y) = \frac{1}{nh} \sum_{i=1}^{n} K(\frac{y - y_i}{h}) \tag{4.2}$$

where K is the Gaussian kernel, h is the bandwidth parameter and n is the number of points considered. Similarly, we can define Kullback Leibler (KL) as a non-symmetric measure of difference between the two probability distributions [43]. In our algorithms, we compute the KL divergence between the two successive KDE's for any class c using the Equation 4.3.

$$D_{kde_1 \| kde_2} = \sum_i ln(\frac{kde_1(i)}{kde_2(i)}) kde_1(i) \tag{4.3}$$

In order to compute the KDE and KLD during the experiments, we use the UCI KDE toolbox [42] which provides us with the implementation of the KDE and KLD functions. Algorithm 2 describes the procedure followed for propagating the label. The algorithm takes input as T_{ij} - transition probability matrix, W_{ij} - weight matrix, Y - probability matrix and the defined constant ϵ. We define the weight matrix W_{ij} using the standard Euclidean distance as shown below:

$$w_{ij} = exp(-\frac{\sum_{d=1}^{D} (x_i^d - x_j^d)^2}{\sigma^2}) \tag{4.4}$$

We can use the weight matrix W_{ij} to compute the transition probability matrix T_{ij}, which gives the probability of moving from node i to node j.

$$T_{ij} = \frac{W_{ij}}{\sum_{k=1}^{l+u} W_{kj}} \tag{4.5}$$

In the Equation 4.5, the parameter σ controls the weight. Finding out a right value of the σ for the data set is a challenging problem. We use the entropy-based method provided by the [49]. The entropy H shown in equation 4.6 can be defined as the sum of entropy on each of the points in the data set. We analyze the effect of the weight parameter σ on the accuracy of labeling. We choose the σ value that gives us the minimum entropy over the entire data set. Given the condition, we can notice that we get minimum entropy when the value of $\sigma \rightarrow 0$. This is because the weights that we have defined on the edges become infinite. This leads us to a situation, which is similar to the one-nearest neighbor propagation algorithm, which is not desirable. Hence we assume the value of $\sigma \neq 0$. On the other hand if we consider the value of $\sigma \rightarrow \inf$ then the edge weights become zero and we approach a state where all the points have similar labels, which is also undesirable.

$$H = -\sum_{ij} Y_{ij}.log(Y_{ij}) \tag{4.6}$$

Now that we have described the basic components of the label propagation algorithm, we try to investigate the effect of the initial number of labeled points in the data set. We ran sev-

eral experiments with number of labeled points ranging from two to twelve for a range of sigma values. In the Figure 4.1, 4.2 we show the entropy curves and the corresponding KLD curves for two classes walking and running. We also show the original KL Divergence (KLD) that we obtain after considering the ground truth labels of the points in the data set. Out main objective is to get the distribution as close to the original distribution. As we can see from Figure 4.1 the KLD between the estimated probability distributions approaches the KLD between the original distribution (assuming the ground truth labels are known), when the entropy in the Figure 4.2 reaches the minimum. Based on the results obtained, we choose the weight parameter σ as 0.65, for all the experiments performed in the following sections.

Algorithm 2: We use this algorithm for propagating the labels from the labeled points to the unlabeled points in the data set. It performs three main tasks 1) To propagate the already known labels based on the proximity of the nodes in the graph, 2) Normalizing the output label matrix, and 3) Check for convergence condition. We repeat these three steps until the algorithm converges. As seen from the algorithm we need to clamp the seed labeled points to make sure that they do not change.

Input: W_{ij} - Weight matrix, Y - Probability matrix, T_{ij} - Probability Transition Matrix, ϵ - Constant

Output: $(y_1, y_2, .., y_k) \in Y$ - matrix that represents the probability of the point belonging to each of the classes

1 **while** *true* **do**
2 $Y \leftarrow TY$;
3 Row normalize the matrix Y.
4 **if** $i \neq 1$ **then**
5 $okde \leftarrow nkde$;
6 **end**
7 **if** $i \geq 2$ **then**
8 $okld \leftarrow nkld$;
9 **end**
10 $nkde = KDE(y_{ci}), \forall c \in C \wedge any\ i \in N$;
11 **if** $i \neq 1$ **then**
12 $nkld = KLD(nkde, okde)$;
13 **if** $(i \geq 2 \wedge (okld - nkld)/(okld) \leq \epsilon)$ **then**
14 break;
15 **end**
16 **end**
17 Clamp the labels of the labeled data points
18 **end**

As described before, we use Algorithm 4.1 for label propagation. It essentially involves four steps; propagating the labels to the nearest neighbor, normalize the matrix, check for the convergence condition, clamp the original labels and repeat the procedure until it converges. In the above algorithm on line 3, the label matrix Y is row normalized using the Equation 4.6. The

