A HMM-BASED PREDICTION MODEL FOR SPATIO-TEMPORAL TRAJECTORIES

by

SAKTHI KUMARAN SHANMUGANATHAN

Presented to the Faculty of the Graduate School of
The University of Texas at Arlington in Partial Fulfillment
of the Requirements
for the Degree of

MASTER OF SCIENCE IN COMPUTER SCIENCE

THE UNIVERSITY OF TEXAS AT ARLINGTON
DECEMBER 2014

Acknowledgements

I would thank Dr. Ramez Elmasri for his support and encouragement throughout my Master's. I sincerely thank him for trusting me and providing opportunities to prove my skills.

I thank Dr. Manfred Huber for his guidance, inspiration and motivation which helped in completing my research. I appreciate his knowledge and skills, and his assistance in writing my thesis. It would have been impossible to complete this thesis without his efforts and time.

I thank Dr. Kulsawasd Jitkajornwanich and Upa Gupta for their co-operation and involvement in a research work which is the base for this thesis.

November 14, 2014

Abstract

A HMM-BASED PREDICTION MODEL

FOR SPATIO-TEMPORAL

TRAJECTORIES

SAKTHI KUMARAN SHANMUGANATHAN, M.S.

The University of Texas at Arlington, 2014

Supervising Professor: Ramez Elmasri

Spatio-temporal trajectories are time series data that represent movement of an object over the time. As these data can represent a large number of phenomena, including weather and temperature pattern, providing them with the ability to model and predict future is an important task. Hidden Markov Models (HMM), a variant of Markov Models (MM), were first applied at a large scale to speech recognition but have also been used in time series prediction by analyzing trends in historical time series data. In this research, we propose a storm prediction model using a HMM built from overall storm trajectories derived from raw rainfall data. This HMM is built by assuming the states are associated with clusters created by clustering the locations of each storm from the overall storm trajectories. Then we learn the variation of transitional information and spatial information present in the given set of trajectories using the Baum-Welch algorithm. This learning is performed by building a HMM that for each cluster contains multiple state instances that represent this cluster and can learn to reflect variations in the information within a cluster. Results from experiments show that the prediction gets better when the number of state instances representing each cluster increases. For example, the average distance value between actual location and location predicted by a model with 5 clusters

and 5 state instances per cluster is approximately 15% smaller than the average distance value between actual location and location predicted by a model with 5 clusters and 3 state instances per cluster, which means that the predicted location gets closer to the actual location with more state instances. It has also been found that the prediction gets better when the number of clusters increases. Apart from introducing a new prediction model for this type of data, we also propose a modified algorithm that creates overall storm trajectories much faster than existing algorithm.

Table of Contents

List of Illustrations

List of Tables

Chapter 1

Introduction

In recent years, due to advancement in technology, there have been large volumes of data generated and used in various fields. These data are recorded at a single instant and provide instantaneous information of an entity .By ordering these data along with the time stamp we get a time sequence. By using the time sequence from the past as historical temporal behavior of an entity we can predict the future behavior of an entity. For example rainfall precipitation data measured over time can be used for predicting future rainfall occurrences in a region, which would help us in preventing floods. Hence predicting the future could help further the welfare of human beings as we can take the necessary precautions before any destruction occurs. Even though most of the predictions are not accurate they are still considered useful as they give a chance to make plans for the future. The accuracy of the prediction depends on the amount of information provided in the data. For example, rainfall occurrences can be predicted well if the wind speed and air pressure data are available in addition to the rainfall precipitation data. Since it is difficult collect all the possible information the prediction models provide probability for all possible outcomes. This would help us developing contingency strategies for future.

The data sequences used as historical data for predicting the future is called time series data. In this research we use a more specific type of time series data called spatio-temporal data which represents movement of an entity over time. We build a model for predicting the movement of a rainfall storm which can be used for rainfall forecasting. Even though there have been other prediction models, we use an approach that can provide better predictions by learning the transitional and spatial variations in the given set of trajectories.

Time series data represents the change of an entity over time. An example of time series data is the stock market price in which the price of a stock varies over time. It should be noted that time series data can be created in two approaches: by recording the data at regular specified time intervals or recording it only when there is a change in the data. The former approach provides only partial information as the data could have changed more than once within the specified time interval while the later approach provides complete behavior of the data. However, the later approach requires constant monitoring of the changes which is complicated to implement whereas the former approach requires only regular monitoring of the data, which can be easily implemented by using an automated timer based recording device.

A specific type of time series data is spatio-temporal data which represents the change in spatial location over time. These data can also be considered as a time sequence of location of an object. A common example of spatio-temporal data is the GPS navigation data generated from cell phones or car GPS devices, which provides the location of the object at any given instant. Each location is associated with a timestamp at which the current location of the object was recorded. If these recorded data are sorted based on the time stamp then they can represent the movement of the object. Hence they are considered as spatio-temporal data. Usually spatial locations are represented using 2 different representations; a 1-dimensional point representation which can be an identification given to the location such as site Id, station id or region name, then a 2 dimensional point that are given by a co-ordinate system such as latitude and longitude or any user defined system such as HRAP [1] [2] used in this paper.

The movement of an object is given using a trajectory. Hence we can consider the spatio-temporal data as trajectories. Even though the trajectories are created by ordering the absolute time of the data, this absolute time is not required once a trajectory

is created. This means the trajectories are created by considering the location at any given time stamp as the starting location of the trajectory and then following it sequentially with the locations of the consecutive time stamps and ending with the location at a future time. Hence a single object can have more than one trajectory, each starting at different time. For example, a trajectory of a person can represents his movement on a single day and can be recorded for a week. This will give us 7 trajectories starting with different time stamps. The length of the trajectory is the number of points in the trajectory which are measured at regular intervals and is often given in time units such as seconds, minutes or hours depending on the units of the time interval between each point in the trajectory.

As a spatio-temporal trajectory provides the movement information of an object and each object can have more than one trajectory, we can obtain the possible historical movement of the object by creating the trajectories from the data recorded in the past. In our research we build a prediction model for the storm trajectories created from the historical rainfall data which can be used for predicting the location of a rainfall storm. The historical rainfall data is converted into set of rainfall storm trajectories where each storm trajectory represents movement of rainfall storm over location. These storm trajectories were created by considering the rainfall in locations. Hence if there is no rainfall in a location then this location would not be part of any of the trajectories. Hence the model can predict the next possible location at which rainfall will occur given a sequence of locations at which rainfall has occurred so far.

Apart from predicting spatial location of storms we have also used this approach for predicting direction of the storms. Also this approach can be extended to predict other non-spatial parameters such as speed or any other application specific parameters. The main aim of the research is to propose a prediction model for time series data by using a

learning approach that provides a better representation of transitional and spatial variations.

.

Chapter 2

Related Works

This chapter discusses about the existing time series prediction models, describes various applications of Hidden Markov Models and Markov Models along with their contribution to spatio-temporal data, and explains the adaptations of the current work from previous work.

Time series prediction has been an important research over the years. Artificial Neural Networks (ANN) [3] , Hidden Markov Models (HMM) [4] , and Autoregressive Models (AR) [5] are among the most commonly used prediction models. These models predict the future based on the statistical or probabilistic information obtained from historical time series data. The application of time series predictions extends to various fields such as weather forecasting, stock market price prediction [6], object movement prediction [7], network traffic prediction [8], rainfall prediction [9] [10] [11] and so on. The works of Patrick et. al. [11] and Nugroho et. al. [10] use variants of Autoregressive models, such as Autoregressive Integrated Moving Average (ARIMA) and Autoregressive Moving Average (ARMA), respectively, for predicting rainfall precipitation over time in a given region. Both these models consist of 2 parts: Autoregressive part- given a time series, the current value of the series is linearly dependent on previous values in the series with a constant error; Moving average part - given a time series, the current value of the series is a mean of the previous values in the series. These models are used for predicting future values in a stationary time series (common variations between series value). Both authors, Patrick et. al. [11] and Nugroho et. al. [10], use rainfall precipitation time series data recorded in a region for predicting future rainfall precipitation in the same region using the ARMA model. However, Patrick et. al. [11] uses a modified ARMA model called ARIMA, which performs an extra step for converting a non-stationary time series

into a stationary time series. Ultimately our research performs prediction for time series data similar to these works. However we focus on performing probabilistic prediction while both these models use linear statistics, such as mean and variance between consecutive values of the series, for calculating the future rainfall precipitation. Abhishek et. al. [9] developed a model for predicting rainfall precipitation using ANN, which is more accurate in predicting rainfall, as neural networks use frequencies for prediction rather than predicting future rainfall precipitation from the linear statistics. In this work a 3-layered feed forward artificial neural network is used for analyzing trends in temporal variations between humidity and wind speed in order to predict the amount of rainfall precipitation in a region. The 3 layers are input layer, output layer and hidden layer. The input layer consists of 2 variables humidity and wind speed, and the output layer is the future rainfall precipitation value. The middle layer is the hidden layer which learns the relationship between variations in humidity and wind speed and rainfall precipitation. This approach predicts rainfall by using the assumption that rainfall storms occur due to temporal variations in humidity and wind speed. Our approach is similar to this approach as we learn the trends in temporal variations of rainfall storm location data to predict future storm location.

In our research we use Markov Models [12] and its variant Hidden Markov Model [4] for predicting time series data. A Markov Chain is a Markov Model from a single time sequence. An application of a Markov Chain is the page rank algorithm of Google [13] which is used for ranking the web pages based on the number of incoming page links. However, this perspective of page rank is applicable only if the internet web is considered as time series data of pages. Hidden Markov Models have been used in various fields. Rabiner et. al. [4] proposed an approach using HMM for speech recognition, in which words were predicted using the audio voice signal. This approach considers speech as

time series data of words or phonemes. The thesis work of Hassan et. al. [6] discusses the application of HMM for predicting stock price change over time using historical stock price variations. Also, in the field of image recognition Yamato [14] has used HMMs for predicting images from a time series of images. Another interesting work was done by Krishner [15] in which a hybrid of HMM [4] and AR [3] model called AR-HMM was created for predicting rainfall precipitation in different locations. It uses the AR model built from precipitation data recorded in each site combined with an HMM to create a network of dependent site precipitation variable, and then uses the change in precipitation value to predict the future precipitation amount and its corresponding site id.

The above mentioned models are used for predicting temporal change of an entity. A time series is a time sequence of entity values. For example, in stock price prediction [6] the entity is the price of each stock, and in the work of Krishner [15] the entity consist of 2 dimensions; sites and precipitation. Hence, this work focuses on building HMMs for multi-dimensional observations. Our research focuses on building prediction models for spatio-temporal data which represents the temporal change in location of objects and can be considered as time series data with the entity being the location value in a geographical space. In other work Jeung et. al. [16] use HMMs for mining patters from object movement trajectories. In this work, the authors divide the given space into a grid and then use HMMs for identifying common movement patterns of each object by finding the frequent regions visited by the object. Even though this work is similar to our approach as it predicts the most frequent regions, in this work object movements are constrained within the grid system. The work by Gambs et. al. [7] uses Markov Chains for predicting an individual person movement using GPS data. This work is similar to our approach using Markov Models, except that the entities are location names such as “Home”, “Work” and “Others”, while we use a set of spatial clusters. In a

Prasad et. al. [17], has used HMMs for predicting the Access Point (AP) to which a current mobile user is connected to, given the location information of the APs and time series data of the geographical location of users. This problem is similar to our HMM approach, however we are more interested in predicting the exact locations while this model is interested in predicting the AP using the location information. The work of Lu et. al. [18] uses an AR-HMM similar to Krishner [15] for learning the temporal variation of various features of social websites and performs a wide variety of applications such as location prediction, friendship circle identification and place recommendations. Similarly our approach learns the spatio-temporal features of the rainfall storm data but restricts the focuses on location prediction. Another work similar to our research is by Mathew et. al. [19] in which the authors perform animal movement prediction, by building a HMM using Geo life data. In his approach Mathew et. al. [19] has divided the space into a triangular mesh and performs learning by considering one trajectory at a time. In contrast, our work uses clusters instead of a mesh and learns the transitional information between clusters from the given set of trajectories. Mathew et. al. [19] predicts the next possible location as the center of the triangular mesh while the HMM in our approach predicts a more precise location.

The works of Mathew et. al. [19], Krishner [15] learn the transitional information present in the trajectories using Baum-Welch [4] but do not learn the spatial information present in the trajectories as they use discrete HMMs. In our approach, since we use clusters of points, spatial information in the clusters are learned by considering clusters as a Gaussian distribution of the spatial points. In his work, Paul [20] discusses the Expectation and Maximization step used for the Baum-Welch algorithm with a Gaussian distribution. Also, in the work by Mitra et. al. [21], the author uses a similar approach using the Baum-Welch algorithm with Gaussian mixture to perform regime calibration. In

our approach we use the same learning algorithm only with a Gaussian distribution rather than a Gaussian mixture. In his work, Paul [20] also explains restricted HMM network topologies that are similar to the methods used in works for speech recognition [22] [23] [24]. These works perform phoneme segmentation by considering a single HMM for each phone. Later Daines et. al. [25] proposed a constrained Baum-Welch algorithm used for improving phoneme segmentation by restricting the phoneme segments to a certain set of phones. This is achieved by having a binary relation between phone and phonetic segments. Similar to this approach we restrict each point of clusters only to the representations the clusters. However, each cluster can have more than one representation so that t the transitional and spatial variations present in the trajectories are represented much better.

This research focuses on predicting the movement of a storm using overall storm trajectories [1] that were defined based on hydrology concepts given by Jitkajornwanich et. al. in the works [1] [26] [27]. These hydrology based storms are discussed in detail in Chapter 3. It should be noted that the previous works [7] [15] [19] use a discrete number of locations and can only perform approximate location predictions. Even though the raw rainfall data used in our research are measured in stations which are represented using Site Id, these stations can be converted into a specific grid co-ordinate using mathematical formulae which will be described later. These coordinates can be further can be mapped to latitude and longitude. Hence in this approach we predict precise location instead of stations.

Chapter 3

Extracting Overall Storm Trajectories

This chapter provides an overview on works of Jitkajornwanich et. al. [1] [26] [27] to convert raw rainfall data into meaningful storm trajectories. In this chapter we describe the raw rainfall data used, define basic hydrology-related storm concepts and explain the procedure to derive hydrology-related storms.

The raw rainfall data that we use comes from the National Weather Service (NWS) -West Gulf River Forecast Center (WGRFC) [28] [29], a weather reporting organization and provides the rainfall data for the regions covering Texas, Lousiana, New Mexico, Colorado and part of Mexico. The regions are divided into 165,750 sites as a 425x390 grid with sites approximately 4km apart from each other. The rainfall data are given in a file that contains the following fields: observation time, row id, site id, precipitation value. The precipitation value is recorded as the Multi-sensor Precipitation Estimates (MPE) [28] [29] value of sites, indicated by their siteID in the hour represented by observation time, which consists of 2 time stamps. Each data file contains the rainfall data of a single hour. The site id can be converted into a coordinate called HRAP(Hydrologic Rainfall Analysis Project) [28] [29] [2] using Equations (1) and (2).

$$\mathrm{X} = \mathrm{a} + \big((\mathrm{siteID} - \mathrm{c}) \bmod \mathrm{d}\big) \qquad (1)$$

$$\mathrm{Y} = \mathrm{b} + \big((\mathrm{siteID} - \mathrm{c}) \operatorname{div} \mathrm{d}\big) \qquad (2)$$

Here, a is the minimum X coordinate value, b is the minimum Y coordinate value, c is the first siteID, and d is the difference between siteIDs in adjacent rows in the same column. The HRAP coordinate system considers values a, b, c, and d, of 290, 10, 15599, and 1701, respectively. Further, these HRAP coordinates can be converted into latitude and longitude by using the procedure routine provided by NWS [28] [29].

