EXPLOITING LOCAL INFORMATION FOR TRAJECTORY CLASSIFICATION UNDER SURVEILLANCE

Ph.D. THESIS

by

RAJKUMAR SAINI

DEPARTMENT OF COMPUTER SCIENCE & ENGINEERING
INDIAN INSTITUTE OF TECHNOLOGY ROORKEE
ROORKEE - 247 667 (INDIA)
JANUARY, 2019

EXPLOITING LOCAL INFORMATION FOR TRAJECTORY CLASSIFICATION UNDER SURVEILLANCE

A THESIS

Submitted in partial fulfilment of the requirements for the award of the degree

of

DOCTOR OF PHILOSOPHY

in

COMPUTER SCIENCE & ENGINEERING

by

RAJKUMAR SAINI

DEPARTMENT OF COMPUTER SCIENCE & ENGINEERING
INDIAN INSTITUTE OF TECHNOLOGY ROORKEE
ROORKEE - 247 667 (INDIA)
JANUARY, 2019

INDIAN INSTITUTE OF TECHNOLOGY ROORKEE ROORKEE

CANDIDATE'S DECLARATION

I hereby certify that the work which is being presented in the thesis entitled "**EXPLOITING LOCAL INFORMATION FOR TRAJECTORY CLASSIFICATION UNDER SURVEILLANCE**" in partial fulfilment of the requirements for the award of the Degree of Doctor of Philosophy and submitted in the Department of Computer Science & Engineering of the Indian Institute of Technology Roorkee, Roorkee is an authentic record of my own work carried out during a period from December, 2014 to January, 2019 under the supervision of Dr. Partha Pratim Roy, Assistant Professor, Department of Computer Science and Engineering, Indian Institute of Technology Roorkee, Roorkee.

The matter presented in this thesis has not been submitted by me for the award of any other degree of this or any other Institution.

(RAJKUMAR SAINI)
Candidate

This is to certify that the above statement made by the candidate is correct to the best of my knowledge.

(Partha Pratim Roy)
Supervisor

The Ph.D. Viva Voce Examination of **Rajkumar Saini**, Research Scholar, has been held on **25th January 2019**.

Chairman, SRC **Signature of External Examiner**

This is to certify that the student has made all the corrections in the thesis.

Signature of Supervisor **Head of the Department**

Dated:

Dedicated

to

Dr. Partha Pratim Roy

and

Dr. Pradeep Kumar

Acknowledgements

At the onset, I pay my sincere gratitude towards the Almighty for his blessings to pursue PhD at my dream institute, Indian Institute of Technology. I strongly believe that without the strength, courage and determination provided to me by the Almighty, I would have never been able to write this PhD Thesis.

I am extremely grateful to my research supervisor Dr. Partha Pratim Roy for his valuable guidance, providing the required inputs and consistent encouragement throughout the research work. I express my gratitude towards him for trusting in my ideas and let me implement the ideas by providing the required components for experimental study. A person with an amicable and positive disposition, he has always made himself available to clarify my doubts despite his busy schedules. His constant constructive opinions and suggestions about my work has been substantial in improving the quality of the work. I would like to sincerely thank him for providing me the constant support and motivation at my times of distress and giving me ample time and freedom to pursue research.

My gratitude towards my research committee members Dr. Durga Toshniwal, Dr. Balasubramanian Raman and Dr. Debashis Ghosh for assessing me at various stages of PhD and providing me their valuable opinions which helped in improving the research work. My sincere gratitude towards the reviewers of my research papers and thesis work for providing their suggestions and comments to improve my research work. I would like to thank Dr. Debi Prosad Dogra, Assistant Professor, School of Electrical Sciences, IIT Bhubaneswar, and Dr. Kanjar De, Post Doctoral Fellow, IIT Roorkee for providing their valuable suggestions and serving as a mock reviewer and proof reader of our papers. Further, my sincere thanks to Prof. Umapada Pal, CVPR Unit, ISI Kolkata for his constructive comments and suggestions on my work during an internship with him.

I would like to acknowledge my co-research scholars at IIT Roorkee including Pradeep Kumar, Gaurav Varshney, Ajay Choudhary, Nishant Sinha, Vivekraj VK, Vikas Chauhan, Arun Pundir, Anshul Arora, Swati Gupta, Prateek Keserwani, Tofik Ali, Pallavi Kaushik and Anmol Gupta with

whom I enjoyed my stay in the institute and for their support and personal help whenever required.

RAJKUMAR SAINI

Abstract

Object motion trajectory classification is the important task, especially when we aim to detect abnormal movement patterns in order to take appropriate actions for prohibiting unwanted events to occur. Given a set of trajectories recorded over a period of time, they can be clustered for understanding usual flow of movement or segmentation of unusual flows.

With the advancement in low-cost sensors object trajectories can be recorded efficiently with ease. These sensors can be RGB video camera, depth camera like Kinect, and Global Positioning System (GPS). With the use of GPS, the real world coordinates of objects along with other related information are tracked and later processed for the real-time analysis of mass flow, crowd analysis, and anomaly detection in the flow.

Video based trajectory analysis could be on-line or off-line. In on-line, objects are tracked in the live videos and there motion is analyzed immediately to make the higher order decisions like prohibiting the objects to enter restricted area, unstable areas like fire and floods, and situations like violence. Video trajectory classification is also done off-line, where object trajectories are first extracted from the recorded videos. Next, their motion is analyzed for off-line analysis by classifying the trajectories into different classes.

In this thesis, we have focused on off-line analysis for the classification of object trajectories using the publicly available datasets. Using the local information along with global information is an efficient way to improve classification performance. To compute the local cues from trajectories, models could be built by partitioning the trajectories into variable number of segments based on the geometry of the trajectories.

A graph based method for trajectory classification has been proposed. Each trajectory has been partitioned into varying number of segments based on its geometry. Complete Bipartite Graph (CBG) is formed between each trajectory pair and there Dynamic Time Warping (DTW) distance is used as the weight of the edge between them. Local costs are computed from the CBG and then fused (using Particle Swarm Optimization (PSO)) with the global cost (global cost computed using the same full length trajectory-pair) to improve the classification performance.

We have also proposed a kernel transformation followed by trajectory classification framework that make the use of information from local segments. The proposed kernel perform the shrinking of trajectories in such a way they preserve their shape. Modified trajectories have been segmented with the help of segmental HMM and their local responses have been recorded. These local responses along with global responses (from full length trajectories) have been fused using genetic algorithm to make the final decision.

A surveillance scene segmentation has been performed based on the results of trajectory classification using HMM. The scene layout is divided into 10×10 local non-overlapping grids and majority voting based scheme is applied to assign each block a label showing the importance of the blocks with the help of region association graph based features. Such off-line analysis helps to understand the flow of motion within the viewing field of video camera.

Contents

Acknowledgements **i**

Abstract **iv**

List of Figures **ix**

List of Tables **xiii**

1 Introduction **1**

1.1 Motivation . . . 3

1.2 Research Gaps . . . 4

1.3 Problem Statement and Research Scope . . . 6

1.4 Specific Research Contributions . . . 6

1.5 Organization of the Thesis . . . 7

2 Literature Survey **9**

2.1 Trajectory Modeling for Classification or Clustering . . . 9

2.1.1 Pattern Similarity Measures . . . 9

2.1.2 Trajectory Classification . . . 10

2.2 Graph based Trajectory Modeling . . . 13

2.3 Motion Pattern based Scene Segmentation . . . 14

2.4 Research Gaps . . . 15

3 Trajectory Classification using Graph Algorithm **17**

3.1 Introduction . . . 17

3.2 Proposed Work . . . 18

3.2.1 Pre-processing and Feature Extraction . . . 21

3.2.2 Modeling Global Behavior using DTW 21
3.2.3 Modeling Local Behavior 22
3.2.4 Fusion of GC, LC, and MSTC using PSO 26
3.3 Experiments and Results 27
3.3.1 Experimental Results 27
3.4 Discussion 33

4 A Segmental HMM based Trajectory Classification using Genetic Algorithm 35
4.1 Introduction 35
4.2 Proposed Framework on Trajectory Classification 36
4.2.1 Kernel Formation 37
4.2.2 Trajectory Classification using GHMM and SHMM 40
4.3 Experimental Result and Analysis 43
4.3.1 Classification Improvement using the Proposed Kernel 45
4.3.2 Combination of GHMM and SHMM Results using GA 50
4.3.3 Results with Added Gaussian Noise 52
4.3.4 Time Computation and Comparative Study 53
4.3.5 Beyond Trajectory Classification: Signature Recognition 54
4.4 Discussion 58

5 Scene Segmentation based on Motion Trajectory Patterns 61
5.1 Introduction 61
5.2 Trajectory Classification and Scene Segmentation 62
5.2.1 High-Level Feature Extraction 63
5.2.2 Supervised Classification using HMM 64
5.2.3 Surveillance Scene Segmentation 65
5.3 Experimental Results 66
5.3.1 Datasets and Ground Truths 66
5.3.2 HMM Classification Results 66
5.4 Discussion 69

6 Conclusion and Future Work 71

A Supplementary Material **73**

A.1 Dataset Description 73

A.2 Hidden Markov Model (HMM) 75

A.3 Support Vector Machine (SVM) 77

A.4 Random Forest (RF): 77

A.5 Comparison with SVM and RF Classifiers 78

A.6 BLSTM-NN based Sequence Classification 79

A.6.1 CTC based BLSTM-NN 79

A.6.2 CEE based BLSTM-NN 79

Bibliography **81**

List of Figures

1.1 Few examples of trajectories (red, green, blue) plotted over a scene image. They represent the different movement pattern followed by some vehicles in the surveillance area in the field of view of camera. 3

1.2 Samples of two trajectories associated with same class label. The shape and the length of trajectories are different. 4

1.3 Example of trajectories from two classes (a) and (b) containing overlapping boundaries as depicted in (c). Note that, the classes show paths of trajectories from two different roads. Different color represent different trajectory class. 5

3.1 Framework of proposed system. Global and local costs (GC, LC, & MSTC) are extracted using DTW, CBG, and MST from featured trajectories and fused using PSO. 19

3.2 Example of DP algorithm. Trajectory is partitioning into segments using DP (1-6) . Final break points are shown in the last (6). 23

3.3 CBG formulation and cost computation. (a) Two segmented trajectories, (b) Corresponding CBG, (c) Local cost within trajectory segments and (d) Aggregated cost ($\triangledown$) computation. 25

3.4 Examples of trajectory samples after segmentation. (a) T15 (b) LabOmni (c) CROSS. Segments are plotted with different colors. For better visibility of colors please see the pdf version. 28

3.5 Experimental results using proposed method. Fusion of *GC, LC,* and *MSTC* improves the classification results for the datasets used in this study namely, T15, LabOmni, and CROSS. 29

3.6 Qualitative results of the proposed method. (a1), (a2), & (a3) Trajectories with ground truth (b1), (b2), & (b3) Incorrectly classified using full length trajectories (c1), (c2), & (c3) Corrected by proposed method. T15 (row 1) LabOmni (row 2) CROSS (row 3). 30

3.7 Erroneous Trajectories. (a1), (a2), & (a3) Trajectories with ground truth (b1), (b2), & (b3) Incorrectly classified using full length trajectories (c1), (c2), & (c3) Incorrectly classified by proposed method. T15 (row 1) LabOmni (row 2) CROSS (row 3). 31

3.8 Comparison of results of proposed method using ACO, GA, and PSO. 34

4.1 A flow-chart of the framework used in the proposed trajectory classification. 37

4.2 Convex Hull and RDP sketch: CH in red and RDP in blue. 40

4.3 SHMM: Sequence of frames are associated with HMM states. 41

4.4 Example of separation of a trajectory into 3 segments by SHMM (Note that the length of the segments are not equal). Best viewed in color. 41

4.5 Example of chromosome encoding. Here, each chromosome is having length $(h+1)$. 43

4.6 Proposed framework comprises trajectory shrinking, two stage HMM classification and optimization using GA. 44

4.7 Classification Result on T15: First column (a1 & a2) shows the dataset where different colors represent different classes. Column 2 (b1 & b2) shows two different classes and column 3 (c1 & c2) shows a single raw trajectory from the corresponding class shown in column 2. Column 4 (d1 & d2) and column 5 (e1 & e2) show the new trajectories (corresponding to trajectories in column 3) with their predicted class formed by AMKS and the proposed kernel, respectively. Proposed framework correctly classifies these trajectories (column 5) as compared to AMKS (column 4). 46

4.8 Classification Result on MIT Car: First column (a1 & a2) shows the dataset where different colors represent different classes. Column 2 (b1 & b2) shows two different classes and column 3 (c1 & c2) shows a single trajectory from the corresponding class depicted in column 2. Column 4 (d1 & d2) and column 5 (e1 & e2) show the new trajectories (corresponding to trajectories in column 3) with their predicted classes formed by AMKS and the proposed kernel, respectively. Proposed framework correctly classifies these trajectories (column 5) as compared to AMKS (column 4). 46

4.9 HMM Classification over T15 and MIT dataset with varying GMMs and HMM states: (a) Performance of GHMM with varying GMMs, (b) Performance of GHMM with varying HMM states. 48

4.10 HMM Classification over T15 and MIT dataset with varying GMMs and HMM states: (a) Performance of SHMM with varying GMMs, (b) Performance of SHMM with varying HMM states. 49

4.11 Performance with T15(HMM states=3 & GMM=64) and MIT(HMM states=3 & GMM=128) datasets by varying the difference of top two target probabilities, Δp. . 50

4.12 Classification Results using MIT and T15 datasets. Green bar shows the accuracy of GHMM, The accuracy of combined classification of GHMM and SHMM results with equal weightage is and is shown in Blue. Proposed framework improves the classification and is shown in Red. 51

4.13 Performance with T15(HMM states=3 & GMM=64) and MIT(HMM states=3 & GMM=128) datasets by varying population size in GA. 52

4.14 Impact of noise on classification performance. 53

4.15 Signature samples: First row shows three samples of subject S1. Similarly, second row is of subject S2. 56

4.16 Computation of curvature feature using two neighbors P_1 and P_3. 56

4.17 GHMM classification performance over SVC2004 dataset. (a) With HMM states 4 and varying GMM components. (b) With varying number of HMM states using GMM 32. 57

4.18 SVC2004 Results using GHMM and GA based combination. 58

5.1 RAG based segmentation map generated using the method proposed in [48] when applied on IIT dataset. 63

5.2 An example demonstrating the process of association of a block to a particular segment or region. 65

5.3 (a) The background scene of IIT dataset divided into 10×10 number of blocks with overlayed trajectories (b) Labeling of the blocks using the method described in [48] (c) Construction of the graph nodes using labels (d) Color-coded representation of the segmented scene, and (e) RAG corresponding to the segmentation. 67

5.4 (a) The background scene of MIT dataset divided into 10×10 number of blocks with overlayed trajectories (b) Labeling of the blocks using the method described in [48] (c) Construction of the graph nodes using labels (d) Color-coded representation of the segmented scene, and (e) RAG corresponding to the segmentation. 67
5.5 HMM Classification (a) ($k = 2$) for IIT dataset (b) ($k = 4$) for MIT dataset 68
5.6 HMM Classification based Segmentation (a) ($k = 2$) Segmentation of IIT dataset using [48] (b) Segmentation of IIT dataset using our approach (c) ($k = 4$) Segmentation of MIT car dataset using [48] (d) ($k = 4$) Segmentation of MIT car dataset with our approach 69

A.1 Samples of T15 trajectory datasets. Classes are represented by color. For better visibility of colors please see the pdf version. 73
A.2 Samples of LabOmni trajectory datasets. Classes are represented by color. For better visibility of colors please see the pdf version. 74
A.3 Samples of CROSS trajectory datasets. Classes are represented by color. For better visibility of colors please see the pdf version. 75
A.4 MIT Dataset (number of classes=6). Color represent trajectory classes. 75
A.5 Variants of HMM: (a) 3-state ergodic model (b) 3-state let-to-right model. 76
A.6 RF classification process. 78

List of Tables

3.1 Computation time elapsed at different stages of proposed method in seconds (Sec) . 32
3.2 Classification accuracy comparison with other methods 33

4.1 Performance of individual kernels 45
4.2 Classification Result: First row shows the results using T15 dataset and second row shows the results for MIT dataset. 47
4.3 Comparison of time computation between AMKS algorithm and proposed approach 53
4.4 Performance comparison with other methods. 54
4.5 Performance of proposed work using ACO and GA 54

5.1 Results on IIT dataset (Training:50% and Testing: 50%), GT: Ground Truth 68
5.2 Results using MIT car dataset (Training: 50% and Testing: 50%), GT: Ground Truth 69

A.1 Comparison with SVM, and RF classifier 78

Chapter 1

Introduction

Trajectory analysis is the area that deals with the study of objects by analyzing their motion dynamics. It involves the classification or clustering of motion patterns for the understanding of behavior of objects to make higher order decisions such as scene segmentation, lane classification, traffic analysis [1, 3, 4], autonomous driving [5–7], parkinglot system [8], and motion anomaly detection [9, 10]. Objects could be humans, robots, and vehicles. Before making the classification or clustering, first, motion or the movements of the objects are recorded with the help of sensors like Video camera, GPS, and vehicular adhoc networks. Next, trajectory information is tracked [11–16] and extracted from it. Finally, models are built using the extracted trajectories which involve trajectory pre-processing, feature extraction, model training followed by the final classification.

There exist methods to classify the data patterns in supervised fashion through classifiers as well as in unsupervised fashion through clustering [17–20] that employ global or full length trajectories. Such methods are adopted for the classification or clustering of trajectory data. The classifiers such as Hidden Markov Model (HMM) [21], Neural Network [16, 22, 23], Support Vector Machine (SVM) [24], K-NN [25] and clustering algorithm like k-means [27] have been used for images, time series, and trajectory classification. Similarity measures like Dynamic Time Warping (DTW) [28, 29, 32], Longest Common Sub-Sequence (LCSS) [17], adaptive dissimilarity index [36], link structure [37] are widely used for sequence matching like signatures and are capable of matching trajectories. Graph based methods [38, 39] have also been proposed for similarity matching and rating prediction [39]. These classifiers, clustering algorithms, and similarity measures are keys for building a classification system and can be used either directly or indirectly for trajectory analysis.

In addition to these, kernels are defined [41,42] to project the trajectory points into another space so that trajectories can be modified to accomplish the predefined objective. The objective could be to reduce the intra-class variance, or to improve the classification performance. Such kernels use trajectory information like positions, and speed. to make an update to the original trajectories to generate new representation.

Also, the graph based methods can be exploited in several ways for trajectory classification. Like, trajectories can be structured into graphs by considering trajectory points as graph nodes or a trajectory itself can be considered as a single node of the graph and pair-wise similarity can be computed using similarity matching algorithms like DTW [28,32]. In this case, few trajectories can be chosen as template or reference for estimating the matching score of a test trajectory.

Though the global information i.e. use of full length trajectories as a whole is the straight forward solution for trajectory modeling, local information [43] can also be exploited. The cues estimated globally can also be estimated over local segments. Later, the local and global responses can be combined to optimize the performance. The optimization can be done using iterative algorithms like Genetic Algorithm (GA) [44,45], and Particle Swarm Optimization (PSO) algorithm [46,47]. Such a way is a lead to improved models that can classify the trajectories with high classification rate.

Typically, a trajectory $T(\in \mathbb{R}^2)$ of an object can be viewed as a sequence of position coordinates that the object follows as given in Eq. 1.0.1. These coordinates either be image coordinates or they may belong to the real world. Such a few samples of trajectories are depicted in Figure 1.1.

$$T = \{(x_1, y_1), (x_2, y_2), (x_3, y_3), ..., (x_n, y_n)\} \tag{1.0.1}$$

Thus, given a set of trajectories, the aim is to correctly classify them to respective clusters. This would help to understand the underlying motion.

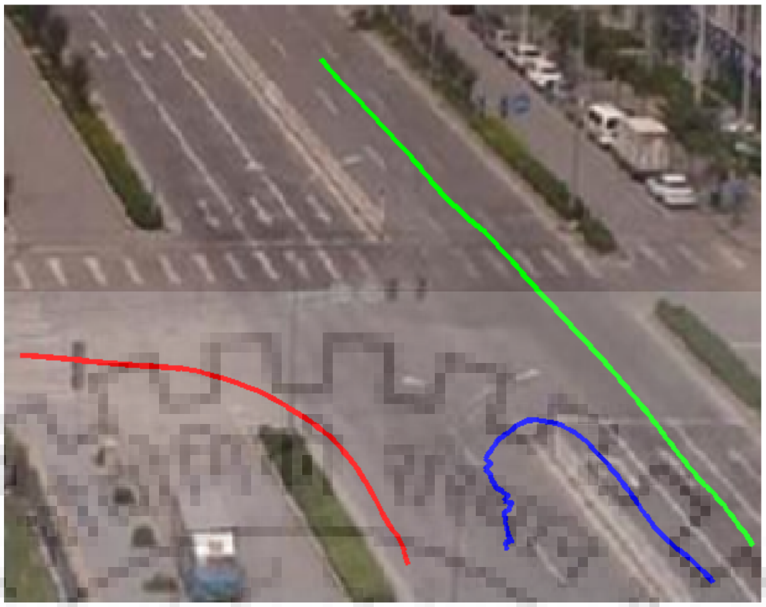


Figure 1.1: Few examples of trajectories (red, green, blue) plotted over a scene image. They represent the different movement pattern followed by some vehicles in the surveillance area in the field of view of camera.

1.1 Motivation

Trajectory classification is an important problem as it helps in understanding the nature of movements of the objects by analyzing their motion pattern. Here, nature may represent the class of a trajectory it should belong to. Thus, trajectory modeling for the classification is the mapping from motion pattern to corresponding class. Trajectory classification models analyze the motion patterns and associate them to some classes. These classes are defined as per the application for which the models are built e.g. lane detection and classification, finding usual motion patterns, and anomaly detection. Such problems are very crucial in today's world and need high attention for making efficient systems like traffic analysis, security, scene understanding, and post event analysis.

Trajectory classification faces various challenges that make it difficult to correctly classify the trajectories into corresponding clusters. First, any two trajectories may not be having identical length in general even if they belong to same class or cluster. Similarly, two trajectories from different classes may have identical length. This may happen due to the motion dynamics during the tracking of the object in motion and obviously the tracking algorithm. Figure 1.2 shows two trajectories from the same class plotted on the scene image. However, they differ in structure and their lengths are also not same. This makes trajectory classification a tedious task.

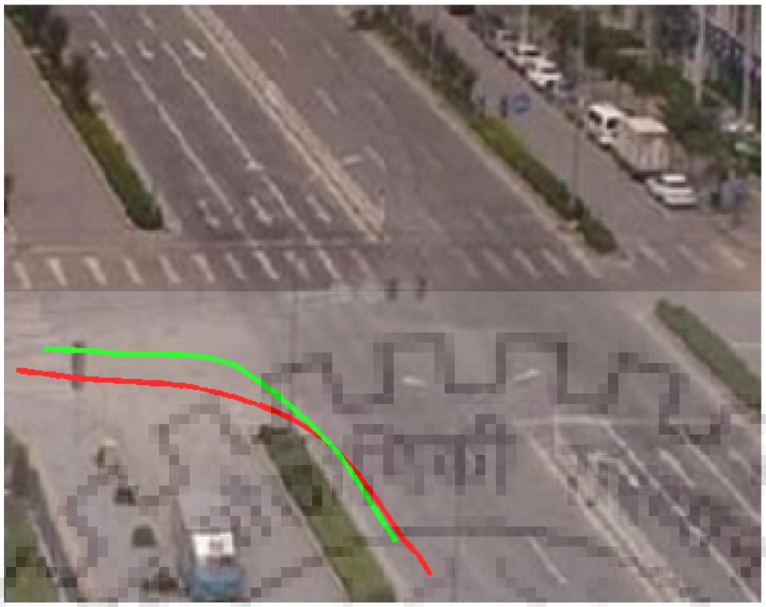


Figure 1.2: Samples of two trajectories associated with same class label. The shape and the length of trajectories are different.

Secondly, given a set of labeled trajectories, it is quite possible that the number of trajectories with a specific class may not match with the number of trajectories belong to the other class labels. This situation is often seen in practice. This may create problem while building the classification models. Let us take a scenario where we have two classes, say C_1, and C_2. Let us assume that the number of trajectories associated with C_1 be 15 and the number of trajectories associated with C_2 be 100. In this particular case, the classification model may be biased towards class C_2. This may cause a high classification error. This is the another issue with trajectory classification.

Thirdly, it is usual with trajectories having overlapping i.e. trajectories from two different classes may overlap with each other. Such classes may have high confusion in the classification results. Figure 1.3 shows the samples of trajectories from two different classes plotted over the surveillance scene image. It can be noticed that there is high overlapping among them.

1.2 Research Gaps

By doing a comprehensive literature survey, following are the research gaps we have come up with.

- There exist datasets on which the previous methods have lower classification rates. Kernel

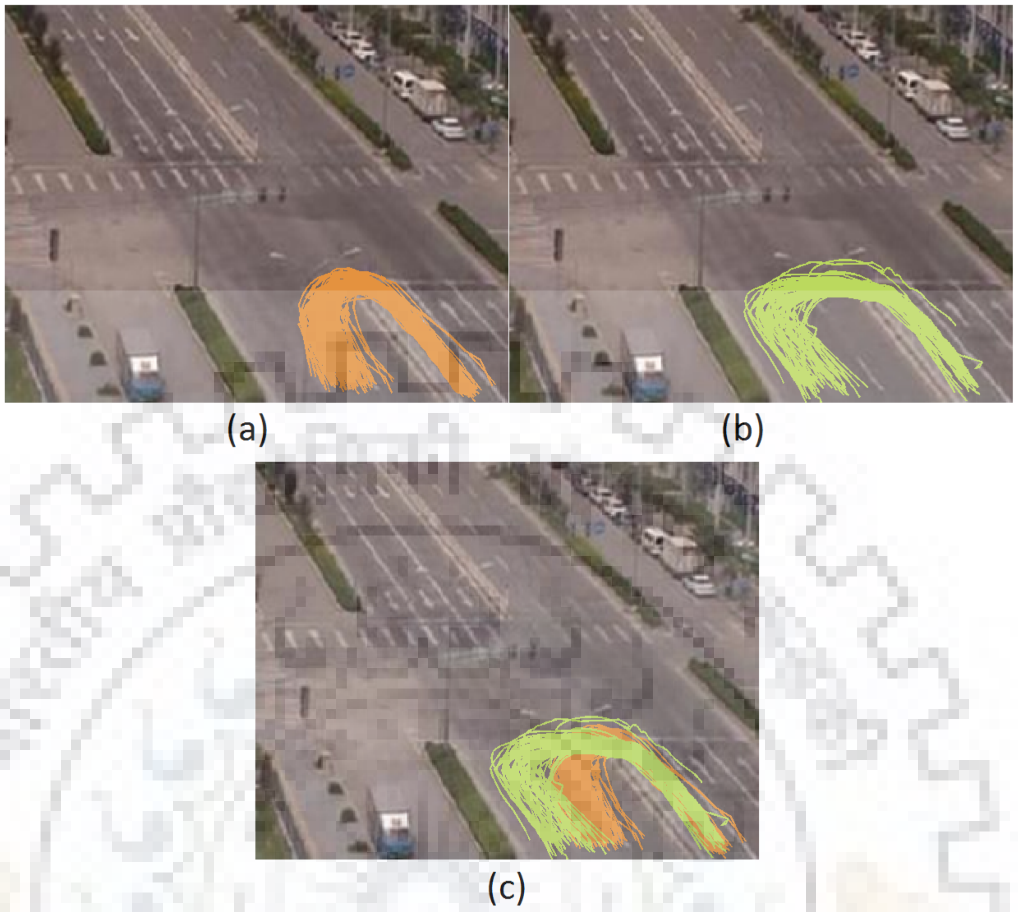


Figure 1.3: Example of trajectories from two classes (a) and (b) containing overlapping boundaries as depicted in (c). Note that, the classes show paths of trajectories from two different roads. Different color represent different trajectory class.

transformations and combinations can be applied here. Hence, there is a need to improve methodology so that higher classification rate can be achieved.

- Previous methods use full length trajectories, thus, cues from local segments can be incorporated to make a better classification system.

- Scene understanding plays a vital role in understanding the movement flow in the field of view of camera which can further be used to predict the upcoming flow. As per our findings there existing scene segmentation is based only on texture information of scene's layout. There is no significant work on scene segmentation based on movement patterns in the area under coverage.

Based on the above mentioned discussion we define the research scope as given below.

1.3 Problem Statement and Research Scope

To study various types of trajectories and then analyzing them for classifications/prediction from recorded trajectories. The following objectives need to be addressed.

1. Improving trajectory classification using local information. Local information can be exploited by partitioning the trajectories into multiple segments.
2. Improving classification performance in standard trajectory datasets using the proposed kernel and classifier combination. Defining a kernel transformation before trajectory classification.
3. Scene segmentation based on motion trajectory patterns in area under surveillance. Extracting the graph based features for scene segmentation.

1.4 Specific Research Contributions

In this thesis, we have proposed local information based methods for trajectory classification using surveillance trajectory datasets. Main contributions of the work are as follows.

- A graph based classification model has been proposed. A global cost is determined for each trajectory pair considering full length. Next, each trajectory is partitioned into local segments. The number of segments may not be same for a trajectory pair, in general. Using the segments, Complete Bipartite Graph (CBG) is formed and two local costs are determined from CBG. The global and local costs are then combined in a linear fashion using Particle Swarm Optimization (PSO). Finally, the classification results are computed from the combination.
- A new kernel along with segment-wise learning and combination based model has been proposed. Segmental Hidden Markov Model base approach has been proposed in this study. The new kernel uses trajectory positions, speed, end points, convex hull points, and douglas-peucker points. HMM partitions the trajectories into fixed number of segments and learns from individual segments along with full length trajectories. Genetic algorithm has been used for the combination of individual scores.

- Scene segmentation into local grids based on motion trajectory patterns in area under surveillance has been proposed. In this work, the surveillance scene has been first divided into 10×10 non-overlapping grids. Two features namely, *block-label* and *node-number* has been extracted based on the work of [48]. Then the featured trajectories have been classified using HMM followed by the scene segmentation using the classification results.

1.5 Organization of the Thesis

The thesis is divided into 6 Chapters. Each Chapter can be read independently without requiring going back and forth. The content of each Chapter is described below.