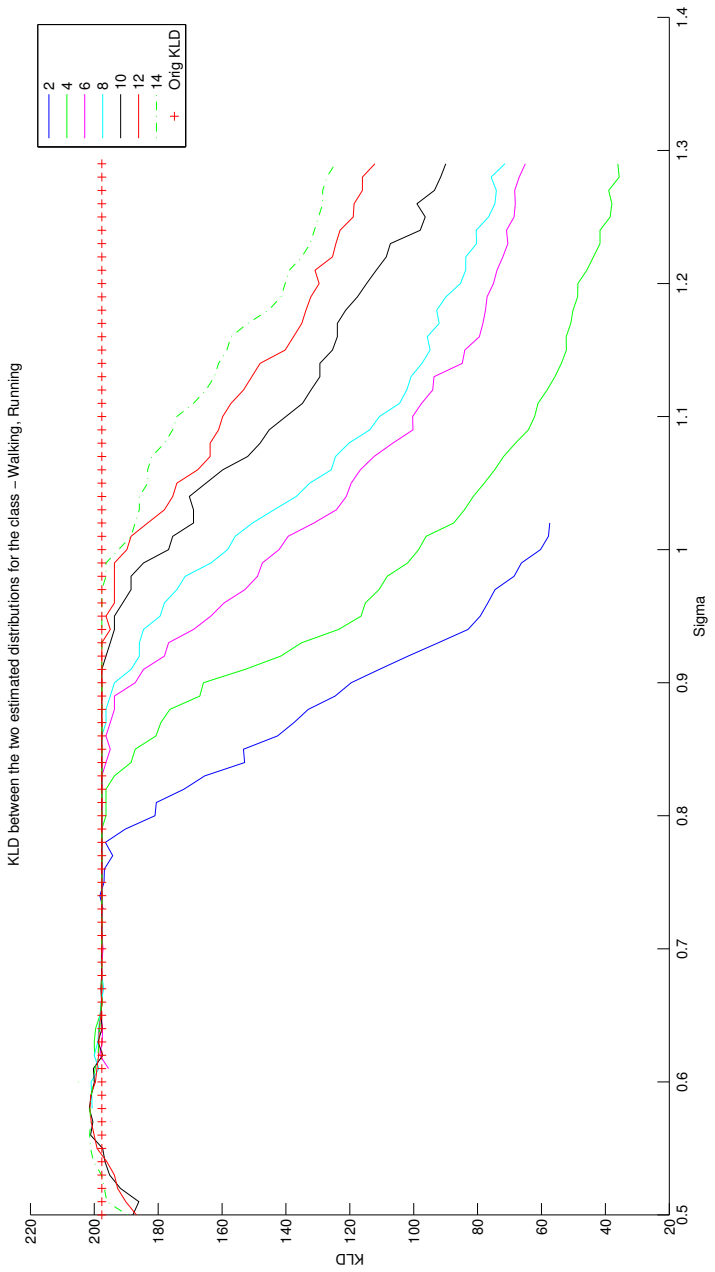


Figure 4.1: Kullback Leibler(KL) Divergence between two estimated KDEs (Kernel Density Estimates) for the classes - Walking, Running. We execute the label propagation algorithm with different number of initial labeled points. As we can notice in all the cases, we get the probability distribution close to the ground truth probability distribution for σ values close to 0.65.