These raw rainfall data are converted into 3 different hydrology-related storm concepts defined below based on the informal definitions coined by Jitkajornwanich et. al [1]. All these storms can be derived from raw rainfall data using 2 approaches: a Depth First Search (DFS) [27] approach applied on database schema called CUHASI (Consortium of Universities for the Advancement of Hydrologic Science, Inc.) [27] [29]; a Map-Reduce approach [1] [30] which is much faster compared to the DFS [27] approach. Hence we will be using only the Map-Reduce approach which performs parallel computation using Key-Value pairs. Apache Hadoop [31], an implementation of Map-Reduce framework [30] , consists of the following components: mapper and reducer, grouping comparator, sorting comparator, combiner and partitioner.

Local Storm

A new local storm occurs in a site when there are *h* consecutive hours with zero precipitation for the same site. Here *h* is the inter event time [32] [1] [26] and defines the number of zero precipitation hours to consider a separate local storm. If a site has *h* consecutive hours of zero precipitation then a single local storm is considered for the hours till the last non-zero precipitation before it, and a new local storm is considered starting from the next non-zero precipitation hour. A typical value of *h* is 6 hours. This is a site-specific storm and a site can have more than one local storm. The local storm defines the temporal characteristics of rainfall storms in a single location. The algorithm using Map-Reduce approach for local storm identification is provided in the work of Jitkajornwanich et al. [26].

Hourly Storm

A new hourly storm occurs in an hour when *n* neighboring sites have zero precipitation in the same hour. Here *n* is the space tolerance [27] and defines the minimum number of neighboring sites with zero precipitation after which a new hourly

storm is created. If a site has non-zero precipitation in an hour, then an hourly storm is considered for those sites that have non-zero precipitation and are separated by at most *n* neighboring sites with zero precipitation. Then a new hourly storm is considered for those sites with non-zero precipitation and are away by more than n neighboring sites with zero precipitation from this current site. Neighboring sites are the sites that are adjacent to a site in the HRAP grid. If a HRAP co-ordinate of a site is given as (X,Y) then its neighboring sites are given by co-ordinates: (X-1, Y), (X, Y-1), (X-1, Y-1), (X+1, Y), (X, Y+1), (X+1, Y+1), (X-1, Y+1) and (X+1, Y-1). An hour can have more than one hourly storm but hourly storms in the same hour should not have common sites.

The typical value of *n* is 0 which means hourly storms are considered only when the neighbors have non-zero precipitation. This is a time-specific storm and an hour can have any number of hourly storms. The hourly storms define the spatial characteristic of rainfall storms in an hour.

An hourly storm consists of parameters such as:

Storm coverage: Total number of sites covered by an hourly storm.

Storm sites total: Sum of precipitation of the sites covered by an hourly storm.

Storm average: The fraction of precipitation per site. Calculated by diving storm sites total by storm coverage

Storm center: The HRAP co-ordinate of the site in an hourly that as maximum precipitation. If 2 or more sites have same precipitation then mean of the HRAP co-ordinates are taken.

Storm centroid: The mean of the HRAP co-ordinates of the sites in an hourly storm.

Storm boundary: The minimum bounding rectangle that can be formed using the given set of sites in an hourly storm. This consists of 2 co-ordinates: lowest values of x and y and highest values of x and y, of HRAP co-ordinates of the sites in an hourly storm.

The algorithm using the Map-Reduce approach for hourly storm identification is provided in the work of Jitkajornwanich et al. [26].

This approach uses only mappers and no reducers. Hence the number of reduce tasks is set to zero. Each map task is assigned a single hourly file of raw rainfall data and outputs a file that consist of all the hourly storms computed for that hour. Hence for a non-leap year there are 8760 mappers while for leap year there are 8784 mappers. Each hourly storm is assigned an hourly storm id value. The mapper emits the hourly storm id as key, and the time stamp of the hour along with the hourly storm parameters defined above as value. A detailed description of various case scenarios for processing the hourly storm are provided by Jitkajornwanich et. al. [26].

Overall Storm

An overall storm occurs when hourly storms within *g* hours overlap spatially by sharing *s* common sites. Here *g* is the grouping window [1] [27] which provides the minimum number of hours of overlap between 2 hourly storms to be considered merged to form a single overall storm, and, *s* is the spatial window [1] [27] which provides the minimum number of common sites shared between 2 hourly storm in order to be considered to be merged into a single overall storm. Hence an overall storm is created by merging one or more hourly storms. The typical value of *s* and *g* are 1 site and 1 hour respectively. An overall storm can be considered as a rainfall trajectory as it defines the spatial movements of rainfall over time. Hence it provides the spatio-temporal

characteristics of rainfall storms. An overall storm has various parameters defined in works by Jitkajornwanich et al. [1] [27]:

Storm coverage: Sum of the storm coverage of all hourly storm that are merged

Storm overall depth: Sum of the storm sites total of each hourly storm of every hour in an overall storm.

Storm overall intensity: Average precipitation per hour of an overall storm. This is calculated by dividing the storm overall depth by number of hour the storm lasted.

Storm overall average: Average precipitation per site of an overall storm. This is calculated by dividing the storm overall depth by storm coverage of overall storm.

*Storm path: S*equence of HRAP co-ordinates of storm centers or storm centroids formed by merging two or more hourly storms. If an hourly storm in a trajectory overlaps with more than one hourly storm in next hour then the average of the storm centers or storm centroids of the hourly storms are calculated and included in the storm path.

The overall storm identification algorithm using the Map-reduce approach is explained by Jitkajornwanich et. al. [1] with s=1 sites and g=1 hour values. The mapper reads the hourly storm data and merges the hourly storms iteratively. In the first iteration the mapper reads the hourly storm values and calculates the hour id for each hourly storm. The hour id is generated by considering hour id of 1st hour of the year as 0 and increasing the hour id value by 1 for each hour. This hour id value is emitted as key by the mapper while the parameters of the hourly storms are emitted as the value. The partitioner sends the 2 consecutive hours id to the same reducer. The reducer uses a data structure called OStorm that is used for merging the hourly storms and creates partial overall storms. The reducer emits (hour id/2) as key while the parameters of the

partial overall storms are emitted as value. In the next iteration the mapper will read the partial overall storm and emits the hour id of the partial overall storm as key and the partial overall storm parameters as value. Then the partial overall storm of consecutive hours is merged by the reducer. The mappers and reducers are iteratively repeated until all the hours are compared with each other. Since there are at most 8784 hours and in each iteration 2 consecutive hours are merged we would overall require $\log_2 8784 \approx 14$ iterations.

In the method proposed by Jitkajornwanich et. al. [1] the number of mappers required in the first iteration is 8760 or 8784 and it reduces logarithmically. Similarly the number of reducers for the first iteration is 4392 and reduces logarithmically. Even though the number of reducer tasks or mapper tasks decreases in each iteration there is always an overhead for creating tasks such as restarting JVM, updating job tracker etc. which will increase the computation time. Hence by making the number of tasks constant and reusing the existing task for different key groups we can improve the computation time of overall storm identification.

The modified algorithm which process overall storm trajectories faster than existing approach is given in Algorithm 1. In this algorithm we reuse the existing map tasks by using a combination of (hour id/2) and hour id as key, and hour id and partial overall storm as value. The sorting comparator sorts the output from the mapper, based on the (hour id/2) key and the grouping comparator performs a secondary sort using the hour id key. The partitioner then sends all the records with (hour id/2) key to the same reducer. In the reducer two consecutive hours are identified using the hour id present as part of the composite value record. The number of reduce tasks is kept constant. The time taken for computing overall storms with 70 reduce tasks was approximately 45

minutes for a year and is better than the 3 hours taken by the earlier implementation. The algorithm uses the following. Functions:

MarkAsFinal(cos) : Marks the overall storm *cos* as final so that they will not be compared further.

IsFinal(cos) : Checks if an overall storm cos is final or not. A storm is final when it cannot grow further.

cos.IsComparable(*os*) : Checks if the current storm *cos* is comparable with a given storm *os*. A storm *os* can be compared with *cos* if *os* starts in an hour consecutive to the ending hour of cos.

cos.IsOverlap(os) : Checks if the current storm *cos* overlaps with the given storm *os* The storm *cos* overlaps with *os* when they share a common site in consecutive hours.

Even though the new implementation speeds up the process, it can cause a node to run out of memory (JAVA Heap Space) when the total size of records processed by all the hour pairs in a node is greater than the RAM size, as the JVM is not restarted for each hour pair. However, this situation is very unlikely to occur as the overall storm identification is executed year by year and usually process less than 500 hourly storms in an hour.

It should be noted that these overall storms contain the spatial HRAP coordinates that represent the location of an hourly storm in each hour. Hence by considering these HRAP coordinates as spatial parameters we get spatio-temporal trajectories that represent the rainfall storm movement.

Algorithm 1: Efficient Map-Reduce-based Overall Storm Identification

```
    Input:
      - hourly storm data text files
    Output:
      - overall storm data text file
class MAPPER:
function MAP(key object, value line):
    if iteration i = 1://value is provided by the main function that creates the job
        V= list of values in a line that are separated by space
        hid = CalHourId(V[0])
        hid1 = hid/2
        key = (hid1, hid)
        value = (hid, list of values of OStorm attributes)
        Emit(key, value)
    else:
            key = (hid1, hid)
        value = (hid, the remaining values, OStorm attributes)
        Emit(key, value)
end function:
end class:
class PARTITIONER:
function PARTITION(key (hid1,hid), value (hid, OStorm)):
    for each record:
            tid = floor(hid1)
            Emit (key, OStorm) to reducer task number tid
    end for:
end function:
end class:
class REDUCER:
function REDUCE(key (hid1,hid), [(hid, OStorm1), (hid, OStorm2), …}):
    for each key do:
        if hid1 remains unchanged then:          //when current call of reduce function has different hid1 value
                                                 from the previous call.
            for each (hid,os) ∈ [(hid,OStorm1), (hid, OStorm2), …}:
                if hid remains unchanged then:          //when current call of reduce function has different
                                                        hid value from the previous call.
                        aList.add(os)
                else:
                        for each storm cos ∈ aList
                        if cos.IsComparable(os) and cos.IsOverlap(os):
                                mos =cos.MergeStormOrder(os)
                                aList.Add(mos, cos.pos
                                CompareWithin(cos.pos)
                            else:
                                if IsFinal(os):
                                MarkAsFinal(os)
                                end if:
                            Emit((hid1/2,hid1),os), to HDFS as input for next iteration
                    end for:
            end if:
            end for:
        else:
            for each storm cos ∈ aList:
                    if IsFinal(cos) then
```

```
                    MarkAsFinal(cos)
                end if:
            Emit((hid1/2,hid1),os), to HDFS as input for next iteration
        end for
    end if:
end for:
end function:
edn class:
```

Thus the raw rainfall data recorded for each hour of each location is converted into a time series data. By considering spatial and time dimensions of a storm path, we get two different storm trajectories: one based on storm centers and another based on storm centroids. (Storm center considers the location with the highest precipitation for each hour.) Each point in a trajectory is a two-dimensional HRAP coordinate. The time interval between each point in a storm trajectory is one hour and the length of the trajectory is in hours.

Chapter 4

Building the Prediction Model

This chapter gives an overview of Markov Models (MM) and Hidden Markov Models (HMM), and, defines and discusses the procedure for building the prediction model along with the evaluation method used for testing the results.

Markov Models (or Markov Chains) and Hidden Markov Models are probabilistic models that are built assuming the Markov property. The Markov property states that, given a sequence of states, the probability of a next state depends only on the current state and not on any other previous states. Hence this property defines a memoryless property for creating the model. A state can represent any entity that changes over time. A MM is built using the state sequences in the training data while a HMM is built using observation sequences where the relationship between states and observations are given as part of the model. The Markov property can be extended by considering more than one previous state and the number of previous states considered is called as the "Order" of the Model. In this paper we use 1-Order Markov Models which means we consider only the previous state. Both models, HMM and MM, can be used to predict the movement of states by learning through a set of sequences that represent the historical transitions between states. The probabilities of the state transitions are learned from a set of sequences that represent the transitions between states. We will discuss more on the learning process and differences between these models and their advantages in more detail in the next few sections.

In this work, the overall storm trajectories are used as the historical sequences for building the model. Each point in an overall storm trajectory, tr, is an hourly storm (or hourly storms that are merged together) and is represented as a point, $P_{t,tr,}$. Here, t represents the relative time of the point in the trajectory tr of length n and can take values

from 1 to *n*. Each point $P_{t,tr}$ contains many attributes that define the state of the hourly storm at time t. Hence, by using these trajectories we can build a model for predicting the temporal state of the hourly storm. For now, we are interested only in the HRAP [2] [1] grid system co-ordinates that represent the location of the storm as a 2D point, (X_t , Y_t) that can be mapped to a corresponding latitude-longitude pair using the subroutines provided by NWS [28] [29]. The model built using this 2D-point can be used to predict the next possible location of the rainfall storm. However, this method can also be extended for predicting the temporal state of other attributes.

The attributes of a point are continuous-valued. Hence building a model with points as states would require an infinite state space. Therefore the state space has to be reduced to a finite number but with minimal loss of data. This is done by grouping similar points into a groups that can be used to approximately represent each point. In the next section, we discuss in more detail the grouping technique used for reducing the state space.

Clustering the Points

Clustering is an unsupervised learning technique that groups a given set of points based on the similarity of their features. Each group is called a cluster. A cluster consists of points that have similar feature values compared to points in the other clusters. This technique is unsupervised as the clustering algorithm is built only with the similarly metric of the features used for grouping, but not with associated cluster labels. After clustering, each point becomes a member of a single cluster. Hence a point can be represented using its cluster and thus a finite approximation of the continuous state space can be formed for building the model. The set of clusters created by clustering algorithm is called a cluster group. The cluster group size (CGS) represents the number

of clusters in a cluster group. The cluster size (CS) represents the number of points in each cluster.

Even though there are many clustering algorithms available, we use the hierarchical clustering algorithm [33] to cluster the points. The main advantage of using hierarchical clustering is that it is much faster than EM-Clustering [34] or K-means [35] clustering which consume more time for convergence. Moreover, hierarchal clustering does not consider any points as outliers, which are ignored by density based clustering algorithms such as DBSCAN [36] or OPTICS [37]. In hierarchical clustering the resulting clusters are represented as dendrograms, which are trees of clusters in which each node represents a cluster and each level of the tree contains new clusters created by merging 2 clusters in the previous level. Hence each level will have one cluster more or less than the previous level depending on the type of hierarchical clustering. This provides us flexibility of choosing cluster groups of different cluster group size in a single clustering process.

There are 2 types of hierarchical clustering; Agglomerative and Divisive. The Divisive technique is a top-down approach in which all the points are part of a single cluster at the start and are then divided into the desired number of clusters iteratively. Agglomerative clustering is a bottom-up approach in which all points are considered as individual clusters at the start and then the most similar cluster pairs are combined in each iteration until the desired number of clusters is reached. In this work, we use Agglomerative clustering since it is the most commonly used technique and also simple to implement. An example dendrogram for Agglomerative clustering is shown in Figure 4.1.