Chapter 1: This Chapter gives the introduction of the proposed work. In particular, the Chapter describes about: what is a trajectory, what is trajectory classification, How is it important, what is the motivation behind it, what are the challenges, and what are the possible applications of trajectory classification. This Chapter also covers the contribution of the work.

Chapter 2: In this Chapter, we present the review of the works related to trajectory classification. How the researchers have tackled the problem of trajectory classification i.e. supervised or unsupervised, similarity based, or classifier based. Based on the review, we found the scope of research and contributed towards it.

Chapter 3: This Chapter presents a new method based on complete bipartite graph (CBG) and PSO for trajectory classification. In this work, we fuse the global and the local costs (Local costs computed from CBG) using PSO to improve the trajectory classification performance. Method is tested using publicly available trajectory datasets.

Chapter 4: A new kernel and a new method based on the segmental HMM has been discussed in this Chapter. The experimental results of the proposed method using public datasets have also been presented. A detailed analysis of the results of this methods has also been discussed along with the future possibilities of this work.

Chapter 5: A method for surveillance scene segmentation into local grids based on region association graph and block importance using HMM is presented in this Chapter. The method has been compared with the existing work by implementation on public datasets.

Chapter 6: In this Chapter, we summarize our findings and the future directions that we plan to focus on.

Chapter 2

Literature Survey

We organize the literature survey into Sections. In the first Section, we mention about the trajectory classification and clustering approaches. In the second Section, we present the scene segmentation related works. Graphs have also been used for trajectory modeling, such works have been discussed in the third Section. The details are as follows.

2.1 Trajectory Modeling for Classification or Clustering

Spatio-temporal information of trajectories has widely been used in anomalous activity recognition as proposed in [9, 10, 49] or detecting unusual movement patterns. In addition to that, moving object trajectories have other applications including behavior analysis of the moving objects as given in [50], traffic planning as in [1], movement pattern mining [51], scene segmentation or classification as in [48]. In this Section, we present some of the existing research works that mainly focus on pattern similarity measure and trajectory classification.

2.1.1 Pattern Similarity Measures

Finding similar patterns in object movement helps to understand the complex movement patterns and localize objects with abnormal behavior. Researchers have used both supervised and unsupervised learning approaches for finding similarity score between trajectories. In unsupervised classification approaches, Dynamic Time Warping (DTW) [30–32] and Longest Common Sub-Sequence (LCSS) have been successfully used for finding pairwise similarity among trajectories which are then grouped into clusters using algorithms like K-means, Gaussian model [52] or similar clustering algorithms [2, 53]. Authors of [28] proposed a path clustering method applicable to video surveillance to identify suspicious behavior of moving objects using DTW. These similarity measures and classifiers have also been used in document processing [33–35] and are now being used in trajectory clustering and classification. Authors of [17] proposed a three-stage hierarchical learning framework for analyzing object activities in surveillance system. This included node learning or extraction of points of interest using Gaussian mixture modeling, spatial learning or routes between points of interest, and spatio-temporal learning for path activity using Hidden Markov Model (HMM). The authors used LCSS distance measure for trajectory clustering using Fuzzy C-Means algorithm.

Supervised classification algorithms adopt a different modeling technique with the help of HMM [21, 54] or Neural Networks [22, 55] for finding similarity in trajectories or gestures. Authors of [21] have proposed a human activity recognition system in video sequences as a two stage process. Firstly, the authors estimated low-level models, i.e. mean translation and covariances using Expectation Maximization (EM) algorithm. Next, the low-level models have been used to estimate transition matrices for each activity class using HMM.

Authors of [56] have proposed a moving object detection and tracking system to detect abnormal events in visual surveillance using Self Organizing Maps (SOM). Detection of moving objects has been done using background modeling. The behavior recognition has been performed using two Kohenen maps that distinguishes normal and abnormal activities based on the local motion and global elliptic Fourier descriptors.

2.1.2 Trajectory Classification

Trajectory classification is a three-step process that includes trajectory estimation (tracking), feature extraction, and grouping. However, majority of the existing methods primarily focus on detection and tracking of objects before modeling their behavior. In [50], the authors have proposed dynamic scene understanding methodology to identify the abnormal behavior of moving objects in video surveillance. The trajectories have been represented as a sequence of symbols with features including shapes, speed and crossed zones and are grouped into homogeneous clusters by using a kernel based clustering algorithm. In [10], authors have proposed a hierarchical framework for video anomaly detection in crowded scene by analyzing global and local spatio-temporal contexts. The authors extracted 8-dimensional motion feature vector using histogram of optical flow which has been used to discover atomic activity patterns using K-means clustering. However, their approach might report an abnormal activity as normal if the objects move slowly because of the low magnitude of optical flow.

In [54], the authors proposed a trajectory pattern learning and an anomaly detection framework applicable to traffic surveillance. The authors used Main Flow Direction (MFD) feature to group trajectories using K-means algorithm for distinguishing outliers. The authors used HMM for route pattern establishing in each cluster that helped in anomaly detection within a new trajectory based on maximum likelihood threshold. The authors of [57] have proposed a model for trajectory analysis and semantic region retrieval using hierarchically linked HMM. Their framework retrieves the semantic regions by modeling temporal dependencies between trajectories using shared transition and emissions. An anomaly detection based framework [58] has been proposed using Hierarchical Dirichlet Process HMM (HDP-HMM). The framework models the human dynamics using GPS trajectories to understand human motion area of dynamic traffic control. The model uses normal trajectory data to detect contextual anomalies present in time series information of human motion using latent states estimated using HDP-HMM.

A time series classification model using Hidden-Unit Logistic Model (HULM) has been proposed in [59]. The model uses binary stochastic hidden units. Temporal dependencies are modeled by connecting the hidden units in chains. The model works well on problems such as action recognition, and character recognition. An online behavior classification framework has been proposed in [60]. The authors use conformal predictor and multi-factor non-conformity measures using multidimensional trajectories. The framework has used multi-factor Hausdorff nearest neighbor conformal classification to classify frequent behaviors. Their framework works as an early warning surveillance system in military and realistic civilian scenario. In [40], authors have proposed a framework for modeling driver's behavior in decision making during phase transition of signal flashes using HMM. The framework observes the vehicles speed and acceleration (or deceleration) while the driver pass the signal or stops during signal flashing (3 flashes green, yellow and red).

Visual surveillance has grasped substantial attention of the computer vision research community. A large number of research [61–65] have been reported on this topic over the last few years. In many of these work, motion trajectories are first extracted from videos and further analyzed to classify movement patterns on the basis of spatio-temporal relation. Classification can be applied to anomalous activity recognition [9, 10, 49] or to filter out unusual patterns. Analysis of moving object trajectories has several applications such as moving object behavior analysis [50, 66], traffic planning [1], pattern mining [51], and scene classification [67].

Existing work can be categorized into two groups i.e. (a) pattern similarity measure and (b) supervised classification. Pattern similarity measures may find similarity between two trajectories using Dynamic Time Warping (DTW) [68], and Longest Common Sub-Sequence (LCSS) [69]. Trajectory classification can be done using K-means, and HMM [70, 71].

Similar movement patterns [52, 72–74] can be useful to understand inherent complex patterns movements. Researchers have used supervised [52,72] as well as unsupervised [69,73–77] methods for classification of trajectories. For unsupervised classification, DTW [68, 69] and LCSS [69, 78] is popularly used to find out the pairwise similarity between trajectories and then group them into clusters using K-means, Gaussian kernel based clustering [52] or other clustering approaches [53]. For supervised classification, researchers have often used Dynamic Bayesian Network (DBN) [79], HMM [54], and Neural Networks [22, 55, 80, 81]. A semantic region modeling framework using hierarchical Bayesian model called dual-Hierarchical Dirichlet Processes (dual-HDP) has been proposed in [76]. In this paper, authors have focused mainly on distinguishing between normal and abnormal trajectories where trajectories with low likelihood have been considered as abnormal.

In general, video-based trajectory classification comprises with three steps: (a) Trajectories are first extracted from the videos (b) Feature extraction from raw trajectories (c) Building a model for the classification of trajectories. Trajectory extraction involves tracking of moving objects. Majority of the tracking approaches assume that the objects are first detected and tracked throughout the sequence [61, 82] and the activities are then modeled as sequences of object movements. However, some authors [83, 84] extract the trajectories by directly extracting motion and appearance based features without relying on the use of a tracking module.

Object size, velocity, mean and standard deviation of trajectory positions are low-level features used to represent the trajectories [85]. Bag of Features (BoF) [86] is also used to find the key point trajectories in classification. Corners and object color can be used as features for tracking objects [87]. Classification can be applied on raw trajectories. Bashir et al. [70] have used HMM for classification, where trajectories are first segmented based on curvature and represented by principal components generated using Principal Component Analysis (PCA).

Nascimento et al. [21] have used Switched Dynamic HMM (SD-HMM). They have modeled trajectories as the sequence of dynamic objects using EM and switching amongst models are realized by HMM chains. A new approach for multivariate time series anomaly detection has been proposed by transforming them into univariate series in [71]. Authors have used Fuzzy C-Means (FCM) algorithm after z-score normalization and then used HMM for anomaly detection.

These work use full length trajectories for modeling. Thus, we have proposed a model that uses both global and local trajectory scores to improve trajectory classification. The model could further be extended for applications like anomaly detection.

2.2 Graph based Trajectory Modeling

Trajectory classification has been addressed by the researchers to solve different real world applications such as traffic analysis [88], scene analysis [89], anomalous activity understanding [10, 90], scene segmentation [91]. It helps in predicting the class label of unknown trajectory based on its path and features. A number of methods have been proposed in literature to model the behavior of trajectories based on the spatial and temporal information. In [58], authors proposed an anomaly detection in human dynamics from time-series gridded data. They used a hierarchical dirichlet process hidden markov model based distribution to decide whether a time-series is anomalous or normal. Thus, authors formulated anomaly detection as a two-class problem. Their method performed decent in terms of precision (62%) and recall (95%), however, experiments were performed using synthetic data.

Graph based matching has been efficiently used in document analysis [26]. However, graph based trajectory classification is a novel research area where different video trajectories are grouped into regions. The construction of such regions can be done with the help of density or distance based approaches which ultimately results into clusters with spatial information. A graph based simultaneous localization and mapping framework to model robot motion has been addressed in [92]. The authors have used least-square minimization for efficiently localizing trajectories to address the problem of simultaneous localization and mapping. Authors of [90] proposed a vehicle inspection framework to find abnormal activity using bipartite graphs. The modeling is done using vehicle's registration district and inspection stations as nodes of bipartite graph and number of vehicle registered making inspection as edge weights. Normalized scores have been used to coin abnormality in vehicle inspection stations. The work has efficiently used bipartite graph, however focused only on abnormality in inspection station.

Authors of [93] proposed a graph based method to find the relevant patterns from the GPS data. The authors focused to reduce the points by finding representative points using graph partitioning. Using delauney triangulation from constructed graphs the prominent regions were extracted. A trajectory analysis and semantic region retrieval using Hierarchical Linked Infinite Hidden Markov Model (HLI-HMM) based framework has been proposed in [57]. The authors have localized the semantic regions using trajectory information. These regions include actual regions and not just trajectory points. The framework works well in finding semantic regions, however, no spatial and temporal dependencies have been modeled in their work.

Authors of [45] compared the performance of parallel genetic algorithm and PSO for the path planning of unmanned aerial vehicles (UAV) in 3D environment. Similarly, in [94], authors have proposed a trajectory optimization for motion planning of three-link robot manipulator using PSO. normalized step cost was used as particles to optimize. Authors of [95] proposed a trajectory generation algorithm using constraint PSO for hypersonic reentry vehicles. Authors used PSO to find the velocity-dependent bank angle profile for reentry of of these vehicles. [96] used PSO of the the optimization of time-jerk trajectory planning for robot motion in 3 degree of freedom.

2.3 Motion Pattern based Scene Segmentation

There are many works on trajectory classification; still, ample scope is there on high-level or context based scene segmentation. As camera based surveillance has reached almost every corner of our society, it has become necessary to automate such systems for tackling the surveillance task with higher efficiency and lesser dependence on human observers. Thus, detection of uncommon movements can be accomplished with computer based systems. To achieve this goal, installed systems must have better understanding of surveillance scene which cannot be achieved without acceptable scene segmentation. Majority of the existing scene segmentation techniques use low level features such as location of the object center [67] or movement pattern learned through velocity or displacement [48].

Unsupervised approach has its own benefits in abnormal activity detection. However, sometimes it is beneficial to use supervised approach for scene understanding since such techniques first learn from the scene and then investigate. In addition to that, supervised learning-based approaches are likely to produce better results because of the availability of ground truths. For learning through supervised approach, one requires training samples and it is desired to have samples within appropriate feature space. It is cumbersome to take decision about the feature space and relevant threshold. Though simple features like time-series representation of an object's trajectory (x_i, y_i, t_i) can work in local pattern analysis, however, complex representation of the scene dynamic may not be possible using such simple features. High level features, such as the block importance or label of a block as proposed by Dogra et al. [48] can be used to represent semantic change in a trajectory over its course of execution. For example, the RAG (Region Association Graph) based segmentation of a scene proposed in their method can be useful to represent the moving object's path. Traces of raw trajectories of two objects, τ_i and τ_j can be completely different even if they move in a similar fashion, therefore, making the classification process tricky. However, property exhibited by the object trajectories moving in similar fashion can be exploited to distinguish the class they belong to.

2.4 Research Gaps

By doing a comprehensive literature survey, following research gaps have been found.

- There exist datasets on which the previous methods have lower classification rate. Kernel transformations and combinations can be applied here. Hence, there is a need to improve methodology so that higher classification rate can be achieved.

- Previous methods use full length trajectories, thus, cues from local segments can be incorporated to make a better classification system.

- Scene understanding plays a vital role in understanding the movement flow in the field of view of camera which can further be used to predict the upcoming flow. As per our findings the existing scene segmentation techniques are based only on the texture information in the scene layout. There is no significant work on scene segmentation based on movement patterns in the area under coverage.

Chapter 3

Trajectory Classification using Graph Algorithm

Graphs can be efficiently used for trajectory classification. We have formulated the trajectory classification problem into graph based similarity problem using Douglas-Peucker (DP) algorithm and complete bipartite graphs. Local behavior of objects has been analyzed using their motion segments and Dynamic Time Warping (DTW) has been used for finding similarity among motion trajectories. Class-wise global and local costs have been computed using DTW and their fusion has been done using Particle Swarm Optimization (PSO) to improve the classification rate.

3.1 Introduction

Graph based consideration of the road structure or the road network can be used as key to understand underlying motion. Road network modeling [97] with GPS and RFID based data define road networks as a graph structure modeling for finding frequent patterns of the data. Trajectories are also analyzed using the mutual information and the sparse reconstruction [98] to facilitate trajectory classification. Learned features from the trajectories are used to address the problem like behavior analysis. From the learned feature representations driver's behavior can be estimated for road lane changing [99], abnormal movement detection [100], and route modeling [101].

Though, modeling global behavior using full length trajectories has its significance, incorporating local behavior can make classification more robust leading to improved performance. Local behavior can be modeled using trajectory segments. Trajectories can be partitioned into segments with the help of curve simplification algorithm such as Douglas-Peucker (DP) [102–104]. Such trajectory segments analyzed individually and their response can be fused in order to optimize classification rate. The optimization of global and local behavior is possible and can be achieved using evolutionary algorithm like Particle Swarm Optimization (PSO) [47, 105]. PSO is the optimization method that mimic the behavior of swarm intelligence to find the possible optima within the search space. Thus, the PSO algorithm can be modeled in corresponds to the problem of trajectory classification by combining individual responses of global and local trajectory segments.

In this work, we have proposed a trajectory classification framework that effectively uses the DTW algorithm to capture global behavior using full length trajectories and segment-wise local behavior from the trajectory segments. Douglas-Peucker (DP), Complete Bipartite Graph (CBG), and Minimum Spanning Tree (MST) have been used to model the local responses between trajectories. Finally, fusion of global and local costs has been coded using Particle Swarm Optimization (PSO) to efficiently improve the trajectory classification performance.

The main contributions of this work are as follows.

1. A trajectory classification approach that uses DTW based Global Cost (GC), and a Complete Bipartite Graph (CBG) and Minimum Spanning Tree (MST) based local behavior of featured trajectories in terms of Local Costs (LC and MSTC) has been proposed.
2. PSO has been implemented to combine global (GC) and local (LC and MSTC) costs for improving the classification rates. Robustness of the proposed method has been tested on publicly available trajectory datasets.

3.2 Proposed Work

A trajectory could be defined in general as an ordered series of object's positions in the monitoring area. Objects such as vehicles, robots, drones or humans can be traced with the help of sensors such as video camera, satellite or proximity sensors. Video trajectories are extracted by tracking objects in video frames, whereas satellite tracking gives the real world coordinates of the object

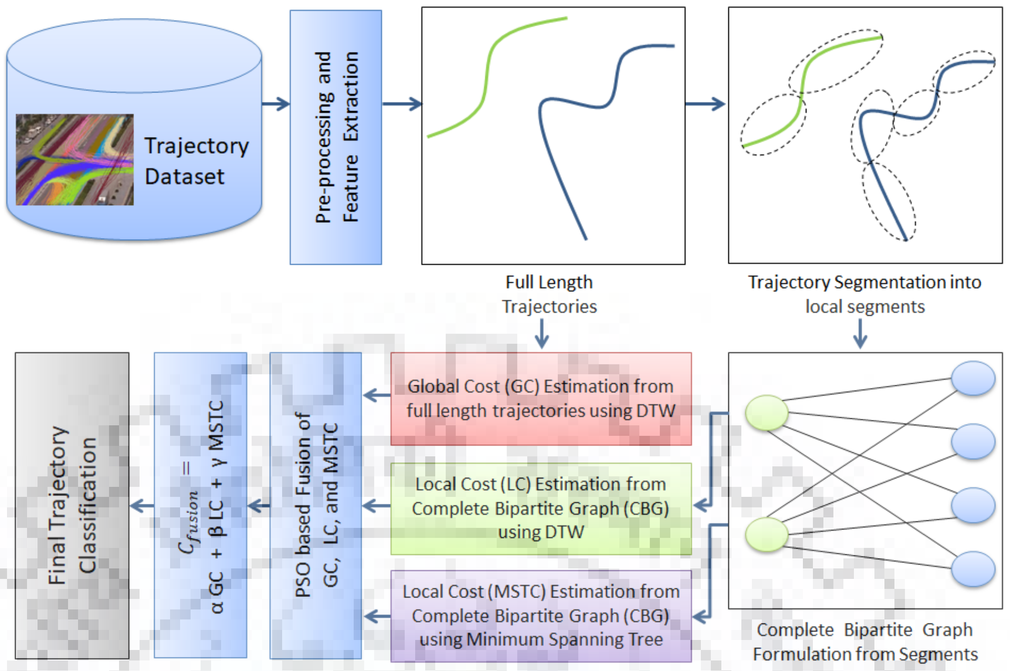

Trajectory Dataset
Pre-processing and Feature Extraction
Full Length Trajectories
Trajectory Segmentation into local segments
Global Cost (GC) Estimation from full length trajectories using DTW
Local Cost (LC) Estimation from Complete Bipartite Graph (CBG) using DTW
Local Cost (MSTC) Estimation from Complete Bipartite Graph (CBG) using Minimum Spanning Tree
Complete Bipartite Graph Formulation from Segments
PSO based Fusion of GC, LC, and MSTC
C_{fusion} = α GC + β LC + γ MSTC
Final Trajectory Classification

Figure 3.1: Framework of proposed system. Global and local costs (GC, LC, & MSTC) are extracted using DTW, CBG, and MST from featured trajectories and fused using PSO.

being tracked. In this way, a trajectory τ^i can be defined using Eq. 3.2.1

$$\tau^i = \{(x_1^i, y_1^i), (x_2^i, y_2^i), \ldots(x_{|\tau^i|}^i, y_{|\tau^i|}^i)\} \tag{3.2.1}$$

where (x_t^i, y_t^i) represents the position of object at time t and $|\tau^i|$ is the number of position vectors (x_t^i, y_t^i) in trajectory τ^i. It is important to note that number of points $|\tau^i|$ is not same for all the trajectories not even in the case when trajectories belong to same class.

Given a set of such trajectories, we have proposed a trajectory classification based on DTW using the fusion of global and local costs. Figure 3.1 shows the block diagram of proposed work. First, trajectories are preprocessed and features are extracted. Pair-wise costs among full length trajectories (referred to as Global Cost, *GC*) are computed using DTW. *GC* is then organized class-wise by taking mean for each trajectory class as described in Section 3.2.2. Next, the local behavior is modeled using trajectory segments and bipartite graph as described in Section 3.2.3. Local cost based on minimum spanning tree (MSTC) is also computed from CBG. Finally, *GC, LC*, and *MSTC* are fused using PSO to optimize the classification rate as described in Section 3.2.4. Algorithm 1 describe flow of the proposed system in detail.

Algorithm 1 Steps of Proposed Framework

```
Input: Reference set $S$ and Test set $R$.
Output: Final classification result (F).
    % Global Cost (GC) estimation
 1: Compute pair-wise DTW distance using full length trajectories from $S$ and $R$ using Eq. 3.2.3
    and Eq. 3.2.4, respectively.
 2: Compute $\Delta$ as in Eq. 3.2.5
 3: Compute Global Cost ($GC$) using Eq. 3.2.6

    % Complete Bipartite Graph (CBG) formulation
 4: Trajectory partitioning into local segments using DP.
 5: Create CBG ($G(\{U, V\}, E)$ for each trajectory pair such that.
    (a) Trajectory segments represents nodes of CBG.
    (b) $U$ is the set of nodes corresponding to one trajectory. Similarly, $V$ corresponds to the other.
    (c) There is no edge between $u_i, u_j \in U$, and $v_k, v_l \in V$.
    (d) Define edge $e(u_l, v_m$ between each pair of nodes $u_l \in U, v_m \in V$ of CBG ($G$) as their DTW
    distance as given in Eq. 3.2.7

    % Local Cost (LC and cost of MST (MSTC) estimation from CBG
 6: Define $W$ as given in Eq. 3.2.8
 7: Compute $\nabla^{col}$, $\nabla^{row}$ for the column and rows of $W$ using Eq. 3.2.9, and Eq. 3.2.10, respectively.

 8: The pair-wise local cost is between two trajectories is computed using Eq. 3.2.11
 9: Compute class-wise local cost $LC$ using Eq. 3.2.13
10: Find the minimum spanning tree (MST) of CBG using Kruskal's or Prim's algorithm [106].
11: Compute the cost of MST termed as MSTC.
12: Combine $GC$, $LC$, and $MSTC$ (Eq. 3.2.16 to find optimal $\alpha, \beta$, and $\gamma$ using PSO using
    validation data.
13: Apply $\alpha$, $\beta$, and $\gamma$ to combine $GC$, $LC$, and $MSTC$ computed using test data.
14: return F.
```

3.2.1 Pre-processing and Feature Extraction

Given the set of trajectories, first, trajectories are normalized using z-score normalization. Next, features are extracted from the normalized trajectories.

3.2.1.1 Cumulative Average Response (CAR)

Given a trajectory τ of finite length, CAR can be seen as average over a growing window. We define CAR as given in Eq. 3.2.2

$$CAR(\tau(k)) = \frac{1}{k}\sum_{i=1}^{k}\tau(i) \tag{3.2.2}$$

3.2.1.2 Convex Hull (CH)

Given a set of points, Convex Hull (CH) may be defined as the smallest convex polygon containing the points within the interior or on the boundary of the polygon. Thus, each trajectory may be represented as a sequence of 6-dimensional feature vector i.e. $< x, y, CAR, CH >$.

3.2.2 Modeling Global Behavior using DTW

The pair-wise global cost is computed using DTW algorithm as defined below.

3.2.2.1 Dynamic Time Warping (DTW)

DTW [28, 107, 108] is used as a similarity measure to find the cost between two sequences $P = \{p_1, p_2 ... p_{|P|}\}$ and $Q = \{q_1, q_2 ... q_{|Q|}\}$ of variable length. It gives the cost of optimal warping path using Eq. 3.2.3

$$DTW(i,j) = \eta(p_i, q_j) + min\begin{cases} 0 & i = 0 \,\&\, j = 0 \\ \infty & i \neq 0 \,\&\, j = 0 \\ \infty & i = 0 \,\&\, j \neq 0 \\ DTW(i-1, j), & \\ DTW(i, j-1), & \\ DTW(i-1, j-1) & otherwise \end{cases} \tag{3.2.3}$$

The algorithm returns $DTW(|P|, |Q|)$ as final cost between P and Q, where $|P|$ and $|Q|$ represent the length of sequences P and Q, respectively. $\eta(p_i, q_j)$ is defined as Euclidean distance between points p_i and q_j.

3.2.2.2 Global Cost (GC) Computation

Given two disjoint trajectory sets R (Testing) and S (Training/Reference) such that $R = \{r^1, r^2 ... r^{|R|}\}$ and $S = \{s^1, s^2 ... s^{|S|}\}$ (trajectories may belong to k classes) a pair-wise cost matrix $\Delta = [\delta^{ij}]_{|R|\times|S|}$ is computed using DTW algorithm where δ^{ij} is the cost between full length trajectories r^i and s^j. Next, the cost matrix Δ is transformed into GC by taking mean of costs per class as expressed in Eq. 3.2.6

$$\delta^{ij} = dtw(r^i, s^j) \quad r^i \in R,\ s^j \in S \tag{3.2.4}$$

$$\Delta = [\delta^{ij}]_{|R|\times|S|} \tag{3.2.5}$$

$$GC(a,b) = \frac{\sum_{s^b \in \ class \ b}(\Delta^{a,b})}{\sum_{s^b \in \ class \ b}(1)} \tag{3.2.6}$$

3.2.3 Modeling Local Behavior

Local behavior is modeled using DP algorithm [103, 104], Complete Bipartite Graph (CBG), and Minimum Spanning Tree (MST). The details are as follows.

3.2.3.1 Douglas-Peucker (DP)

To model the local behavior of trajectories, in the next phase, each trajectory is segmented into sub-parts using DP algorithm. DP algorithm [102–104] or iterative end-point fit algorithm is the most commonly used line simplification algorithm coined by David Douglas and Thomas Peucker [102]. The algorithm selects a set of fewer points that generates equivalent curve. It starts by initializing output set with the two end points of the curve *i.e.* first point and the last point. Then the two end-points are joined by an imaginary straight line. Now the point that is the farthest (from this line) is considered. If the distance of the point from the line is less than a pre-specified threshold (ϵ) then only the first and last points are included in the output set, rest of the points are discarded. If not,

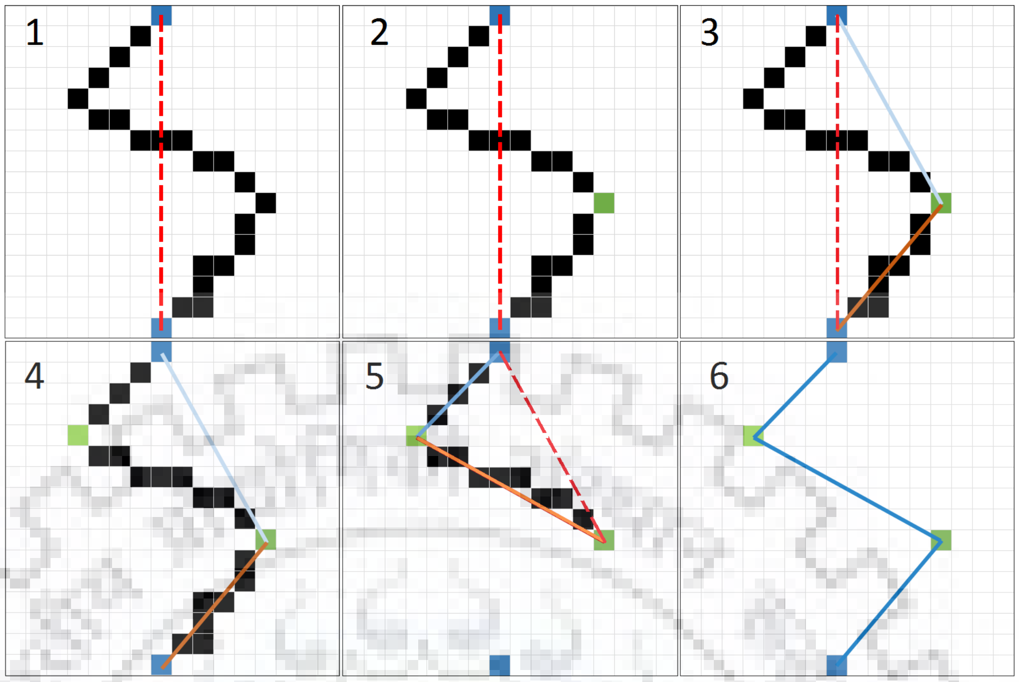


Figure 3.2: Example of DP algorithm. Trajectory is partitioning into segments using DP (1-6) . Final break points are shown in the last (6).

the farthest point is included in the output set and the same procedure is applied on the two parts *i.e* points between first point and the farthest point, and points between the farthest point and last point, recursively. Finally, we get a reduced set of points that represent the curve. Using the reduced curve each trajectory is split into segments. Figure 3.2 shows and example of DP algorithm with $\epsilon = 5$.

3.2.3.2 Complete Bipartite Graph (CBG)

A CBG [109, 110] is defined as $G = (\bar{V}, \bar{E})$ where $\bar{V}$ is the set of nodes and $\bar{E}$ is the set of edges in G. The set $\bar{V}$ is further defined as $\bar{V} = (U, V)$ with the constraints that $U \cup V = \bar{V}$ and $U \cap V = \phi$. Also, there is no edge within the sets U and V. However, there exists an edge e_{lm} between each pair of nodes $u_l \in U$ and $v_m \in V$.