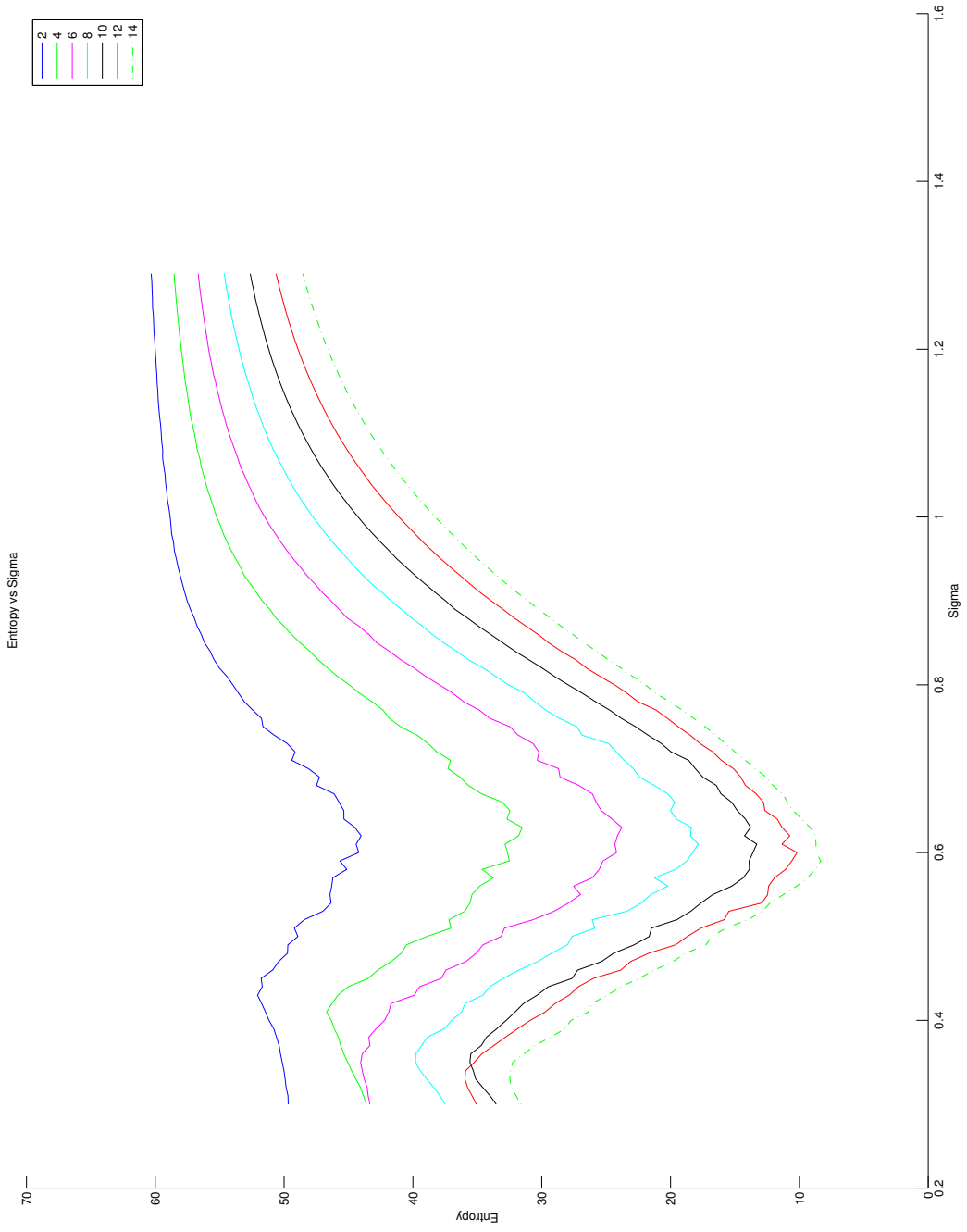


Figure 4.2: Entropy vs Sigma curve for activity classes - Walking, Running. We use this graph for choosing the weight parameter σ. This figure complements the Figure 4.1 by showing us that, we attain the minimum entropy value when the weight parameter σ is close to 0.65.

variable okld stores the value of the KLD at time t-1 and the variable nkld stores the value of KLD at time t. We clamp the labels of the original labelled data points to make sure that they do not change after the label propagation.

$$y_i = \frac{y_i}{\sum_{d=1}^{D} y_{id}}, \forall\, i = (L+1, L+2, ..., L+U) \tag{4.7}$$

4.2.1 Features

We now look at the features that we use for running our experiments. In order to test our proposed approach, we make use of the UCI iPhone data set used in [16]. This data set provides us nine different activities and nine individuals repeat each activity five times. Therefore, for the activities we consider, the total number of instances we have is 45, where each instance in the considered data set has a nine dimensional feature vector. We consider the features considered in [3]. They are -

1. Mean - Mean gives us the average acceleration value in a particular direction. We calculate the mean of all the values captured by the accelerometer in each direction. Therefore, we obtain three features, one in X, Y and Z direction.

2. Standard deviation - Standard deviation is a statistical measure that gives us the sense of how the data is distributed. We calculate standard deviation for the accelerometer values in all the three directions X, Y and Z. Therefore, we obtain three features using this measure.

3. Energy - We use energy to capture the data periodicity, as described in [24]. We calculate the energy value in each of the three directions X, Y and Z. We compute the energy as shown in equation 4.8. Therefore, we obtain three features using this measure.

$$E = \sum_{i=1}^{m} \frac{x_i^2}{|X|} \tag{4.8}$$

   Here $(x_1, x_2, .., x_m) \in X$ are the m points belonging to each of the direction of the accelerometer. As we can notice, we get three features from each type. In the next subsection, we will present some performance metrics that we use to measure the performance of the algorithm.

4.2.2 Performance Metrics

We consider two performance metrics: The KL Divergence, which provides us with a measure of accuracy of the label propagation technique and cost, incurred to achieve that. We define these two metrics as follows:

1. Kullback Leibler Divergence (KLD)

   We compute the KL Divergence between the estimated Kernel density estimation (KDE) for each class in the given data set. As we have defined in the preceding section, the KL Divergence provides us with a measure of how far we are from the probability distribution of the data set assuming the ground truth labels. The ideal case is when the the result is 0, as shown below.

$$D_{kde_1 \| kde_2} = \sum_i ln(\frac{kde_1(i)}{kde_1(i)}) kde_1(i) \Rightarrow \sum_i (0) * kde_1(i) \Rightarrow 0 \tag{4.9}$$

2. Cost of asking the oracle

   We add points to our labeled data set to increase the prediction accuracy. After the label propagation procedure, we select the next set of points and ask the oracle to provide us with the ground truth labels for these points. Every time we ask the oracle to reveal the ground truth label of a point, we incur a unit cost. Our objective is to converge with minimum cost.

4.3 Seed selection

In this section, we describe the way we choose the initial set of points for label propagation. One of the simplest approaches to pick the seed points is to randomly pick c points from the data-set where c is the number of classes. However, due to randomness the picked seed point might lie anywhere between the decision boundary and the cluster center which will result in unpredictable behavior. Hence we propose the Algorithm 3 to select our initial seed points. In this approach we make use of the K-Means [40] algorithm to provide us with k different clusters. Here we choose k equal to the number of classes. Now we select one point from each of the clusters, which is close to the cluster centroid, and add it to the list of seed points.

Algorithm 3: This algorithm describes the seed selection procedure, where we perform k-means clustering and choose the points that are closest to the cluster centroids. This technique tries to pick the most representative points for each class in the data set.

Input: $(x_1, x_2, .., x_N) \in X$ - the set of N points, k - Number of clusters needed.
Output: $(l_1, l_2, .., l_c) \in L$ - set of seed data-points

Let $[S_1, S_2, .., S_K] = kmedoids(X, k)$;
$i \leftarrow 1$;
for *each* $c \in C$ **do**
    if $Oracle(S_i) \in c$ **then**
        $l_i \leftarrow S_i$
    end
    $i \leftarrow i + 1$;
end

4.4 Baseline Algorithm

In the process of label propagation, we propagate the labels from a small set of labeled instances to a large number of unlabeled instances. Since the algorithm considers the proximity to the labeled points during label propagation, it becomes essential to choose the proper set of seed labeled data points. In the previous section, we proposed an algorithm to choose the seed points from the data set. The number of seed points chosen might be very small and the classification accuracy we obtain might not be the best. In order to improve the accuracy, we would want to iteratively consider additional points and add them to the labeled set. We can do this by asking the oracle for the label. Using this approach, we incur some cost and we need to make sure that we get the best accuracy by using minimum number of labeled points while incurring the least cost.