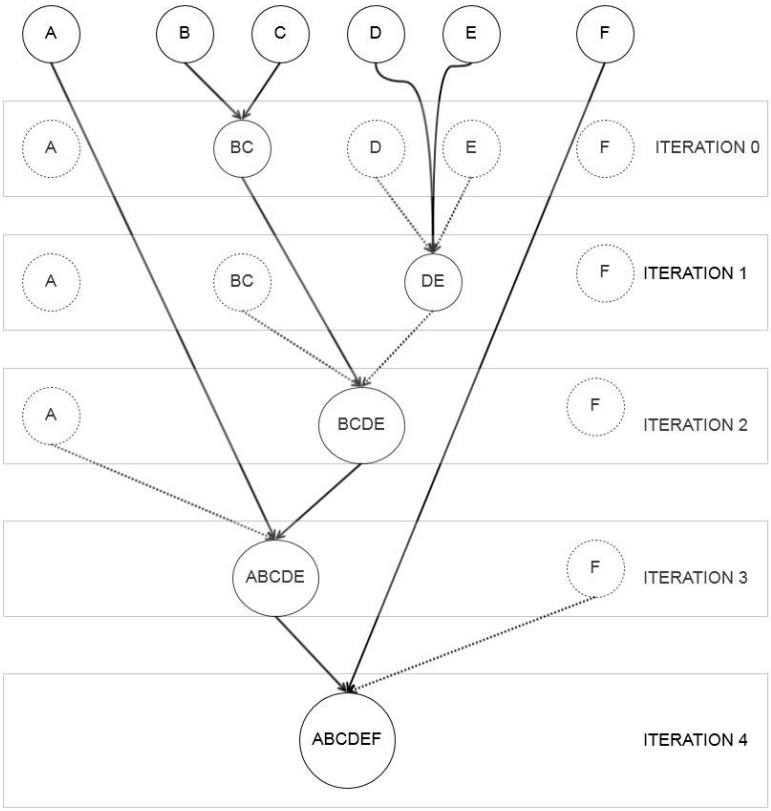


Figure 4.1 An example of a Hierarchical clustering process and dendrogram

This figure illustrates the process of hierarchical clustering for 5 points A, B, C, D, E and F. In this figure solid circles represent the new clusters and solid lines represent the links between the new clusters. Hence solid lines and solid circles form the dendrograms. The dotted circles represent old clusters, i.e. clusters from previous iterations, and dotted lines represent the link between the old clusters and the new cluster. At the beginning of the process all points are considered as individual clusters. At iteration 0(1st iteration) the 2 clusters B and C are merged to form a new cluster, represent as solid circle BC. Hence a solid line is drawn from B and C to BC. In the next iteration clusters D and E are merged to get new cluster DE. Hence a solid line is drawn

from the clusters D and E present before iteration 0 to new cluster DE and a dotted line is drawn from clusters D and E present at iteration 0. The dotted lines represent the iteration at which the clusters merged while solid line represents the link between clusters in the dendrogram. Similarly the other clusters are merged until we get only one cluster. Now the cluster group can be chosen from the results of each iteration. For example if the cluster group is chosen from iteration 1 then we get 4 clusters A, BC, DE, F. Hence we can choose a cluster group with the required number of clusters from the iteration of the process.

In hierarchical clustering each cluster is considered to be linked with the others. The similarity measure between two clusters in agglomerative clustering is called linkage. There are 3 types of linkages: single linkage - the nearest distance between 2 clusters; complete linkage - the farthest distance between 2 clusters; average linkage - average distance between 2 clusters. We use the average linkage criterion as it provides an easier metric used for evaluating the clusters. The average linkage measured between a cluster pair is provided by Equation (3)

$$d_C(C_i, C_j) = \frac{1}{(\,|C_i| \times |C_j|\,)} \times \sum_{L_i \in C_i} \sum_{L_j \in C_j} d_p(L_i, L_j) \qquad (3)$$

Here C_i and C_j are the 2 clusters with size | C_i | and | C_j |, and, L_i and L_j are the points that are members of cluster C_i and C_j. The function $d_p(L_i, L_j)$ represents the distance between 2 points L_i and L_j. The point L_i is represented as (X_i, Y_i) and L_j is represented as (X_j, Y_j) and distance between the spatial attributes of the points is calculated as the Euclidean distance measure given by Equation (4)

$$d_S(L_i, L_j) = d\left((X_i, Y_i), (X_j, Y_j)\right) = \sqrt{(X_i - X_j)^2 + (Y_i - Y_j)^2} \qquad (4)$$

Clustering only the spatial attributes is straightforward as we can use the distance metric given by Equation (4) to find the distance between members of 2 different clusters. But if the points contain other attributes apart from spatial attributes, then clustering is performed using a weighted distance metric. Let us consider an example for clustering with an extra parameter which represents the angular movement between 2 consecutive points in a trajectory. This parameter is only an example; hence, we do not discuss the derivation or any properties of the attribute for now. Equation (5) provides the weighted distance metric for 3D points, (X_i, Y_i, θ_i) and (X_j, Y_j, θ_j), with the third dimension θ representing the directional attribute. The function $d_D(\theta_i, \theta_j)$ gives the distance between the directional attribute and W_D is weight for directional distance, while W_S is the weight for the spatial distance. The values for these weights are chosen such that $W_D+W_S=1$. With weights, $W_S=1$ and $W_D=0$, we get clusters that represents only the spatial regions. A value of $W_D>0$ will provide clusters that have points that might not be spatially closer to points in the other clusters but have similar angular movements. This equation can also be extended for points with more dimensions by providing the weights for each dimension and their distance function.

$$d_P(L_i, L_j) = \left(W_D \times d_D(\theta_i, \theta_j)\right) + \left(W_S \times d_S\left((X_i, Y_i), (X_j, Y_j)\right)\right) \qquad (5)$$

To implement the algorithm, an M x M similarity matrix D, is created, where M is the total number of points from all the trajectories. Each entry of the matrix $D_{i,j}$ is the average linkage criterion value between the clusters C_i and C_j, calculated using Equation (3). Since the value $D_{i,j} = D_{j,i}$, the matrix D is restricted as an adjacency list to avoid redundant computation. Hence the distance metric $D_{i,j}$ is calculated only for all i<j and the remaining values are assigned infinity. Clustering is performed iteratively, by merging the clusters C_i and C_j that have least $D_{i,j}$ values. After merging, the matrix entries for merged cluster i; $D_{i,k}$ are updated for all i<k and $D_{k,i}$ are updated for all k<i. Also, entries of column

and row j are changed to infinity. The complete pseudo code of the algorithm is given in Algorithm 2.

The center point L_{c_i} of the cluster is calculated as the mean value of individual attributes of all the points L_i in cluster C_i. If the center point is represented as (X_{c_i} , Y_{c_i}), then the mean value of the spatial attributes, X_{c_i} and Y_{c_i}, is calculated as the linear average value using Equation (6).

$$L_{c_i} = (X_{c_i}, Y_{c_i}) = \left(\frac{1}{|C_i|} \times \sum_{X_i \in C_{i,X}} X_i \, , \, \frac{1}{|C_i|} \times \sum_{Y_i \in C_{i,Y}} Y_i \right) \qquad (6)$$

Here $C_{i,X}$ and $C_{i,Y}$ represents the X and Y values in cluster C_i .

In this algorithm TRL is the list of trajectories. Each trajectory tr in this list consists of points $P_{t,tr}$ at time t. Each point in the trajectory is added to list LST. Now each point in the list LST are considered as individual clusters with a one member. The distance between these clusters are calculated and entered in matrix D. Then at each iteration we merge clusters C_i and C_j with lowest distance value $D_{i,j}$. In agglomerative clustering we can get cluster groups with different number of clusters in a single clustering process. Hence we maintain a list CGS that contains the list of numbers of clusters required so that we can get cluster groups with different number of clusters.

In our approach we evaluate the clusters by assigning a point from testing data to a cluster C_i with the lowest distance from center of each cluster L_{c_i} . The clusters created using the training data are used for building the prediction model. It should also be noted that the center point of clusters move as the clusters grow. This means a member L_1 of cluster C_1 can be closer to a point L_{c_2} which is the center of cluster C_2 than to point L_{c_1} which is the center of cluster C_1. To address this and provide a consistent way to assign new points to clusters, the clusters found are re-mapped into clusters with closest center corresponding. Hence each point in the training data is reassigned to a cluster $C_i$with

Algorithm 2: Hierarchical Clustering

```
Input:
        Set of Trajectory of points- TRL
        set of required cluster group size- CGS
Output:
        Trajectory of points along with clusters for each point- TRL

function cluster (TRL, CGS)

        for each trajectory tr in TRL:
                for each point at time t of trajectory tr:
                        LST.add(P_t,tr)
                end for:
        end for:
        for each Cluster C_i in L:
                for each Cluster C_j in L and i<j:
                        D_i,j = d_c(C_i, C_j)
                end for:
        end for:
        while LST.size==1:
                for each Cluster C_i in LST:
                        min = infinity
                        for each Cluster C_j in L and i<j:
                                if D_i,j <min:
                                        k = i
                                        h = j
                                end if:
                        end for:
                end for:
                C_k =Merge clusters C_k and C_h
                for each Cluster C_i in LST and i<k:
                        D_i,k = d_c(C_i, C_k)
                end for:
                for each Cluster C_i in LST and k<i:
                        D_k,i = d_c(C_i, C_k)
                end for:
                if CGS.contains(L.size):
                        for each Cluster C_i in LST:
                                Caculate L_ci
                        end for:
                        for each trajectory tr in TRL:
                                for each point at time t of trajectory tr:
                                        min = infinity
                                        for each Cluster C_i in LST:
                                                dis = d_p (L_ci , P_t,tr)
                                                if dis<min:
                                                        k = i
                                                end if:
                                        end for:
                                        tr.add(P_t,tr, k)
                                end for:
                                TRL.add(tr)
                        end for:
                end if:
        end while:
end function:
```

least distance value from center point L_{c_i} . This will give us new set of clusters that can be used for building prediction model. The average of points in these new cluster is considered to be centroid of the cluster. Hence a cluster consists of 2 points: Center and Centroid.

Even though clustering is performed using individual points from trajectories, we make sure that the trajectories of points are retained, so that we can get the trajectory of points along with the cluster of each point. This provides the trajectories of clusters, which can be used in computing the model parameters. This trajectory of clusters is presented in Figure 4.2

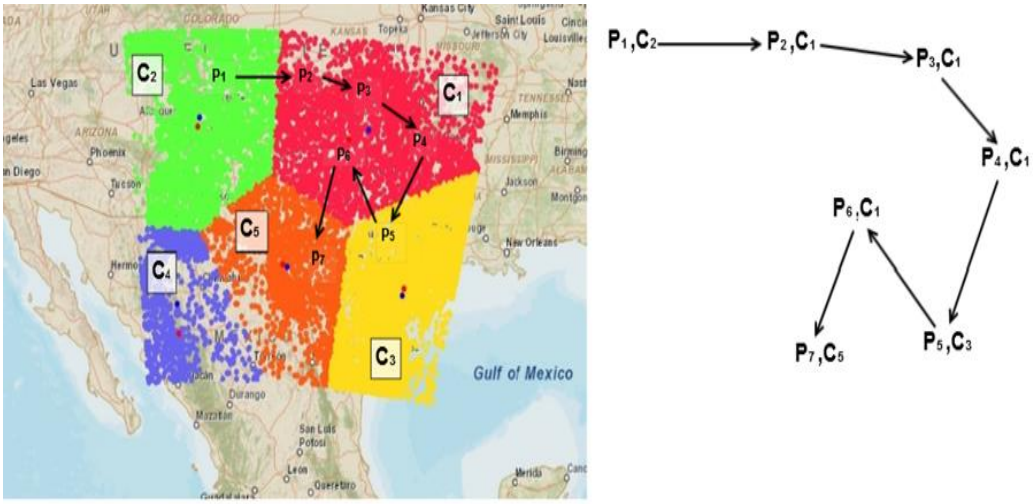


Figure 4.2 An example trajectory of points plotted over a map visualizing the clusters (left) and correponding output trajectory with the clusters of each points (right)

A trajectory of points, P_1, P_2, P_3, P_4, P_5, P_6, P_7, is plotted on the ArcGIS Map [38] and is given on the left side of Figure 4.2. Also the points of 5 clusters C_1, C_2, C_3, C_4 and C_5 are plotted on the map and are differentiated using different colors. The output from the clustering algorithm for this trajectory is shown in the figure on the right side and contains the clusters corresponding to each point on the trajectory.

Building Prediction Model Using Markov Model (MM)

Markov Model [12] is a mathematical model that uses a set of probabilistic parameters and a predefined set of states to represent a set of sequences, where each sequence is made up of one or more transitions between states. In our work the states of the Markov Model corresponds to the clusters of the points that were created in the previous section. Hence the transitions between clusters in the set of cluster trajectories that were obtained in the previous section correspond to state transition and can be used for calculating the model parameters. Throughout this work the states are represented using notation S_i. Hence in a MM cluster C_i corresponds to state S_i. Also, the notation $S_{t,tr}$ represents the state corresponding to an observation $P_{t,tr}$ at time t in a trajectory tr.

A Markov Model has two sets of parameters as described below.

Initial probability (Π_i): The probability that a trajectory starts in a given state, S_i. Since the states correspond to clusters this value can be calculated from the cluster trajectories as the probability of each cluster C_i to be present at time t=1 in any of the trajectories in the given set of trajectories. This is given by Equation (7).

$$\Pi_i = \frac{\sum_{tr}(P_{1,tr} \in S_i)}{\sum_{tr}\sum_j (P_{1,tr} \in S_j)} \qquad (7)$$

Here $P_{1,tr}$ represents a point at time t=1 in trajectory tr and $P_{1,tr} \in S_i$ gives the binary boolean value, either 0 or 1, representing that a point at time t=1 in a trajectory tr belongs to cluster C_i. If the point $P_{1,tr}$ belongs to cluster C_i it gives a value 1 else it gives a value 0. As mentioned, cluster C_i is represented using state S_i since clusters uniquely correspond to states.