3.2.3.3 Local Cost (LC) Computation

For each pair of trajectories, a CBG is defined by considering their segments from DP as nodes in it. An edge e_{lm} between nodes u_l and v_m is weighted using DTW distance between segments corresponding to nodes u_l and v_m (Eq. 3.2.8 where u_l belongs to one trajectory and v_m belongs to the other.)

$$w(e_{lm} = dtw(u_l, v_m) \tag{3.2.7}$$

$$W(\tau^i, \tau^j) = [w(e_{lm})] \tag{3.2.8}$$

where w represent the weight of the edge between nodes u and v and W is weight matrix of CBG formed by the segments of trajectories τ^i and τ^j. Next, the costs from local segments (nodes) for trajectories τ^i and τ^j are accumulated from CBG and an aggregated local cost ∇. ∇^{ij} is defined for each pair of trajectories as given in Eq. 3.2.11

$$min_{col} = \min_{c} W(:, c),\ \nabla^{col} = mean(min_{col}) \tag{3.2.9}$$

$$min_{row} = \min_{r} W(r, :),\ \nabla^{row} = mean(min_{row}) \tag{3.2.10}$$

$$\nabla^{ij} = \frac{\nabla^{col} + \nabla^{row}}{2} \tag{3.2.11}$$

Finally, LC is defined for each pair of trajectories $\tau^i \in R$ and $\tau^j \in S$ as given in Eq. 3.2.13

$$\nabla = [\nabla^{ij}]_{|R|\times|S|} \tag{3.2.12}$$

$$LC(a, b) = \frac{\sum_{s^b \in\ class\ b}(\nabla^{a,b})}{\sum_{s^b \in\ class\ b}(1)} \tag{3.2.13}$$

The CBG formulation is depicted in Figure 3.3. Figure 3.3(a) shows two segmented trajectories and corresponding CBG is shown in Figure 3.3(b). The aggregated cost matrix ∇ is calculated as shown in Figure 3.3(c) and Figure 3.3(d). The cost matrix ∇ is similar to cost matrix Δ as in Section 3.2.2.2 and has same size. Next, LC is computed by averaging costs per class as given in Eq. 3.2.13. *GC* represents the pair-wise global cost computed from full length trajectories and LC represents the pair-wise local cost computed using trajectory segments.

3.2.3.4 Minimum Spanning Tree (MST)

MST gives a optimal tree structure of a graph that traces the edges with the minimum possible total cost by covering all the nodes of the graph. Thus, trajectory classification can be benefited by considering the cost of MST structure. Typically, MST of a graph $G = (\bar{V}, \bar{E})$ is a connected

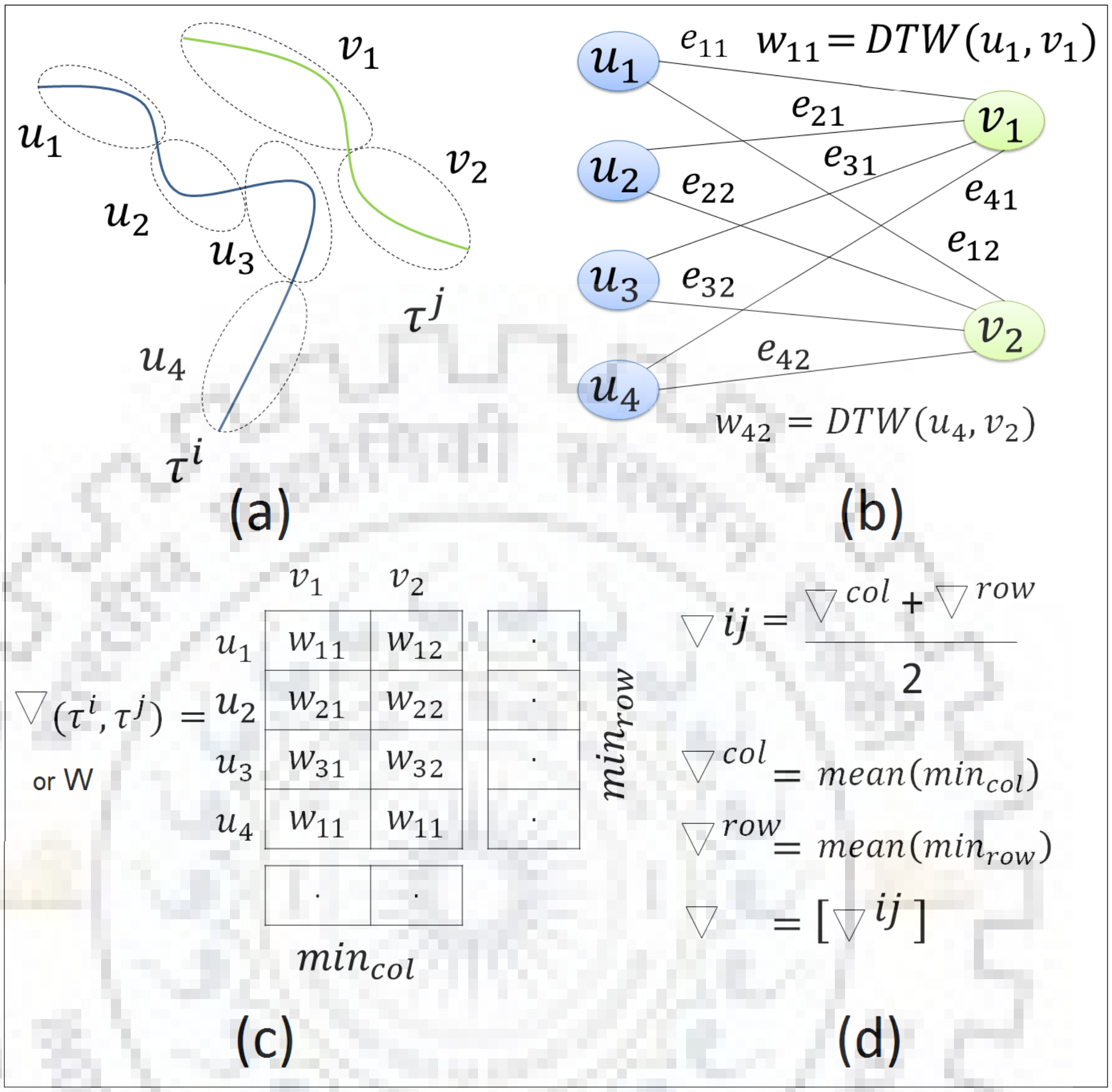

v_1
u_1
v_2
u_2
u_3
τ^j
u_4
τ^i
(a)
e_{11} $w_{11} = DTW(u_1, v_1)$
e_{21} e_{31} e_{22} e_{41} e_{12} e_{32} e_{42}
$w_{42} = DTW(u_4, v_2)$
(b)
$\nabla(\tau^i, \tau^j) =$ or W
w_{11} w_{12} w_{21} w_{22} w_{31} w_{32} w_{11} w_{11}
min_{row}
min_{col}
(c)
$\nabla ij = \frac{\nabla col + \nabla row}{2}$
$\nabla col = mean(min_{col})$
$\nabla row = mean(min_{row})$
$\nabla = [\nabla ij]$
(d)

Figure 3.3: CBG formulation and cost computation. (a) Two segmented trajectories, (b) Corresponding CBG, (c) Local cost within trajectory segments and (d) Aggregated cost (∇) computation.

tree with minimum possible cost. It is a tree structure $T = (\bar{V}, \bar{E}')$ contains all the nodes of G connected through edges such that $\bar{E}' \in \bar{E}$ and summation of weights of all the edges in T, $\sum w(\bar{E}')$ is minimum. MST can be estimated using Kruskal's or Prim's algorithm [106]. MST is estimated from CBG and the average of the total weight of edges in MST is computed and termed as MSTC.

GC is computed using full length trajectories for modeling global behavior of trajectories. LC and MSTC are calculated using DP and CBG for modeling local segment-wise trajectory behavior. Finally, the cost matrices *GC, LC*, and *MSTC* having identical size are fused using PSO.

3.2.4 Fusion of GC, LC, and MSTC using PSO

In this phase, fusion is performed using PSO. The details are discussed as follows.

3.2.4.1 Particle swarm optimization (PSO)

PSO [47, 105] is a biologically-inspired population-based metaheuristic. PSO simulates behavior of bird swarming for solving optimization problems. Consider a flock of birds, searching for food in a region, with food at only one position. The location of the food is not known, but the birds know how far they are from food and the position of every other bird in the flock. Now, the movement of a bird in the flock is influenced by the best location the bird has found so far and the best location anyone in its neighbor has found. PSO follows the same strategy to find optima of non-linear continuous function. PSO starts with a group of particles (known as *swarm*). The position of the particles are randomly initialized. Optimization function is evaluated at each particle's position (fitness value). Velocity of a particle is updated considering two positions: the position where the particle has found its best fitness value so far (personal best or *pBest*) and the location where the best fitness value is achieved by its neighbor (neighborhood best or *nBest*). When the whole swarm is considered as neighborhood then *nBest* is called global best or *gBest*. Let, $P_i,\ V_i,\ pBest_i,\ nBest_i$ be position, velocity, pBest position and nBest position of a particle, respectively in i^{th} iteration and V_{i-1} be particle's velocity at $(i-1)^{th}$ iteration. So, the velocity V_i in i^{th} iteration is given by Eq. 3.2.14

$$\begin{aligned} V_i = w \times V_{i-1} + c_1 \times r_1 \times (pBest_i - P_i) \\ +c_2 \times r_2 \times (nBest_i - P_i) \end{aligned} \tag{3.2.14}$$

w is a weighting factor. c_1 and c_2 are learning factors that influence contribution of $pBest$ and $nBest$ determining particle's new position, respectively. $r1$ and $r2$ are two independently generated random values. Particle's new position at $(i+1)^{th}$ iteration is updated by Eq. 3.2.15

$$P_{i+1} = P_i + V_i \tag{3.2.15}$$

3.2.4.2 Linear Fusion

The global cost *GC* and local cost *LC* is linearly fused using Eq. 3.2.16

$$C_{fusion} = \alpha \times GC + \beta \times LC + \gamma \times MSTC \tag{3.2.16}$$

$$where\ \alpha + \beta + \gamma = 10 \tag{3.2.17}$$

The parameters α, β, and γ are learned with the help of PSO algorithm by using classification error rate as objective function to optimize. The error rate is computed using fused cost matrix C_{fusion} as given in Eq. 3.2.18. The test trajectory is assigned class labels associated with the least cost in C_{fusion}. PSO over the iterations seeks to find minima of error rate by choosing the best values of α, β, and γ using swarm theory.

$$\%_{error} = 100 - \%_{classification} \tag{3.2.18}$$

Thus, the proposed framework can be summarized using algorithm 1.

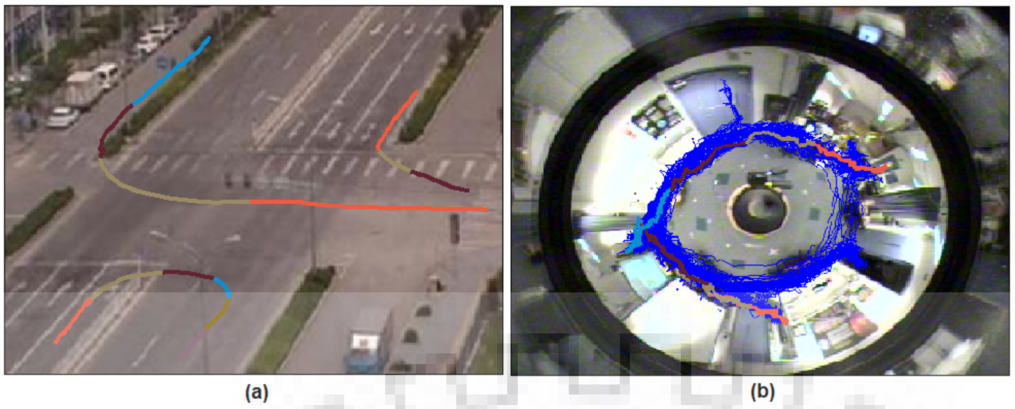
3.3 Experiments and Results

In this Section, we present the details of the experiments and the results using the proposed work.

3.3.1 Experimental Results

Here, we show the trajectory segmentation and classification results. We have used three public datasets, namely T15, LabOmni, and CROSS (details are available in the appendix A.1). DP algorithm has been used for partitioning of trajectories into local segments. Some examples of segmented trajectories from all the datasets are shown in Figure 3.4. The number of trajectory segments varies from 2 to 14 depending upon the curvature of trajectories. Here, samples shown are having 3 or more segments. The color changes when a new segment starts.

Experiments have been conducted by partitioning the data into train (50%), validation (25%),



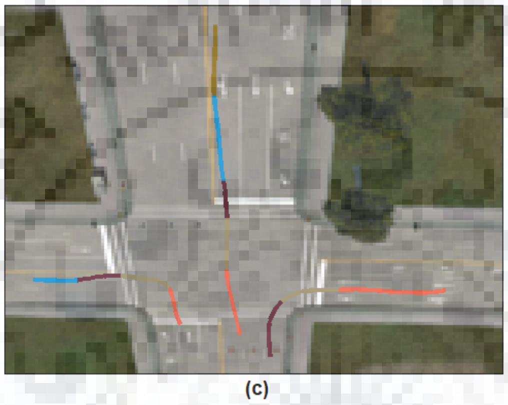


Figure 3.4: Examples of trajectory samples after segmentation. (a) T15 (b) LabOmni (c) CROSS. Segments are plotted with different colors. For better visibility of colors please see the pdf version.

and test (25%) sets. The cost *GC* has been computed using full length trajectories, whereas *LC*, and *MSTC* have been computed using segmented trajectories. The costs *GC, LC,* and *MSTC* have been fused using PSO. Fusion weights learned from validation data have been applied to test set to compute final classification results. Figure 3.5 shows the classification results using T15, LabOmni, and CROSS datasets. Accuracies of 90.23 %, 91.67 %, and 98.74 % have been recorded when tested using full length trajectories, whereas accuracies of LC(MSTC) have been recorded as 90.79(78.32) %, 71.67(71.67) %, and 97.89(95.58) % on T15, LabOmni, and CROSS datasets, respectively. The proposed method improves the performance with the classification rates of 92.95%, 95.38%, and 99.58% using T15, LabOmni, and CROSS datasets, respectively. We have noticed that different values of α, β, and γ have been recorded for different datasets. The values of $\alpha = 2.7462$, $\beta = 6.9638$, and $\gamma = 0.2900$ have been recorded on T15 dataset when tested using proposed method, whereas the proposed method come up with the values of $\alpha = 6.7267$, $\beta = 2.1043$, and $\gamma = 1.1690$ on LabOmni dataset. Values of α, β, and γ have been recorded as 1.2903, 8.4987, and 0.2110, respectively when the proposed method was tested using CROSS dataset.

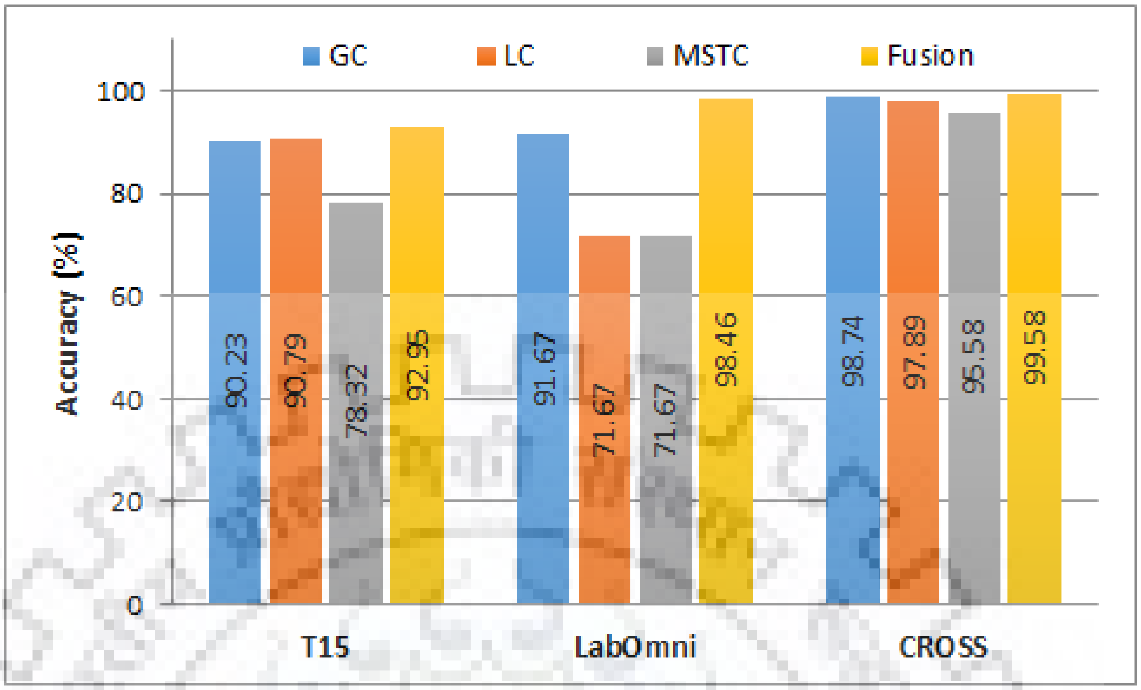


Figure 3.5: Experimental results using proposed method. Fusion of *GC, LC,* and *MSTC* improves the classification results for the datasets used in this study namely, T15, LabOmni, and CROSS.

3.3.1.1 Result Analysis

Here, we analyze the classification results obtained by the proposed work.

We notice from the results of the proposed method that there is an improvement of 2.72%, 6.79%, and 0.84% using T15, LabOmni, and CROSS datasets, respectively when compared the trajectory classification results using full length trajectories. Some of the samples classified incorrectly using full length trajectories are shown in the second column of Figure 3.6, whereas these samples are correctly classified by the proposed method (shown in column 3).

First row of Figure 3.6 shows few such samples of the T15 dataset, where *(a1)* is the trajectories with ground truth (color represents ground truth classes) *(b1)* is the trajectories with predicted classes (color represents predicted classes) using full length trajectories and *(c1)* is the prediction by proposed method. It can be noticed from *(a1)* and *(c1)* that proposed method recognizes the correct classes of trajectories while they are incorrectly classified to the other classes when full length trajectories are used (shown in *(b1)*). Second and third rows depict the similar results for LabOmni, and CROSS datasets, respectively. Here, column 1 is showing few samples (color is depicting the ground truth classes), column 2 is showing the incorrect classification of trajectories

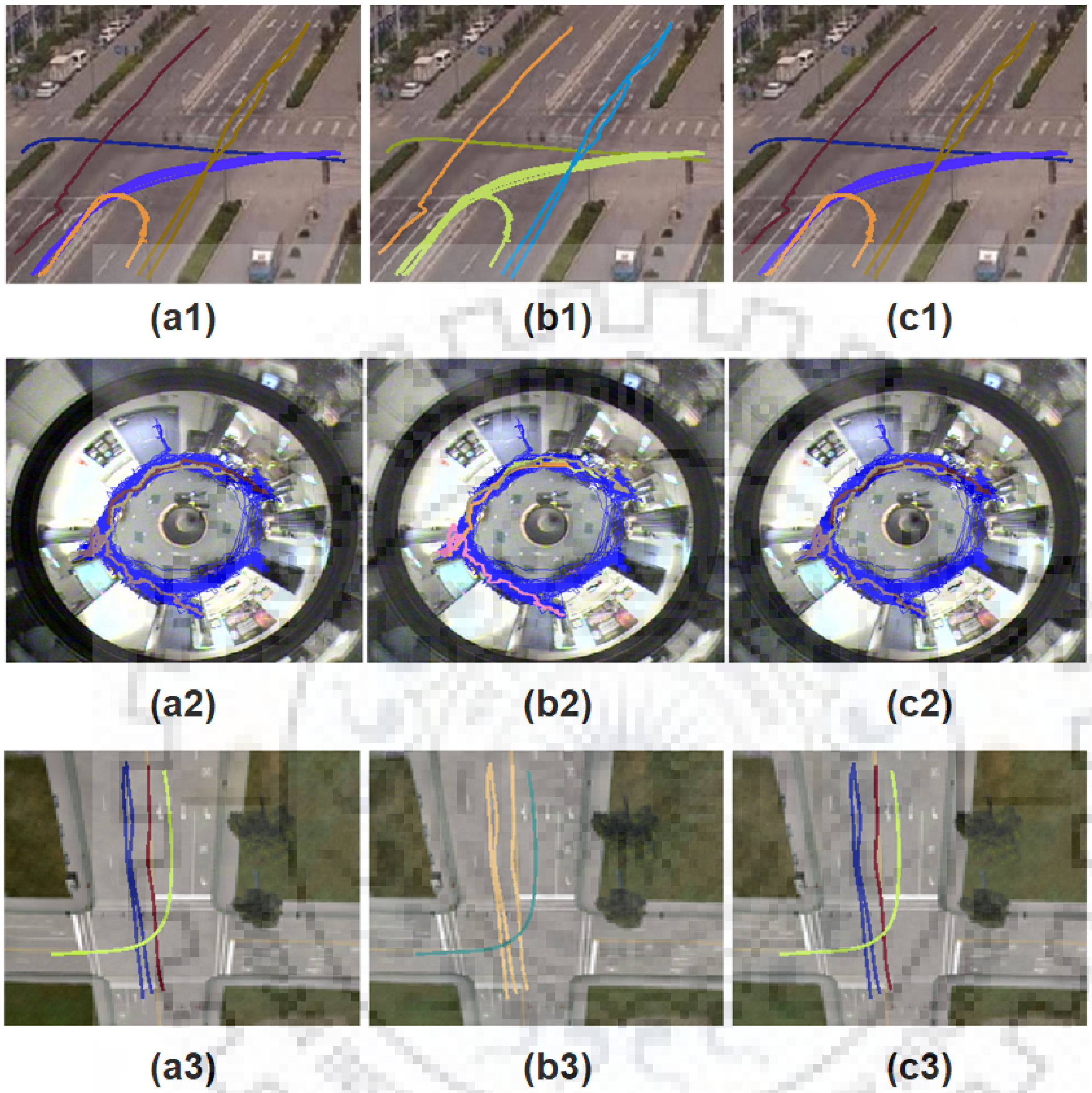


Figure 3.6: Qualitative results of the proposed method. (a1), (a2), & (a3) Trajectories with ground truth (b1), (b2), & (b3) Incorrectly classified using full length trajectories (c1), (c2), & (c3) Corrected by proposed method. T15 (row 1) LabOmni (row 2) CROSS (row 3).

when full length trajectories are used, and column 3 is showing their correct classification result using the proposed method for T15, LabOmni, and CROSS datasets, respectively.

Though the proposed method improves the performance by correctly identifying the true classes, we have noticed that some of the trajectories have not been correctly classified into respective classes. Figure 3.7 (column 3) shows few such trajectories that have been incorrectly classified into other classes. Here, column 1 show the ground truth trajectories in color (color depicting ground

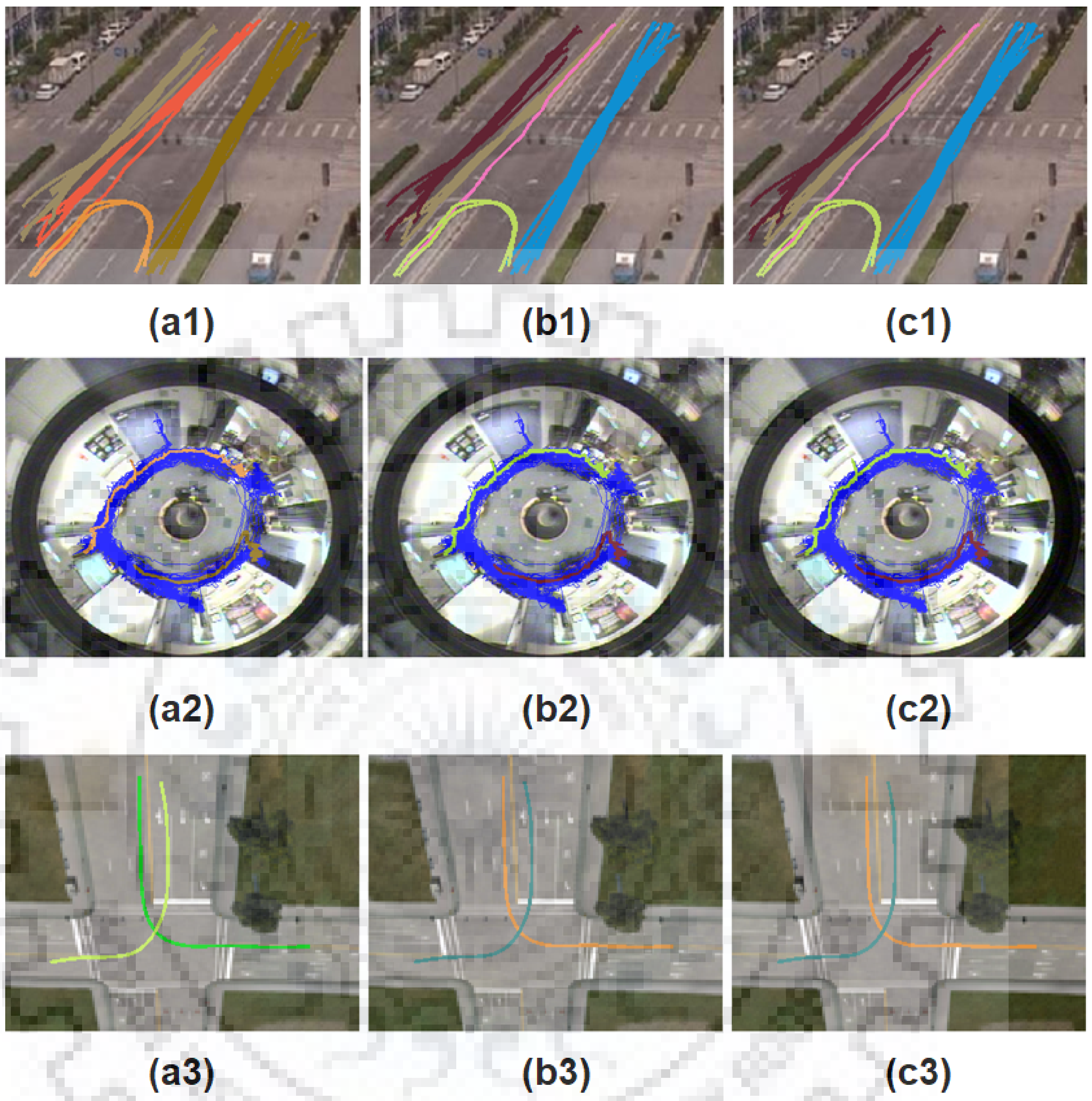


Figure 3.7: Erroneous Trajectories. (a1), (a2), & (a3) Trajectories with ground truth (b1), (b2), & (b3) Incorrectly classified using full length trajectories (c1), (c2), & (c3) Incorrectly classified by proposed method. T15 (row 1) LabOmni (row 2) CROSS (row 3).

truth), column 2 & 3 show the prediction by full length trajectories and by the proposed method, respectively. It can be noticed from column 2 & 3 that the trajectories have not been correctly classified into respective classes by both the methods i.e. by full length trajectories and by proposed method. Such incorrect results are due to the similar shape of trajectories and the high overlapping between their ground truth and predicted class.

We noticed that (Figure 3.5) the accuracy of *LC* and *MSTC* in case of LabOmni dataset are much

Table 3.1: Computation time elapsed at different stages of proposed method in seconds (Sec)

Dataset	GC (Sec)	DP (Sec)	CBG+LC (Sec)	MSTC (Sec)	PSO (Sec)	Total (Sec)
T15	644.92	12.76	98.71	1765.54	617.74	3139.67
LabOmni	44.21	10.10	94.12	1123.72	149.25	1421.40
CROSS	760.61	18.56	127.16	2123.29	710.18	3739.80

lesser as compared to *GC*. This is due to very high deviation in the number of trajectories belong to different classes of LabOmni dataset. Some to the classes only have 3-4 trajectories, whereas other classes have more than 8 trajectories. However, the results of the proposed fusion is promising for all the dataset used in this study.

3.3.1.2 Time Computation and Comparative Study

In this Section, we present the average running time per fold consumed in proposed system and comparison to the other methods. The proposed work has been implemented using computer system with 3^{rd} generation Intel core $i3$ processor and 4 GB of RAM. The time consumption at different stages of the proposed work is shown in Table 3.1. The average running time in the computation of *GC* using DTW distances using full length trajectories have been recorded as 644.92, 44.21, and 760.61 seconds for T15, LabOmni, and CROSS datasets, respectively. The trajectory segmentation using DP algorithm has consumed 12.76 seconds for T15, 10.1 seconds for LabOmni, and 18.56 seconds for CROSS datasets. The minimum and maximum number of segments using DP algorithm has been recorded as 2 and 14, respectively. The total running times including GC, DP, CBG, LC, MSTC, and PSO optimization have been recorded as 3139.67, 1421.4, and 3739.8 seconds for T15, LabOmni, and CROSS datasets, respectively. We noticed that proposed system takes around 1 second to assign a class label for a test trajectory which can be optimize with the systems with higher processing capabilities. Thus, the proposed framework could be used in modeling trajectories for better classification performance in appreciable time.

The trajectory learning frameworks for clustering/classification have been proposed in [41,112]. Authors of [112] have compared various distance measures such as DTW, and LCSS for trajectory clustering. The accuracies as high as 83.83% and 90.91% have been recorded using DTW and LCSS, respectively. Authors of [41] have proposed a shrinking framework for trajectory clustering and classification using multi-kernels. The framework came up with the accuracies of 84.45% using

HMM and 87.60% using k-means algorithm. Table 3.2 shows the comparison with other methods. The proposed method has better performance as compared to the methods mentioned in the Table.

Table 3.2: Classification accuracy comparison with other methods

T15 Dataset	
Xu et al. [41]	87.60%
Proposed Work	**92.05%**
LabOmni Dataset	
Morris and Trivedi [113]	93.84%
Proposed Work	**98.46%**
CROSS Dataset	
Morris and Trivedi [113]	98.52%
Proposed Work	**99.58%**

The performance of proposed method has also been evaluated and compared with Ant Colony Optimization (ACO) [114] and Genetic Algorithm (GA) [115] for all the datasets. ACO is an optimization method that searches for an optimal path by mimicking the behavior of ants while ants searching for food. Ants release a certain amount of pheromones as they explore. Gradually, the pheromone concentration on the shortest path becomes higher and attracts more ants, thus forming a positive feedback which leads to the discovery of the best path towards food in our case the best possible values of α, β, and γ. Whereas, GA is the evolutionary algorithm that is inspired from Darvin's theory of survival of fittest. It involves the crossover and mutation of genes and reproduction of new offsprings that are the possible solution to the search. Both ACO and GA iteratively search for the best possible solution and stop when they meet the convergence criteria. Figure 3.8 shows the comparative results of them. GA and PSO has performed better as compared to ACO. However, results using PSO are promising for all the datasets.