In this section, we show the baseline algorithm with which we compare our proposed approaches. One of the simplest baseline procedures is when we randomly select the next set of labeled points from the unlabeled data set. The result of this approach might be undesirable depending on the points picked by the system. We propose a smarter baseline algorithm. We will explain the working of the algorithm for two class problems, but we can use it for multi-class problems too.

In our proposed baseline algorithm we first run the K-Means algorithm to choose the seed points. Running K-Means gives us K clusters, each having a centroid. In our case, we would want the number of clusters to be equal to the number of classes (C) present in the data set. This ensures that we have at least one point from each of the class in the data set. Using the seed points we run the label propagation algorithm discussed above. Once the algorithm

converges, we now have the estimated labels for all the unlabeled data points. At this stage, the accuracy obtained is low as we started out with minimum number of seed points. Hence, we add K additional points to the labeled data set by asking the oracle to provide ground truth label to some of the points. Now that we have C sets of labeled points, where C is the number of classes, we run the K-Means algorithm on each of these C sets, so that we obtain K points for each of the classes. We now select one point from each of the classes, such that they all are distinct and are closest to the cluster centroid.

We now formally define the working of the algorithm. Let there be the N data points and C_1 and C_2 be the sets containing the indices of data points whose ground truth labels are class C_1 and class C_2 respectively. First, we select the seed points for label propagation using the Algorithm 3. Now to select the next pair of points we use the following technique. Let $x_{11}, x_{12}, .., x_{1m} \subset X_{1i}$ belong to class 1 after label propagation and $x_{21}, x_{22}, .., x_{2n} \subset X_{2j}$ belong to class 2 after label propagation such that $X_{1i} \cup X_{2j} = X$. We now run the K-Means algorithm on X_{cr} where $c \in C \ \wedge \ \forall r \in (1, 2, ..., n(c))$. Let M_c represents the points which are closest to each of the K centroids in the labeled class c. Finally we randomly select one point from M_c which has the same label as the ground truth label obtained from the oracle. The output of this algorithm is a list of L points from each of the classes respectively, which are added to the original seed set for next iteration of label propagation.

Algorithm 4 shows the idea discussed above. Here we can notice that we randomly pick one point from the set of K points selected from each class c. In order for us to add it to the seed set, we make sure that the ground truth label of the point is similar to the class c, which we assign during the label propagation. We do not consider the same point twice. As seen from the algorithm we are asking the oracle for the ground truth labels and every time we choose to add the point to the seed set and we incur a unit cost. We show the results and compare its performance with our proposed algorithm in Section 4.7. In the following section, we will look at how we can improve on the baseline algorithm.

4.5 Proposed Approach 1

In this section, we will discuss a different approach of choosing the next pair of points. In the previous section, we randomly choose one of the points from a list of points classified as label c_1 and c_2 in a two-class problem. Following this approach may not give us the best result as we may are may not be choosing the points with maximum entropy. Points having the maximum entropy are the one that have the maximum probability of misclassification. These points are

Algorithm 4: This algorithm describes the baseline procedure to select next set of points. After label propagation procedure, we run the K-Means algorithm on the points assigned to each of the classes. We now list the K points in each class, which are closest to their respective centroid. Finally, we randomly pick the point from this list whose ground truth label matches the assigned label.

Input: $(x_1, x_2, .., x_N) \in X$ - the set of N points, k - Number of clusters needed and ensure $k \leq n(X_c)\ \forall\ c \in C$, X_{Lc} - Set of points assigned label c.
Output: $(l_1, l_2, .., l_c) \in L$ - next set of data-points

Let M_c be the list containing the k Medoids for each class c.
for *each* $c \in C$ **do**
    while *true* **do**
        S_c = Pick a random point from the list M_c;
        if $Oracle(S_c) \in c$ **then**
            $l_c \leftarrow S_c$;
            break;
        end
        $M_c = (M_c \backslash M_c[S_c])$;
    end
end

hard to predict as they lie on the decision boundary. Since we are asking the oracle for additional set of seed points, we would get a better classification accuracy if we ask the oracle to provide us with the labels for these high entropy points.

We now discuss the working of our proposed algorithm. We again start with the label propagation process by using the seed label points obtained from Algorithm 3. After label propagation, we have C sets of labelled points, where C is the number of classes in the data set. As we have discussed previously we calculate the misclassification probability given by Equation 4.10, for each point in the all of the C sets and sort them in descending order. Sorting makes sure that the top most point has the maximum mis-classification probability. We now select these top most points from each of the C sets and add them to the seed set for next round of label propagation. Here we notice that the points selected might not belong to distinct classes.

$$P \leftarrow max(Y_j \backslash max(Y_j))\ \forall\ c \in C\ \wedge y_i \in Y \tag{4.10}$$

We formally present our idea in Algorithm 5. Let us consider a set of N data points in a D dimensional hyperspace, where a vector of D values represents each point. We normalize the values in each dimension so that they lie in the range of $[0 - 1]$. The seed selection procedure is the same as the one described in Algorithm 3. Once we run the label propagation algorithm

using these seed points and we have an updated label vector $\vec{y_i} = [\varphi_{i1}, \varphi_{i2}, .., \varphi_{iC}]\ \forall x_i \in X \wedge c \in C$ which gives the probability of the point being assigned to each class. We now assign the label to the point as the one that has the maximum probability. We add all the points having label c into set S_c. For example all the label vectors (y_i) of the points having maximum probability for class c_1 are assigned to set S_1 and similarly all the label vectors of the points having maximum probability for class c_2 are assigned to set S_2 in a two class problem. Now we choose the point associated with maximum probability of mis-classification from each set present in S_c and add them to the initial seed set for the next round of label propagation.