Transition probability ($A_{i,j}$): The probability of moving from one state to another state. This is represented as a (CGS x CGS) matrix, A, where CGS is the number of states (clusters). Each entry $A_{i,j}$ represents the transition from cluster C_i to C_j and is

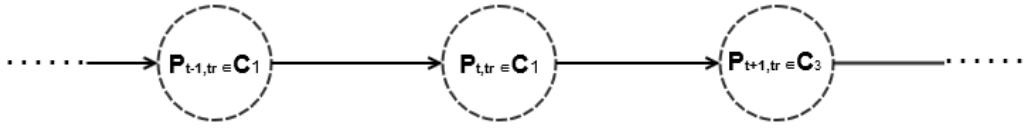
calculated from cluster trajectories as the probability of seeing cluster C_i at time t and C_j at time t+1 in each trajectory tr and is given by Equation (8)

$$A_{i,j} = \frac{\sum_{tr}\sum_{t}\left((P_{t,tr}\in S_i)\wedge(P_{t+1,tr}\in S_j)\right)}{\sum_{tr}\sum_{t}\sum_{k}\left((P_{t,tr}\in S_i)\wedge(P_{t+1,tr}\in S_k)\right)} \qquad (8)$$

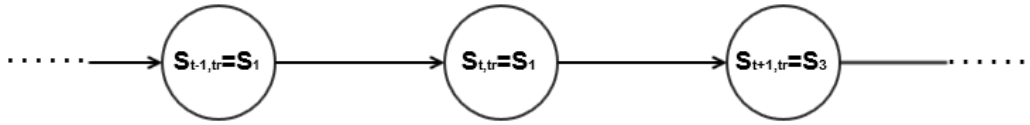
The transition probability matrix provides a probability distribution of cluster- to-cluster transitions. Also the initial probability provides the probability of a trajectory beginning from a cluster. Let us consider a trajectory tr as $P_{1,tr} \in C_i, P_{2,tr} \in C_k, P_{3,tr} \in C_j$ Now the probability of the trajectory tr can be obtained by applying the chain rule as given in Equation (9)

$$P(tr) = \Pi_i \times A_{i,k} \times A_{k,j} \qquad (9)$$

Hence from the transition probability the likelihood of a cluster at time t can be predicted if the cluster at time t-1 is known.



a)



b)

Figure 4.3 a) Sequence of observations (points) along with the cluster of each point in the sequence. b) Sequence of states in which states corresponding clusters representing the Markov Model for the observation sequence given in a)

In this figure a dotted circle represents a point along with the cluster of the point and a solid circle represents a state in the Markov Model. As mentioned, each state

corresponds to one cluster we can see that the cluster C_1 corresponds to state S_1. The Figure 4.3 b) provides a state sequence in a Markov Model corresponding to an observation sequence provided in Figure 4.3 a). This state sequence is created for all the trajectories in given set of trajectories of points and then used for calculating the model parameters using Equations (7) and (8).

Prediction using MM

The model is built using training data and the experiments are carried out using testing data. To test a transition we choose a point P_t at time t from a given trajectory and assign it to the closest cluster, C_{cl}. This is done by finding the distance between the point P_t and the center of cluster L_{c_i} using Equation (3). The same procedure is followed to find the cluster of the next point P_{t+1} in the trajectory as C_{actual}. From the transition probability matrix we can find the probability distribution for reaching all the clusters from cluster, C_{cl}. Now we choose the cluster with higher transition probability as the cluster $C_{predicted}$. Then the difference between the center of the cluster $C_{predicted}$ and the cluster C_{actual} is found using Equation (4). We will discuss about the results in the Chapter 5.

The Markov Model predicts the cluster at time t+1 given the cluster at time t. Since the clusters are created using the attributes of hourly storms, they represent the state of the storms (either spatial or spatio-directional). Hence we predict the state of the rainfall storms. However, the cluster provides only the closest state of the storm but not the actual attribute values. Moreover we need to be very careful when choosing the number of clusters since a Markov Model with a single cluster will provide 100% correct predictions in terms of cluster identities but is not helpful. Hence to get a better prediction model that is much more useful. We use a Hidden Markov Model (HMM) that provides a better representation of the transitional and spatial variation in the given set of trajectories by learning the hidden information present in the set of trajectories.

Building Prediction Model Using Hidden Markov Model (HMM)

A Hidden Markov Model [4] is similar to a Markov Model except that the states are hidden. Each state is associated with a set of observations that are visible. Hence we use sequences that represent observation transitions rather than state transitions. In this work, observations are the 2D points and the states correspond to the clusters. The main aim of using HMM is to learn the spatial and transitional information present in the given set of trajectories. This can be achieved by having more than one state for representing each cluster so that each state can represent different information in the cluster. A representation of a cluster is considered as the state instance and is assigned with a partial contribution value from each member of the cluster. However the sum of this contribution value over all the state instances for a given observation will be 1. If the number of the instances is 1, then each observation will contribute to only one state which is the cluster itself, thereby making the model similar to a MM.

Since each cluster corresponds to more than one state, the state S_i will no longer represent the cluster C_i and the identifier of the state representing an cluster depends on the number of instances. If INS is the number of instances for each cluster C_i, the states corresponding to this cluster are given as $S_{(1+INSx(i-1))}$, $S_{(2+INSx(i-1))}$, $S_{(3+INSx(i-1))}$... $S_{(INS+INSx(i-1))}$. Since each member of a cluster provides only a partial contribution to each state instance a state can have only a fraction of each observation which makes the amount of observations in a state as the sum of fractions of all observations contributing to this state. This amount of observation provides the weight of the state.

A HMM uses both the parameter sets of the Markov Model; transition probabilities and initial probabilities; along with a new parameter set called observation probability, which provides the relationship between the state and the observation. We

have already discussed transition probability and initial probability calculations in the case of a Markov Model. Hence, we define only the observation probability here.

Observation probability (B_i ($P_{t,tr}$)): This parameter provides the conditional probability of obtaining an observation $P_{t,tr}$ in state S_i. This probability is also called emission probability as it is considered that a state emits the given observation. A single state can emit more than one observation and one observation can be emitted by more than one state. However in our research an observation can be emitted only by the state instances that represent the cluster corresponding to the observation. An emission probability of 1 for a given observation represents that the state S_i emits only this observation and not any other observation while a value of 0 represents that the state S_i do not emit this observation. In the HMM used in our approach the states correspond to clusters of points. Each dimension of a point are continuous and are dependent on each other. Hence, to make the model learning feasible it will have to be assumed that the observations of each state follow a Multivariate Gaussian distribution [39]. The probability density function of a Multivariate Gaussian distribution, for an observation $P_{t,tr,}$ is given by Equation (10).

$$N(P_{t,tr}; \mu_i; V_i) = \frac{1}{\sqrt{2\Pi(V_i)^D}} \times e^{\frac{-1}{2} \times (P_{t,tr} - \mu_i)^T {V_i}^{-1} (P_{t,tr} - \mu_i)} \qquad (10)$$

Here, μ_i and $V_{i,}$ are the mean and covariance of the state S_i, and, D is the number of dimensions, which is the number of attributes of a point, $P_{t,tr}$.

An example of HMM with 3 state instances for each cluster is given in Figure 4.4 This figure depicts the relationship between observations and the states for an observation sequence, ($P_{1,tr}$ ……$P_{t-1,tr}$, $P_{t,tr}$, $P_{t+1,tr}$ ……. $P_{n,tr}$), where n is the length. In this figure, the observations are represented using dotted circles and the states are represented using solid circles. The dotted lines represent the observations emitted by

each state while the solid line represents the transitions. It should be noted that there are no solid lines between observation which means the HMM considers only state transitions obtained from observation transitions as we assume that observations are emitted by the states. The states S_1, S_2, S_3 can emit more than one observation given as $P_{t,tr}$ and $P_{t+1,tr}$. These observations are the members of a cluster which is represented using three states S_1, S_2, S_3. For the observation sequence, (……$P_{t-1,\,tr}$, $P_{t,tr}$, $P_{t+1,tr}$ …….), state sequences are created for all possible combinations of states.

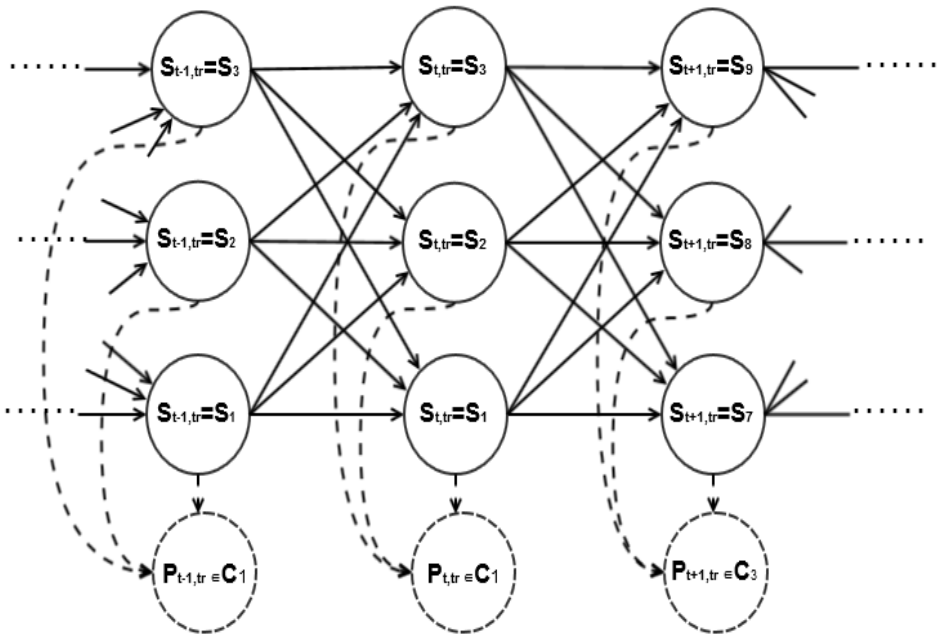


Figure 4.4 Sequence of states represnting HMM for the seuqnece of observations given in Figure 4.3 a)

This is done by keeping track of all states that emit an observation at each time t in the given trajectory, tr. In our approach an observation can be emitted by the states corresponding to a cluster and a cluster is represented by its state instances. Hence for each observation we keep track of its state instances.

The point $P_{t-1,tr}$ belongs to cluster C_1 .Hence state instances corresponding to these clusters are given by states S_1,S_2, S_3. Similarly the active states for points $P_{t,tr}$ and $P_{t+1,tr}$ are states S_1,S_2, S_3 and S_7,S_8, S_9 , respectively. From these states we get 3x3x3=27 state sequences. Now the probability of a trajectory is calculated using the model parameters from all possible state sequence combinations. The number of state sequences corresponding to a single observation sequence is exponential in the number of state instances representing each cluster and is given as INS. Hence calculating the probability of a trajectory by considering all the state sequence combinations increases space and time complexity. However this complexity is reduced by using the Forward or Backward algorithm which uses a dynamic programming approach. We discuss on these algorithms in detail later. For now we discuss the use of the algorithm. Both Forward and Backward algorithms are combined as Forward-Backward algorithm, which finds the probability of being in state, S_i, at time, t in trajectory tr and is represented as P ($S_{t,tr}$=S_i), for a given observation trajectory.

Our main aim is to find model parameters that maximize the sum of the probability of all trajectories in the given set of trajectories. To achieve this, the model parameters for each state are initialized using randomly generated P($S_{t,tr}$=S_i) values and then maximized by a learning algorithm called Baum-Welch [4]. Baum-Welch is an EM algorithm that maximizes the generative precision of the HMM model by modifying its parameters through learning using a given set of trajectories.

The calculation of the starting values for the model parameters can be explained using the following example. Let us consider a trajectory tr of length n. The observations $P_{t,tr}$ and $P_{t+1,tr}$ of the trajectory tr, belong to clusters C_i and C_j respectively. The number of state instances for representing each cluster in the model is given as INS. Now, for each observation, $P_{t,tr}$, INS number of random probability values are generated and used as

the probability value P ($S_{t,tr}$=S_i). This probability represents the weight of state S_i at time t in a trajectory tr and is used for calculating the starting values of the model parameters.

The Initial probability Π_i of state S_i is calculated as the average weight of a state S_i starting any trajectory in the given set of trajectories.

$$\Pi_i = \frac{\sum_{tr} \mathrm{P}\,(\mathrm{S}_{1,\mathrm{tr}}=\mathrm{S}_\mathrm{i})}{\sum_{tr}\sum_j \mathrm{P}\,(\mathrm{S}_{1,\mathrm{tr}}=\mathrm{S}_\mathrm{j})} \qquad (11)$$

A single transition from one observation, $P_{t,tr}$ to another observation, $P_{t+1,tr}$ in trajectory tr contributes to INS x INS transitions between states. Since the number of instances in the example is give as INS=3 we generate (INS x INS)=9, transition probabilities for a single observation transition, $P_{t,tr}$. If P ($S_{t+1,tr}$ =S_j | $S_{t,tr}$=S_i) represents the randomly generated value for a point $P_{t,tr}$ in states S_i and point $P_{t+1,tr}$ in state S_j, then the transition probability is given as:

$$A_{i,j} = \frac{(\sum_{tr}\sum_t \mathrm{P}(\mathrm{S}_{\mathrm{t+1,tr}}=\mathrm{S}_\mathrm{j}|\mathrm{S}_{\mathrm{t,tr}}=\mathrm{S}_\mathrm{i}))}{\sum_{tr}\sum_t\sum_k \mathrm{P}(\mathrm{S}_{\mathrm{t+1,tr}}=\mathrm{S}_\mathrm{k}|\mathrm{S}_{\mathrm{t,tr}}=\mathrm{S}_\mathrm{i})} \qquad (12)$$

The mean of observation distribution of state S_i, μ_i is a point given by the following equation.

$$\mu_i = \frac{\sum_{tr}\sum_t \mathrm{P}\,(\mathrm{S}_{\mathrm{t,tr}}=\mathrm{S}_\mathrm{i})\times P_{t,tr}}{\sum_{tr}\sum_t \mathrm{P}\,(\mathrm{S}_{\mathrm{t,tr}}=\mathrm{S}_\mathrm{i})} \qquad (13)$$

If the observation $P_{t,tr}$ is given as {X_t, Y_t}, then the mean of spatial values is calculated using Equation (14) and Equation (15).

$$\mu_{X_i} = \frac{\sum_{tr}\sum_t \mathrm{P}\,(\mathrm{S}_{\mathrm{t,tr}}=\mathrm{S}_\mathrm{i})\times X_t}{\sum_{tr}\sum_t \mathrm{P}(\mathrm{S}_{\mathrm{t,tr}}=\mathrm{S}_\mathrm{i})} \qquad (14)$$

$$\mu_{Y_i} = \frac{\sum_{tr}\sum_t \mathrm{P}\,(\mathrm{S}_{\mathrm{t,tr}}=\mathrm{S}_\mathrm{i})\times Y_t}{\sum_{tr}\sum_t \mathrm{P}(\mathrm{S}_{\mathrm{t,tr}}=\mathrm{S}_\mathrm{i})} \qquad (15)$$

The covariance of the distribution of observations in state S_i is a (D x D) matrix, V_i, which is calculated using the following equation.

$$V_i = \frac{\sum_{tr}\sum_t \mathrm{P}\,(\mathrm{S}_{\mathrm{t,tr}}=\mathrm{S}_\mathrm{i})\,(P_{t,tr}-\mu_i)^T(P_{t,tr}-\mu_i)}{\sum_{tr}\sum_t \mathrm{P}\,(\mathrm{S}_{\mathrm{t,tr}}=\mathrm{S}_\mathrm{i})} \qquad (16)$$

The value ($P_{t,tr} - \mu_i$) is a point obtained by subtracting each dimension of point $P_{t,tr}$ from the corresponding dimensions in the mean point μ_i and is given as:

$$\left(P_{t,tr} - \mu_i\right) = \left(\left(X_t - \mu_{X_i}\right), \left(Y_t - \mu_{Y_i}\right)\right) \qquad (17)$$

The model parameters are learned such that the probabilities of the trajectories are maximized. This algorithm involves 2 steps.

Expectation Step: In this step we calculate the probability for each trajectory from the given distribution of the states. As we mentioned earlier the probability of a trajectory can be calculated either using the Forward algorithm or the Backward algorithm.