3.4 Discussion

In this work, we have used complete bipartite graph to incorporate local behavior of trajectory using DTW along with the global dynamics. The global dynamics has been captured using full length trajectories, whereas DP algorithm has been used for segmentation before incorporating local dynamics. Global (GC) and local (LC) costs are computed using pair-wise DTW matching and MSTC using MST of CBG. Finally, *GC, LC,* and *MSTC* have been fused linearly using PSO. The PSO optimization improves the classification rates of trajectory datasets used, namely T15, LabOmni, and

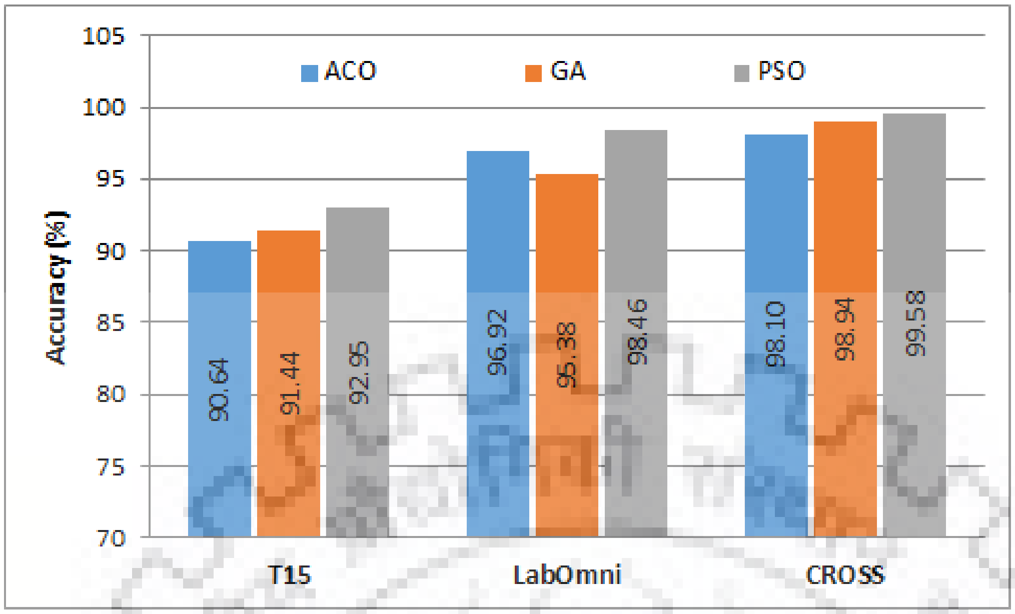


Figure 3.8: Comparison of results of proposed method using ACO, GA, and PSO.

CROSS in this work. The proposed work shows accuracies as high as 92.95%, 98.46%, and 99.58% using datasets T15, LabOmni, and CROSS, respectively. This Chapter emphasize that the graph based methods have capability to improve the classification performance in trajectory data and the same idea could be extended further. In future, the proposed work with robust features could be used to improve the trajectory classification performance. In future, we will extend this work on sign gestures and signature based recognition.

Chapter 4

A Segmental HMM based Trajectory Classification using Genetic Algorithm

Trajectory classification techniques face various challenges due to varying length and lack of the presence of clear boundaries among the trajectory classes. To overcome such challenges, a trajectory shrinking framework using Adaptive Multi-Kernel based Shrinkage (AMKS) can be used. However, such a strategy often results in over-shrinking of trajectories leading to poor classification. To improve classification performance, we introduce two additional kernels that are based on convex hull and Ramer-Douglas-Peucker (RDP) algorithm. Next, we present a supervised trajectory classification approach using a combination of global and Segmental Hidden Markov Model (HMM) based classifiers. In the first stage, HMM is used globally for classification of trajectory to provide state-wise distribution of trajectory segments. In the second stage, state-wise trajectory segments are classified and combined with global recognition performance to improve the classification results. Combination of Global HMM and Segmental HMM is performed using a genetic algorithm (GA) based framework in the final stage.

4.1 Introduction

In static camera based video surveillance systems, the required actions can be taken on the basis of the movements occurred in the camera's field of view. For this purpose, trajectories are first extracted [116] from the captured video which may contain various types of noises due to the limitations of the underlying tracker. Moreover, the tracking results may incur confusion because of the speed and appearance of the objects. Therefore, tracking of nearby objects may be difficult. Though a number of solutions [42, 43] exist, presence of noise in the trajectory sequences may not be removed completely. This essentially leads to wrong classification. It happens because, most of the existing algorithms [42, 43, 72, 117] do not focus adequately on the classification approaches.

In [41], authors have proposed shrinking of trajectories to improve the trajectory classification performance. They have shown the classification results using Hidden Markov Model (HMM) as well as of clustering using k-means algorithm. Trajectory shrinking [41] makes the classification task easier and it can successfully group similar trajectories. Shrinking similar trajectories into one class may help in understanding the activities or actions taken by moving objects such as humans or vehicles. A portion of trajectory may be spatially close to a trajectory of different class. Moreover, the start and end segments of a trajectory may significantly vary from the rest part which may create confusion with other trajectories during the classification. Thus, inter-class boundaries may not be prominent.

To minimize such errors in shrinking, we have introduced two popular kernels that are based on convex hull (CH) [118] and Ramer-Douglas-Peucker (RDP) algorithm [102]. In our framework, trajectories are first modified with the proposed multi-kernel framework as given in Eq. 4.2.4. Next the modified trajectories are trained using a two-stage HMM classification framework using Global HMM (GHMM) and Segmental HMM (SHMM). In the first stage, HMM is used globally for classification of trajectory to provide state-wise distribution of trajectory segments. In the second stage, state-wise trajectory segments are classified and combined with global recognition performance to improve the classification results. Later, these classification results from GHMM and SHMM are combined using GA. GA optimizes the combination weightage and produces the final classification results. Thus, key idea of proposed work is the combination of classification from both i.e. from full length trajectories and from their segments where combination is performed using GA.

4.2 Proposed Framework on Trajectory Classification

In this Section, we present our proposed framework of trajectory classification. Our framework comprises of trajectory shrinking, independent classification using conventional HMM and SHMM, and finally a GA guided method for combination of GHMM and SHMM results. The flow-chart of

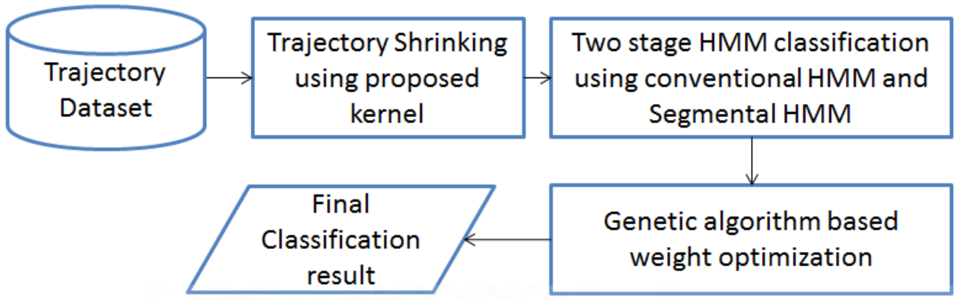

Trajectory Dataset
Trajectory Shrinking using proposed kernel
Two stage HMM classification using conventional HMM and Segmental HMM
Genetic algorithm based weight optimization
Final Classification result

Figure 4.1: A flow-chart of the framework used in the proposed trajectory classification.

the proposed framework is depicted in Figure 4.1. Given a set of trajectories, proposed shrinking model produces compact trajectories that are referred to as shrunken trajectories. These trajectories are then fed to a two-stage HMM classifier to find global classification and perform state-wise trajectory separation. In the next step, a linear combination is obtained using GA to improve the accuracy. Each of these steps are discussed in following sub-sections.

4.2.1 Kernel Formation

A trajectory may be defined as a spatio-temporal sequence representing an object's instantaneous position. This spatio-temporal sequence is represented by coordinates or points in d-dimensional Euclidean space (R^d). Let $T = \{T_1, T_2, ...T_N\}$ be the set of N such trajectories, where each trajectory is represented by $T_i = \{p_{i,1}, p_{i,2}, ...p_{i,M}\}$, such that $p_{i,t}$ is the position of i^{th} object at time t.

The proposed kernel is inspired from AMKS kernel proposed by Xu et al. [41] due to its good performance in various trajectory datasets. AMKS kernel is briefly explained below. Here, we introduce two new kernels using CH and RDP algorithm [102] respectively and combine these kernels with AMKS kernel to improve the performance.

4.2.1.1 AMKS Kernel

The AMKS kernel proposed by Xu et al. [41] involves the computation of new position as discussed hereafter. For each point $p_{i,t} \in T_i$, neighboring points $\{p_k\}_{k=1}^{K}$ are first searched within the neighborhood. These neighbors can be part of other trajectories. The new estimates of position $p_{i,t}$ and speed $v_{i,t}$ can be defined as given respectively in Eq. 4.2.1 and Eq. 4.2.2

$$\hat{p}_{i,t} = \frac{\sum_{k=1}^{K} f(p_{i,t}, p_k) p_k}{\sum_{k=1}^{K} f(p_{i,t}, p_k)} \tag{4.2.1}$$

$$\hat{v}_{i,t} = \frac{\sum_{k=1}^{K} f(p_{i,t}, p_k) p_k}{\sum_{k=1}^{K} f(p_{i,t}, v_k)} \tag{4.2.2}$$

where $v_{i,t} = (p_{i,t+1} - p_{i,t})/\delta t$ and δt is the time step.

Let $p_m \in T_i$ and $p'_m \in T_j$, then composite kernel $f(:,:)$ is defined as given in Eq. 4.2.3

$$f_{AMKS}(p_m, p'_m) = f_p(p_m, p'_m)\ f_s(p_m, p'_m)\ f_a(p_m, p'_m) \tag{4.2.3}$$

where $f_p(p_m, p'_m) = e^{(-\|p_m - p'_m\|_2^2/(2\alpha^2))}$ is the position kernel which finds the consistency in location. The points with similar positions are close to each others in the kernel space.

$f_s(p_m, p'_m) = e^{(-\|v_m - v'_m\|_2^2/(2\beta^2))}$ is the speed kernel that finds the consistency in speed. The points with similar speed are close in the kernel space.

$f_a(p_m, p'_m) = e^{(-\|a_i - a_j\|_2^2/(2\gamma^2))}$ is the kernel for aim points (i.e. end points of trajectories) which finds the consistency in aim, where $a_i = \{p_{i,1}, p_{i,M}\}$.

Next, shrunken trajectory estimates are further optimized to generate final positions that preserve the shape of the trajectories using a speed-regularization scheme [41].

4.2.1.2 Proposed Kernels

AMKS [41] finds the new shrunk positions of trajectories. We extend AMKS algorithm by adding two new kernels using CH [118] and RDP [102]. It has been observed that, majority of regular trajectories belonging to one class have similar curvature or shape. This is quite useful in distinguishing the trajectories of different classes. Therefore, we select the key points from each trajectory that derive the curvature. The task is accomplished using CH and RDP algorithm. Both algorithms provide important key points on the trajectories. The nearby trajectories from different classes might create problem in classification. However, the points from CH/RDP are likely to pertain the shape of trajectories that could be distinct in this case. Thus, it could help to improve the system. It may be verified from Figure 4.2 that CH and RDP come up with different set of points for a given trajectory. Therefore, both algorithms have been used in kernel formation in this framework. Hence, the proposed multi-kernel consists of five sub kernel as given in Eq. 4.2.4.

Convex Hull (CH) and Ramer-Douglas-Peucker Algorithm (RDP): CH of a given set of points S in euclidean space can be defined as a minimal convex set. A convex set is the region such that any straight line segment is fully contained within the region. CH is the smallest area convex polygon that encloses all the points from S. CH has been used successfully in solving problems such as character recognition [119] and roman numeral recognition [120]. Figure 4.2 shows a toy example of CH, where the closed polygon in red color is a CH of all the points lying on green path.

RDP [102] tries to approximate a curve with fewer points. It keeps first and last point of the curve and finds the farthest point (say, $P_{farthest}$) from the line joining first and last points. If the distance between the farthest point and the line segment is greater than a threshold ϵ then $P_{farthest}$ is selected for processing and the search is then split in two parts, i.e. from first point to $P_{farthest}$ and from $P_{farthest}$ to the last point. The process is carried out on both the parts separately. The process is continued over the iterations and stops only when the end points of every segment have no point(s) having distance greater than ϵ. The algorithms results in a similar curve with lesser number of points. The blue line path in Figure 4.2 represents the RDP walk.

Note that, points given by RDP algorithm [102] may be different from convex-hull points, i.e. points from RDP algorithm may not include all the hull points as shown in Figure 4.2. Hence, we consider them separately in two different kernels. New modified kernel is given in Eq. 4.2.4

$$f(p_m, p'_m) = f_{AMKS}(p_m, p'_m)\ f_{hull}(p_m, p'_m)\ f_{RDP}(p_m, p'_m) \tag{4.2.4}$$

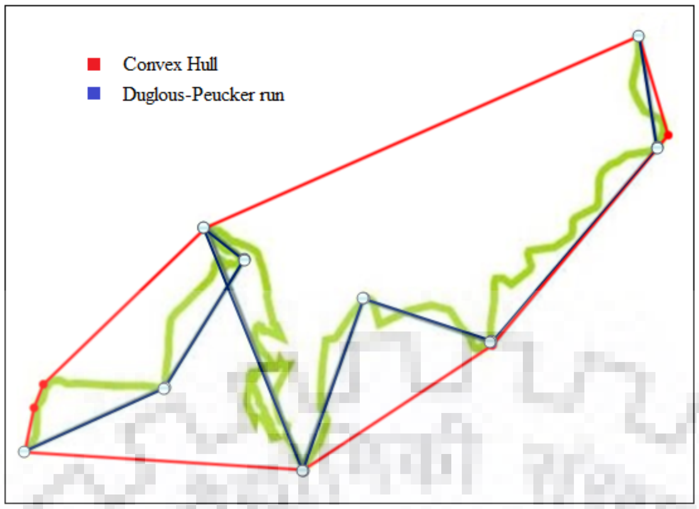


Figure 4.2: Convex Hull and RDP sketch: CH in red and RDP in blue.

$f_{hull}(p_m, p'_m) = e^{(-\|hull_i - hull_j\|_2^2/(2\theta^2))}$ is the CH kernel that are consistent with the hull points. $hull_i$ represents the CH points for the i^{th} trajectory.

$f_{RDP}(p_m, p'_m) = e^{(-\|RDP_i - RDP_j\|_2^2/(2\delta^2))}$ is the kernel that are consistent with the RDP points. RDP_i represents the key points provided by RDP algorithm for the i^{th} trajectory.

The parameters $r, \alpha, \beta, \gamma, \theta$ and δ are configured as given in [41]. With the above mentioned kernels, the trajectories are shrunk. Next, these shrunk trajectories are classified using HMM as discussed in next Section.

4.2.2 Trajectory Classification using GHMM and SHMM

HMM [121] is a supervised classifier that has been successfully applied to model time series sequences [21, 54, 70]. In this study, we propose a two stage trajectory classification using HMM as discussed in the following Section.

During GHMM based classification stage, if the difference between the top two classification scores of a trajectory is high, it is considered as a true class. However, if the difference is low between top two probability scores (Δp), we consider such trajectories for SHMM-based classification. Lower value of the probability difference is mainly due to high confusion between the classes. For such sequences, a separate SHMM based classification strategy is used that is based on the segmentation of sequences. It is to identify the state-wise segments in trajectory and improve

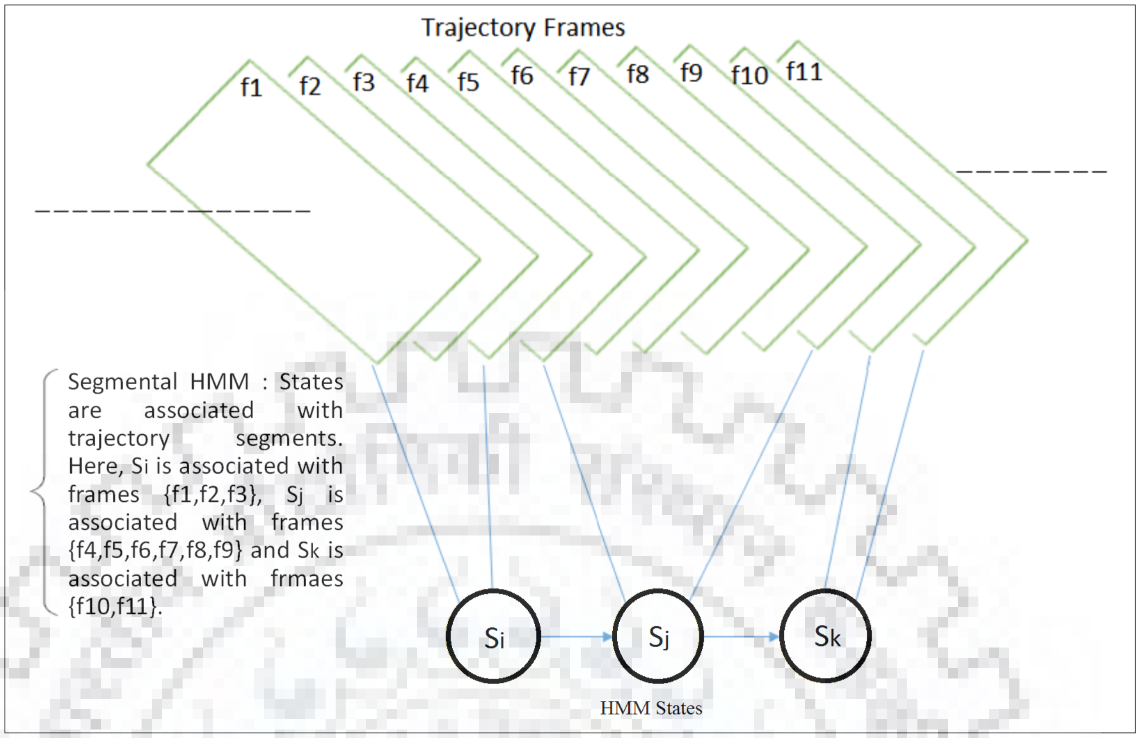

Trajectory Frames
f1
f2
f3
f4
f5
f6
f7
f8
f9
f10
f11
Segmental HMM : States are associated with trajectory segments. Here, Si is associated with frames {f1,f2,f3}, Sj is associated with frames {f4,f5,f6,f7,f8,f9} and Sk is associated with frmaes {f10,f11}.
Si
Sj
Sk
HMM States

Figure 4.3: SHMM: Sequence of frames are associated with HMM states.

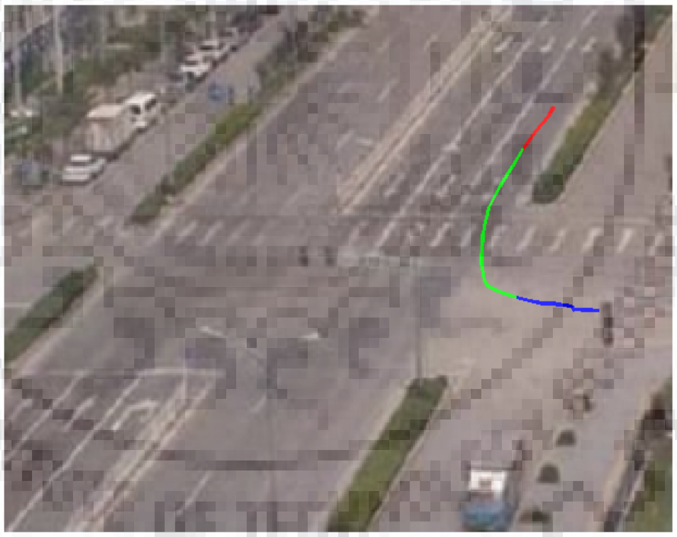


Figure 4.4: Example of separation of a trajectory into 3 segments by SHMM (Note that the length of the segments are not equal). Best viewed in color.

the overall classification using these trajectory-segment classifications. For this purpose, we have considered SHMM [122–124] based approach for trajectory classification. SHMM differs from conventional HMM. The conventional HMM associates the state with observation feature vector, whereas SHMM associates the state with trajectory segments that belong to corresponding sequence of frames as shown in Figure 4.3. The length of the trajectory segments for each states (say S_i) may differ. An example of segmented trajectory is shown in Figure 4.4.

4.2.2.1 Combining GHMM and SHMM using GA

Now, the prediction scores obtained from GHMM and SHMM are linearly combined in an efficient way. To optimize the weighted linear combination (Eq. 4.2.6) of GHMM and SHMM, we use GA. GA has been used to put efforts to correctly classify the trajectories that have fallen into incorrect class in top-1 choices however, their correct class is in top-2 choices. The weighted linear combination using GA could help to accomplish this task.

$$\hat{C_i} = (w_0 * C_i^{Global} + \sum_{j=1}^{h} w_j * C_i^{Sj}), \;\; given \tag{4.2.5}$$

$$\sum_{j=0}^{h} w_j = 1 \tag{4.2.6}$$

In Eq. 4.2.6, $\hat{C_i}$ is a row vector that consists of weighted score for testing sequence i for all test classes. Finally, the class with highest weighted score is selected as final label as given in Eq. 4.2.7

$$\hat{\omega}_i = arg\; max(\hat{C_i}) \tag{4.2.7}$$

After classification of each segment, a weighted scheme is used to combine the global classification in the first stage and local classification results in the second stage. On the basis of this weighted score, a final label is obtained. Weights are assigned using GA based optimization technique. The process is explained in Figure 4.6.

4.2.2.2 Genetic algorithm (GA)

GA [125] is a method for solving both constrained and unconstrained optimization problems based on a natural selection process that mimics biological evolution. The algorithm repeatedly modifies a population of individual solutions. At each step, the algorithm randomly selects individuals from the current population and uses them as parents to produce the children for the next generation. Over successive generations, the population evolves toward an optimal solution. The process of GA starts

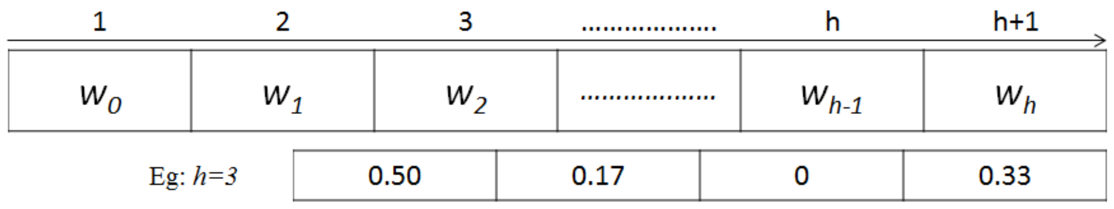


Figure 4.5: Example of chromosome encoding. Here, each chromosome is having length $(h+1)$.

with an initial population. The size of initial population may vary depending to the optimization problem, however, it is usually taken randomly as hundreds or thousands. Then comes selection of the proportion of population for evaluation of objective function. During each successive generation, a proportion of the existing population is selected to breed a new generation. Individual solutions are selected through a fitness criteria (fitness function). The next step is to generate a second generation population of solutions from those selected through a combination of genetic operators: crossover, and mutation. Crossover refers to the process of combining pair of parents to generate new population. Mutation generates children through random change in single parent. New generation is again fed into selection step and the process continues until the termination criteria is fulfilled. The termination condition may be based on the number of generation, time limit, tolerance threshold, etc.

As in stage two, each trajectory is partitioned into h segments and it implies the linear combination of classification of GHMM and SHMM as shown in Eq. 4.2.6 which requires $(h+1)$ variables or weights. Hence, each chromosome in GA is having length $(h+1)$. An example of chromosome encoding is shown in Figure 4.5.

After encoding chromosomes, we then generate initial population of chromosomes randomly. Over iterations, GA optimizes the weights w_i $(i = 0, 1, 2....h)$ using Eq. 4.2.6. Finally, trajectories are labeled into its true class according to the weights obtained from the algorithm.

4.3 Experimental Result and Analysis

In this Section, we present the experiment results obtained using our framework on publicly available trajectory datasets. In our system, trajectories have been extracted from videos using an existing target detection and tracking algorithm proposed by [116]. The trajectories have been classified

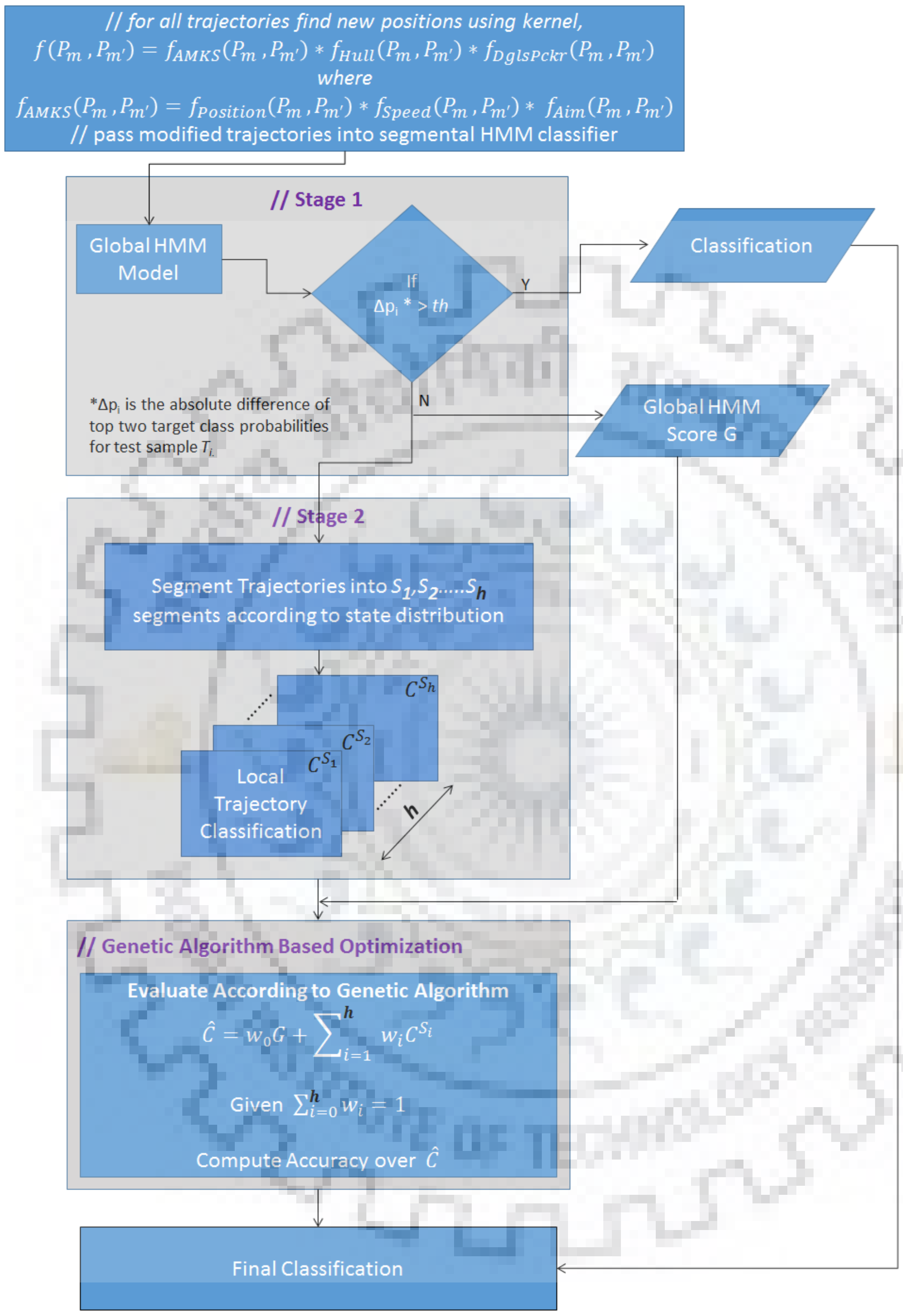

// for all trajectories find new positions using kernel,
$f(P_m, P_{m'}) = f_{AMKS}(P_m, P_{m'}) * f_{Hull}(P_m, P_{m'}) * f_{DglsPckr}(P_m, P_{m'})$
where
$f_{AMKS}(P_m, P_{m'}) = f_{Position}(P_m, P_{m'}) * f_{Speed}(P_m, P_{m'}) * f_{Aim}(P_m, P_{m'})$
// pass modified trajectories into segmental HMM classifier
// Stage 1
Global HMM Model
If Δp_i * > th
Y
N
Classification
Global HMM Score G
*Δp_i is the absolute difference of top two target class probabilities for test sample T_i
// Stage 2
Segment Trajectories into $S_1, S_2 \dots S_h$ segments according to state distribution
C^{S_h}
C^{S_2}
C^{S_1}
Local Trajectory Classification
h
// Genetic Algorithm Based Optimization
Evaluate According to Genetic Algorithm
$\hat{C} = w_0 G + \sum_{i=1}^{h} w_i C^{S_i}$
Given $\sum_{i=0}^{h} w_i = 1$
Compute Accuracy over $\hat{C}$
Final Classification

Figure 4.6: Proposed framework comprises trajectory shrinking, two stage HMM classification and optimization using GA.

Table 4.1: Performance of individual kernels

Kernel	T15 (%)	MIT (%)
AMKS [41]	84.40	94.25
AMKS + CH	84.40	94.55
AMKS + RDP	89.45	94.60
AMKS + CH + RDP	92.80	94.75

using our framework. The results are compared with Xu et al. [41]'s work and other existing approaches.

To test the robustness of our framework, we have tested two publicly available surveillance datasets namely T15 and MIT car. The details are available in the appendix A.1.

4.3.1 Classification Improvement using the Proposed Kernel

We present experimental results on T15 and MIT datasets. We show some qualitative results using proposed framework and compare AMKS kernels. The classification results are shown on shrunk datasets plotted over respective surveillance scene.

AMKS [41] kernel shrunk the nearby trajectories together, though they belong to different classes which mainly depend on position, speed and aim points, whereas the proposed kernel includes key points from CH and RDP algorithm that minimizes over shrinking. Table 4.1 shows the performance of individual kernels on trajectory classification. Classification rate is highest when all kernels are used together.