We present the results obtained using this algorithm in section 4.7 and compare it to the baseline algorithm. In the next section, we describe a variation of this algorithm where we make sure that the set of points selected belong to distinct classes.

Algorithm 5: This algorithm describes the proposed way to select next set of seed points for label propagation. We choose the points based on the mis-classification probability of the points. We select one point from each class whose mis-classification probability is the highest. Here we do not care if the selected points have distinct ground truth labels. We incur a unit cost when we ask oracle to reveal the ground truth label.

Input: $(x_1, x_2, .., x_N) \in X$ - the set of N points, S_c - Set of all the points having label $c\ \forall\ c \in C$

Output: $(l_1, l_2, .., l_c) \in L$ - set of new seed points

$P_c \leftarrow argmax_j(max(y_j \backslash max(y_j)))\ \forall\ c \in C\ \wedge\ y_j \in Y$;
$M_S \leftarrow \{P_1 \cup P_2 \cup .. \cup P_c\}$;
$totalcost \leftarrow 0$;
$i \leftarrow 0$;
while $i \leq n\{M_S\}$ **do**
    $totalcost \leftarrow totalcost + 1$;
    if $Oracle(M_S(i)) \in c$ **then**
        $l_i \leftarrow M_S(i)$;
        $i \leftarrow i + 1$;
    end
    else
        $M_S(i) \rightarrow point\ with\ next\ highest\ probability\ of\ misclassification.$
    end
end

4.6 Proposed Approach 2

The previous algorithm choose C points with the maximum probability of mis-classification, where C is the number of classes. We did not care if the selected points belonged to distinct classes. In this section, we present a new approach to selecting the new set of seed points as shown in Algorithm 6. In this algorithm, we make sure that the points chosen for the next iteration of label propagation are all distinct.

Formally, we can define the algorithm as follows. Let us consider a set of N data points in a D dimensional hyperspace. We select the initial set of seed points using Algorithm 3. Next, we perform label propagation using these seed points. After the label propagation algorithm we add all the points having label c into set $S_c \forall c \in C$. For example in a two class problem, all the label vectors (y_i) of the points having maximum probability for class c_1 are assigned to set S_1 and similarly all the label vectors of the points having maximum probability for class c_2 are assigned to set S_2 in a two class problem. As we have discussed previously we calculate the misclassification probability given by Equation 4.10, for each point in the set S_1 and S_2 and sort them in descending order. Now we choose one point form set S_1 whose ground truth label is c_1 and one point from set S_2 whose ground truth label is c_2 and add them to the initial seed set for the next round of label propagation. To generalize this, in a multi-class problem we keep picking the points from the each of the set S_c until all the points obtained have distinct class labels.

We need to ask the oracle for the ground truth labels and this operation has a unit cost associated with it. As stated in the beginning of this chapter, our goal is to minimize this cost while maximizing the prediction accuracy. In this procedure, we might see a considerable increase in accuracy and cost. The cost incurred by the system increases if the classes in the data set have a significant overlap in the distribution of points. This is due to the increasing amount of labels queried from the oracle to get the distinct set of points. Even though the cost incurred is high, we get better accuracy when compared to our previous approach. In the following section, we show the results obtained on the data set considered using the proposed algorithms and we will compare it to the baseline approach.

4.7 Results and Analysis

In the previous sections, we investigated two approaches to select the next set of points to add to the seed data set. Algorithm 5 describes a way to choose these points based on the maximum probability of misclassification to pick the points on the decision boundary. Algorithm 6 modifies the above procedure to make sure that all the points that we select belong to distinct classes. The main advantage of using the Algorithm 5 is that we get significantly lower cost and a slight increase in accuracy compared the base line algorithm. However, when we consider Algorithm 6, we achieve a significant increase in accuracy, but we incur more cost. We can make another observation, when there is a significant overlap in the distribution of points from different classes then the result obtained using both the algorithms converge.

To analysis our proposed algorithms, we consider the UCF data set [16] containing

Algorithm 6: This algorithm describes the proposed way to select next set of seed points from the data set. We choose the points based on the mis-classification probability of the points. We select one point from each class whose mis-classification probability is the highest. However, the selected points should belong to distinct classes.

Input: $(x_1, x_2, .., x_n) \in X$ - the set of N points, S_c - Set of all the points having label $c \; \forall \; c \in C$

Output: $(l_1, l_2, .., l_c) \in L$ - next set of data-points

$S_c \leftarrow max(Y_j \backslash max(Y_j)) \; \forall \; c \in C \; \wedge y_i \in Y$;
$S_c = sort(S_c, desc)$;
$totalcost \leftarrow 0$;
for *each* $c \in C$ **do**
    $i \leftarrow 0$;
    while $i \leq n\{S_c\}$ **do**
        $totalcost \leftarrow totalcost + 1$;
        if $Oracle(S_c(i)) \in c$ **then**
            $l_c \leftarrow S_c(i)$;
            break;
        end
        $i \leftarrow i + 1$
    end
end

four activities each with 45 instances. For the purpose of this experiment, we consider these four classes - running, climbing up the stairs, walking and descending the stairs. As discussed in the previous sections, we represent each instance using a nine dimensional feature vector. Although in these sets of experiments, we consider the transport mode verification problem, we can use the same techniques for label propagation on the recycle bin verification problem.

We measure the performance of the algorithm based on how close the predicted density estimate is from the original density assuming we know the ground truth. We measure this 'distance' measure using the Kullback Leibler divergence (KLD) introduced in the above section. In the following section, we show the results obtained for each pair of these classes and show the results obtained for multi-class problem.