The calculation of forward and backward probabilities is explained using a given trajectory tr of length n represented as,($P_{1,tr}$, $P_{2,tr}$, $P_{3,tr}$....$P_{n,tr}$). The Forward algorithm finds the forward probability of state S_i at time t in a trajectory tr as $\alpha_{i,t,tr}$. This probability is defined as the joint probability of being in state S_i at time t and seeing the observation sequence (P_1, P_2...P_{t-1}). Equation (18) gives the forward probability of state S_i using a single observation, $P_{t+1,tr}$.

$$\alpha_{j,t+1,tr} = B_j\left(P_{t+1,tr}\right) \sum_i \left(\alpha_{i,t,tr} \times A_{i,j}\right) \qquad (18)$$

The forward probability for the observation at time t=1 is given as:

$$\alpha_{i,1,tr} = \Pi_i \times B_i\left(P_{1,tr}\right) \qquad (19)$$

The backward algorithm provides the backward probability of a state S_i at time t in a trajectory tr as $\beta_{i,t,tr}$, and is calculated as the conditional probability of ending an observation sequence as (…$P_{t+1,tr}$, $P_{t+2,tr}$, $P_{t+3,tr}$....$P_{n,tr}$) given a state S_i at time t. This probability is calculated by the following Equation.

$$\beta_{i,t,tr} = \sum_j \beta_{j,t+1,tr} \times A_{i,j} \times B_j\left(P_{t+1,tr}\right) \qquad (20)$$

The backward probability for of state S_i at time t=n in a trajectory tr, where n is the length of the trajectory tr, is gives as:

$$\beta_{i,n,tr} = 1 \qquad (21)$$

The probability of the trajectory can be calculated, either by using the forward probability or the backward probability and is given as:

$$P(tr) = \sum_i (\alpha_{i,n,tr}) \qquad (22)$$

The probability P ($S_{t,tr} = S_i$) is calculated as the product of both the forward and the backward probability.

$$P(S_{t,tr} = S_i) = \frac{\alpha_{i,t,tr} \times \beta_{i,t,tr}}{P(tr)} \qquad (23)$$

The calculation of forward and backward probability can be explained using Figure 4.5.

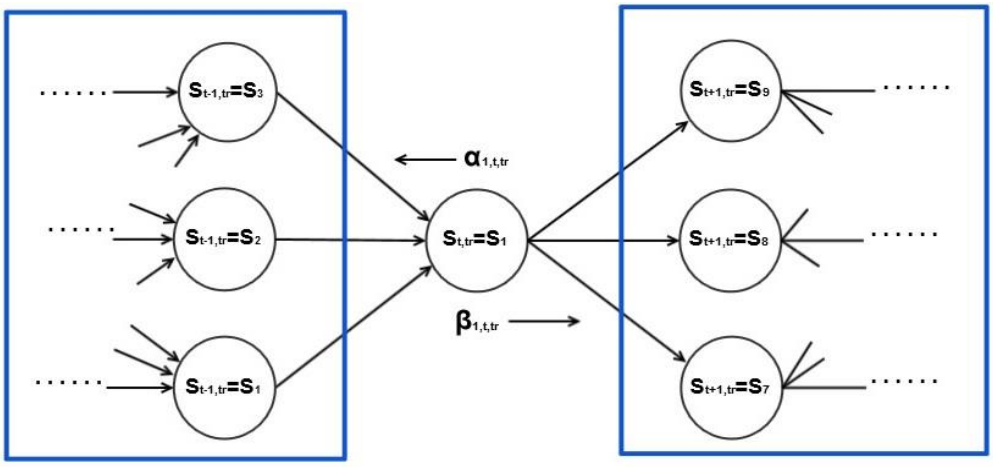


Figure 4.5 Forward and Backward probability

This figure illustrates forward probability and backward probability calculation for state S_1 at time t in a given trajectory tr. The forward probability, $\alpha_{1,t,tr}$, of the state S_1 is calculated by moving left to right from all possible states at time t-1 to state S_1 at time t. The backward probability, $\beta_{1,t,tr}$ is calculated by moving right to left from all possible states at time t+1 to state S_1 at time t. The forward probability is calculated using the

observation seen at time t while the backward probability is calculated using observation at time t+1 as we have not seen the observation at t.

Maximization Step: In this step the model parameters are calculated so that the sum of the probabilities of all trajectories in the given set of trajectories is maximized given the model parameters calculated in the Expectation step. The model parameters Π_i, μ_i, V_i are calculated using the following equations.

$$\Pi_i = \frac{\sum_{tr} \frac{\alpha_{i,1,tr} \times \beta_{i,1,tr}}{P(tr)}}{\sum_{tr} \frac{\sum_j \alpha_{j,1,tr} \times \beta_{j,1,tr}}{P(tr)}} \qquad (24)$$

$$\mu_i = \frac{\sum_{tr} \frac{\sum_t \left(\alpha_{i,t,tr} \times \beta_{i,t,tr}\right) \times P_{t,tr}}{P(tr)}}{\sum_{tr} \frac{\sum_t \left(\alpha_{i,t,tr} \times \beta_{i,t,tr}\right)}{P(tr)}} \qquad (25)$$

$$V_i = \frac{\sum_{tr} \frac{\sum_t \left(\alpha_{i,t,tr} \times \beta_{i,t,tr}\right) \times \left(P_{t,tr} - \mu_i\right)^T \left(P_{t,tr} - \mu_i\right)}{P(tr)}}{\sum_{tr} \frac{\sum_t \left(\alpha_{i,t,tr} \times \beta_{i,t,tr}\right)}{P(tr)}} \qquad (26)$$

The calculation of the transition probabilities is given by Equation (27).

$$A_{i,j} = \frac{\sum_{tr} \sum_t \frac{\alpha_{i,t,tr} \times A_{i,j} \times B_j\left(P_{t+1,tr}\right) \times \beta_{j,t+1,tr}}{P(tr)}}{\sum_{tr} \sum_t \sum_k \frac{\alpha_{i,t,tr} \times A_{i,k} \times B_k\left(P_{t+1,tr}\right) \times \beta_{k,t+1,tr}}{P(tr)}} \qquad (27)$$

Both steps are repeated until the model parameters converged. The pseudo code of the Baum-Welch algorithm is given as Algorithm 3.

Algorithm 3: Building HMM

Input:
Trajectories of points along with corresponding clusters-TRL
Number of instances-INS

Output:
Model parameters- $\{\hat{\Pi}, \hat{A}, \hat{\mu}, \hat{V}\}$ of maximized model

function Model parameters HMM(TRL)**:**

Obtain set of clusters CS from the trajectory TRL
Create set of states M as set with INS instances for each cluster in CS
Create set of states $STS_{t,tr}$ for each point $P_{t,tr}$ containing instances of cluster C_i where $P_{1,tr} \in C_i$
for each trajectory tr in TRL**:**
Generate random values by restricting $P_{t,tr}$ to states in set $STS_{t,tr}$
for each state S_i**:**
Calculate the starting values of model parameters as $\{\Pi_i, A_{i,j}, \mu_i, V_i\}$
end for:
end for:
do :
for each trajectory tr in TRL**:**
$\hat{P}(tr) = P(tr)$
end for:
for each trajectory tr **in** TRL**:**
Calculate forward and backward probability, α_{tr} and β_{tr}, by restricting $P_{t,tr}$ to states in set $STS_{t,tr}$
Calculate the probability of trajectory $\hat{P}(tr)$
for each state S_i **in** M**:**
Calculate the new values of model parameters as $\{\hat{\Pi}_i, \hat{A}_{i,j}, \hat{\mu}_i, \hat{V}_i\}$ from $\alpha_{i,tr}$and $\beta_{i,tr}$
end for:
end for:
$\{\Pi_i, A_{i,j}, \mu_i, V_i\} = \{\hat{\Pi}_i, \hat{A}_{i,j}, \hat{\mu}_i, \hat{V}_i\}$
while $\left(\Sigma_{tr}\hat{P}(tr) - \Sigma_{tr}P(tr)\right) == 0$**:**
return $\{\hat{\Pi}, \hat{A}, \hat{\mu}, \hat{V}\}$**:**
end function:

One common problem in these calculations is that the probability for a longer trajectory may go lower than the minimum floating point value supported by the machine. To avoid this issue the probabilities are represented as log likelihood values and multiplication and division operations are converted into addition and subtraction respectively.

The Baum-Welch algorithm provides only local maximum values depending on the starting values of the model parameters. Hence the results of the learning process is different for different starting values. The converged model parameters provide us with a

new representation for states that gives a better representation for observations. For example, after learning, a State S_i which is an instance of cluster C_k, can represent hourly storms that occurred in a single region of the cluster C_k, or hourly storms that are moving in same direction, or hourly storms that move towards the same region or any other representation depending on the learning process. This new representation of clusters is obtained by modifying (or weighing) the relevant parts of the cluster to each state instances based on the spatial and transitional variations. In other words, clusters are further diversified based on the space and movement. Hence this approach looks similar to clustering or classification. But it should be noted that this learning is different from clustering since in HMM learning an observation may not be represented by a single state while in most clustering techniques each strictly observation belongs to one cluster. Moreover, clustering does not consider transitions between observations but only the observations, while HMM learning uses transition probabilities apart from the observation probabilities.

Smoothing the Gaussian Using Uniform Distribution

Some of the state distributions are created using few points. For example, a state distribution could be created using a single point which will lead to a zero covariance matrix which will give a density value of infinity if the point is same as the mean and of zero for all other points. This state with zero covariance is not useful as we may not get any observation identical to the mean in the future. Hence this distribution is modified by adding noise to the existing state distribution.

This is done by generating random observations within the given range (actually by generating values for each dimension of an observation from a random number generator limited within the range of the dimension) and assigning them to the closest

cluster. For example if the range of X and Y values in a co-ordinate system is (X_{min}, X_{max}) and (Y_{min} , Y_{max}) then a random observation L_{rand} is given as below.

$$L_{rand} = (X_{rand}, Y_{rand})$$
$$= ((X_{min} + rand() \times X_{max}),)(Y_{min} + rand() \times Y_{max})) \quad (28)$$

The point L_{rand} is assigned to a cluster C_i for which the distance between random point L_{rand} and the center of cluster $L_{c_i}$is the minimum. In this work we generate n random observations and assign them to the closest cluster. Then the mean and variance for these uniformly distributed random observations of each cluster is calculated.as μ_{rand_i} and σ_{rand_i} , respectively for each cluster C_i. These random distributions are added to each state instance of the cluster C_i by performing weighted addition with mean μ_k and variance $V_{k,d,d}$ of the states S_k for all states that represent cluster C_i and is given by Equations (29) and (30).

$$\mu_k = \mu_{rand_i} \times W_{rand_i} + \mu_k \times \left(1 - W_{rand_i}\right) \quad (29)$$

$$V_{k,d,d} = \sigma_{\mathrm{rand}_i} \times W_{rand_i} + V_{k,d,d} \times \left(1 - W_{rand_i}\right) \quad (30)$$

Here d represents a dimension identifier and W_{rand_i} is the weight for the uniformly distributed samples of the cluster C_i and is calculated using Equation (31).

$$W_{rand_i} = K/(W_{uniform_i} + W_k) \quad (31)$$

Here, W_k represents the weight of the state, S_k, which is a state instance that represents the cluster C_i. It should be noted that state, S_k is assumed to follow a Gaussian distribution of observations present in the given set of trajectories and belong to cluster C_i. $W_{uniform_i}$ is the weight of the cluster C_i created by uniformly distributed random observations and is the number of random observations which are closer to the center of the cluster C_i than any other cluster. Also K represents the number of uniformly distributed random observations added to the Gaussian distribution. A value $W_k < K$ will

make the state biased towards randomly generated uniformly distributed observations while a $W_k > K$ will make the state biased towards observations from the input data that follow a Gaussian distribution. Hence we need to be careful in choosing the K value. We discuss experiments for different values of K in Chapter 5.

Prediction using HMM

Once the model is created, we can predict the behavioral attributes of the next possible observation in any given trajectory. In a HMM the next possible observation depends on the entire trajectory and not only on the current state, unlike in a MM. Let us consider a trajectory tr as ($P_{1,tr}$, $P_{2,tr}$, $P_{3,tr}$.... $P_{n+1,tr}$). In order to evaluate the model we predict the observation $P_{n+1,tr}$ given the trajectory ($P_{1,tr}$, $P_{2,tr}$, $P_{3,tr}$.... $P_{n,tr}$) and compare the predicted location with the actual location. The probability of this trajectory is calculated using the forward algorithm. Since the observation probability of the next possible observation $P_{n+1,tr}$ is unknown, the forward probability at time t=n is calculated and then the transition probability is used for calculating the probability of being in state S_i at time t=n+1 for all states S_i in the model. This is given by Equation (32).

$$W_j = \sum_i \alpha_{i,n,tr} \times A_{i,j} \qquad (32)$$

Now, by using the probability as weights, the observation, $P_{n+1,tr}$, is predicted based on the weighted sum of all the states. Since each state observation is assumed to follow a Gaussian distribution, the weighted sum of all the states will result in an observation following a Mixture of Gaussians and the predicted observation is the mode of the mixture. The PDF of a Gaussian Mixture with M component, and, weights W_i for each component i, for a point $P_{t,tr}$ is given by Equation (33)

$$N\left(P_{t,tr}; \mu_{1\ldots M}; V_{1\ldots M}; W_{1\ldots M}\right) = \sum_{i=1}^{M} W_i \times N\left(P_{t,tr}; \mu_i; V_i\right) \qquad (33)$$

Here, $\sum_{i=1}^{M} W_i = 1$. The value of $N(P_t; \mu_i; V_i)$ is calculated using the multivariate Gaussian PDF given in Equation (10).

Figure 4.6 visually illustrates the difference between the PDF of a 2D-Gaussian and a 2D Gaussian mixture with 5 components and different weights for each component.

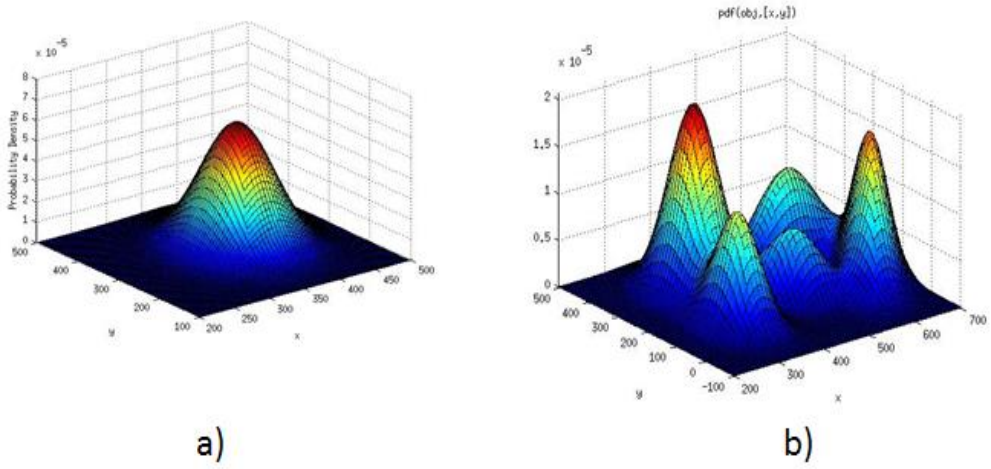


Figure 4.6 a) PDF of a 2D Multivariate Gaussian b) PDF of 2D Multivariate Gaussian mixture with 5 compnenst and unequal weights.

The mode of a distribution is the point with the maximum probability in the distribution. Hence, for a Gaussian distribution with no skewness, the mean is equal to the mode. The mode of a Gaussian mixture depends on the weight and mean of each component in the mixture. The mean of the mixture is the weighted sum of the means of the individual components of the mixture. But this mean is not the peak point (mode) of the mixture. In order to find the mode of a mixture we need to search for the peak of the mixture. There are many methods to perform mode search. We use the Gradient-Quadratic Mode Search method proposed by Miguel A. [40]. This method uses the quadratic maximization technique (Newton's method) combined with a gradient ascent method to search for a peak point in a mixture.