Figure 4.7 and Figure 4.8 depict a few wrongly classified trajectories of T15 and MIT datasets, respectively. In Figure 4.7 and Figure 4.8, column 1 shows T15 and MIT Car datasets, respectively. Here, trajectories are shown in colors to distinguish the classes and the trajectories with identical color belong to the same class. Column 2 of Figure 4.7 and Figure 4.8 show two different classes of T15 and MIT datasets, respectively. A single trajectory from corresponding the class is shown in column 3. Column 4 of both the Figures show the trajectories modified by AMKS [41] kernel that are incorrectly classified, whereas column 5 shows the modified trajectories using the new kernel.

The quantitative results in these datasets are shown in Table 4.2. Note that, the proposed multi-kernel outperforms the AMKS approach. AMKS framework has an overall accuracy of 87.60%

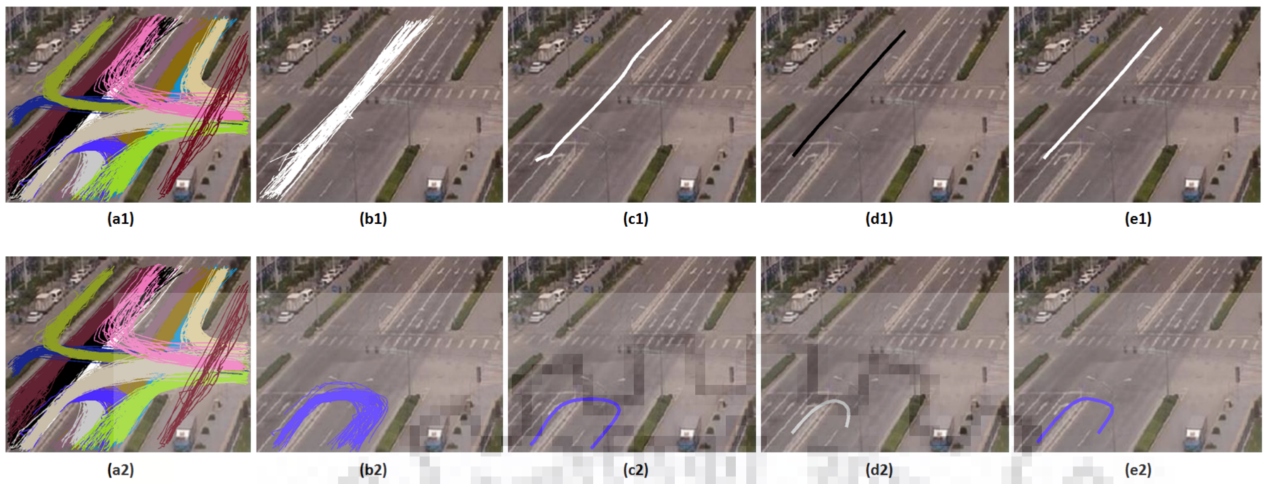


Figure 4.7: Classification Result on T15: First column (a1 & a2) shows the dataset where different colors represent different classes. Column 2 (b1 & b2) shows two different classes and column 3 (c1 & c2) shows a single raw trajectory from the corresponding class shown in column 2. Column 4 (d1 & d2) and column 5 (e1 & e2) show the new trajectories (corresponding to trajectories in column 3) with their predicted class formed by AMKS and the proposed kernel, respectively. Proposed framework correctly classifies these trajectories (column 5) as compared to AMKS (column 4).

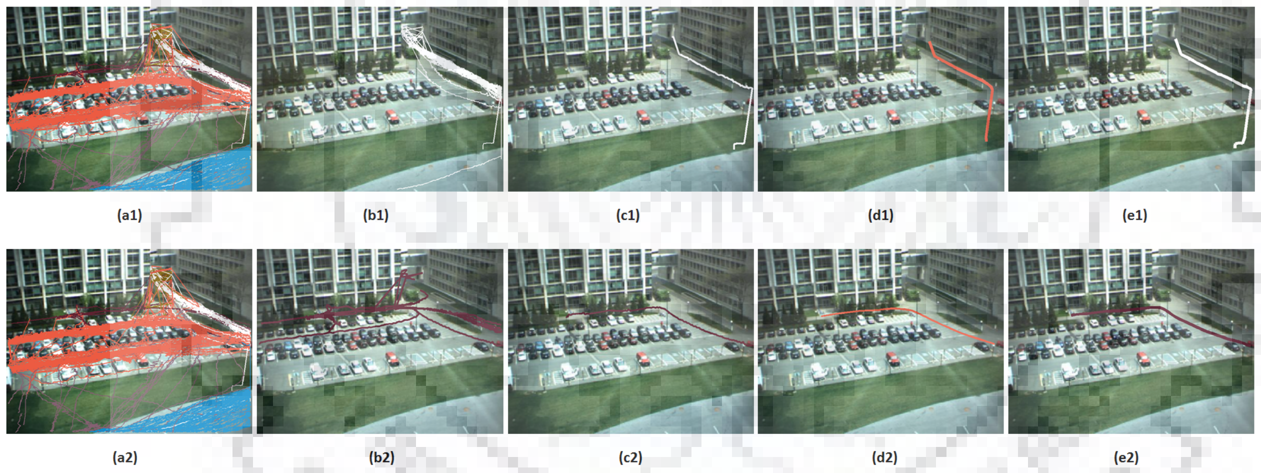


Figure 4.8: Classification Result on MIT Car: First column (a1 & a2) shows the dataset where different colors represent different classes. Column 2 (b1 & b2) shows two different classes and column 3 (c1 & c2) shows a single trajectory from the corresponding class depicted in column 2. Column 4 (d1 & d2) and column 5 (e1 & e2) show the new trajectories (corresponding to trajectories in column 3) with their predicted classes formed by AMKS and the proposed kernel, respectively. Proposed framework correctly classifies these trajectories (column 5) as compared to AMKS (column 4).

using k-means algorithm and 84.40% using HMM on T15 and 94.25% on MIT dataset using traditional HMM-based classification. Our proposed method achieves accuracy of 94.80% and 96.75% on T15 and MIT datasets, respectively. Here, in GA based optimization, population was taken in double vector encoding and roulette wheel was used for selection. The crossover and mutation probability was set to 0.8 and 0.01, respectively and the objective function tolerance (difference between

Table 4.2: Classification Result: First row shows the results using T15 dataset and second row shows the results for MIT dataset.

Dataset	**Trajectories Per Class**	**AMKS (%)**	**Proposed Work (%)**
Name: T15 #Class=15 #Trajectories=1500	C1=57, C2=271, C3=141, C4=137, C5=128, C6=53, C7=109, C8=59, C9=127, C10=105, C11=53, C12=65, C13=141, C14=35, C15=19	87.60	94.80
Name: MIT #Class=6 #Trajectories=400	C1=173, C2=52, C3=19, C4=87, C5=44, C6=25	94.25	96.75

two successive observation) was set to 1e-7.

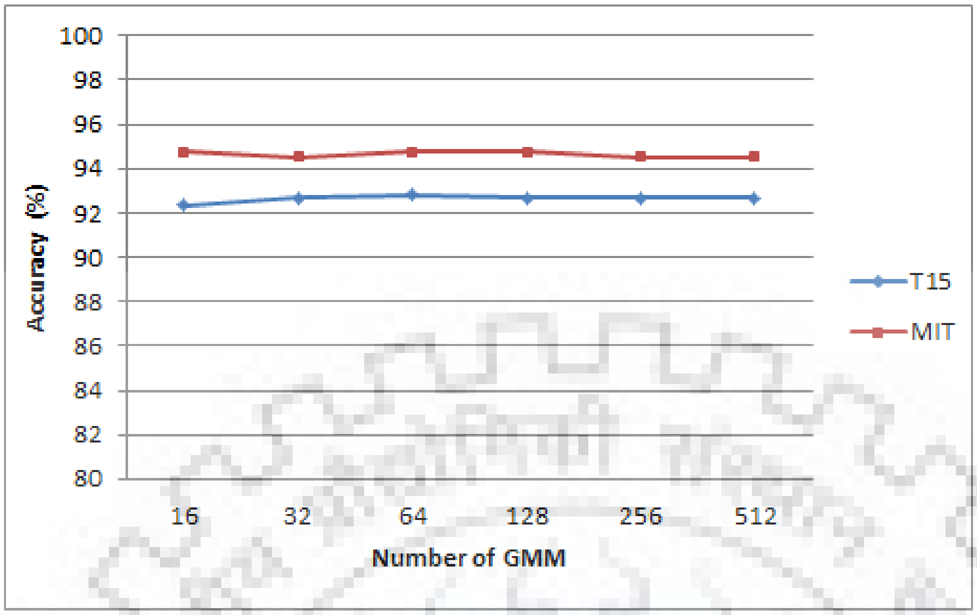
In experimental setup, we have used four parameters, namely number of HMM states h, GMMs, difference of top two target class probabilities Δp and $w_i,\ (i = 0, 1, 2..h)$. In stage 2, the SHMM requires the value of 'h' to segment each trajectory and then to classify the samples. The value of h in second stage is chosen as per the GHMM classification results in the first stage.

4.3.1.1 Results using GHMM:

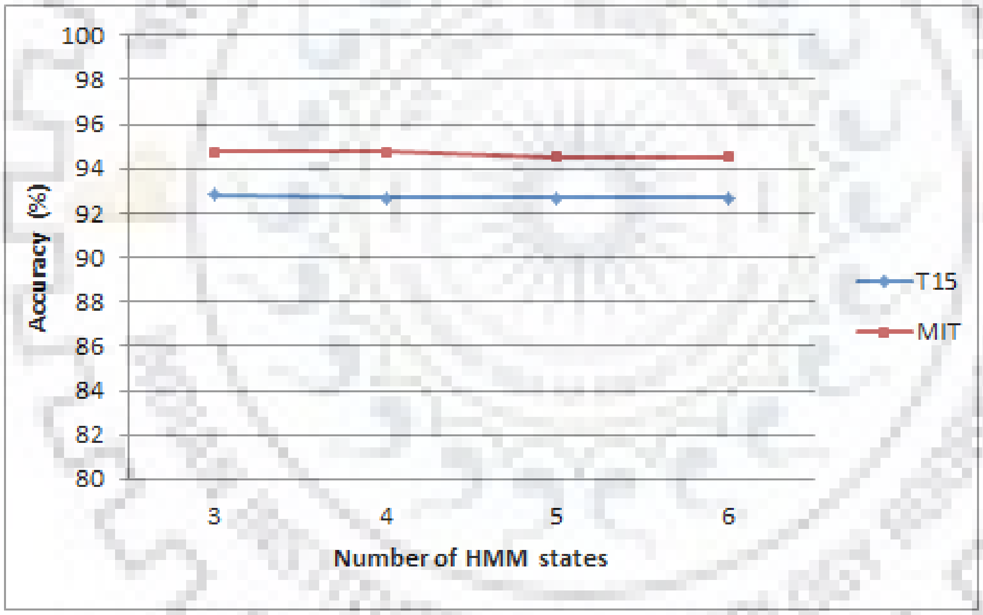
Figure 4.9 shows the result of classification over T15 and MIT dataset using GHMM. Experiments have been carried out by varying GMMs and number of HMM states. Figure 4.9(a) and Figure 4.9(b) show that GHMM provides the best performance with 92.80% accuracy on T15 dataset with HMM states=3 and GMM=64 and accuracy of 94.75% for MIT dataset with HMM states=3 and GMM=128. It is to be noted from Figure 4.9 that GHMM performs better with states $h = 3$. Thus, the value of h is chosen to be 3 for SHMM.

4.3.1.2 Results using SHMM:

Figure 4.10 shows the result of classification for the sample passed into the second stage over T15 and MIT dataset using SHMM. Experiments have been carried out by varying GMMs and number of HMM states. Figure 4.10.(a) and Figure 4.10.(b) depict that SHMM produces highest accuracy of 71.88% on T15 dataset with HMM states=3 and GMM=64 and accuracy of 93.55% for MIT dataset with HMM states=3 and GMM=128.



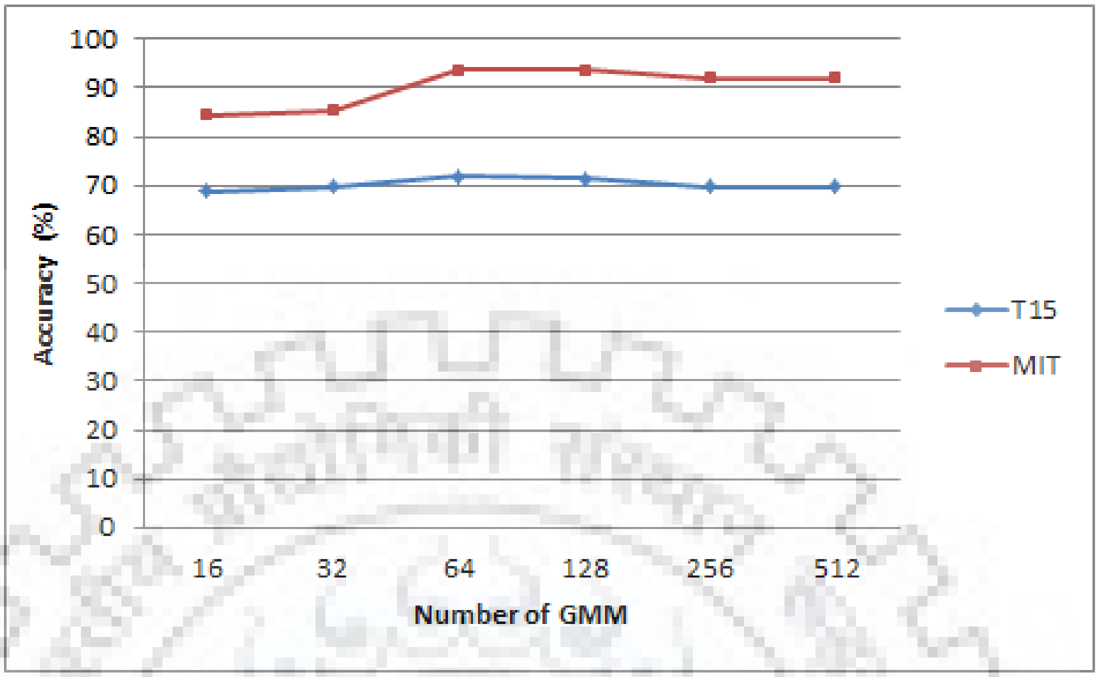
(a)



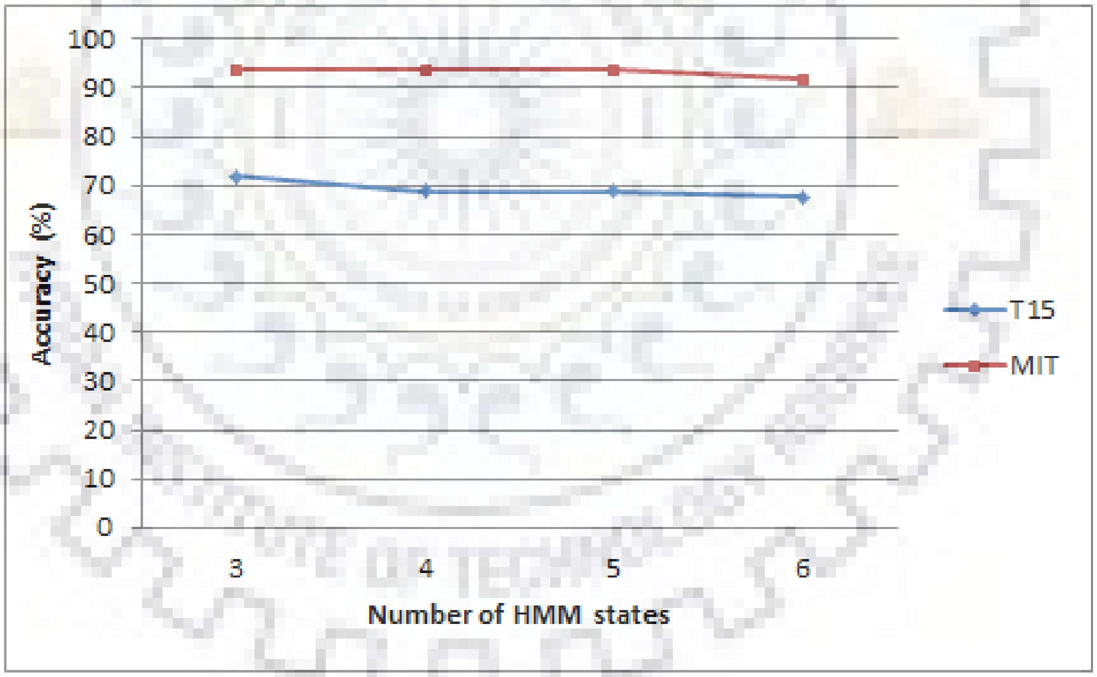
(b)

Figure 4.9: HMM Classification over T15 and MIT dataset with varying GMMs and HMM states: (a) Performance of GHMM with varying GMMs, (b) Performance of GHMM with varying HMM states.

We have tested our framework by varying the difference of top-two target probabilities, Δp. Figure 4.11 shows that the highest accuracy is achieved at $\Delta p = 0.008,\ 0.009,\ 0.01$ on T15 dataset and $\Delta p = 0.008,\ 0.009$ on MIT dataset.



(a)



(b)

Figure 4.10: HMM Classification over T15 and MIT dataset with varying GMMs and HMM states: (a) Performance of SHMM with varying GMMs, (b) Performance of SHMM with varying HMM states.

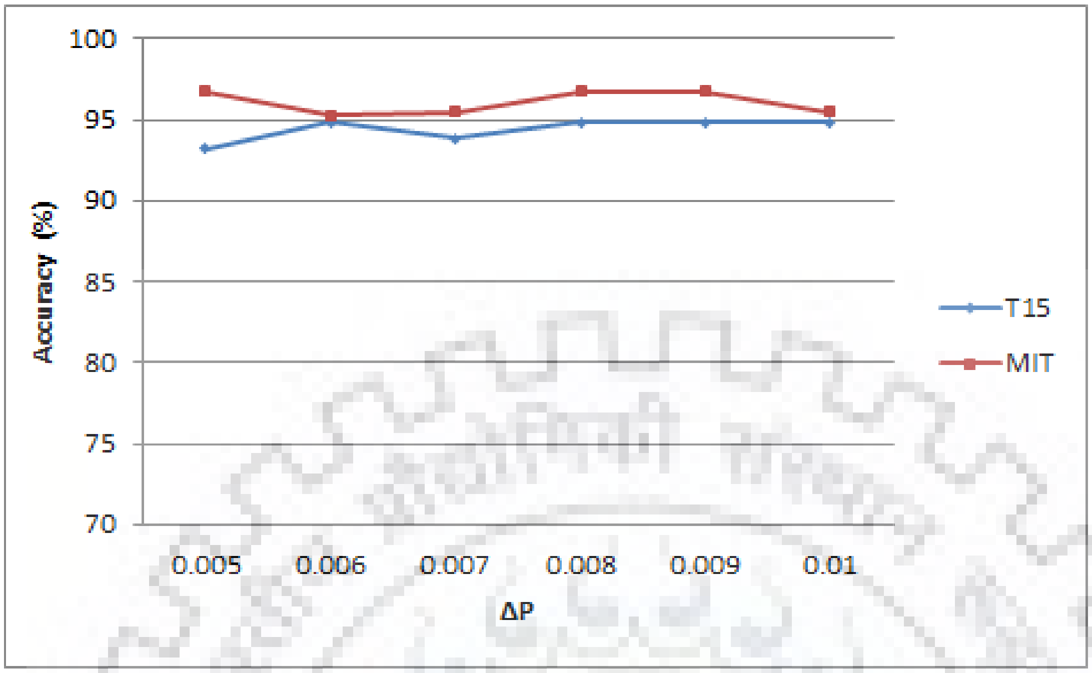
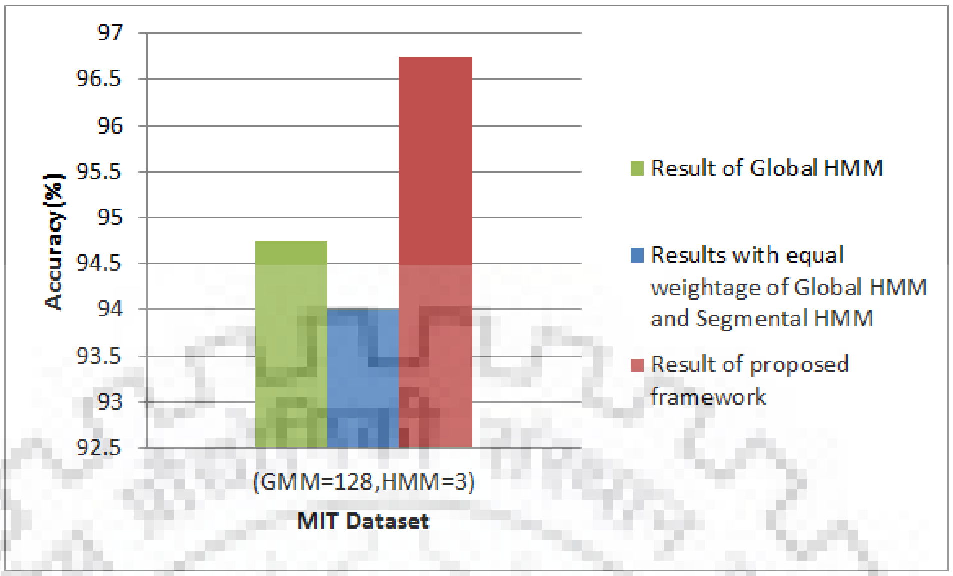


Figure 4.11: Performance with T15(HMM states=3 & GMM=64) and MIT(HMM states=3 & GMM=128) datasets by varying the difference of top two target probabilities, Δp.

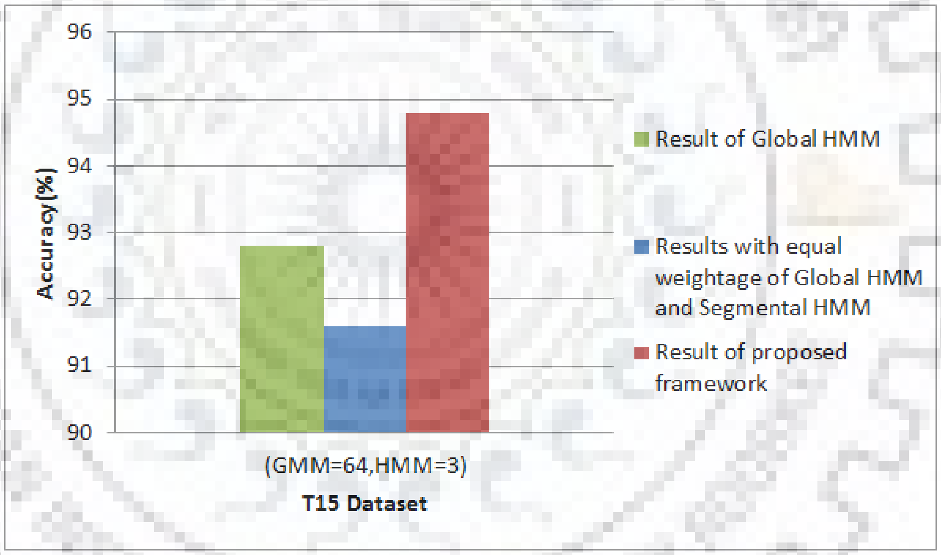
4.3.2 Combination of GHMM and SHMM Results using GA

We have used GA for optimization of combination of GHMM and SHMM results and to find out the values of weights $'w_i,\ i = 0, 1, 2...h\ where \sum_{i=0}^{h} w_i = 1'$. Here, roulette wheel selection has been used for selection of chromosomes. The crossover rate has been set to 0.8 and the mutation probability has been set as 0.01 in GA optimization. Over the iterations, GA optimizes the weights and improves the classification. GA itself gives the optimal values of the weights. The optimized values have been recorded as $w_0 = 0.5288,\ w_1 = 0.0073,\ w_2 = 0.2210,\ w_3 = 0.2429$ for T15 dataset and $w_0 = 0.5714,\ w_1 = 0.1429,\ w_2 = 0.143,\ w_3 = 0.1427$ for MIT dataset.

Figure 4.12(a) and Figure 4.12(b) show the comparison of classification of GHMM and combination of GHMM and SHMM with equal weight and the proposed work applied on T15 and MIT datasets. In T15 dataset, GHMM achieves the accuracy of 92.80%, whereas performance degrades to 91.60% when the classification from GHMM and SHMM is combined using equal weights. The proposed framework with GA improves the classification accuracy to 94.80% on T15 dataset. In MIT dataset, GHMM here achieves the accuracy of 94.75%, whereas the performance degrades to 94% when the classifications from GHMM and SHMM are combined with equal weight. The proposed framework improves the classification and we have recorded accuracy of 96.75% on MIT



(a)



(b)

Figure 4.12: Classification Results using MIT and T15 datasets. Green bar shows the accuracy of GHMM, The accuracy of combined classification of GHMM and SHMM results with equal weightage is and is shown in Blue. Proposed framework improves the classification and is shown in Red.

dataset.

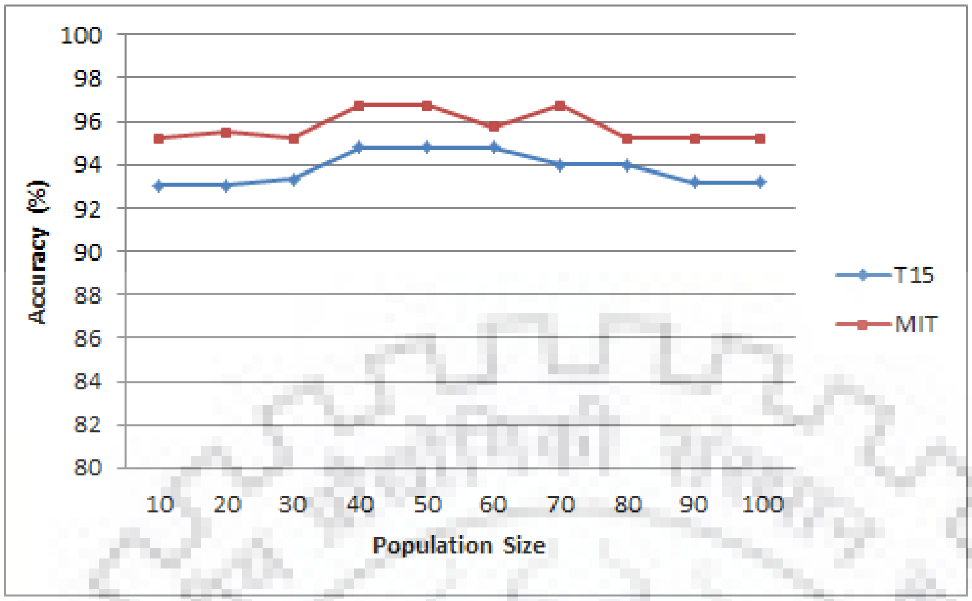


Figure 4.13: Performance with T15(HMM states=3 & GMM=64) and MIT(HMM states=3 & GMM=128) datasets by varying population size in GA.

4.3.2.1 Population size in GA

Figure 4.13 shows the performance of proposed framework by varying the population size. Highest accuracy of 94.80% (HMM states=3 & GMM=64) and 96.75% (HMM states=3 & GMM=128) have been recorded on T15 and MIT datasets, respectively. The highest accuracy has been recorded at population size=40 and 50. It is probably because as the population size increases, performance may degrade. At population size 100, performance decreases to 95.25% on MIT dataset.

4.3.3 Results with Added Gaussian Noise

Experiments have also been conducted to analyze the robustness of proposed framework by adding Gaussian noise with mean=0 and standard deviation=$0.5 \times R$, where R is the dynamic range of the trajectories. A decrement of 1.62% and 1.05% have been recorded with accuracies of 93.18% and 95.70% for T15 and MIT dataset, respectively. Figure 4.14 shows the results on T15 and MIT datasets using the noisy data.

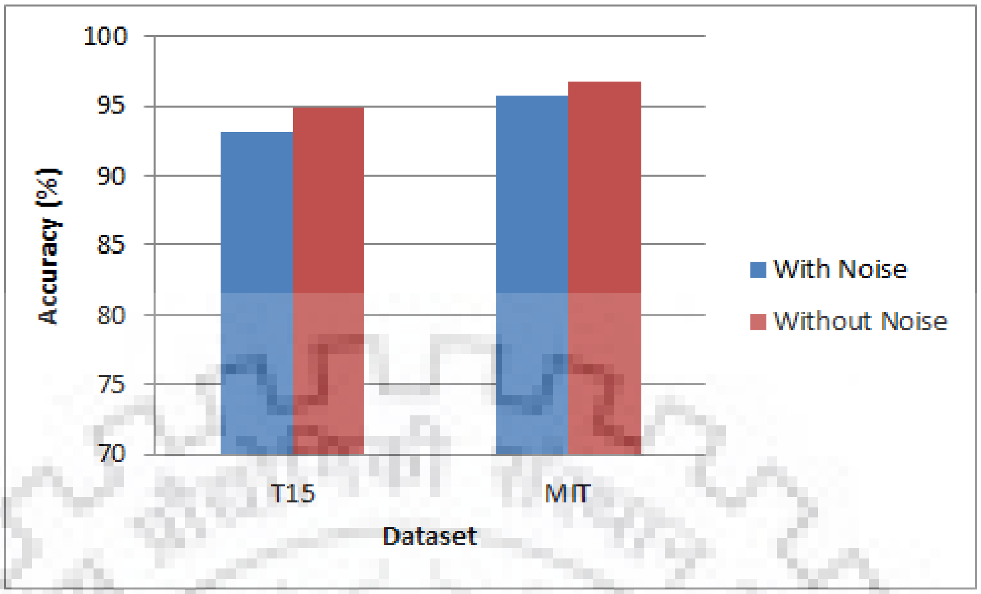


Figure 4.14: Impact of noise on classification performance.

Table 4.3: Comparison of time computation between AMKS algorithm and proposed approach

	T15 (minutes)	MIT (minutes)
AMKS	21.25	5.66
Proposed Kernel	32.291	8.44

4.3.4 Time Computation and Comparative Study

Table 4.3 shows the computation time taken in generating shrunk trajectories. As we introduce two more kernels, the running time is likely to be high. The shrinking time using the proposed kernel is however reasonable.