Two class problem In this sub-section, we compare the results obtained using our proposed algorithm with the baseline algorithm. As we can notice from the table 4.1, using Algorithm 5 we incur the least cost and the density estimated after label propagation is much close to the ground truth distribution when compared to the baseline algorithm. If we consider the proposed Algorithm 6, we see that it takes more cost but gives a much closer density estimate. We incur high cost because we always make sure that the points chosen after executing the algorithm must belong to distinct classes. In the table given below, we show the results obtained

on two pairs of classes. In the first pair of classes, i.e. climbing up the stairs and descending the stairs there is a significant overlap in distribution of points. On the other hand, jumping and running are very distinct activities, and we can see that we get much better performance on these classes. We can use both the algorithms proposed by us on multi-class data set as well.

		Divergence between the estimated KDE and original for class 1	Divergence between the estimated KDE and original for class 2	Cost
climbing,descending	Baseline	13.62	00.87	05.84
	Algorithm 5	**08.59**	**00.07**	**04.00**
	Algorithm 6	09.16	00.69	06.00
jumping,running	Baseline	22.52	04.02	08.38
	Algorithm 5	16.01	04.04	06.00
	Algorithm 6	**06.42**	**0.94**	**26.00**

Table 4.1: Performance of our proposed algorithm with respect to the baseline algorithm for a two-class data set. We observe that Algorithm 5 gives us a cost effective way to propagate the labels among the unlabeled data points. However, Algorithm 6 gives us the probability density that is the closest to the ground truth probability density, but we incur high cost.

In this section, we have examined the performance of the proposed techniques for choosing the next set of points. We observe a 71% increase in performance compared to the baseline technique in a two-class example. In the next section, we conclude this chapter by providing the summary of this chapter.

4.8 Conclusion

In this chapter, we look at the semi-supervised learning techniques to solve the problem of activity verification. Semi-supervised learning techniques makes use of both the labeled and unlabeled data for predicting the labels. This is well suited for a problem like ours where the amount of labeled data available is small, and the cost of manual labeling is high. Given the

small amount of labeled data, human intervention is essential for accurate prediction. Every time we ask the oracle (human) for the ground truth label, we incur some cost and our objective is to minimize that cost and achieving high levels of prediction accuracy.

We start this chapter by introducing the problem of sparse labeled data sets and the concept of label propagation. We then discussed the work done in the area of semi-supervised learning in the recent past. We then introduce the graph-based approach for label propagation, where we assign the labels to the unlabeled data point based on its proximity to the labeled point. We introduce the entropy-based technique to assign the edge weights to this graph effectively. We start the propagation process with a minimum number of labeled points from each class. However, in large data sets it is unlikely for us to achieve good accuracy using this small initial set of points; hence we increase the size of seed set by picking unlabeled points from the data set and asking oracle for their label. We provide a baseline algorithm for picking this additional set of points. In the second half of this chapter, we provide two approaches to pick the points based on its misclassification probability effectively. Both these approaches outperform the baseline algorithm in terms of the distance of predicted probability distribution from the ground truth probability distribution and the cost incurred.

We tested the baseline and the proposed algorithms on the accelerometer data set. We examined four activities and performed by nine different people. In our experiments, we have tested our algorithm on two class problems, but the algorithms provided in this chapter work for multi-class problems as well. In the next chapter, we will conclude the thesis by providing the summary of the work done and the provide some of the future directions that could be taken by this research.

Chapter 5

CONCLUSION AND FUTURE WORK

In this research, we look at the problem of activity verification using mobile phones. Activity verification is the process of proving the user assertion pertaining to some activity performed by the user. In the first chapter, we provide some background on the problem of activity verification along with the motivation for investigating it. Our motivation lies in incentivizing the users who try to reduce their carbon footprint. There are lot of different activities that help us reduce our carbon footprint. However, two activities that have the greatest impact are the use of public transport and recycling. Hence, in this research we try to tackle these two problems. The other big motivation is the growing adoption of smart phones. With ubiquitous presence of smartphones having multiple sensors like accelerometer, barometer, gyroscope, microphone etc. we can record the impact the individual has on the environment. Given the evidence provided by the user, we investigate the different approaches we can take to verify this evidence.

First, we look at the supervised learning models and try to build a classifier for transportation mode and recycle verification. We obtain the data for transportation problem using the accelerometer present on the smart phone and use the images of recycle bin captured by the user as evidence for recycling. We train three classifiers- Naive Bayes, Decision Trees and Random Forest on the data set containing all the four modes of transport - Bike, Walk, Rail and Bus. For transport mode problem, we achieve an average accuracy of 93% and for recycling problem; we achieve an average accuracy of 75%. In both the cases random forest classifier outperforms the other two. We obtain a good accuracy for the transport problem because the data captured by the user provides higher confidence of classification and we do not need any additional evidence to prove the assertion. However, in the case of recycle bin verification it is hard to determine if the object in the image is a recycle bin. We further notice that, with this approach we need gather a large amount of labeled training data, which is time-consuming and expensive process. In the following chapter attempt to tackle the problem by asking the user for next best piece of evidence which maximizes the classification accuracy. We experiment with combining different data sets and we observe that the overall classification accuracy jumps from 80% to 93% when we consider the next best piece of evidence is considered.

Due to the expensive, time-consuming and impractical nature of acquiring large amounts of labeled data, we look at an alternative approach called semi-supervised learning. Here we

start with the assumption that the amount of labeled data is extremely small compared to the unlabeled data. We then propagate these labels to the unlabeled data points by using the proximity measure. In this research, we first provide the approach for selecting the labeled seed points from the data set. Once the seed points are selected, we provide a couple of approaches that we use to select the next best seed data points from the data set, which help us in achieving the label density close to the original density. Every time we pick an unlabeled data-point from the data set, we consult the oracle to know its true label and we incur some cost. Our goal is to reduce the overall cost by selecting the best set of points in the process of label propagation. We use our purposed technique to achieve the maximum of 71% increase in the performance when compared to the baseline algorithm for a two-class problem.