A gradient is the first derivative of the given function. For a Multivariate Gaussian the first derivative is given by Equation (34).

$$Gradient\ g = d\frac{N(P_{t,tr};\mu_{1..M};V_{1..M};W_{1..M})}{dx}$$

$$=\sum_{i=1}^{M} W_i \times N(P_{t,tr};\mu_i;V_i) \times (V_i)^{-1} \times (P_{t,tr}-\mu_i) \quad (34)$$

Gradient ascent [40] is a local optimization technique used for finding the local maximum in a given function. We describe the gradient ascent technique used by Miguel [40] for performing mode search. In this technique, we start from a point P_{start} and move to a point which is certain number of steps away from P_{start}. The number of steps moved is given as step size, SZ. This movement is repeated by reducing the step size value by half until a point with higher probability than the probability of P_{start} is reached.

Quadratic search proposed by Miguel [40] for mode finding, is similar to gradient ascent except that the step size is not constant but is given as the ratio of the second derivative (Hessian function) and the first derivative (gradient function). The second derivative of the Multivariate Gaussian is given by Equation (35).

$$Hessian\ H = d^2\ \frac{N(P_{t,tr};\mu_{1..M};V_{1..M};W_{1..M})}{dx}$$

$$= \sum_{i=1}^{M} W_i \times N(P_{t,tr};\mu_i;V_i)$$

$$\times V_i^{-1}\left((P_{t,tr}-\mu_i)(P_{t,tr}-\mu_i)^T - V_i\right)V_i^{-1} \quad (35)$$

To address resolution in use with small probabilities, Miguel A. [40] uses log density of these functions, lg and lh, is given by Equation (36) and Equation (37) respectively.

$$lg = \frac{1}{p}g \quad (36)$$

$$lh = \frac{1}{p^2}gg^t + \frac{1}{p}H \quad (37)$$

The prediction algorithm is given in Algorithm 4.

Algorithm 4: Predicting Observation

Input:
Trajectory of points tr,($P_{1,tr}$, $P_{2,tr}$, $P_{3,tr}$....$P_{n,tr}$),
Model parameters- $\{\hat{\Pi}, \hat{A}, \hat{\mu}, \hat{V}\}$
Output:
Observation $P_{n+1,tr}$

function PO(tr, $\{\hat{\Pi}, \hat{A}, \hat{\mu}, \hat{V}\}$**:**

for each state S_i**:**
Calculate the weights W_i using forward probability and transition probability A
end for:
for each state S_i**:**
$P_{old}=\hat{\mu}_i$
do:
$pr_{old}= N(P_{old}; \mu ; V ; W)$
Calculate lg and lh
if lh < 0**:**
$P_{new} = P_{old}$ + lg/lh
$pr_{new} = N(P_{new}; \mu ; V ; W)$
else
pr_{new}= -1
end if:
while pr_{new}< pr_{old}**:**
$P_{new} = P_{old}$ + (SZ x lg)
$pr_{new} = N(P_{new}; \mu ; V ; W)$
SZ= SZ/2
end while:
$P_{old}= P_{new}$
while (lg>0) **:**
if pr_{new}>min **:**
$P_{predicted}=P_{new}$
end if:
end for:
return $P_{predicted}$**:**
end function:

The mode search starts from a point μ_i, the mean of the observations of state S_i of the mixture. The Hessian, lh, and gradient, lg, for this point are calculated using Equations (37) and (36) respectively. A value of lh>0 represents that we are on the hill cap of the mixture and a new point is reached by moving through $\frac{lg}{lh}$ steps. But if the value of lh<0 is reached then the current point is not on the hill cap of a peak in the mixture and a new point is reached by gradient ascent. If the new point has a probability lower than the current point then we reach another new point by reducing the step size value. Then the lg and lh values are calculated for this new point. Ideally the process stops when we reach a point with g=0, which means the peak of the mixture is reached. But experiments

have shown that condition g=0 is never reached. Hence the process stops when g reaches a threshold value close to zero or when a certain number of iterations has been processed. The process is repeated by starting from the mean of all the remaining components of the mixture hence giving more than one point. We choose the point with the highest probability as the predicted point, $P_{predicted}$.

Once the observation is predicted the model is evaluated by finding the distance between the predicted observation point $P_{predicted}$ and the actual observation point $P_{n+1,tr}$ in the trajectory from the testing data. The distance is measured separately for each attribute, so that the prediction accuracy of each attribute can be analyzed separately. We discuss the results more in Chapter 5.

Predicting More Than One Step in the Future

Both MM and HMM models can also be extended to predict more than one step in the future. This can be done by modifying the transition probability matrix $A_{i,j}$ using the Chapman-Kolmogorov Equation given as Equation (38).

$$A_{i,j}^{(nx)} = \sum_{S_k} A_{i,k}^{(m)} \times A_{k,j}^{(nx-m)} \qquad (38)$$

Here, m represents the number of steps transitioned so far and nx is the number of steps to be predicted in the future. The probability $A_{i,j}^{(nx)}$ gives the probability of transitioning from state S_i to S_j, nx steps in the future. The probability of transitioning from state S_i to S_j in 3 steps, $A_{i,j}^{3}$ is calculated from the value $A_{k,j}^{2}$ which is further calculated from the value $A_{i,k}^{1}$, which is the transition probability $A_{i,k}$ This means that the probability of transitioning from state S_i to S_j after nx transitions, depends on the transition probability of all the states at steps 1 to nx-1.

From Equation (38) we can see that the value $A_{i,j}^{(nx)}$ is the same as the entry of a matrix at row i and column j which was obtained by multiplying the transition probability

matrix $A_{i,j}$ n times with itself. Hence by multiplying the transition probability matrix A, nx times we get the new transition probability matrix $A^{(nx)}$ which can be used for predicting nx steps in the future. Given a trajectory ,($P_{1,tr}$, $P_{2,tr}$, $P_{3,tr}$....$P_{n,tr}$), this matrix can be used for a MM to predict the most probable cluster at time (n+nx) similar to predicting the cluster at n+1 as explained earlier. In a HMM the forward probability of a trajectory is calculated using the transition matrix A and then the weights for each cluster at time n+nx are obtained using the transition matrix $A^{(nx)}$. These weights can be used for finding the most probable observation using mode search explained earlier for predicting a single step transition.

Directional attribute

Apart from the spatial attribute, we also use an attribute that represents the directional movement between each point in the trajectory. The direction attribute is the direction taken by a storm at time t from position (X_t , Y_t) in trajectory tr to reach a position (X_{t+1} , Y_{t+1}). This direction is then calculated by considering the point (X_t , Y_t) as the origin of a unit circle and the angle between the x-axis of the circle and a line drawn from the origin of this circle to a point given as $((X_{t+1} - X_t), (Y_{t+1} - Y_t))$ and is given as line ln. The direction is given as the angle between the x-axis and this line ln and is represented in degrees. We consider only anti-clockwise rotation and restrict the directional parameters to [0,360) degrees. Figure 4.7 shows the direction parameter at time t=4 calculated as angle between point at time t=3 and t=4 in a given trajectory.

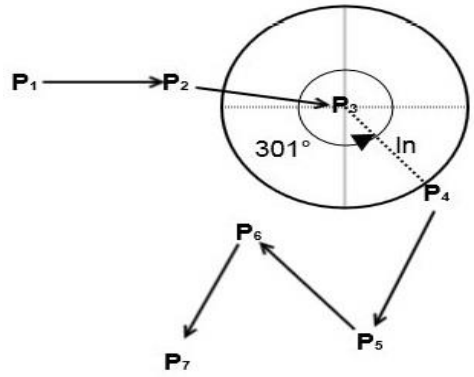


Figure 4.7 Caculating the angle between 2 points in a trajectory.

This angle is calculated using Equation (39)

$$\theta_{t+1} = atan2\big((Y_{t+1} - Y_t), (X_{t+1} - X_t)\big) \quad (39)$$

where (X_t , Y_t) and(X_{t+1} , Y_{t+1}) are the points $P_{t,tr}$ and $P_{t+1,tr}$ at time t and t+1, respectively, and the angle θ_{t+1} represents the directional attribute at time t+1 in a trajectory tr. The atan2 function provides the angle value in radians between (-π, π] which is then converted into degrees in the range [0,360). The starting point of a trajectory, P_1, will have a directional attribute value of -1, representing an unknown angle.

This directional attribute is included as the 3rd dimension of each point, $P_{t,tr}$, in the trajectory. Hence the point, $P_{t,tr}$, is given as a 3D point, (X_t , Y_t, θ_t).

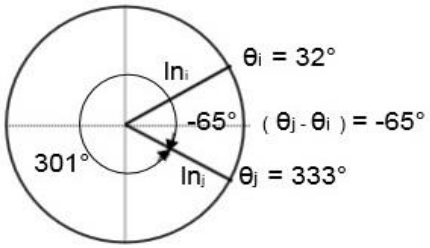


Figure 4.8 Calculating the diffrence between 2 angles

It should be noted that angles are restricted to lie.in the range [0,360) and any angle above or equal to 360 is considered to start from 0 again. For example, a value of

720 is the same as 0 or 360 or 1080. Hence angles are considered wrapped parameters as they repeat after 360 is reached .Since angles are wrapped, the distance measure for the directional attribute is a special case. Angles can be well understood by considering a unit circle as provided in Figure 4.8. In this circle the origin is given as (0,0) and to find the difference between angles θ_j and θ_i we draw 2 lines, inclined relative to the x-axis of the circle at angles θ_j and θ_i and represent them as ln_j and ln_i respectively. The difference between these 2 angles is given as the angle between these 2 lines. This angle is calculated as the minimum angular rotation required for ln_j.to reach line ln_i. In a circle a line can rotate either clockwise or anti-clockwise. An anti-clockwise rotation is considered to give a positive angle while clock wise rotation gives a negative angle. The linear difference between these 2 angles, $abs(\theta_i - \theta_j)$, gives the unwrapped difference and $\left(360 - abs(\theta_i - \theta_j)\right)$ gives the wrapped difference. The distance between the angles is the minimum of the absolute value of these rotational angles. Distances do not consider direction hence only the absolute value is considered. This distance measure is given by Equation (40)

$$d_D(\theta_i, \theta_j) = \min\left(abs\left(\theta_i - \theta_j\right), \left(360 - abs(\theta_i - \theta_j)\right)\right) \quad (40)$$

In this equation abs is the absolute function which considers only the total amount and not the sign. This distance metric is used in Equation (5) for creating clusters with both spatial and directional attributes.

Once the clusters are created a HMM is created using these clusters. It should be noted that the HMM is built considering the assumption that the clusters follow a multivariate Gaussian distribution. Since clusters have both spatial and directional attributes we need to use a Gaussian distribution that includes both attributes.

The directional attribute in the clusters follows a special distribution called a wrapped distribution. If μ_{θ_i} and σ_{θ_i} are the mean and variances of the wrapped distribution, then the PDF of a wrapped Gaussian is given as:

$$N_w(\theta_t; \mu_{\theta_i}; \sigma_{\theta_i}) = \sum_{\omega=-\infty}^{\infty} N(\theta_t - 2\pi\omega; \mu_{\theta_i}; \sigma_{\theta_i}) \quad (41)$$

$$N_w(\theta_t - 2\pi\omega; \mu_{\theta_i}; \sigma_{\theta_i}) = \frac{1}{\sqrt{2\pi\sigma_{\theta_i}^2}} \times e^{\frac{\frac{-1}{2}\times\left|\theta_t - \mu_{\theta_i} - 2\pi\omega\right|^2}{\sigma_{\theta_i}^2}} \quad (42)$$

Since the summation of the values between (+∞, -∞) is computationally impossible, we use an approximation of this distribution called Mixture of Approximated Wrapped Gaussian (MoAWG) distribution proposed by Balham [41] . The density function of a MoAWG distribution with M components is given in Equation (43).

$$N_w(\theta_t; \mu_{\theta_i}; \sigma_{\theta_i}) = \sum_{i=1}^{M} W_i \times \frac{1}{\sqrt{2\pi\sigma_{\theta_i}^2}} \times e^{\frac{\left(\frac{-1}{2}\times\left|\left(\theta_t - \mu_{\theta_i}\right) mod 2\pi\right|^2\right)}{\sigma_{\theta_i}^2}} \quad (43)$$

Here μ_{θ_i} and σ_{θ_i} are the mean and standard deviation of the component S_i and W_i is the weight of the component S_i in the mixture.

This distribution was proposed considering the angles are in radians in the range (-π, π]. The function (θ_t - μ_{θ_i}) mod 2π is modified to make sure that the directional difference is in the range (-π , π]. However, we represent the angles in degrees and use (θ_t - μ_{θ_i})mod (360) instead. Here, (θ_t - μ_{θ_i}) mod (360) represents displacement between the angle and the mean of the state and lies in the range (-180, 180]. The displacement measure for two given angles is similar to the distance measure provided by Equation (41) except that the displacement includes direction also. Hence displacement between 2 angles θ_t and μ_{θ_i} is given by Equation (44). A negative displacement value represents a clockwise rotation while positive displacement represents anti-clockwise rotation.

$$displ(\theta_t, \mu_{\theta_i}) = \text{absminWithSign}\left(\left(\left(\theta_t - \mu_{\theta_i}\right), \left(\theta_t - \mu_{\theta_i} - 360\right), \left(\theta_t - \mu_{\theta_i} + 360\right)\right)\right) \quad (44)$$

In this equation, the function $absminWithSign$ finds the absolute minimum value among the input values given but also retains the sign of the value.

The approximated mean, μ_{θ_i}, of state S_i is calculated in the maximization step or for starting value using the Equation (45).

$$\mu_{\theta_i} = atan2\left(\frac{\sum_{tr}\sum_{t} P(S_{t,tr}=S_i)\times \sin(\theta_t)}{\sum_{tr}\sum_{t} P(S_{t,tr}=S_i)}, \frac{\sum_{tr}\sum_{t} P(S_{t,tr}=S_i)\times \cos(\theta_t)}{\sum_{tr}\sum_{t} P(S_{t,tr}=S_i)}\right) \quad (45)$$

Also, by the modified Equation (46) we get the mean of angular attribute of the points in cluster C_i;

$$\theta_{c_i} = atan2\left(\frac{1}{|C_i|}\times \sum_{\theta_i \in C_{i,\theta}} sin(\theta_i), \frac{1}{|C_i|}\times \sum_{\theta_i \in C_{i,\theta}} \cos(\theta_i)\right) \quad (46)$$

MoAWG represents a univariate wrapped distribution while our model is built by assuming that spatial and directional attributes are dependent on one other. Hence, we need to use a distribution which is a combination of both wrapped and unwrapped components. Simone Calderara [42] proposed a semi-directional distribution by combining MoAWG with a multivariate Gaussian. The difference between the density function of this multivariate Gaussian and a linear multivariate Gaussian is only the calculation of the displacement value ($P_{t,tr} - \mu_i$). The proposed method uses the new displacement value given by Equation (47). This equation is also used for calculating the covariance matrix V_i using the directional parameter as a dimension.

$$(P_{t,tr} - \mu_i) = \left((X_t - \mu_{X_i}), (Y_t - \mu_{Y_i}), displ(\theta_t, \mu_{\theta_i})\right) \quad (47)$$

Also, by modifying the observation $P_{t,tr}$ using Equation (48) we can find the probability density of the 3D point (X_t, Y_t, θ_t) using the multivariate Gaussian density function defined in Equation (10).

$$P_{t,tr} = \left(X_t, Y_t, displ(\theta_t, \mu_{\theta_i}) + \mu_{\theta_i}\right) \quad (48)$$

Chapter 5

Experimental Results

In this chapter we describe the data sets used for performing experiments, discuss the results of clustering, and, compare and analyze the results of MM and HMM models using statistics.