Table 4.4 shows the comparative performance of state of the art methods. The proposed approach outperforms other methods and achieves accuracy of 94.80% and 96.75% on T15 and MIT datasets, respectively.

A visual behavior framework to detect abnormality in trajectories has been proposed in [66] using sparse reconstruction analysis. Authors have used Least-squares Cubic Spline Curves Approximation (LCSCA) to define trajectories as fixed number of control point representation. Sparse reconstruction analysis has been performed using LCSCA represented trajectories and threshold was decided to predict abnormal behavior of trajectories.

Table 4.4: Performance comparison with other methods.

Method	T15(%)	MIT(%)
AMKS+HMM [41]	84.40	94.25
MS [127]	85.30	–
MBMS [42]	86.60	–
AMKS+K-means [41]	87.60	–
Visual Behavior [66]	93.26	95.44
Proposed approach	**94.80**	**96.75**

Table 4.5: Performance of proposed work using ACO and GA

	T15 (%)	MIT (%)
Proposed Approach + ACO	94.25	96.40
Proposed Approach + GA	**94.80**	**96.75**

4.3.4.1 Ant Colony Optimization (ACO)

The ACO is inspired from some species of ants. There are ants that follow the particular way while they move. Ants deposit pheromone on the ground while moving. Ants do that to indicate some favorable path that other ants follow. The similar idea is simulated to find the desired solution for optimization problems. Artificial ants try to find optimal solution by exchanging information via communication links. Every iteration ACO tries to optimize the given problem and come up with the optimal solution once the convergence criteria is fulfilled. More details could be found in [128, 129].

The combination of GHMM and SHMM has also been performed using ACO [128, 129]. Table 4.5 shows the performance of proposed framework having combination performed using ACO and GA. The GA based combination performs well as compared to ACO.

4.3.5 Beyond Trajectory Classification: Signature Recognition

Signature is one of the most common biometric traits used for user authentication. It involves identification of user by user's signatures. Signatures could either be on-line or off-line. On-line signatures [130] are usually recorded via stylus on an electronic pad that records stylus tip position, pressure, azimuth and altitude angles. Whereas, off-line signatures [131] are usually drawn on paper and their images are used to extract features for recognition purpose.

We have tested SHMM using SVC2004 [132] on-line signature dataset for signature recognition. It has been noted that SHMM works well for signatures.

4.3.5.1 SVC2004 Dataset Details

The dataset [132] consists of 1600 signatures of 40 different subjects. There are 40 instance of signatures of each user. Out of 40 samples, 20 samples are genuine and rest are forged. Hence, the total number of genuine samples are 800 ($40\times$ 20=800). Each signature is represented as an ordered sequence of the points. Figure 4.15 shows signatures of two different subjects from SVC2004 dataset, where three samples are shown as row-wise for subjects S1 and S2, respectively.

- X-coordinate - scaled cursor position along the x-axis
- Y-coordinate - scaled cursor position along the y-axis
- Time stamp - system time at which the event was posted
- Button status - current button status (0 for pen-up and 1 for pen-down)
- Azimuth - clockwise rotation of cursor about the z-axis
- Altitude - angle upward toward the positive z-axis
- Pressure - adjusted state of the normal pressure

4.3.5.2 Feature Extraction for Signatures

We have extracted two features from raw (x,y) signatures, i.e. velocity and curvature [133]. Velocity is calculated as the difference of two consecutive points. Curvature based feature is calculated as follows:

Curvature [133] of point P_2 is calculated using two neighboring points P_1 and P_3 as shown in Figure 4.16.

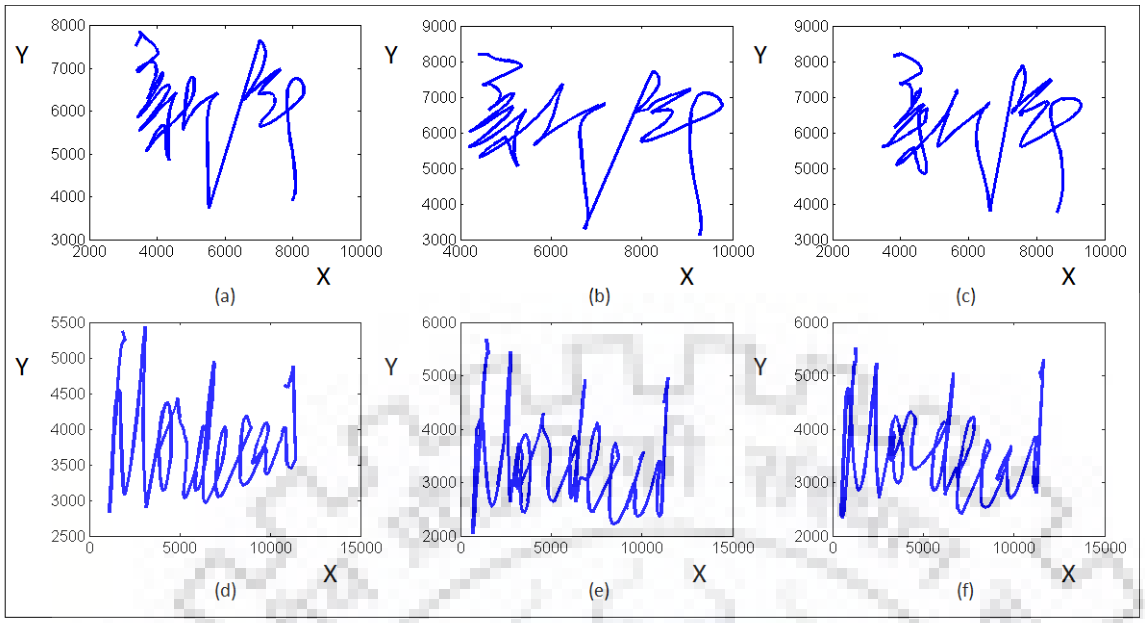


Figure 4.15: Signature samples: First row shows three samples of subject S1. Similarly, second row is of subject S2.

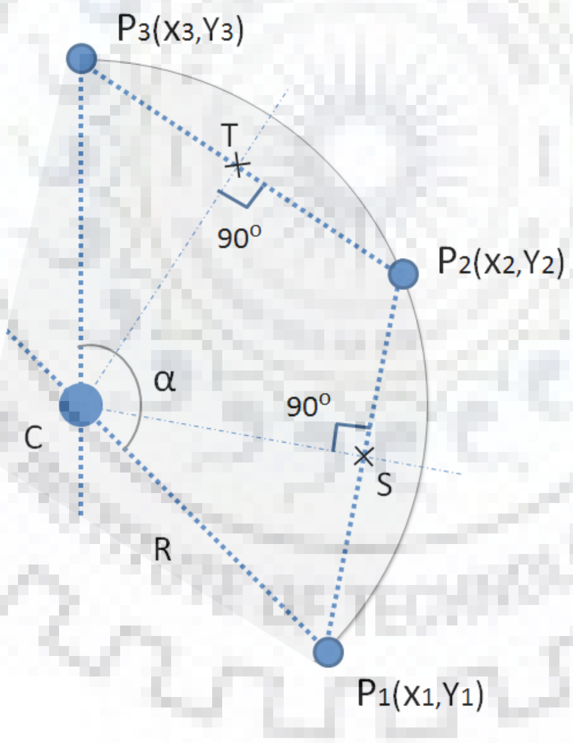


Figure 4.16: Computation of curvature feature using two neighbors P_1 and P_3.

A circle is fitted using three consecutive non-linear points of signature trajectories P_1, P_2 and P_3. It includes vectors $\overrightarrow{CT}$, $\overrightarrow{CS}$, circle center $\overrightarrow{C}$, angle α and radius R. $\overrightarrow{C}$ is calculated using Eq. 4.3.1

$$\vec{C} = \frac{Sin2P_1\ \vec{P_1} + Sin2P_2\ \vec{P_2} + Sin2P_3\ \vec{P_3}}{Sin2P_1 + Sin2P_2 + Sin2P_3} \tag{4.3.1}$$

4.3.5.3 Signature Recognition Results

Signatures have been classified using GHMM and SHMM, respectively. Figure 4.17(a) shows the recognition rates when tested with 4 HMM states and with GMM components $\{16, 32, 64, 128, 256\}$. Similarly, Figure 4.17(b) shows the accuracies by varying number of HMM states $\{3, 4, 5, 6, 7\}$ with GMM components as 32. The highest accuracy of 93.13% has been recorded with HMM state 4 and 32 GMMs.

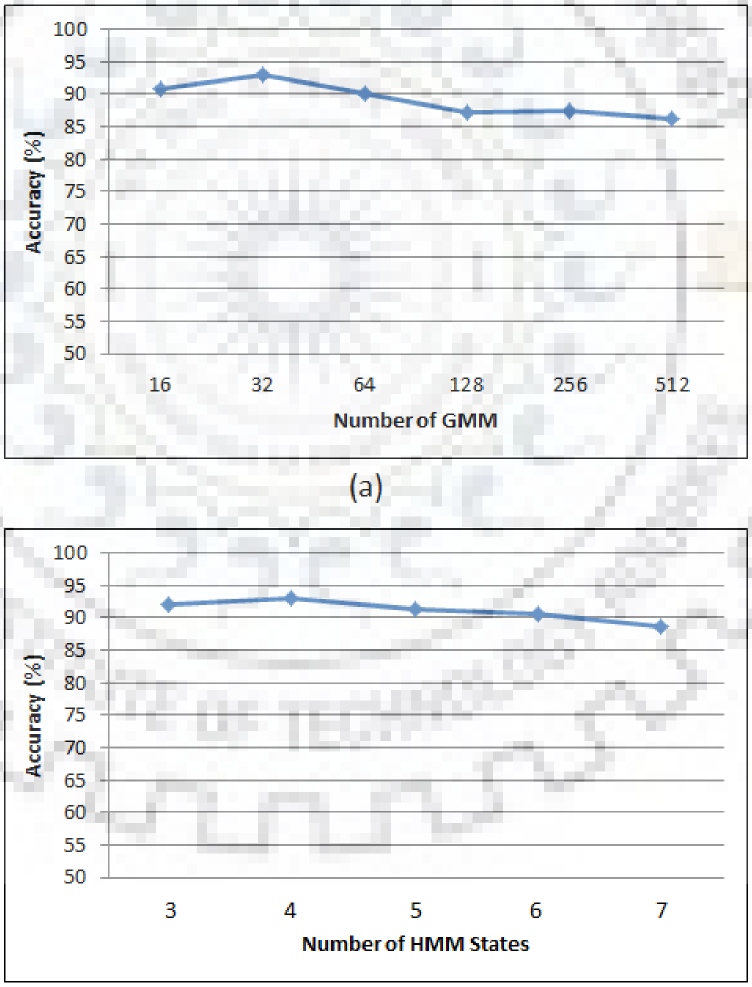


Figure 4.17: GHMM classification performance over SVC2004 dataset. (a) With HMM states 4 and varying GMM components. (b) With varying number of HMM states using GMM 32.

Next, state-wise separation of signatures has been carried out on 4 segments and classified using

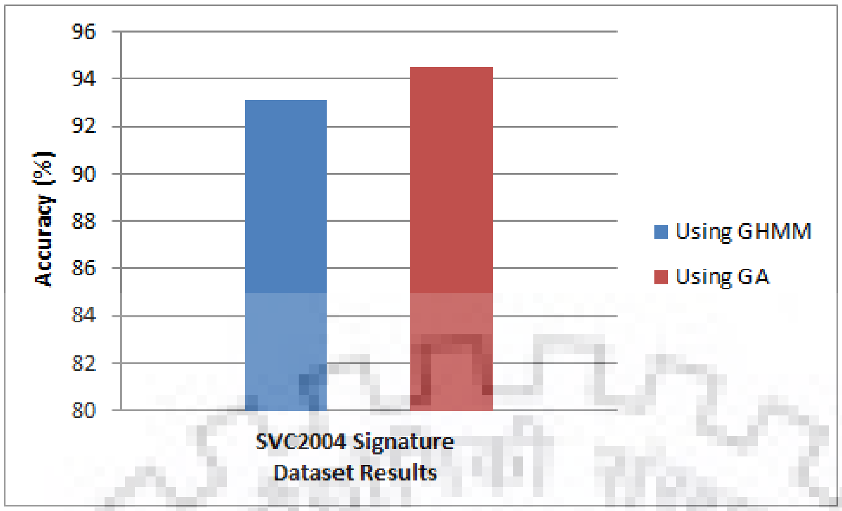


Figure 4.18: SVC2004 Results using GHMM and GA based combination.

SHMM. The genetic framework is used to combine recognition scores from GHMM and SHMM. Hence, the chromosome length becomes 5 for the signatures. The size of initial population has been set to 20. The optimized weights from GA have been recorded as $w_0 = 0.5192$, $w_1 = 0.0410$, $w_2 = 0.3258$, $w_3 = 0.0088$, and $w_4 = 0.1049$. Figure 4.18 shows the classification results using GHMM and GA based combination of GHMM and SHMM. An improvement of 1.37% is recorded with an accuracy of 94.50% using GA.

4.4 Discussion

We propose a new approach for trajectory shrinking and classification, where shrinking is based on the position, speed, starting/ending points, and CH/RDP algorithm, and the classification has been done based on GHMM and local SHMM with a GA based optimization. A two stage HMM classification is used in this work, where HMM is used for trajectory classification and GA is used for weight optimization to improve the classification. The Accuracy of AMKS framework has been recorded as 87.60% on T15 dataset and 94.25% on MIT dataset, whereas the accuracy of our approach has been recorded as 94.80% and 96.75%, respectively on T15 and MIT dataset. Experiments reveal that our approach produces better result as compared to AMKS framework. The proposed kernel prevents over-shrinking by preserving the shape of the trajectories leading to the better classification performance. Also, the idea is to learn from individual trajectory segments and later combining their results. In this way, there is the scope of improvement in classification performance using the fusion based techniques such as GA, and ACO. SHMM is not restricted to trajectory classification and it can be used for other machine learning tasks such as signature recognition. We have shown the application of SHMM for signature recognition, where an accuracy of 94.50% has been recorded. In future, we aim to use SHMM framework for action recognition in RGB and 3D depth videos.

Chapter 5

Scene Segmentation based on Motion Trajectory Patterns

Trajectory classification is a precursor step towards scene segmentation, given a set of trajectories. Scene segmentation is helpful for understanding the flow of motion in the different regions of area under surveillance. It plays a vital role in the field of visual surveillance and security where we aim to classify surveillance scenes based on two important information, namely scene's layout and activities or motions within the scene. In this work, we propose a supervised learning based method to segment surveillance scenes with the help of high-level features extracted from object trajectories.

5.1 Introduction

Self regulating visual vigilance systems are highly dependent upon motion patterns of moving objects to find out distinct types of activities occurring within the range of sensor. Motion patterns of the objects can be quite explanatory and often used for various assignments, such as scene semantic analysis [82], highway traffic management [1], and atypical activity detection [134].

Computer vision aided object detection and tracking has advanced significantly during last two decades. Hence, applications of object detection and tracking has increased in leaps and bounds. For example, it is being used for foreground segmentation [91] semantics analysis [82, 137], scene classification, segmentation [48, 67], and interest area localization [135], Region labeling [138].

Camera based surveillance has reached almost every corner of our society. It is crucial to develop systems that are capable of automatically taking care of the surveillance task efficiently without external support from human observers. The goal can be achieved with the help of systems having better understanding of surveillance scene that can be achieved with scene segmentation. Majority of the existing scene segmentation techniques use low level features such as location of the object center [67] or movement pattern learned through velocity or displacement [48].

The problem of abnormal activity detection can be exploited with supervised/unsupervised manner given the trajectory features. Though simple features like time-series representation of an object's trajectory (x_i, y_i, t_i) can work in local pattern analysis, however, high-level representation of the scene dynamic may not be possible using such simple features. The work of Dogra et al. [48] can be used to get a new representation using the RAG (Region Association Graph) and block importance based features. For example, the RAG based segmentation of a scene proposed in their method can be useful to represent the moving object's path.

In this work, we have used two high-level features proposed in the work of [48], namely block **label** and **node-number** to classify object trajectories. The modified representation of the trajectories or paths with respect to RAG representation of the scene are then fed to an HMM classifier. Next, a heuristic has been applied to assign the blocks to grow meaningful segments. We compared our technique with the method proposed by Dogra et al. [48] using two datasets, namely IIT and MIT car datasets. Our proposed method gives better results on scene segmentation as compared to the method proposed in [48].

In the next Section, we present the proposed work of trajectory classification and scene segmentation. In Section 5.3, experimental results obtained using two surveillance datasets are presented. We conclude in the Section 5.4 by highlighting some of the possible extensions of the present work.

5.2 Trajectory Classification and Scene Segmentation

We have processed the raw trajectories to extract the high-level features using a methodology introduced by the authors of [48]. Their method is fundamentally built upon a hypothesis introduced by Dogra et al. [135]. They have assumed a novel theoretical model of object motion and tested their supposition with benchmark datasets. Their model shows that, the probabilistic importance distribution of distinct localized areas or block of a given surveillance scene, follows a predefined pattern. They have partitioned a surveillance scene into $N \times N$ rectangular blocks and calculated the probabilistic importance or label of each block based on parameters such as, number of objects visiting a block and the total time spent by all moving objects inside the block. Next, a weighted graph referred to as RAG, has been constructed. They have used this for detection of suspicious or anomalous movements.

In this work, we have used high-level features, namely **label** and **node-number**, to represent a coarse trajectory. It has been verified that, these two features are important to decide the pattern of movements inside a scene. In Figure 5.1, we show a sample RAG based segmentation map of the surveillance scene taken from the IIT human trajectory dataset [48], wherein red regions represent rarely or never visited blocks, blue areas represent moderately visited zones, and green segments are frequently visited blocks. These blocks are encoded with decimal values, e.g. $\{4, 3, 2\}$, and a region growing technique has been adopted to construct the RAG. Scene segmentation solely based on movement patterns is discussed in this work without any texture based information.

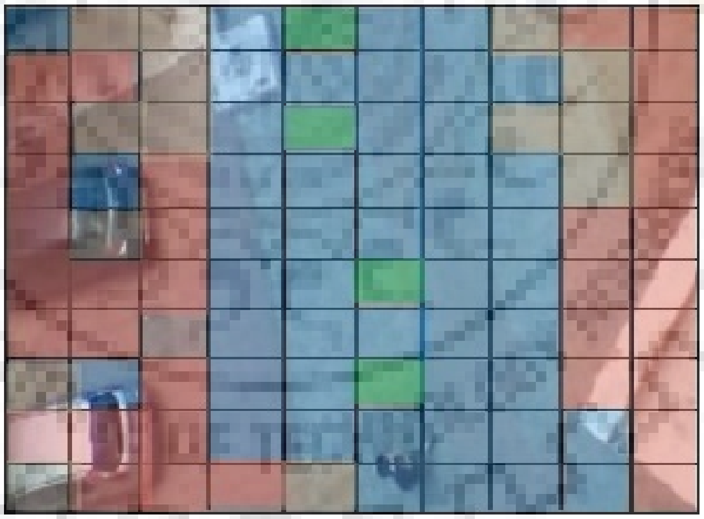


Figure 5.1: RAG based segmentation map generated using the method proposed in [48] when applied on IIT dataset.

5.2.1 High-Level Feature Extraction

For the purpose of trajectory smoothing and removal of discontinuities or outliers, we have used RANSAC based approach proposed by [136]. Outliers exist in the signal because of tracking error. Therefore, we need to remove these outliers for better segmentation results. Dogra et al. [135] have extracted some high-level features, such as **label** of a block (b). In their approach, first they have calculated average velocity of a target object from its uniformly sampled trajectory segment. In the next step, they have calculated the total number of times a block is visited by various targets, which is referred to as **global count** or σ_b. Using this count, they filtered-out some unimportant blocks. Lastly, they have estimated block importance (τ_b) using Eq. 5.2.1 and Eq. 5.2.2, respectively.

$$\omega_b = \omega_b + \frac{\overline{v}^{t_i} - v^{t_i}}{\overline{v}^{t_i}} \tag{5.2.1}$$

where ω_b represents the weight of the block b computed from average ($\overline{v}^{t_i}$) and instantaneous (v^{t_i}) velocities of a target object, t_i.

$$\tau_b = \frac{\omega_b}{\sigma_b} \tag{5.2.2}$$

In this work, we have used the above block importance (τ_b) to construct a high-level feature vector comprising with **label** together with the original (x, y) coordinate values of the trajectory, and the **node-number** of a block as per the RAG proposed in [48]. Hence each point of a trajectory of arbitrary length can be replaced by the four dimensional feature point as given in Eq. 5.2.3, where $x(i)$, $y(i)$, $i)_b$, and $node - number(i)_b$, represent value of x coordinate, y coordinate, label of block b, and node-number of block b at sequence number i of the time-series representation of the original trajectory.

$$F(i) = [x(i), y(i), label(i)_b, node - number(i)_b] \tag{5.2.3}$$

5.2.2 Supervised Classification using HMM

As HMM is a supervised approach, therefore, we first need to manually classify some of the trajectories to prepare a training set and provide this set as input to the HMM. Classification of trajectories highly depends upon the scene geometry as well as on movement patterns. In general, classification criteria likely to change as the scene and motion patterns change. Based on this, the number of classes may vary. Remaining set of trajectories are fed to the classifier to grouping. using these grouped trajectories, we apply a heuristic to merge blocks of the image. Finally, we get the segmentation of the scene. Segmentation is described in the following Section.

5.2.3 Surveillance Scene Segmentation

The scene segmentation heuristic is described in this Section. As mentioned earlier, a surveillance scene is first needs to be partitioned into non-overlapping local grids or blocks. Next, we apply the following methodology to assign or group these blocks based on the number of trajectories passing through these blocks. We may recall, our HMM based classification has already grouped the trajectories into desired numbers of classes. Therefore, the whole scene is finally segmented into regions depending upon the number of classes in the trajectories plus one. This extra region is introduced because, there may be some blocks that have not visited by any target. The method of segmentation is as follows.

Let T_i represents the i^{th} trajectory of length n and b_j denotes the j^{th} block of the scene such that B is the set of all blocks. We assume, there are k classes of trajectories, where c_k represents the k^{th} class. Now, for every block, say $b_j \in B$, we determine the set of trajectories λ_{b_j} passing through block b_j. Now, we partition the trajectories from set λ_{b_j} into respective classes. In the next step we find the dominating class within that block having highest number of trajectory footfalls. Suppose c_k be the dominating class for the block b_j, then we assign class c_k to this block. An example of the above block labeling is depicted in Figure 5.3. Finally, simple region growing algorithm has been used to merge blocks having similar cluster assignment and connected through 8-connectivity rule.

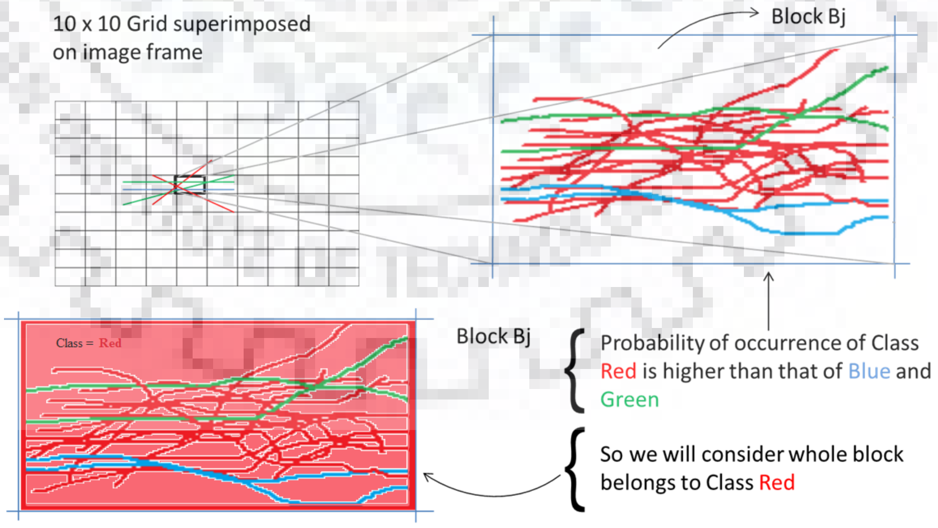

10 x 10 Grid superimposed
on image frame
Block Bj
Class = Red
Block Bj
Probability of occurrence of Class Red is higher than that of Blue and Green
So we will consider whole block belongs to Class Red

Figure 5.2: An example demonstrating the process of association of a block to a particular segment or region.

5.3 Experimental Results

In this Section, we present the outcomes obtained by applying our method on public datasets and we also show comparative performance evaluation carried out against the method proposed by authors of [48]. We extracted trajectories using the target detection and tracking algorithm proposed by Dinh et al. [116].

5.3.1 Datasets and Ground Truths

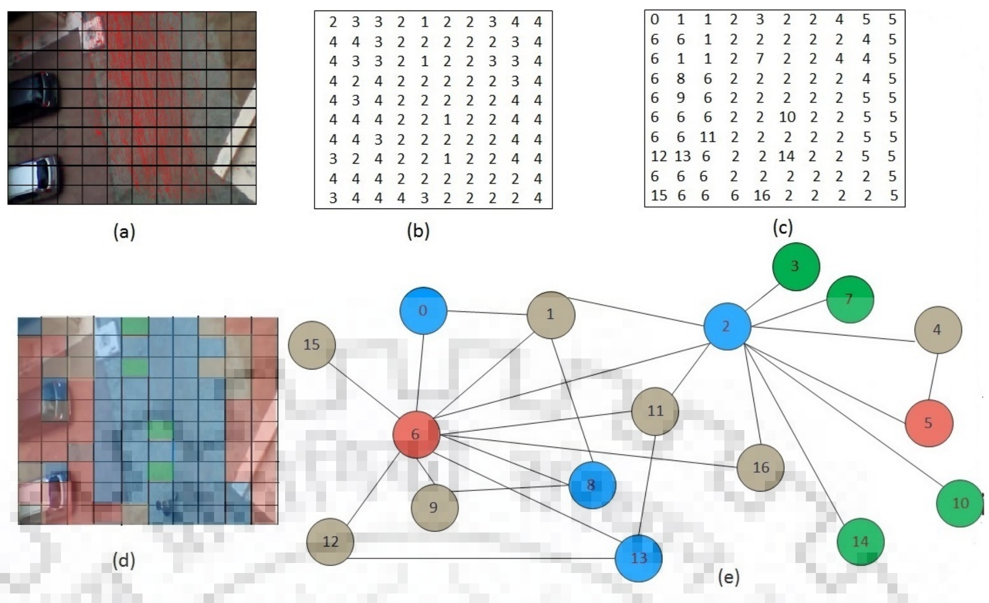
Through experiments, we have tested our methodology applied on IIT surveillance dataset made by the authors of [48] and MIT car dataset [126]. The IIT dataset consists of a total of 191 distinct human trajectories and it was made for testing scene segmentation and anomaly detection algorithms. Car dataset is huge in volume (≥ 40k trajectories), out of which we have randomly selected 400 trajectories to verify our algorithm. Dogra et al. [48] have shown that, the anomaly detection algorithm proposed in their work performs satisfactorily on IIT dataset as well as other public datasets, namely VISOR[1] and CAVIAR[2]. However, these datasets are not suitable for our application because of insufficient number of trajectories. Thus, we have not tested our method using these datasets.

We initiate our discussion on outcomes by showing original scenes of both IIT and MIT datasets with trajectories plotted, as shown in Figure 5.3(a) and Figure 5.4(a). Corresponding label map, node-number map, scene-segmentation map, and RAG representations are shown in subsequent diagrams of the Figures.

5.3.2 HMM Classification Results

Trajectories were manually labeled and then used for training and testing. HMM was trained and tested using high level features *'label'* and *'node-number'*. Based on the HMM classification, we segmented the surveillance scene of both datasets. We have trained and tested HMM with varying number of classes. Finally, we have observed that, HMM based classification produces exciting results for four classes when applied on MIT car dataset. We have classified the IIT dataset tra-

[1]http://www.openvisor.org
[2]http://homepages.inf.ed.ac.uk/rbf/CAVIARDATA1/



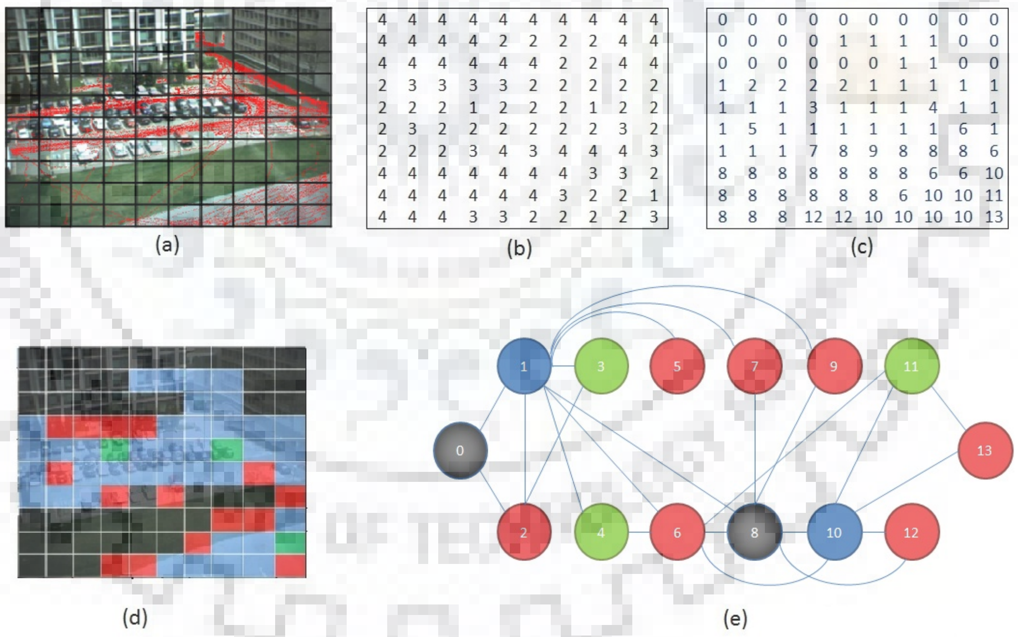
(b)

2	3	3	2	1	2	2	3	4	4
4	4	3	2	2	2	2	2	3	4
4	3	3	2	1	2	2	3	3	4
4	2	4	2	2	2	2	2	3	4
4	3	4	2	2	2	2	2	4	4
4	4	4	2	2	1	2	2	4	4
4	4	3	2	2	2	2	2	4	4
3	2	4	2	2	1	2	2	4	4
4	4	4	2	2	2	2	2	2	4
3	4	4	4	3	2	2	2	2	4

(c)

0	1	1	2	3	2	2	4	5	5
6	6	1	2	2	2	2	2	4	5
6	1	1	2	7	2	2	4	4	5
6	8	6	2	2	2	2	2	4	5
6	9	6	2	2	2	2	2	5	5
6	6	6	2	2	10	2	2	5	5
6	6	11	2	2	2	2	2	5	5
12	13	6	2	2	14	2	2	5	5
6	6	6	2	2	2	2	2	2	5
15	6	6	6	16	2	2	2	2	5

Figure 5.3: (a) The background scene of IIT dataset divided into 10×10 number of blocks with overlayed trajectories (b) Labeling of the blocks using the method described in [48] (c) Construction of the graph nodes using labels (d) Color-coded representation of the segmented scene, and (e) RAG corresponding to the segmentation.