We can extended this research work in many ways. Here we have looked into only two of the modalities provided by the smart phones, which is image and accelerometer data. We would like to see the system decide on the right modality when asking the user for next piece of evidence. Viewpoint correlation was one such effort towards making the system direct the user towards the best piece of evidence. We would also like to explore other activities like verifying if the user consumes locally grown food or if the user turns off the air conditioning when not required etc. which are inherently hard to verify.

[1] Ali N Akansu, Wouter A Serdijn, and Ivan W Selesnick. Emerging applications of wavelets: A review. *Physical Communication*, 3(1):1–18, 2010.

[2] Ian Anderson and Henk Muller. Practical activity recognition using gsm data.

[3] Ling Bao and Stephen S Intille. Activity recognition from user-annotated acceleration data. In *Pervasive Computing*, pages 1–17. Springer, 2004.

[4] Nicola Bicocchi, Marco Mamei, and Franco Zambonelli. Detecting activities from body-worn accelerometers via instance-based algorithms.

[5] Leo Breiman. Random forests. *Machine learning*, 45(1):5–32, 2001.

[6] Chih-Chung Chang and Chih-Jen Lin. LIBSVM: A library for support vector machines. *ACM Transactions on Intelligent Systems and Technology*, 2:27:1–27:27, 2011.

[7] US EPA. United states, environmental protection agency, 2013. [Online; accessed 30-May-2013].

[8] Ákos Fábián, Norbert Győrbíró, and Gergely Hományi. Activity recognition system for mobile phones using the motionband device. In *Proceedings of the 1st international conference on MOBILe Wireless MiddleWARE, Operating Systems, and Applications*, MOBILWARE '08, pages 41:1–41:5, ICST, Brussels, Belgium, Belgium, 2007. ICST (Institute for Computer Sciences, Social-Informatics and Telecommunications Engineering).

[9] Andrew B Goldberg and Xiaojin Zhu. Seeing stars when there aren't many stars: graph-based semi-supervised learning for sentiment categorization. In *Proceedings of the First Workshop on Graph Based Methods for Natural Language Processing*, pages 45–52. Association for Computational Linguistics, 2006.

[10] Mark Hall, Eibe Frank, Geoffrey Holmes, Bernhard Pfahringer, Peter Reutemann, and Ian H. Witten. The weka data mining software: an update. *SIGKDD Explor. Newsl.*, 11(1):10–18, November 2009.

[11] Jonathan Lester, Phil Hurvitz, Rohit Chaudhri, Carl Hartung, and Gaetano Borriello. Mobilesense-sensing modes of transportation in studies of the built environment. *UrbanSense08*, pages 46–50, 2008.

[12] David G. Lowe. Object recognition from local scale-invariant features. In *Proceedings of the International Conference on Computer Vision-Volume 2 - Volume 2*, ICCV '99, pages 1150–, Washington, DC, USA, 1999. IEEE Computer Society.

[13] Jun Luo, Y. Ma, E. Takikawa, Shihong Lao, M. Kawade, and Bao-Liang Lu. Person-specific sift features for face recognition. In *Acoustics, Speech and Signal Processing, 2007. ICASSP 2007. IEEE International Conference on*, volume 2, pages II–593–II–596, 2007.

[14] Beatriz Maeireizo, Diane Litman, and Rebecca Hwa. Co-training for predicting emotions with spoken dialogue data. In *Proceedings of the ACL 2004 on Interactive poster and*

demonstration sessions, ACLdemo '04, Stroudsburg, PA, USA, 2004. Association for Computational Linguistics.

[15] Marianthi Markatou, Hong Tian, Shameek Biswas, and George Hripcsak. Analysis of variance of cross-validation estimators of the generalization error. *Journal of Machine Learning Research*, 6(2):1127, 2006.

[16] Corey McCall, Kishore Reddy, and Mubarak Shah. Macro-class selection for hierarchical k-nn classification of inertial sensor data. *Proc. of PECCS*, 2012.

[17] Emiliano Miluzzo, Nicholas D. Lane, Kristóf Fodor, Ronald Peterson, Hong Lu, Mirco Musolesi, Shane B. Eisenman, Xiao Zheng, and Andrew T. Campbell. Sensing meets mobile social networks: the design, implementation and evaluation of the cenceme application. In *Proceedings of the 6th ACM conference on Embedded network sensor systems*, SenSys '08, pages 337–350, New York, NY, USA, 2008. ACM.

[18] Jean-Michel Morel and Guoshen Yu. Asift: A new framework for fully affine invariant image comparison. *SIAM Journal on Imaging Sciences*, 2(2):438–469, 2009.

[19] M Mun, Deborah Estrin, Jeff Burke, and Mark Hansen. Parsimonious mobility classification using gsm and wifi traces. In *Proceedings of the 5th Workshop on Embedded Networked Sensors, HotEmNets*, volume 2008. Citeseer, 2008.

[20] Stefan Munder and Dariu M Gavrila. An experimental study on pedestrian classification. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, 28(11):1863–1868, 2006.

[21] Eric Nowak, FrÃľdÃľric Jurie, and Bill Triggs. Sampling strategies for bag-of-features image classification. In AleÅą Leonardis, Horst Bischof, and Axel Pinz, editors, *Computer Vision âĂŞ ECCV 2006*, volume 3954 of *Lecture Notes in Computer Science*, pages 490–503. Springer Berlin Heidelberg, 2006.