Overall storm trajectories were obtained for each year separately, from years between 2004 and 2008. Each of these years contains at least 200,000 trajectories with lengths varying from 1 to 200 hours. Trajectories with very small length are not temporal. Hence we derive a data set for each year that contains trajectories that are at least 10 hours long. The experiments are performed for each year separately by considering any given year as training data and the remaining years as testing data. Training data are used for building the prediction model while the testing data are used for evaluating the model. Even though the testing data set consists of trajectories from more than one year, we perform evaluation for each year separately. Moreover, training data is also evaluated so that we can analyze the prediction accuracy of the model by comparing evaluation results of training data with the evaluation results of testing data.

Before analyzing the results of the experiments conducted we will define terms and abbreviations that will be used.

Cluster Group (CG) : Set of clusters obtained for the given trajectories.

Cluster Group Size (CGS) *:* Number clusters in a Cluster Group

Cluster Size (CS) : Number of points in a cluster.

Mean Cluster Size (MCS) : Mean of Cluster Size of all the clusters in a Cluster Group.

Distance Accuracy (DA) : The distance between actual point and predicted point.

Diameter : The distance between the 2 farthest points in a cluster.

Mean Diameter Distance (MDD) : The mean of the diameter of each cluster in a given Cluster Group.

INS : number of state instances representing each cluster in a HMM

Mean Distance Accuracy (MDA) : Mean of the distance between actual point and predicted point for given set of evaluated distances for the training data.

Training Data Year (TrDY) : Year of the training data used to create the model

Testing Data Year (TDY) : Year of the testing data used for evaluating the model.

Log Likelihood (LL) : sum of log likelihoods of the trajectories in a given set of trajectories using HMM.

Analyzing clusters

Experiments are performed with 2004 trajectories as training data and the data from years between 2005 to 2008 as testing data. The training data set, year 2004, consist of 713 trajectories with 14,831 points in them. The points are clustered into cluster groups with 5, 10, 25, 50, 75 and 100 clusters. The number of clusters in a cluster group is called cluster group size and the number of points in each cluster is called the cluster size. The maximum distance between any 2 points in the cluster is called the diameter of the cluster. The clusters of the points are visualized using ArcGIS maps [38] and are presented in the Figure 5.1, Figure 5.2 and Figure 5.3 for clusters groups with 5, 25 and 100 clusters respectively. The members of the clusters are differentiated using different colors. However, due to limitation of colors some colors either repeat or are a slightly modified version of colors used for other clusters.

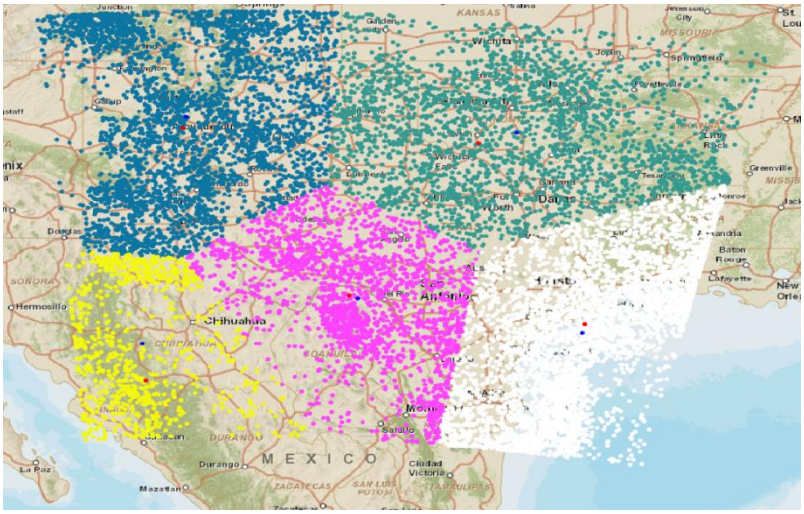


Figure 5.1 Visualizing cluster group with 5 clusters.

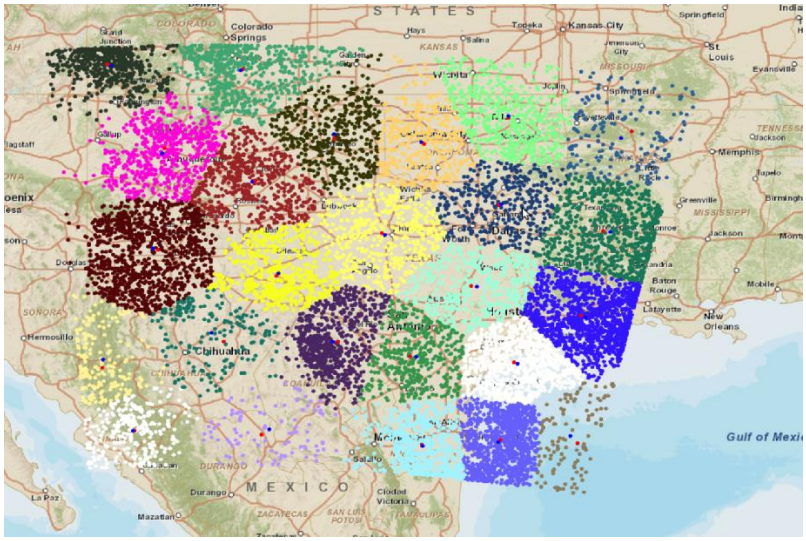


Figure 5.2 Visualizing cluster group with 25 clusters.

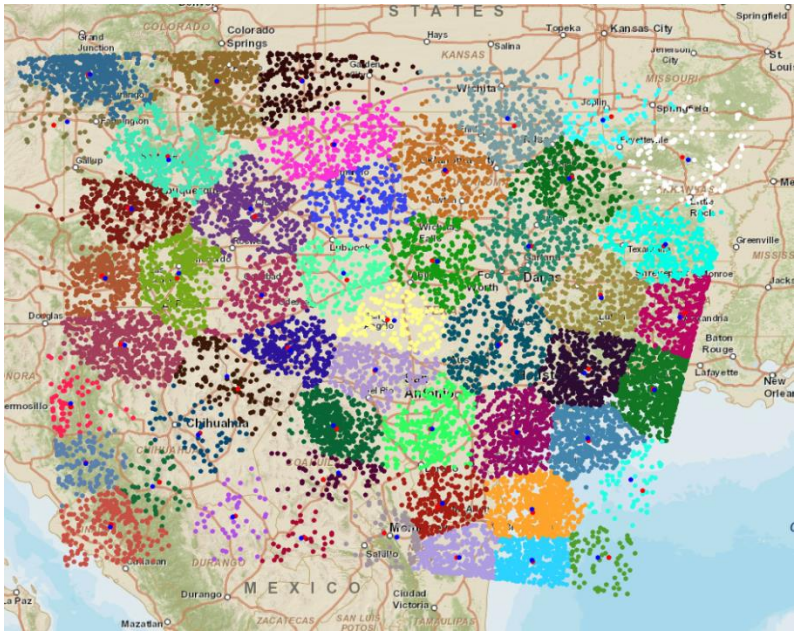


Figure 5.3 Visualizing cluster group with 100 clusters.

Apart from the members, each cluster contains 2 points; the blue colored point represents the center of the cluster and the red colored point represents the centroid of the cluster. These 2 points have already been defined in Section 4.1. Also, we can see that the points of a cluster do not overlap with any other cluster. Table 5.1 provides the mean cluster size for cluster groups with different cluster group size and shows that the mean cluster size decreases when the cluster group size increases.

Table 5.1 Mean Cluster Size (MCS) for different clusters groups

TrDY	CGS	MCS
2004	5	343.6
2004	10	245.9
2004	25	157.16
2004	50	102.46
2004	75	76.70
2004	100	63.13

Table 5.2 Mean Diameter Distance (MDD) for different clusters groups

TrDY	CGS	MDD
2004	5	1059.45
2004	10	699.97
2004	25	428.97
2004	50	296.60
2004	75	236.12
2004	100	198.91

Also, from Table 5.2, which provides Mean Diameter Distance for cluster groups with different cluster group size, we can see that of the average diameter decreases when the cluster group size increases.This is because in hierarchical clustering at each iteration clusters are created by merging two existing clusters which represent 2 spatially closer regions. Since the clusters do not overlap, the cluster size of the new cluster will be the sum of the cluster size of the individual clusters that were merged to create the new cluster. This is also evident from the map visuals as we can see that each cluster in Figure 5.1 can be spatially broken down to get the set of clusters in Figure 5.2 and in Figure 5.3.

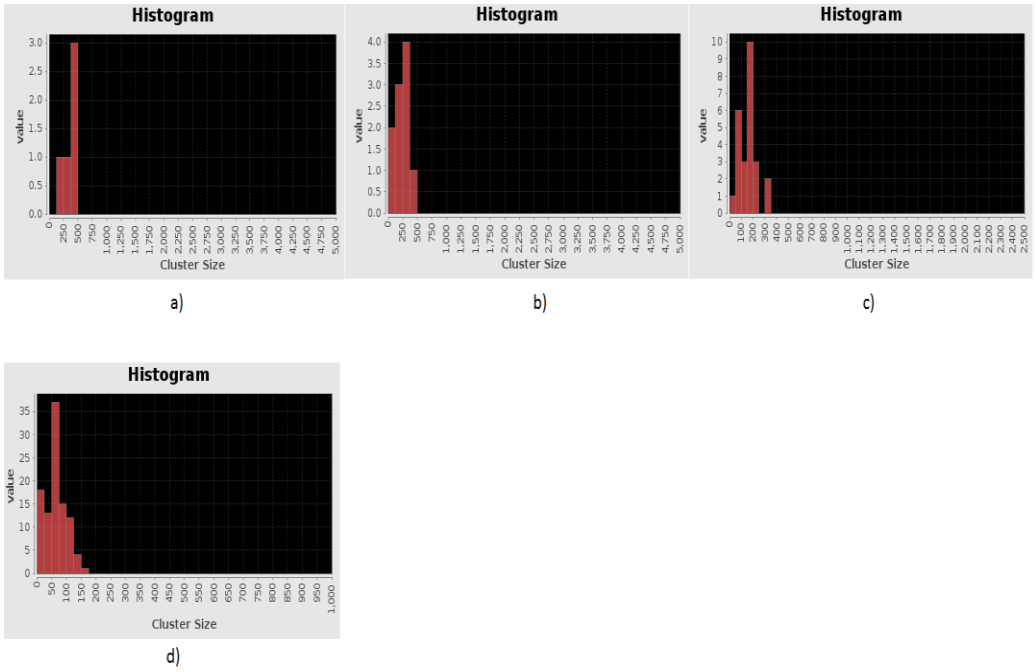


Figure 5.4 CS for CGS with a) 5 clusters b) 10 clusters c) 25 clusters d) 100 clusters

The histograms of cluster size for cluster groups with different cluster group size are provided in the Figure 5.4 . From these histograms we can see that in the cluster group with 100 clusters almost all the clusters have the same cluster size but one of the cluster sizes is less than 10. But in the cluster group with 25 or less clusters the minimum cluster size is always greater than 50. Also, from these histograms we can see that cluster size is not uniformly distributed for clusters with smaller cluster group size. Also, the cluster visualization in Figure 5.1, Figure 5.2, Figure 5.3 shows that the cluster group with 25 clusters and 100 clusters have more than one cluster that is less dense while cluster group with 5 clusters has no significantly less dense clusters.

We should remember that the prediction model is built by learning through the information present in the clusters. A cluster with fewer points contains less internal structure information, which will require more external information that can be obtained from the transition between clusters. We will discuss prediction results in more detail by

comparing the results of models built using cluster groups with 5, 10 and 25 clusters. We do not choose larger cluster groups size values because cluster groups with 50 or more clusters have at least one cluster with cluster size less than 100.

Analyzing Prediction Results

The MM and HMM are built using the cluster trajectories obtained from these cluster groups. The procedure for building the prediction models has already been discussed in Chapter 4. In the MM the distance between the center of the predicted cluster $L_{c_{predicted}}$ and the point P_{n+1} is used to evaluate the model, while in the HMM evaluation is performed by using the distance between the point $P_{predicted}$ and point P_{n+1}. This distance is called Distance Accuracy (DA). A lower DA value means the predicted observation is closer to the actual observation and represents a good prediction while a higher distance value represents a bad prediction. This distance value is found for each transition in all the trajectories of the testing data. Then the mean of these distances is calculated as Mean Distance Accuracy (MDA). Table 5.3 provides the number of transitions evaluated for each year. For example, while evaluating year 2008 the DA value for 13,231 transitions are found and their average is considered as the MDA value of that year.

Table 5.3 Number of transitions (NT) evaluated in each year

TDY	NT
2004	14138
2005	13408
2006	13026
2007	14717
2008	13231

Analyzing MM Prediction Results

Table 5.4 provides the results obtained using a MM with cluster group sizes of 5, 10 and 25. Figure 5.5 provides histograms for the prediction results obtained by

predicting the storms of the year 2005 using the Markov Model built from data of the year 2004 for different cluster groups.

Table 5.4 Mean Distance accuracy for MM built using different CGS values

CGS=5		CGS=10		CGS=25	
TDY	**MDA**	**TDY**	**MDA**	**TDY**	**MDA**
2004	309.22	2004	225.83	2004	166.94
2005	316.08	2005	221.83	2005	158.81
2006	314.50	2006	220.00	2006	162.68
2007	319.00	2007	235.56	2007	176.83
2008	340.41	2008	230.52	2008	169.28

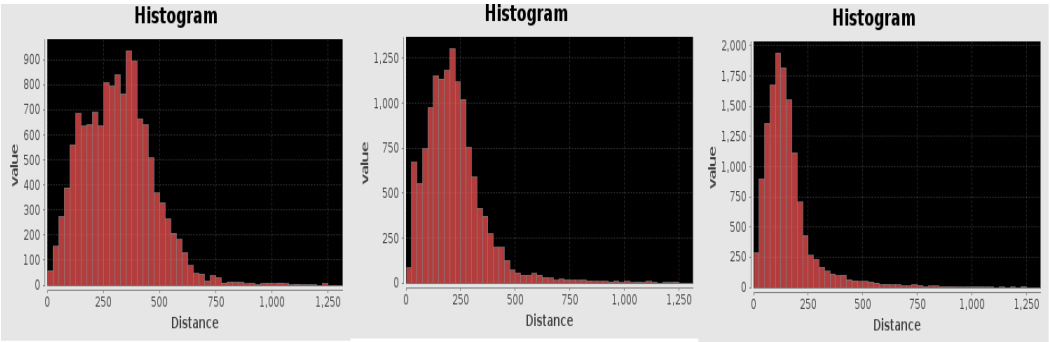


Figure 5.5 Histogram of distance accuracy for predicting 2005 data using MM built with 2004 data with a) 5 b) 10 c) 25 clusters

From Table 5.4 we can see that the MM with a cluster group with 5 clusters has a mean distance value of 310 km for year 2005 while the cluster group size of 25 clusters has a mean distance value of 122 km for the same year. Also, from the histogram we can see that the peak distance value decreases for the cluster group with larger cluster group size. The peak for MM built using 5 clusters is around 300 while for the one built using 10 clusters and 25 clusters it is around 150. This is because the diameter of the clusters

decreases with cluster group size. This means predictions are better for cluster groups with more clusters. However, it should also be noted from the histograms that a distance value greater than 1000 km can still be found for all groups with almost the same frequency. The main reason for these distance values is that certain storms move fast occasionally and could be considered as outliers. The prediction model can never predict outliers since the probability of these longer movements is very low as we can see from the histogram.