(b)

4	4	4	4	4	4	4	4	4	4
4	4	4	4	2	2	2	2	4	4
4	4	4	4	4	4	2	2	4	4
2	3	3	3	3	2	2	2	2	2
2	2	2	1	2	2	2	1	2	2
2	3	2	2	2	2	2	2	3	2
2	2	2	3	4	3	4	4	4	3
4	4	4	4	4	4	4	3	3	2
4	4	4	4	4	4	3	2	2	1
4	4	4	3	3	2	2	2	2	3

(c)

0	0	0	0	0	0	0	0	0	0
0	0	0	0	1	1	1	1	0	0
0	0	0	0	0	0	1	1	0	0
1	2	2	2	2	1	1	1	1	1
1	1	1	3	1	1	1	4	1	1
1	5	1	1	1	1	1	1	6	1
1	1	1	7	8	9	8	8	8	6
8	8	8	8	8	8	8	6	6	10
8	8	8	8	8	8	6	10	10	11
8	8	8	12	12	10	10	10	10	13

Figure 5.4: (a) The background scene of MIT dataset divided into 10×10 number of blocks with overlayed trajectories (b) Labeling of the blocks using the method described in [48] (c) Construction of the graph nodes using labels (d) Color-coded representation of the segmented scene, and (e) RAG corresponding to the segmentation.

jectories into two classes. It has been observed that, for both datasets, trajectories of suspicious or off-the-track nature have been classified with reasonably high accuracy. Figure 5.5 presents the

HMM classification of both datasets.

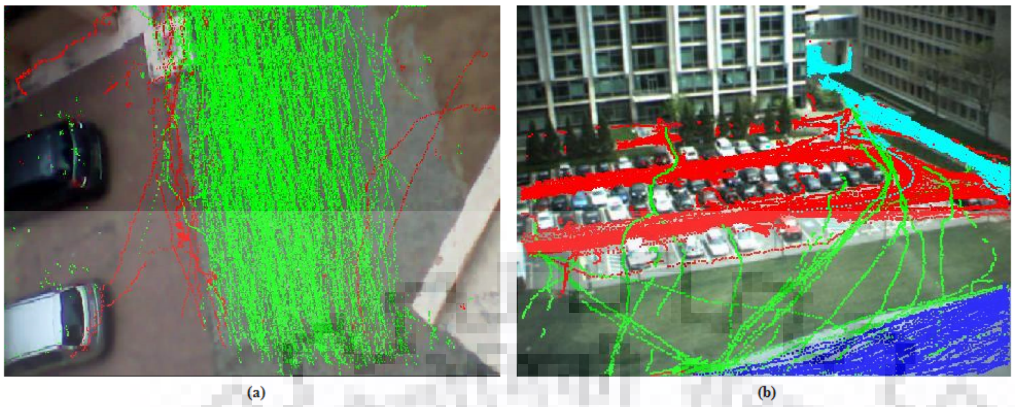


Figure 5.5: HMM Classification (a) ($k = 2$) for IIT dataset (b) ($k = 4$) for MIT dataset

Figure 5.5 shows that, our proposed classification-based segmentation produces better scene segmentation as compared to segmentation obtained by Dogra et al. [48]. Trajectories were pre-processed by a method proposed in [136] to remove the effect of outliers. We mark the unvisited blocks with **black** color after outlier removal. For IIT dataset, frequently visiting or regular blocks are marked as green and suspicious bocks are marked as red. For MIT dataset, frequently visiting nodes are marked as green, moderately visited nodes with sky blue color, rarely or abnormally visited blocks with red color and rest of nodes are marked with purple color.

It can be easily verified from Figure 5.6 that, the approach of [48] identifies some blocks as frequently visiting, however, in reality, these blocks have never been visited by any moving target. On the other hand, our proposed segmentation algorithm does a better job. Ground truths and classification results using IIT surveillance dataset and MIT car dataset are presented in Table 5.1 and Table 5.2, respectively.

Table 5.1: Results on IIT dataset (Training:50% and Testing: 50%), GT: Ground Truth

Dataset: IIT Trajectory=191; GT: C1=190; GT: C2=10; K=2;	C1	C2	Result
	91	4	Accuracy=96.84% Precision=75% Recall=60%

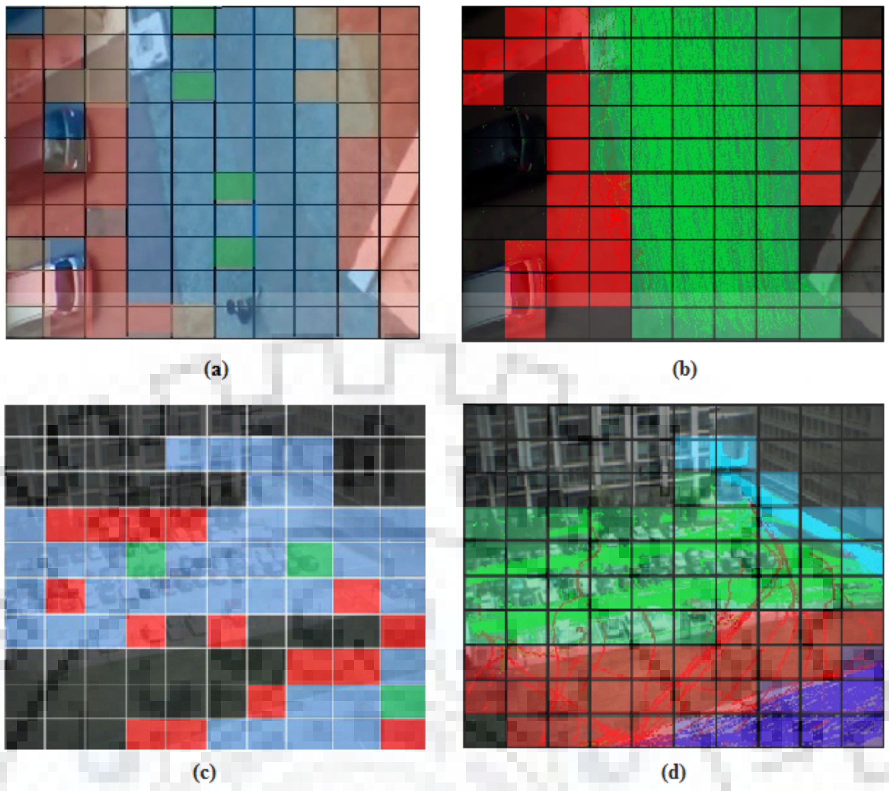


Figure 5.6: HMM Classification based Segmentation (a) ($k = 2$) Segmentation of IIT dataset using [48] (b) Segmentation of IIT dataset using our approach (c) ($k = 4$) Segmentation of MIT car dataset using [48] (d) ($k = 4$) Segmentation of MIT car dataset with our approach

Table 5.2: Results using MIT car dataset (Training: 50% and Testing: 50%), GT: Ground Truth

Dataset: MIT car	C1	C2	C3	C4	Result

Trajectory=400;
GT:C1=195;
GT:C2=103;
GT:C3=77;
GT:C4=25;
K=4;
	92	55	38	13	Accuracy=97.47%
					Precision=
					92.31%
					Recall=100%

5.4 Discussion

In this work, surveillance scene segmentation with the help of trajectory classification using HMM, is introduced. High-level features have been obtained from raw object trajectories and then trajectories were classified into normal and abnormal movements on MIT parking-lot and IIT Bhubaneshwar datatsets. High-level features are obtained using a recently proposed unsupervised technique to label segments of a given surveillance scene partitioned into non-overlapping blocks. The proposed method produces better results (Figure 5.6) as compared to the method of [48]. The proposed method is a preliminary work towards the automatic scene segmentation where the grid size will be selected automatically. The proposed trajectory classification and scene segmentation methodology has many applications including traffic management, anomalous activity detection, and crowd flow analysis.

Chapter 6

Conclusion and Future Work

In this thesis, we have worked on basically two types of trajectories i.e. the vehicle and human trajectories recorded in surveillance area such as cross-road, and a lab environment. We have used publicly available datasets in this thesis. A computer system with Intel i3 3^{rd} generation processor and 4 GB of RAM has been used to implement the work presented in this thesis. We believe that the system performance will be better in terms of computation time with the use of high-end machines.

In the very first work presented in Chapter 3, the concepts of graph theory for trajectory classification. A global cost was determined using DTW algorithm between each pair of trajectories. Next, each trajectory was partitioned into variable number of segments based on their geometry. A complete bipartite graph was formulated between each pair of trajectories and two local costs were determined from it. These global and local costs are combined using particle swarm optimization to improve the classification rates.

In the second work presented in Chapter 4, we proposed a new kernel and a segmental HMM based trajectory classification model. The kernel uses position vectors, speed, end points, convex hull points and RDP points to shrink the trajectories. Next, segmental HMM breaks the trajectories into fixed number of segments and individually learns from them. Finally, individual responses are combined using genetic algorithm to improve the classification rates.

In the third work presented in Chapter 5, we used RAG based features for trajectory representation and used them to train HMM classifier. Using the HMM classifier results, we segmented the surveillance scene into $M \times N$ local non-overlapping grids for analyzing the motion through each local grid followed by classification using the features extracted grid-wise. The grid size of 10×10 has been opted from state of the art method to compare with. The model outperformed the state of the art method.

The first work described in Chapter 3 has the capability to exploit other problems such as signature modeling for user identification and verification. This work can be extended with deep learning based framework to analyze the local dynamics of trajectories and can be used to address the problems like action/activity recognition. The work presented in Chapter 4 may be suitable for modeling online signatures, online gestures, dynamic sign language recognition with proper features extraction methods. As this work presents a segmental HMM based method, such strategy can be applied to LSTMs for gesture and signature modeling. The work discussed in Chapter 5 uses motion trajectories. The work can be extended be combining the texture based information for the scene segmentation.

The work presented in this thesis has dealt with the classification of object trajectories using the supervised approach. Thus, in future, we will focus on building the unsupervised or semi-supervised version of the work discussed in this thesis because there might be situations where the trajectory patterns may not be available with their ground truths i.e. the class labels. Also, we will focus on building Generative Adversarial Network (GAN) based methods to improve the trajectory classification performance. The work discussed in this thesis has very important applications like traffic management, lane classification, scene segmentation, and anomaly detection.

Appendix A

Supplementary Material

A.1 Dataset Description

Here, we present the details of the datasets used in this thesis.

T15 Dataset: The trajectories of T15 dataset [111] are labeled into 15 different classes. It consists of 1500 noisy trajectories that have been collected over a span of time. The size of each class is not fixed and it varies from 19 to 278. The samples of T15 dataset are shown in Figure A.1.

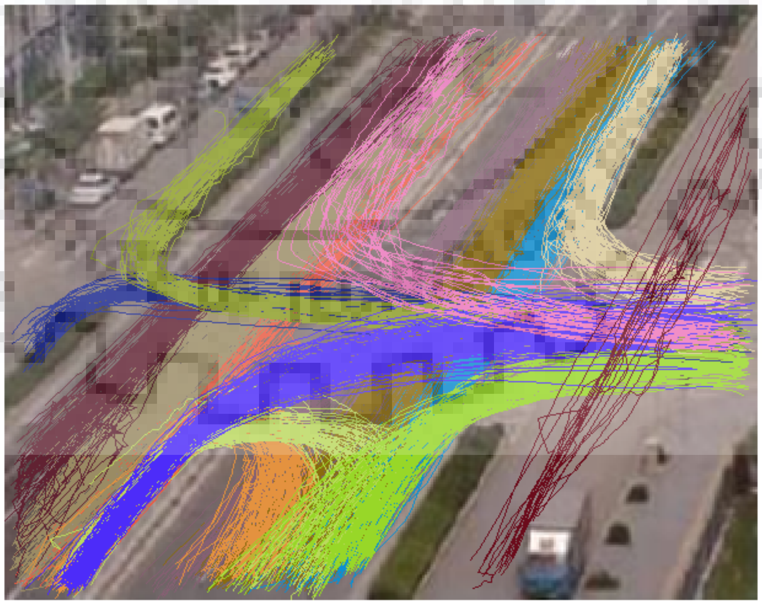


Figure A.1: Samples of T15 trajectory datasets. Classes are represented by color. For better visibility of colors please see the pdf version.

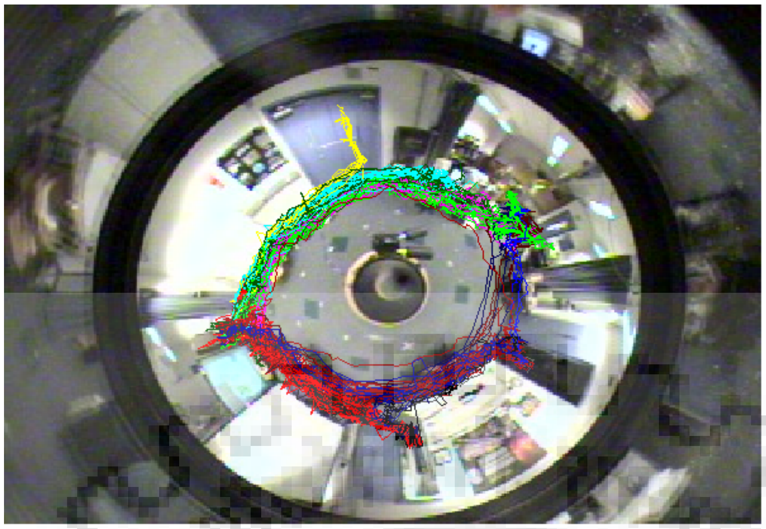


Figure A.2: Samples of LabOmni trajectory datasets. Classes are represented by color. For better visibility of colors please see the pdf version.

LabOmni Dataset: LabOmni [112] dataset consists of human trajectories walking through a lab captured using an omni-directional camera. The dataset consists of 209 trajectories from 15 classes. The number of trajectories in a class varies from 3 to 36. Trajectory points are the pixels of captured video. Samples of LabOmni dataset are shown in Figure A.2.

CROSS Dataset: CROSS [112] dataset consists of a total number of 1900 trajectories from 19 different classes. Each class contains 100 trajectories. The length of trajectories vary from 5 to 23. Figure A.3 shows the samples of CROSS dataset with trajectories from different classes are plotted in color.

MIT Car Dataset: MIT car dataset [126][1] consists of 400 very noisy trajectories extracted from videos. MIT car trajectories are labeled in six classes. Here, the number of trajectories in each class is not fixed and varies from 19 to 173. MIT Car trajectories are shown in Figure A.4.(b). MIT dataset contains several noisy trajectories and the length of trajectories vary not only among different classes, but within the same class too.

[1]MIT dataset has many trajectory out of which many trajectories are very noisy and/or very short in length. So only 400 trajectories were selected for experiments.

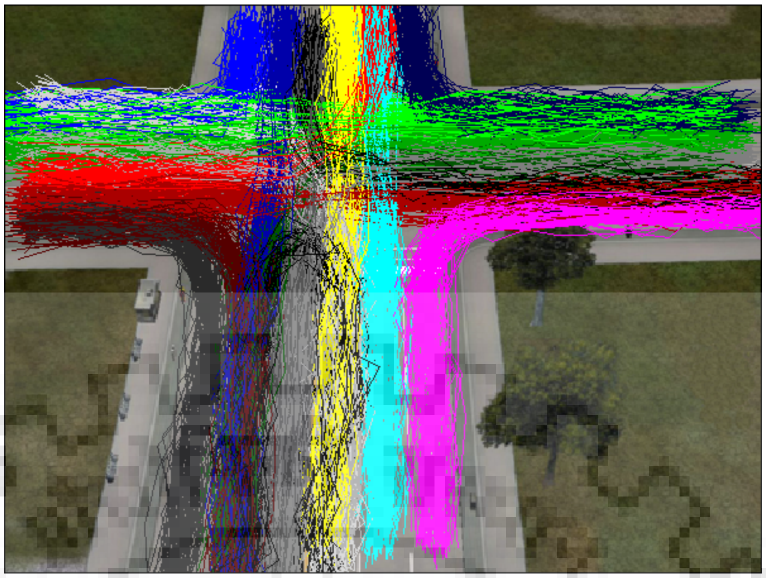


Figure A.3: Samples of CROSS trajectory datasets. Classes are represented by color. For better visibility of colors please see the pdf version.

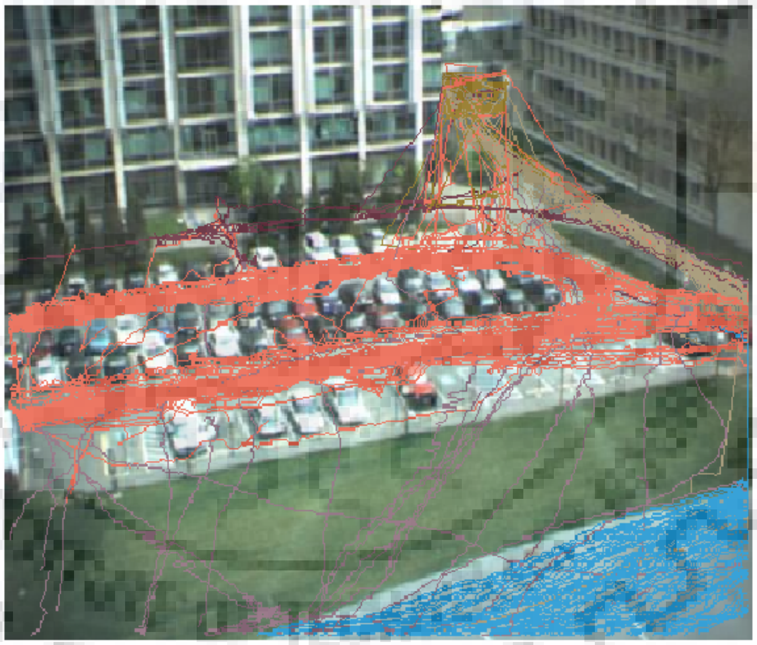


Figure A.4: MIT Dataset (number of classes=6). Color represent trajectory classes.

A.2 Hidden Markov Model (HMM)

HMM has been known for its capability to model the sequential dependencies. A HMM model can be defined by using three tuples i.e. π, A, B where π defines the initial state probabilities, π, $A = [a_{ij}]$, $i, j = 1, 2, \ldots N$ is a state transition matrix which denotes the transition probability a_{ij} from state i to state j and B denotes the observation probability $b_j(O_k)$ which is modeled with

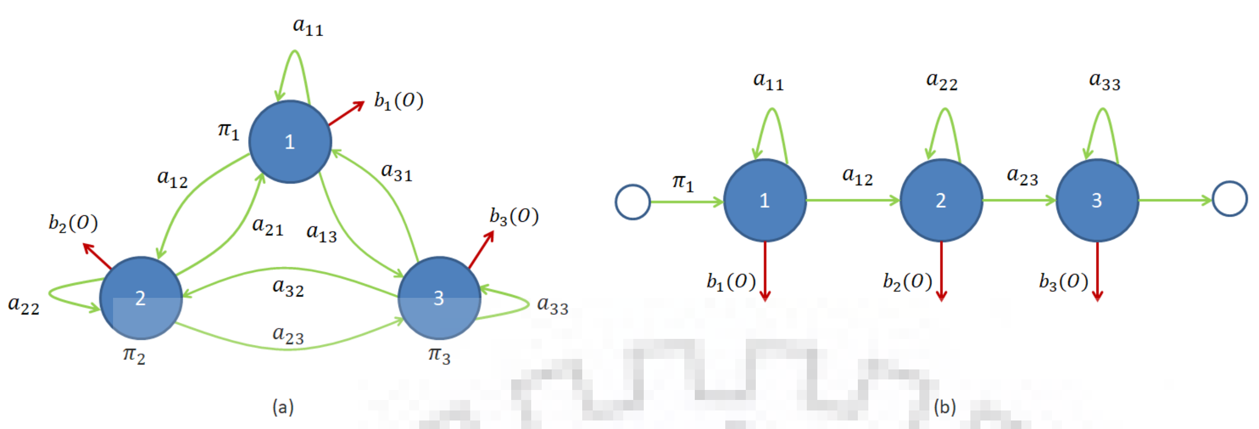


Figure A.5: Variants of HMM: (a) 3-state ergodic model (b) 3-state let-to-right model.

the continuous probability density function from state j and observing a symbol O_k. The density function is represented by $b_j(x)$, where x denotes the k dimensional feature vector. For each state of the model, a Gaussian Mixture Model (GMM) is defined separately and the output probability density of state j can be defined using Eq. A.2.1

$$b_j(x) = \sum_{k=1}^{M_j} c_{jk}\aleph(x, \mu_{jk}, \Sigma_{jk}) \tag{A.2.1}$$

where M_j denotes the number of Gaussian alloted to j, and $\aleph(x, \mu, \Sigma)$ denotes a Gaussian with mean μ and co-variance matrix $\sum$ and c_{jk} denotes weight coefficient of the Gaussian with component k of state j. For a model λ, if O is an observation sequence $O = (O_1, O_2, \ldots O_T)$ and is assumed to be generated by a state sequence $Q = Q_1, Q_2, \ldots Q_T$ of length T, we can calculate the probability of observation or likelihood using Eq. A.2.2, where π_{q_1} denotes the initial probability of start state.

$$P(O, Q|\lambda) = \sum_{Q} \pi_{q_1} b_{q_1}(O_1) \prod_{T} a_{q_{T-1}q_T} b_{q_T}(O_T) \tag{A.2.2}$$

Figure A.5 shows the two variants of HMM where (a) is a 3-state ergodic model, whereas (b) is a 3-state left-to-right model. To train a model, the original label and corresponding feature vector sequences are fed together in training phase. Baum-Welch algorithm has been used for re-estimation of the initial output probability distributions of $b_i(O)$ for maximizing the likelihood of the training set. Viterbi decoding algorithm is used for recognition purpose which finds the character sequence having the best likelihood using a given feature vector sequence.

A.3 Support Vector Machine (SVM)

SVM is a supervised learning based classifier and is widely used by the researchers to perform classification of trajectories [142] using linear and non-linear kernels. The classifier maps the data into a feature space where a hyperplane separates the classes [142]. The general solution is presented in Eq. A.3.1, where k represents the kernel function that SVM uses to add both linear and non-linear classification functionality.

$$f(x) = \sum_i a_i y_i k(x_i, x) \tag{A.3.1}$$

A.4 Random Forest (RF):

RF [143] is a supervised classifier that uses bootstrapping of Features into multiple training subsets. Next, it builds classification trees for each training subset. The final classification is made by collecting decisions from all the trees and choosing the final class having maximum votes. The voting can be done by assigning equal shares to the decisions of all trees or a weighting scheme can be adopted to assign unequal weights to the decisions of all trees. Figure A.6 shows the procedure of the RF classifier to decide the final class through voting using decisions of M number of trees.

The selection of root nodes and splitting of features are done on the basis of information gain (I_G) and entropy of features. The nodes are split only if there is a positive I_G. The I_G of splitting training data (S) into subsets (S_j) could be done using Eq. A.4.1.

$$I_G = -\sum_j \frac{|S_j|}{|S|} E(S_j) \tag{A.4.1}$$

where $|S_j|$ and $|S|$ are the size of the sets S_j and S, respectively. $E(S_j)$ is the entropy of set S_j.

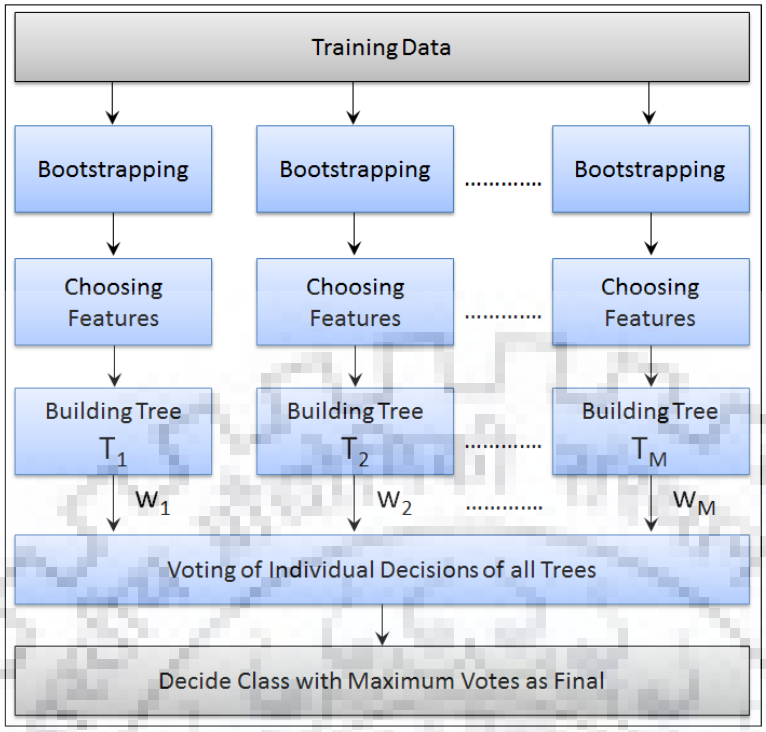


Figure A.6: RF classification process.

A.5 Comparison with SVM and RF Classifiers

Trajectories are sequential in nature. Thus, to model the sequential dependencies of trajectories, sequential classifier is required. The static classifiers like SVM, and RF only work on static data i.e. position vectors in corresponding feature space. So, the performance of static classifiers may be lesser than that of sequential classifiers. However, we have compared the performance of proposed work (Chapter 3) with the SVM, and RF classifiers on *T15, LabOmni*, and *CROSS* datasets, respectively in this thesis. For SVM, radial basis function kernel has been used. The feature like Convex Hull, angular feature, and curvature [144–146] have been extracted and their mean is taken to feed into these classifiers. Classification rates are mentioned in the Table A.1.

Table A.1: Comparison with SVM, and RF classifier

Dataset	**SVM(%)**	**RF(%)**	**Proposed(%)**
T15	83.43	90.58	92.05
LabOmni	85.19	93.32	98.46
CROSS	90.37	95.30	99.58

A.6 BLSTM-NN based Sequence Classification

BLSTM-NN is a popular sequence classification model and has been used in various hand-writing [139] and gesture recognition problems [141]. BLSTM-NN has two hidden layers that process a given input sequence in both directions [140]. One layer is used to process the sequence in forward direction, whereas, the other processes the input in backward direction. Both hidden layers are attached with the same output layer that has one node for each possible activity present in the input sequence. A special node ϵ in the output layer is used to indicate '$no - activity$' which refers that no decision has been made at that position. Two different models can be trained using Cross Entropy Error (CEE) based objective function and Connectionist Temporal Classification (CTC) based objective function.

A.6.1 CTC based BLSTM-NN

The Connectionist Temporal Classification (CTC) objective function (O) is used as negative *log* probability of correct labeling of the entire training set. If S is the given training set with a pair of input and target sequence as (p, q), then the O can be described using Eq. A.6.1

$$O = -\ln\left(\prod_{(p,q)\epsilon S} p(q|p)\right) = -\sum_{(p,q)\epsilon S} \ln\left(p(q|p)\right). \tag{A.6.1}$$

Here, O has the ability to model a label sequence with given inputs. In some cases, it has been found that BLSTM-NN performs better than HMM classifier for handwriting recognition by providing a long range of context accessible in both input directions. The model also resolves the problem of vanishing gradient using Long Short-Term Memory (LSTM) hidden layers which contains memory blocks to provide access to the information for longer time.

A.6.2 CEE based BLSTM-NN

Softmax function has been used in output layer in CEE based BLSTM-NN, which ensures that the network outputs have been normalized between 0 and 1; and sums to 1. The network has K output units, one for each class of the gesture sequence [140]. CEE for K classes are defined in Eq. A.6.2

$$CEE = -\sum_{(x,z)\epsilon T}\sum_{i=1}^{K} z_i \ln y_i \tag{A.6.2}$$

where (x, z) is the input pair with x as the input sequence and z as the target sequence from the training set T. The term y defines the probability such that the input belongs to a particular class and it can be computed using Eq. A.6.3

$$p(z|x) = \prod_{i=1}^{K} y_i^{z_i} \tag{A.6.3}$$

In future, we will model trajectories using LSTM based networks.

Bibliography

[1] Melo, J., Naftel, A., Bernardino, A., Santos-Victor, J.: Detection and classification of highway lanes using vehicle motion trajectories. IEEE Transactions on Intelligent Transportation Systems **7**(2), 188–200 (2006)

[2] K. Santosh, C. Nattee, and B. Lamiroy. Spatial similarity based stroke number and order free clustering. In *2010 12th International Conference on Frontiers in Handwriting Recognition*, pages 652–657. IEEE, (2010).