[22] Greg Pass, Ramin Zabih, and Justin Miller. Comparing images using color coherence vectors. In *Proceedings of the fourth ACM international conference on Multimedia*, MULTIMEDIA '96, pages 65–73, New York, NY, USA, 1996. ACM.

[23] C. Randell and H. Muller. Context awareness by analysing accelerometer data. In *Wearable Computers, The Fourth International Symposium on*, pages 175–176, 2000.

[24] Nishkam Ravi, Nikhil D, Preetham Mysore, and Michael L. Littman. Activity recognition from accelerometer data. In *In Proceedings of the Seventeenth Conference on Innovative Applications of Artificial Intelligence(IAAI*, pages 1541–1546. AAAI Press, 2005.

[25] Sasank Reddy, Min Mun, Jeff Burke, Deborah Estrin, Mark Hansen, and Mani Srivastava. Using mobile phones to determine transportation modes. *ACM Transactions on Sensor Networks (TOSN)*, 6(2):13, 2010.

[26] Chuck Rosenberg, Martial Hebert, and Henry Schneiderman. Semi-supervised self-training of object detection models. *Seventh IEEE Workshop on Applications of Computer Vision*, 2005.

[27] Burr Settles. Active learning literature survey. *University of Wisconsin, Madison*, 2010.

[28] Catarina Silva and Bernardete Ribeiro. On text-based mining with active learning and background knowledge using svm. *Soft Computing*, 11(6):519–530, 2007.

[29] Timothy Sohn, Alex Varshavsky, Anthony LaMarca, Mike Y Chen, Tanzeem Choudhury, Ian Smith, Sunny Consolvo, Jeffrey Hightower, William G Griswold, and Eyal De Lara. Mobility detection using everyday gsm traces. In *UbiComp 2006: Ubiquitous Computing*, pages 212–224. Springer, 2006.

[30] Carolin Strobl, Anne-Laure Boulesteix, Achim Zeileis, and Torsten Hothorn. Bias in random forest variable importance measures: Illustrations, sources and a solution. *BMC bioinformatics*, 8(1):25, 2007.

[31] Michael J Swain and Dana H Ballard. Color indexing. *International journal of computer vision*, 7(1):11–32, 1991.

[32] M. Tico, T. Haverinen, and P. Kuosmanen. A method of color histogram creation for image retrieval. In *ProceedingsoftheMordicSig. Proc. Symposium, Kolmarden, Sweden*, pages 157–168, 2000.

[33] Simon Tong and Edward Chang. Support vector machine active learning for image retrieval. In *Proceedings of the ninth ACM international conference on Multimedia*, MULTIMEDIA '01, pages 107–118, New York, NY, USA, 2001. ACM.

[34] Simon Tong and Daphne Koller. Support vector machine active learning with applications to text classification. *J. Mach. Learn. Res.*, 2:45–66, March 2002.

[35] Aditya Vailaya, Anil Jain, and Hong Jiang Zhang. On image classification: City images vs. landscapes. *PATTERN RECOGNITION*, 31:1921–1935, 1998.

[36] Chieh-Chih Wang and Ko-Chih Wang. Hand posture recognition using adaboost with sift for human robot interaction. In *Recent progress in robotics: viable robotic service to human*, pages 317–329. Springer, 2008.

[37] T. Wiedmann and J. Minx. A definition of âĂŸcarbon footprintâĂŹ. *CC Pertsova, Ecological Economics Research Trends*, 2:55–65, 2007.

[38] Wikipedia. Decision tree — wikipedia, the free encyclopedia, 2013. [Online; accessed 15-May-2013].

[39] Wikipedia. F1 score — wikipedia, the free encyclopedia, 2013. [Online; accessed 16-May-2013].

[40] Wikipedia. K-means clustering — wikipedia, the free encyclopedia, 2013. [Online; accessed 20-May-2013].

[41] Wikipedia. Kernel density estimation — wikipedia, the free encyclopedia, 2013. [Online; accessed 16-May-2013].

[42] Wikipedia. Kernel density estimation toolbox for matlab (uci), 2013. [Online; accessed 15-May-2013].

[43] Wikipedia. KullbackâĂŞleibler divergence — wikipedia, the free encyclopedia, 2013. [Online; accessed 16-May-2013].

[44] Wikipedia. Naive bayes classifier — wikipedia, the free encyclopedia, 2013. [Online; accessed 31-May-2013].

[45] Wikipedia. Precision and recall — wikipedia, the free encyclopedia, 2013. [Online; accessed 16-May-2013].

[46] Wikipedia. Random forest — wikipedia, the free encyclopedia, 2013. [Online; accessed 15-May-2013].

[47] Wikipedia. Support vector machine — wikipedia, the free encyclopedia, 2013. [Online; accessed 15-May-2013].

[48] Guoshen Yu and Jean-Michel Morel. ASIFT: An Algorithm for Fully Affine Invariant Comparison. *Image Processing On Line*, 2011, 2011.

[49] Xiaojin Zhu. Semi-supervised learning literature survey. *Book*, 2005.

[50] Xiaojin Zhu and Zoubin Ghahramani. Learning from labeled and unlabeled data with label propagation. Technical report, Technical Report CMU-CALD-02-107, Carnegie Mellon University, 2002.

[51] Xiaojin Zhu, Zoubin Ghahramani, and John Lafferty. Semi-supervised learning using gaussian fields and harmonic functions. In *MACHINE LEARNING-INTERNATIONAL WORKSHOP THEN CONFERENCE-*, page 912, 2003.