[3] Sekar, V., Anuradha, J., Arya, A., Balusamy, B., Chang, V.: A framework for smart traffic management using hybrid clustering techniques. Cluster Computing pp. 1–16 (2017)

[4] Liao, D., Li, H., Sun, G., Zhang, M., Chang, V.: Location and trajectory privacy preservation in 5g-enabled vehicle social network services. Journal of Network and Computer Applications (2018)

[5] Bae, I., Kim, J., Kim, S.: Steering rate controller based on curvature of trajectory for autonomous driving vehicles. In: Intelligent Vehicles Symposium (IV), pp. 1381–1386. IEEE (2013)

[6] Lee, S., Kim, J.C.S.: A 3-d real-time simulation for autonomous driving with v2v communications. In: International Conference on Connected Vehicles and Expo (ICCVE), pp. 800–801. IEEE (2013)

[7] Lee, S., Kim, J., Song, H., Kim, S.: A 3-dimensional real-time traffic simulator considering the interaction among autonomous and human-driven vehicles. In: 17th International Conference on Intelligent Transportation Systems (ITSC), pp. 1917–1918. IEEE (2014)

[8] Moon, J., Bae, I., Cha, J., Kim, S.: A trajectory planning method based on forward path generation and backward tracking algorithm for automatic parking systems. In: 17th International Conference on Intelligent Transportation Systems (ITSC), pp. 719–724. IEEE (2014)

[9] Suzuki, N., Hirasawa, K., Tanaka, K., Kobayashi, Y., Sato, Y., Fujino, Y.: Learning motion patterns and anomaly detection by human trajectory analysis. In: Proceedings of the International Conference on Systems, Man and Cybernetics, pp. 498–503 (2007)

[10] Xu, D., Wu, X., Song, D., Li, N., Chen, Y.: Hierarchical activity discovery within spatio-temporal context for video anomaly detection. In: Proceedings of International Conference on Image Processing, pp. 3597–3601 (2013)

[11] Krishna, K., Kalra, P.: Detection, tracking and avoidance of multiple dynamic objects. Journal of Intelligent and Robotic Systems **33**(4), 371–408 (2002)

[12] Krishna, K., Kalra, P.: Perception and remembrance of the environment during real-time navigation of a mobile robot. Robotics and Autonomous Systems **37**(1), 25–51 (2001)

[13] Murala, S., Wu, Q., Balasubramanian, R., Maheshwari, R.: Joint histogram between color and local extrema patterns for object tracking. In: Video Surveillance and Transportation Imaging Applications, vol. 8663, p. 86630T. International Society for Optics and Photonics (2013)

[14] Kim, B., Hong, G., Kim, J., Choi, Y.: An efficient vision-based object detection and tracking using online learning. Journal of Multimedia Information System **4**(4) (2017)

[15] Kim, B., Park, D.: Unsupervised video object segmentation and tracking based on new edge features. Pattern Recognition Letters **25**(15), 1731–1742 (2004)

[16] Saqib, M., Khan, D.S., Sharma, N., Blumenstein, M.: A study on detecting drones using deep convolutional neural networks. In: 14th IEEE International Conference on Advanced Video and Signal Based Surveillance (AVSS), pp. 1–5. IEEE (2017)

[17] Tran, M.B., Trivedi, M.M.: Trajectory learning for activity understanding: Unsupervised, multilevel, and long-term adaptive approach. IEEE Transactions on Pattern Analysis and Machine Intelligence **33**(11), 2287–2301 (2011)

[18] Guru, D., Kiranagi, B., Nagabhushan, P.: Multivalued type proximity measure and concept of mutual similarity value useful for clustering symbolic patterns. Pattern Recognition Letters **25**(10), 1203–1213 (2004)

[19] Pandey, S., Khanna, P.: Content-based image retrieval embedded with agglomerative clustering built on information loss. Computers & Electrical Engineering **54**, 506–521 (2016)

[20] Pandey, S., Khanna, P.: A hierarchical clustering approach for image datasets. In: 9th International Conference on Industrial and Information Systems (ICIIS), pp. 1–6. IEEE (2014)

[21] Nascimento, J., Figueiredo, M.A.T., Marques, J.S.: Trajectory classification using switched dynamical hidden markov models. IEEE Transactions on Image Processing **19**(5), 1338–1348 (2010)

[22] Appiah, K., Hunter, A., Lotfi, A., Waltham, C., Dickinson, P.: Human behavioural analysis with self-organizing map for ambient assisted living. In: IEEE International Conference on Fuzzy Systems, pp. 2430–2437 (2014)

[23] Yadav, R., Kalra, P., John, J.: Time series prediction with single multiplicative neuron model. Applied soft computing **7**(4), 1157–1163 (2007)

[24] Kisku, D.R., Gupta, P., Sing, J., Nasipuri, M.: Fusion of gaussian mixture densities for face and ear biometrics using support vector machines. In: International Conference on Future Generation Information Technology, pp. 344–351. Springer (2010)

[25] Szilárd, V., Santosh, K.: A fast k-nearest neighbor classifier using unsupervised clustering. In: International Conference on Recent Trends in Image Processing and Pattern Recognition, pp. 185–193. Springer (2016)

[26] K. Santosh. g-dice: graph mining-based document information content exploitation. *International Journal on Document Analysis and Recognition (IJDAR)*, 18(4):337–355, (2015).

[27] Ferrer, M., Valveny, E., Serratosa, F., Bardají, I., Bunke, H.: Graph-based k-means clustering: A comparison of the set median versus the generalized median graph. In: International Conference on Computer Analysis of Images and Patterns, pp. 342–350. Springer (2009)

[28] Siang, K., Khor, S.: Path clustering using dynamic time warping technique. In: International Conference on Computing Technology and Information Management, vol. 1, pp. 449–452 (2012)

[29] Wirotius, M., Ramel, J.Y., Vincent, N.: Improving dtw for online handwritten signature verification. In: International Conference Image Analysis and Recognition, pp. 786–793. Springer (2004)

[30] K. Santosh. Character recognition based on dtw-radon. In *International Conference on Document Analysis and Recognition (ICDAR)*, pages 264–268. IEEE, (2011).

[31] K. Santosh, B. Lamiroy, and L. Wendling. Dtw for matching radon features: a pattern recognition and retrieval method. In *International Conference on Advanced Concepts for Intelligent Vision Systems*, pages 249–260. Springer, (2011).

[32] Santosh, K., Lamiroy, B., Wendling, L.: Dtw–radon-based shape descriptor for pattern recognition. International Journal of Pattern Recognition and Artificial Intelligence **27**(03), 1350,008 (2013)

[33] M.-R. Bouguelia, S. Nowaczyk, K. Santosh, and A. Verikas. Agreeing to disagree: Active learning with noisy labels without crowdsourcing. *International Journal of Machine Learning and Cybernetics*, 9(8):1307–1319, (2018).

[34] S. M. Obaidullah, A. Bose, H. Mukherjee, K. Santosh, N. Das, and K. Roy. Extreme learning machine for handwritten indic script identification in multiscript documents. *Journal of Electronic Imaging*, 27(5):051214, (2018).

[35] K. Santosh and L. Wendling. Pattern recognition based on hierarchical description of decision rules using choquet integral. In *International Conference on Recent Trends in Image Processing and Pattern Recognition*, pages 146–158. Springer, (2016).

[36] Chouakria, A., Nagabhushan, P.: Adaptive dissimilarity index for measuring time series proximity. Advances in Data Analysis and Classification **1**(1), 5–21 (2007)

[37] De, A., Desarkar, M., Ganguly, N., Mitra, P.: Local learning of item dissimilarity using content and link structure. In: Proceedings of the 6th ACM Conference on Recommender systems, pp. 221–224. ACM (2012)

[38] Kisku, D., Gupta, P., Sing, J.: Multibiometrics feature level fusion by graph clustering. International Journal of Security and Its Applications **5**(2), 61–74 (2011)

[39] Desarkar, M., Sarkar, S., Mitra, P.: Aggregating preference graphs for collaborative rating prediction. In: Proceedings of the 4th ACM Conference on Recommender systems, pp. 21–28. ACM (2010)

[40] Tang, K., Zhu, S., Xu, Y., Wang, F.: Modeling drivers' dynamic decision-making behavior during the phase transition period: An analytical approach based on hidden markov model theory. IEEE Transactions on Intelligent Transportation Systems **17**(1), 206–214 (2016)

[41] Xu, H., Zhou, Y., Lin, W., Zha, H.: Unsupervised trajectory clustering via adaptive multi-kernel-based shrinkage. IEEE International Conference on Computer Vision pp. 4328–4336 (2015)

[42] Wang, W., Carreira-Perpinan, M.A.: Manifold blurring mean shift algorithms for manifold denoising. In: IEEE Conference on Computer Vision and Pattern Recognition, p. 1759–1766 (2010)

[43] Park, J., Zhang, Z., Zha, H., Kasturi, R.: Local smoothing for manifold learning. In: IEEE Conference on Computer Vision and Pattern Recognition, vol. 2, pp. II–452–II–459 (2004)

[44] Goldberg, D., Deb, K., Kargupta, H., Harik, G.: Rapidaccurate optimization of difficult problems using fast messy genetic algorithms. In: International Conference on Genetic Algorithms, vol. 5, pp. 56–64 (1993)

[45] Roberge, V., Tarbouchi, M., Labonte, G.: Comparison of parallel genetic algorithm and particle swarm optimization for real-time uav path planning. IEEE Transactions on Industrial Informatics **9**(1), 132–141 (2013)

[46] Bonyadi, M., Michalewicz, Z.: Particle swarm optimization for single objective continuous space problems: a review (2017)

[47] Du, K., Swamy, M.: Particle swarm optimization. In: Search and Optimization by Metaheuristics, pp. 153–173. Springer (2016)

[48] Dogra, D., Reddy, R., Subramanyam, K., Ahmed, A., Bhaskar, H.: Scene representation and anomalous activity detection using weighted region association graph. In: Proceedings of the 10th International Conference on Computer Vision Theory and Applications, pp. 104–112 (2015)

[49] Piciarelli, C., Micheloni, C., Foresti, G.: Trajectory-based anomalous event detection. IEEE Transactions on Circuits and Systems for Video Technology **18**(11), 1544–1554 (2008)

[50] Brun, L., Saggese, A., Vento, M.: Dynamic scene understanding for behavior analysis based on string kernels. IEEE Transactions on Circuits and Systems for Video Technology **24**(10), 1669–1681 (2014)

[51] Lee, A.J., Chen, Y.A., Ip, W.C.: Mining frequent trajectory patterns in spatial–temporal databases. Information Sciences **179**(13), 2218–2231 (2009)

[52] Kihwan, K., Dongryeol, L., Irfan, E.: Gaussian process regression flow for analysis of motion trajectories. In: International Conference on Computer vision, pp. 1164–1171 (2011)

[53] Mehta, P., Shah, H., Kori, V., Vikani, V., Shukla, S., Shenoy, M.: Survey of unsupervised machine learning algorithms on precision agricultural data. In: ICIIECS, pp. 1–8 (2015)

[54] Cai, Y., Wang, H., Chen, X., Jiang, H.: Trajectory-based anomalous behaviour detection for intelligent traffic surveillance. IET Intelligent Transport Systems **9**(8), 810–816 (2015)

[55] Jan, T.: Neural network based threat assessment for automated visual surveillance. In: IEEE International Joint Conference on Neural Networks, vol. 2, pp. 1309–1312 (2004)

[56] Dahmane, M., Meunier, J.: Real-time video surveillance with self-organizing maps. In: Canadian Conference on Computer and Robot Vision, vol. 2, pp. 136–143 (2005)

[57] Kwon, Y., Kang, K., Jin, J., Moon, J., Park, J.: Hierarchically linked infinite hidden markov model based trajectory analysis and semantic region retrieval in a trajectory dataset. Expert Systems with Applications **78**, 386–395 (2017)

[58] Fuse, T., Kamiya, K.: Statistical anomaly detection in human dynamics monitoring using a hierarchical dirichlet process hidden markov model. IEEE Transactions on Intelligent Transportation Systems (2017)

[59] Pei, W., Dibeklioglu, H., Tax, D.M.J., van der Maaten, L.: Multivariate time-series classification using the hidden-unit logistic model. IEEE Transactions on Neural Networks and Learning Systems **99**,pp. 1–12 (2017)

[60] Pan, X., Wang, H., He, Y., Xiong, W., Jian, T.: Online classification of frequent behaviours based on multidimensional trajectories. IET Radar, Sonar & Navigation (2017)

[61] Stauffer, C., Eric, W., Grimson, L.: Learning patterns of activity using real-time tracking. IEEE Transactions on Pattern Analysis and Machine Intelligence **22**(8), 747–757 (2000)

[62] Kim, E., Helal, S., Cook, D.: Human activity recognition and pattern discovery. Pervasive Computing **9**(1), 48–53 (2010)

[63] Johnson, N., Hogg, D.: Learning the distribution of object trajectories for event recognition. Image and vision computing **14**(8), 609–615 (1996)

[64] Tao, X., Shaogang, G.: Video behaviour profiling and abnormality detection without manual labelling. In: International Conference on Computer Vision, vol. 2, pp. 1238–1245 (2005)

[65] Tran, T., Phung, D., Svetha, V., Bui, H.: Adaboost. mrf: Boosted markov random forests and application to multilevel activity recognition. In: IEEE Conference on Computer Vision and Pattern Recognition, vol. 2, pp. 1686–1693 (2006)

[66] Li, C., Han, Z., Ye, Q., Jiao, J.: Visual abnormal behavior detection based on trajectory sparse reconstruction analysis. Neurocomputing **119**, 94–100 (2013)

[67] Morris, B., Trivedi, M.: Learning and classification of trajectories in dynamic scenes: A general framework for live video analysis. In: International Conference on Advanced Video and Signal Based Surveillance, Proceedings of the, pp. 154–161 (2008)

[68] Athanassios, K., John, M., Paul, T.: Synchronization of batch trajectories using dynamic time warping. American Institute of Chemical Engineers Journal **44**(4), 864–875 (1998)

[69] Michail, V., George, K., Dimitrios, G.: Discovering similar multidimensional trajectories. In: International Conference on Data Engineering, pp. 673–684 (2002)

[70] Bashir, F.I., Khokhar, A.A., Schonfeld, D.: Object trajectory-based activity classification and recognition using hidden markov models. IEEE Transactions on Image Processing **16**(7) (2007)

[71] Li, J., Pedrycz, W., Jamal, I.: Multivariate time series anomaly detection: A framework of hidden markov models. Applied Soft Computing **60**, 229 – 240 (2017)

[72] Alvina, G., René, V.: Segmenting motions of different types by unsupervised manifold clustering. In: IEEE Conference on Computer Vision and Pattern Recognition, pp. 1–6 (2007)

[73] Lei, C., Raymond, N.: On the marriage of lp-norms and edit distance. In: Proceedings of the 30th International Conference on Very large data bases, vol. 30, pp. 792–803 (2004)

[74] Tasin, N., Andrea, C., Bernhard, R.: Trajectory clustering for motion pattern extraction in aerial videos. In: IEEE International Conference on Image Processing, pp. 1016–1020 (2014)

[75] Eamonn, K., Michael, P.: Scaling up dynamic time warping for datamining applications. In: Proceedings of the 6th ACM SIGKDD International Conference on Knowledge discovery and data mining, pp. 285–289 (2000)

[76] Wang, X., Ma, K., Ng, G., Grimson, W., Eric, L.: Trajectory analysis and semantic region modeling using nonparametric hierarchical bayesian models. International Journal of Computer Vision **95**(3), 287–312 (2011)

[77] Yoshua, B., Courville, A., Vincent, P.: Unsupervised feature learning and deep learning: A review and new perspectives. Computing Research Repository **1** (2012)

[78] Lasse, B., Harri, H., Timo, R.: A survey of longest common subsequence algorithms. In: 7th International Symposium on String Processing and Information Retrieval, pp. 39–48 (2000)

[79] Pavlovic, V., Frey, B.J., Huang, T.S.: Time-series classification using mixed-state dynamic bayesian networks. In: IEEE Conference on Computer Vision and Pattern Recognition, vol. 2, pp. 609–615 (1999)

[80] Owens, J., Hunter, A.: Application of the self-organising map to trajectory classification. In: IEEE International Workshop on Visual Surveillance, vol. 3, pp. 77–83 (2000)

[81] Foresti, G.L., Vanzella, W.: Generalized neural trees for outdoor scene understanding. In: International Conference on Image Processing, vol. 3, pp. 336–339 (2000)

[82] Wang, X., Tieu, K., Grimson, E.: Learning semantic scene models by trajectory analysis. In: Proceedings of the European Conference on Computer Vision, pp. 110–123 (2006)

[83] Xiang, T., Gong, S.: Beyond tracking: Modelling activity and understanding behaviour. International Journal of Computer Vision **67**(1), 21–51 (2006)

[84] Zhong, H., Shi, J., Visontai, M.: Detecting unusual activity in video. In: Proceedings of IEEE Conference on Computer Vision and Pattern Recognition, pp. 819–826 (2004)

[85] Li, H., Bull, D.R., Achim, A.: Unsupervised feature based abnormality detection. In: Sensor Signal Processing for Defence, pp. 1–5 (2010)

[86] Takahashi, M., Naemura, M., Fujii, M., Satoh, S.: Human action recognition in crowded surveillance video sequences by using features taken from key-point trajectories. In: IEEE International Conference on Computer Vision and Pattern Recognition, pp. 9–16 (2011)

[87] Huang, L., Barth, M.: Real-time multi-vehicle tracking based on feature detection and color probability model. In: IEEE Intelligent Vehicles Symposium, vol. 4, pp. 981–986 (2010)

[88] Salvi, G.: An automated nighttime vehicle counting and detection system for traffic surveillance. In: Proceedings of the International Conference on Computational Science and Computational Intelligence (2014)

[89] Lao, W., Han, J., n. De With, P.H.: Automatic video-based human motion analyzer for consumer surveillance system. IEEE Transactions on Consumer Electronics **55**(2), 591–598 (2009)

[90] Qiu, C., Xu, H., Liu, W.: Supervision on abnormal activities in vehicle inspection service by anomaly detection in bipartite graph. In: International Conference on Intelligent Transportation Engineering, pp. 68–71 (2016)

[91] Fradi, H., Dugelay, J.: Robust foreground segmentation using improved gaussian mixture model and optical flow. In: Proceedings of International Conference on Informatics, Electronics and Vision, pp. 248 – 253 (2012)

[92] Grisetti, G., Kummerle, R., Stachniss, C., Burgard, W.: A tutorial on graph-based slam. IEEE Intelligent Transportation Systems Magazine **2**(4), 31–43 (2010)

[93] Guo, D., Liu, S., Jin, H.: A graph-based approach to vehicle trajectory analysis. Journal of Location Based Services **4**(3-4), 183–199 (2010)

[94] Kim, J.J., Lee, J.J.: Trajectory optimization with particle swarm optimization for manipulator motion planning. IEEE Transactions on Industrial Informatics **11**(3), 620–631 (2015)

[95] Zhao, J., Zhou, R.: Particle swarm optimization applied to hypersonic reentry trajectories. Chinese Journal of Aeronautics **28**(3), 822 – 831 (2015)

[96] Lu, S., Zhao, J., Jiang, L., Liu, H.: Solving the time-jerk optimal trajectory planning problem of a robot using augmented lagrange constrained particle swarm optimization. Mathematical Problems in Engineering **2017**(1921479), 1–10 (2017)

[97] Lee, J., Han, J., Li, X., Cheng, H.: Mining discriminative patterns for classifying trajectories on road networks. IEEE Transactions on Knowledge and Data Engineering **23**(5), 713–726 (2011)

[98] Zhang, Y., Chen, Z., Wu, C., Jiang, J., Ran, B.: Vehicle trajectory analysis system via mutual information and sparse reconstruction. Transportation Research Record: Journal of the Transportation Research Board **2645**, 195–202 (2017)

[99] Liu, P., Kurt, A., Özgüner, Ü.: Trajectory prediction of a lane changing vehicle based on driver behavior estimation and classification. In: 17th International IEEE Conference on Intelligent Transportation Systems (ITSC), pp. 942–947 (2014)

[100] Coşar, S., Donatiello, G., Bogorny, V., Garate, C., Alvares, L., François, B.: Toward abnormal trajectory and event detection in video surveillance. IEEE Transactions on Circuits and Systems for Video Technology **27**(3), 683–695 (2017)

[101] González, D., Pérez, J., Milanés, V., Nashashibi, F.: A review of motion planning techniques for automated vehicles. IEEE Transactions on Intelligent Transportation Systems **17**(4), 1135–1145 (2016)

[102] David, D., Thomas, P.: Algorithms for the reduction of the number of points required to represent a digitized line or its caricature. Cartographica: International Journal for Geographic Information and Geovisualization **10**, 112 –122 (1973)

[103] Domínguez, R., Onieva, E., Alonso, J., Villagra, J., González, C.: Lidar based perception solution for autonomous vehicles. In: 11th International Conference on Intelligent Systems Design and Applications (ISDA), pp. 790–795 (2011)

[104] Guichang, S., Jaehwa, C., Yunsick, S.: 3d uav flying path optimization method based on the douglas-peucker algorithm. In: Advanced Multimedia and Ubiquitous Engineering, pp. 56–60 (2017)

[105] Yılmaz, D., Aydoğan, E., Özcan, U., İlkay, M.: A modified particle swarm optimization algorithm to mixed-model two-sided assembly line balancing. Journal of Intelligent Manufacturing **28**(1), 23–36 (2017)

[106] Jothi, R., Mohanty, S., Ojha, A.: Fast approximate minimum spanning tree based clustering algorithm. Neurocomputing **272**, 542–557 (2018)

[107] Senin, P.: Dynamic time warping algorithm review. Information and Computer Science Department University of Hawaii at Manoa Honolulu, USA **855**, 1–23 (2008)

[108] Vaughan, N., Gabrys, B.: Comparing and combining time series trajectories using dynamic time warping. Procedia Computer Science **96**, 465–474 (2016)

[109] Stauffer, M., Tschachtli, T., Fischer, A., Riesen, K.: A survey on applications of bipartite graph edit distance. In: International Workshop on Graph-Based Representations in Pattern Recognition, pp. 242–252. Springer (2017)

[110] Zhang, Y., Jiang, F., Rho, S., Liu, S., Zhao, D., Ji, R.: 3d object retrieval with multi-feature collaboration and bipartite graph matching. Neurocomputing **195**, 40–49 (2016)

[111] Weiming, H., Xi, L., Guodong, T., J., M.S., Zhongfei, Z.: An incremental dpmm-based method for trajectory clustering, modeling, and retrieval. IEEE Transactions on Pattern Analysis and Machine Intelligence **35**(5), 1051–1065 (2013)

[112] Morris, B., Trivedi, M.: Learning trajectory patterns by clustering: Experimental studies and comparative evaluation. In: Computer Vision and Pattern Recognition, pp. 312–319 (2009)

[113] Morris, B.T., Trivedi, M.M.: Trajectory learning for activity understanding: Unsupervised, multilevel, and long-term adaptive approach. IEEE Transactions on Pattern Analysis and Machine Intelligence **33**(11), 2287–2301 (2011)

[114] Mahi, M., Baykan, O., Kodaz, H.: A new hybrid method based on particle swarm optimization, ant colony optimization and 3-opt algorithms for traveling salesman problem. Applied Soft Computing **30**, 484–490 (2015)

[115] Ghamisi, P., Benediktsson, J.A.: Feature selection based on hybridization of genetic algorithm and particle swarm optimization. IEEE Geoscience and Remote Sensing Letters **12**(2), 309–313 (2015)

[116] Dinh, T., Vo, N., Medioni, G.: Context tracker: Exploring supporters and distracters in unconstrained environments. In: Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition, pp. 1177–1184 (2011)

[117] Dollár, P., Belongie, S., Rabaud, V.: Learning to traverse image manifolds. In: Advances in Neural Information Processing Systems 19, Proceedings of the 20th Annual Conference on Neural Information Processing Systems, pp. 361–368 (2006)

[118] Bradford, C., Dobkin, D., Huhdanpaa, H.: The quickhull algorithm for convex hulls. ACM Transactions on Mathematical Software **22**(4), 469–483 (1996)

[119] Das, N., Pramanik, S., Basu, S., Saha, P.K., Sarkar, R., Kundu, M., Nasipuri, M.: Recognition of handwritten bangla basic characters and digits using convex hull based feature set. Computing Research Repository (CoRR) abs/1410.0478 (2014)

[120] Das, N., Pramanik, S., Basu, S., Saha, P., Sarkar, R., Kundu, M.: Design of a novel convex hull based feature set for recognition of isolated handwritten roman numerals. Computing Research Repository (CoRR) abs/1501.05494 (2015)

[121] Rabiner, R.L.: Readings in speech recognition. chap. A Tutorial on Hidden Markov Models and Selected Applications in Speech Recognition, pp. 267–296 (1990)

[122] Kim, S., Smyth, P.: Segmental hidden markov models with random effects for waveform modeling. Journal of Machine Learning Research **7**, 945 – 969 (2006)

[123] Ostendorf, M., Digalakis, V., Kimball, O.A.: From hmm's to segment models: A unified view of stochastic modeling for speech recognition. IEEE Transections on Speech Audio Processing **4**, 360-378 (1996)

[124] Russell, M.J., Holmes, W.J.: Linear trajectory segmental hmms. IEEE Signal Processing Letters **4**, 72 – 74 (1997)

[125] Goldberg, D.E.: Genetic Algorithms in Search, Optimization and Machine Learning, 1st edn. Addison-Wesley Longman Publishing Co., Inc. (1989)

[126] Xiaogang, W., Keng, T., Gee-Wah, N., Grimson, W.: Trajectory analysis and semantic region modeling using a nonparametric bayesian model. In: Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition, pp. 1–8 (2008)

[127] Comaniciu, D., Meer, P.: Mean shift: A robust approach toward feature space analysis. IEEE Transactions on Pattern Analysis and Machine Intelligence **24**(5), 603–619 (2002)

[128] Dorigo, M., Birattari, M., Stutzle, T.: Ant colony optimization. IEEE Computational Intelligence Magazine **1**(4), 28–39 (2006)

[129] Jovanovic, R., Tuba, M., Voss, S.: An ant colony optimization algorithm for partitioning graphs with supply and demand. Applied Soft Computing **41**, 317–330 (2016)

[130] Toker, K., Kucuk, S., Catalbas, M.: Online signature recognition. In: 22nd Signal Processing and Communications Applications Conference (SIU), pp. 1754–1757 (2014)

[131] Zhang, B.: Off-line signature verification and identification by pyramid histogram of oriented gradients. International Journal of Intelligent Computing and Cybernetics **3**(4), 611–630 (2010)

[132] Yeung, D.Y., Chang, H., Xiong, Y., George, S., Kashi, R., Matsumoto, T., Rigoll, G.: SVC2004: 1st International Signature Verification Competition, pp. 16–22. Springer Berlin Heidelberg (2004)

[133] Kumar, P., Saini, R., Roy, P.P., Dogra, D.P.: Study of text segmentation and recognition using leap motion sensor. IEEE Sensors Journal **17**(5), 1293–1301 (2017)

[134] Piciarelli, C., Foresti, G.: On-line trajectory clustering for anomalous events detection. Pattern Recognition Letters **27**(15), Vision for Crime Detection and Prevention 1835 – 1842 (2006).

[135] Dogra, D., Ahmed, A., Bhaskar, H.: Interest area localization using trajectory analysis in surveillance scenes. In: 10th International Conference on Computer Vision Theory and Applications, Proceedings of the, pp. 31–38 (2015)

[136] Sugaya, Y., Kanatani, K.: Outlier removal for motion tracking by subspace separation. IEICE Transactions on Information and Systems **86**, 1095–1102 (2003)

[137] K. Santosh, B. Lamiroy, and J.-P. Ropers. Inductive logic programming for symbol recognition. In *Tenth International Conference on Document Analysis and Recognition-ICDAR'2009*, pages 1330–1334. IEEE, (2009).

[138] K. C. Santosh, L. Wendling, S. Antani, and G. R. Thoma. Overlaid arrow detection for labeling regions of interest in biomedical images. *IEEE Intelligent Systems*, 31(3):66–75, (2016).

[139] Graves, A., Liwicki, M., Fernández, S., Bertolami, R., Bunke, H., Schmidhuber, J.: A novel connectionist system for unconstrained handwriting recognition. IEEE Transactions on Pattern Analysis and Machine Intelligence **31**(5), 855–868 (2009)

[140] Graves, A., Schmidhuber, J.: Framewise phoneme classification with bidirectional LSTM and other neural network architectures. Neural Networks **18**(5), 602–610 (2005)

[141] Kumar, P., Gauba, H., Roy, P.P., Dogra, D.P.: A multimodal framework for sensor based sign language recognition. Neurocomputing (2016)

[142] O. Chapelle, V. Vapnik, O. Bousquet, and S. Mukherjee. Choosing multiple parameters for support vector machines. *Machine learning*, 46(1-3):131–159, 2002.

[143] Breiman, L.: Random forests. Machine Learning 45(1), 5–32 (2001)

[144] R. Saini, P. Kumar, P. P. Roy, and D. P. Dogra. An efficient approach for trajectory classification using fcm and svm. In *2017 IEEE Region 10 Symposium (TENSYMP)*, pages 1–4, (2017).

[145] K. Santosh and P. P. Roy. Arrow detection in biomedical images using sequential classifier. *International Journal of Machine Learning and Cybernetics*, 9(6):993–1006, (2018).

[146] K. Santosh, N. Alam, P. P. Roy, L. Wendling, S. Antani, and G. R. Thoma. A simple and efficient arrowhead detection technique in biomedical images. *International Journal of Pattern Recognition and Artificial Intelligence*, 30(05):1657002, (2016).

Achievements and Publications on the Research Work

Achievements

1. Received travel support to present paper in Asian Conference on Pattern Recognition, Nanjing, China from ACPR 2017 organizers, Nanjing, China.
2. Attended Internship with Prof. Umapada Pal at Computer Vision & Pattern Recognition (CVPR) Unit of Indian Statistical Institute (ISI) Kolkata, India during April, 2017.

International Peer-Reviewed Journals

1. Rajkumar Saini, Partha Pratim Roy, and Debi Prosad Dogra. "A segmental HMM based trajectory classification using genetic algorithm." Expert Systems with Applications, Elsevier, 93 (2018): 169-181.
2. Rajkumar Saini, Pradeep Kumar, Partha Pratim Roy, and Umapada Pal. "Modeling Local and Global Behavior for Trajectory Classification using Graph Based Algorithm." Pattern Recognition Letters, Elsevier, 2018. (Minor revision).

International Conferences

1. Rajkumar Saini, Pradeep Kumar, Saikat Dutta, Partha Pratim Roy, and Umapada Pal. "Local Behavior Analysis for Trajectory Classification using Graph Embedding." In: Asian Conference on Pattern Recognition (ACPR), 2017.
2. Rajkumar Saini, Arif Ahmed, Debi Prosad Dogra, and Partha Pratim Roy. "Surveillance Scene Segmentation Based on Trajectory Classification Using Supervised Learning." In: International Conference on Computer Vision and Image Processing (CVIP), 2016.