Analyzing HMM Prediction Results

The main aim of using HMMs is to increase the prediction accuracy by increasing the probability of all trajectories in the given set of trajectories. This can be achieved by increasing the number of states. Table 5.5 provides the log likelihood values of models with different cluster group sizes and different INS values.

Table 5.5 Sum of log likelihood of trajectories for HMMs with different CGS and INS values

TrDY	CGS	INS	LL
2004	5	1	-165225.9622
2004	5	3	-151794.8092
2004	5	5	-146218.3089
2004	10	1	-157799.4102
2004	10	3	-145722.2486
2004	10	5	-140674.7263
2004	25	1	-149333.5611
2004	25	3	-138114.3425
2004	25	5	-134002.1491

It can be seen from the table that the log likelihood value increases with the number of instances. For example, the log likelihood value of a cluster group with 10 clusters and 1 instance is -157799.4102 which increases with 3 instances to -145722.24 and further increases to -140674.72 with 5 instances. Similarly with larger clusters groups

sizes the log likelihood values increases with number of instances. This means with more number of instances the probability of trajectories increases. This could be because the observation probability of an observation increase with more states as the distribution of observations in each state would have been created using fewer observations that are closer to each other. From the table we can see that the log likelihood of the model with 10 clusters and 3 instances is greater than the log likelihood of the model with 5 clusters and 5 instances or 25 clusters and 1 instance.

Table 5.6 provides the results obtained for experiments using a HMM with Cluster Group Sizes of 5, 10, 25, INS values of 1, 3, 5 for each cluster groups and K=5.

Table 5.6 Mean Distance accuracy for HMM with different CGS and INS with K=5

CGS=5			CGS=10			CGS=25		
TDY	**INS**	**MDA**	**TDY**	**INS**	**MDA**	**TDY**	**INS**	**MDA**
2004	1	304.15	2004	1	221.96	2004	1	165.77
2005	1	306.62	2005	1	217.89	2005	1	158.21
2006	1	305.95	2006	1	217.02	2006	1	161.98
2007	1	309.06	2007	1	232.76	2007	1	175.36
2008	1	328.08	2008	1	227.52	2008	1	167.61
2004	3	198.57	2004	3	163.56	2004	3	133.69
2005	3	193.69	2005	3	158.11	2005	3	128.03
2006	3	196.37	2006	3	159.93	2006	3	132.67
2007	3	209.66	2007	3	171.11	2007	3	145.78
2008	3	204.64	2008	3	164.70	2008	3	138.18
2004	5	168.40	2004	5	146.27	2004	5	125.99
2005	5	163.23	2005	5	140.54	2005	5	122.35
2006	5	163.44	2006	5	143.87	2006	5	126.91
2007	5	178.91	2007	5	157.16	2007	5	136.33
2008	5	172.19	2008	5	150.80	2008	5	132.71

From Table 5.6 we can see that the results of a HMM with a single state instance (INS=1) is similar to the results of the MM for the same cluster group. This is because in a HMM with a single instance, the states represent the clusters, which means the states are observable, thereby making the model identical to a MM. Figure 5.6 provides the

histograms for the prediction results obtained by predicting the storms of the year 2005 using the model built from data of the year 2004 for cluster groups with 5 clusters and different numbers of instances.

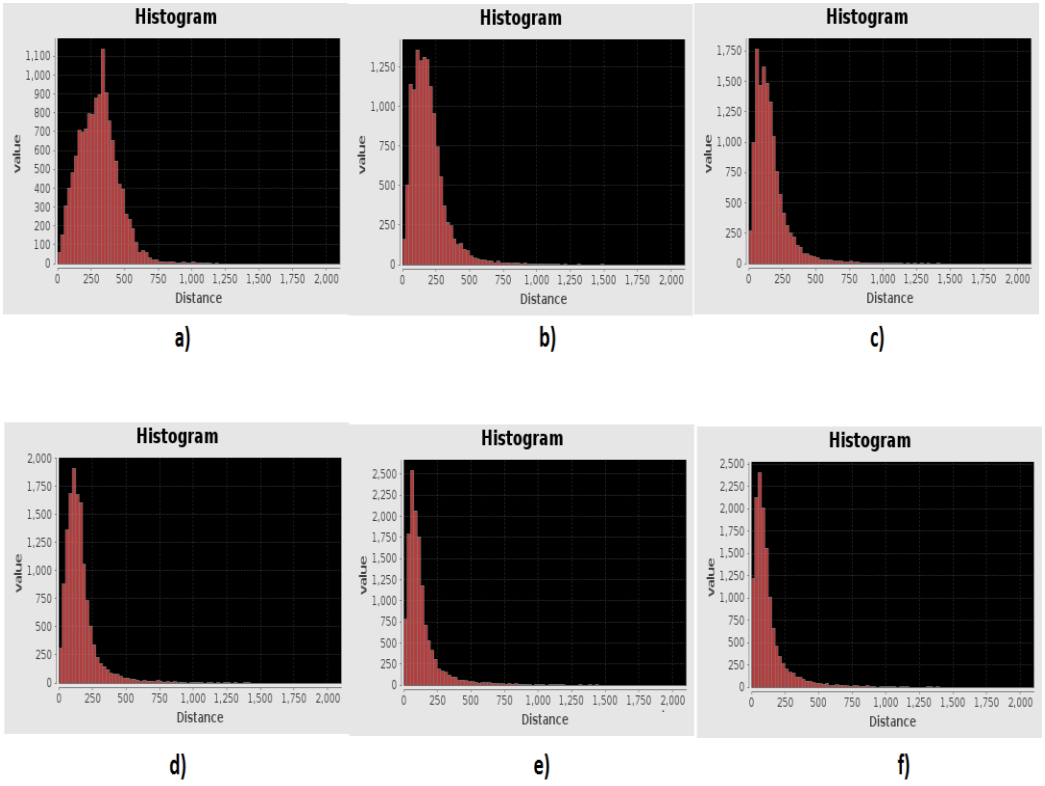


Figure 5.6 Histogram of distance accuracy for predicting 2005 data using HMMs built from 2004 data with a) CS=5 and INS=1 b) CS=5 and INS=3 c) CS=5 and INS=5 d) CS=25 and INS=1 e) CS=25 and INS=3 f) CS=25 and INS=5

However, the distance measurements are different for MM and HMM. The HMM with 5 clusters and 1 instance has a mean distance of 309 km while the HMMs for the same cluster group size with 3 and 5 instances have mean distance of 198 km and 165 km, respectively. Hence with more instances, the HMM provides better prediction.

However, there are still distance values greater than 1000 km with almost the same frequency even for larger INS values. This means that outliers are still not predictable even with a large number of instances. Table 5.7 provides the results obtained using the same values of INS and Cluster Group Size but with a value of K=100.

Table 5.7 Mean Distance Accuracy for HMMs with different CGS and INS values with K=100

CGS=5			CGS=25		
TDY	**INS**	**MD**	**TDY**	**INS**	**MD**
2004	1	304.178	2004	1	166.0644
2005	1	306.7934	2005	1	158.3079
2006	1	305.8073	2006	1	162.2134
2007	1	309.0883	2007	1	175.4753
2008	1	327.9021	2008	1	167.8482
2004	3	200.3555	2004	3	143.3752
2005	3	196.3853	2005	3	138.471
2006	3	199.7797	2006	3	142.7885
2007	3	212.306	2007	3	156.9269
2008	3	209.4565	2008	3	148.1394
2004	5	187.6726	2004	5	142.9421
2005	5	184.3579	2005	5	140.1061
2006	5	185.5374	2006	5	144.2488
2007	5	199.1041	2007	5	158.7434
2008	5	193.1906	2008	5	149.7333

From this table we can see that the HMM built using 25 clusters and 3 instances has almost the same results as a HMM built using the same cluster group size and 5 instances. This is evident from the table as we can see the MDAs of the HMMs with 25 clusters and 5 instances and 3 instances for testing year 2005 are 138.471 km and 140.1061 km respectively. On the other hand the same experiment with K=5 shows improvement in prediction when the number of instances is increased. This is because with a larger cluster group size the weights of certain states become low and the uniform distribution dominates the Gaussian distribution of the states. Hence for larger Cluster Group Size and higher K values, the instances of the clusters are dominated by the

uniform distribution. This means the instances do not contain any structural information from the cluster, hence reducing the benefit of a larger number of instances.

Analyzing Prediction Capability Over Years

The main reason for evaluating each year separately is to analyze the prediction capability of the model over years. From Table 5.4 and Table 5.6 we can see that both models, MM and HMM, built using 2004 data, for the same values of INS, K and CGS, provide very almost similar results for all the years, including 2004. HMM built using 5 clusters, 3 instances and K=5 gives a mean of 198 km for the year 2004 while the mean for 2005 using the same model is 198.5 km. Also, the mean distance values obtained using a HMM and a MM with the same CGS, INS values and K value are in the range of 198-210 km for all the years.

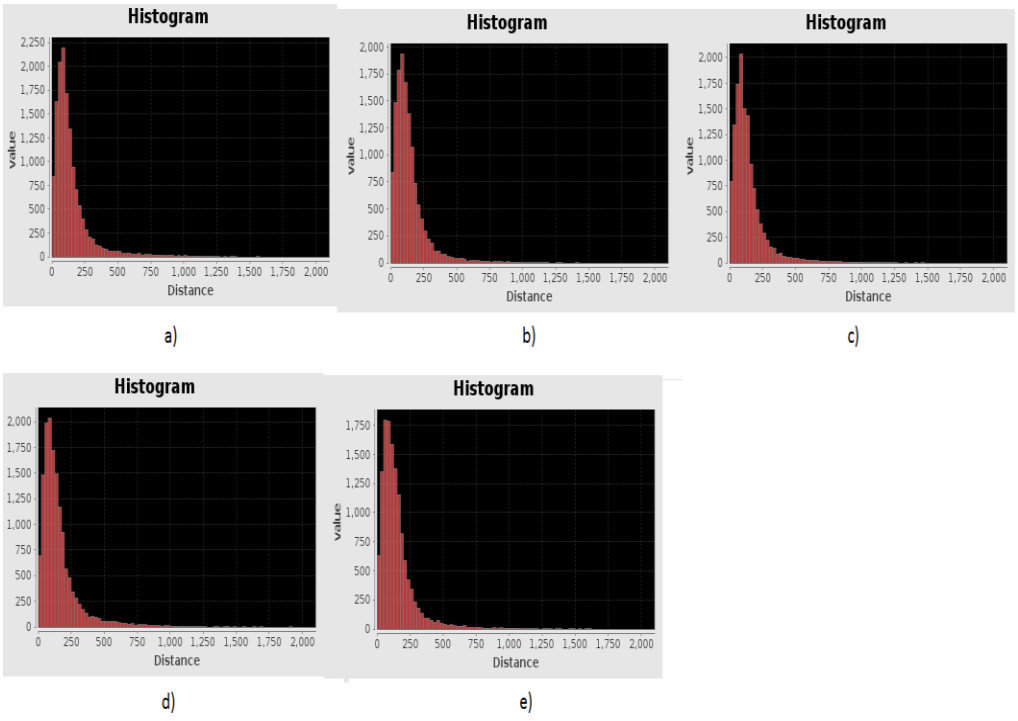


Figure 5.7 Histograms of distance accuracy measured for predicting years a) 2004 b) 2005, c) 2006 d) 2007 e) 2008 using the HMM built from the 2004 data with 10 clusters and 5 instances

This similarity is found for all the cluster group sizes and INS values, in both the MM and HMM. Figure 5.7 provides the histograms of mean distance values predicted by HMM with INS=5, cluster group size=10 and K=5, for different years. From this histogram we can see that the peak values and the height of the peak values are in the range of 50-100 km for all the years including 2004 (training data). Also, the distribution of the values is almost equal for all the years since they have a similar bell curve. Hence, from these results it seems that the model trained using a single year has similar prediction ability for all the years, including the training year. However, we would need more results of other years to derive a conclusion on the prediction capability of the model over years.

Prediction With No Intra-Cluster Transitions

It should be noted that the storms were moving very slow, which resulted in very high intra-cluster transition probabilities. A prediction model with higher intra-cluster transition will predict the next probable observation as an observation which belongs to the cluster of the current observation. This means that the model predicts that a storm did not move out of the cluster. Hence, in order evaluate the movement of storms better we are performing experiments with zero intra cluster transitions. This approach is similar to the works [7] [19], in which only the transitions that move out of the cluster are considered and all the transitions that remain in the same cluster are thrown out. This is done by preprocessing the cluster trajectories by replacing the consecutive observations that belong to the same cluster with one single observation in the cluster. While this removes the intra-cluster transitions, it also results in significant loss of data since most of the transition remain in the same cluster. The reduction in the number of transitions for each year is given in Table 5.8. For example, preprocessing the trajectories for data of the year 2004 with 5 clusters gave us in almost 89.76% reduction in data. This reduction in data reduces when the number of clusters in a cluster group increases. This is

because with more clusters the diameter of the cluster decreases and hence more storms move into other clusters.

Table 5.8 Loss of data for each year for different CGS

TDY	CGS	NT	RNT	%Reduction
2004	5	14138	1447	89.7651719
2004	10	14138	2115	85.0403169
2004	25	14138	3453	75.5764606
2004	50	14138	4563	67.7252794
2004	75	14138	5164	63.4743245
2004	100	14138	5709	59.6194653

The results for the prediction with no intra cluster transitions are given in Table 5.9 for the Markov Model and in Table 5.10 for the HMM. It can be seen that the mean distance value for MM still decreases with increase in cluster size however the value is greater than the value with intra-cluster transitions. Table 5.9 shows that the value for a cluster group with 5 clusters is 700-750 for all the years whiles the values for the same cluster group is 300-330 in the Table 5.4.The same results can be found for different cluster groups.

Table 5.9 Mean Distance Accuracy for a MM with no intra cluster transition

CGS=5		CGS=10		CGS=25	
TDY	MDA	TDY	MDA	TDY	MDA
2004	731.58	2004	564.15	2004	340.00
2005	735.81	2005	560.99	2005	341.42
2006	756.95	2006	578.15	2006	342.23
2007	741.22	2007	572.20	2007	353.84
2008	767.29	2008	589.62	2008	353.05

From Table 5.10 we can find that the mean distance value decreases with increase in the number of instances too. However the prediction using an HMM with no

intra cluster transitions is worse than the one with intra cluster transitions as found in Table 5.6. The main reason for this is that most storms move slow and they stay in the same cluster. Hence when the intra transition probability is zero most predictions are not accurate due to loss of data. However, the prediction improves with a larger number of clusters and instances.

Table 5.10 Mean Distance Accuracy for HMM with no intra cluster transition

CGS=5			CGS=10			CGS=25		
TDY	**INS**	**MDA**	**TDY**	**INS**	**MDA**	**TDY**	**INS**	**MDA**
2004	1	734.55	2004	1	497.75	2004	1	327.19
2005	1	756.46	2005	1	499.75	2005	1	326.34
2006	1	783.84	2006	1	502.62	2006	1	330.72
2007	1	764.47	2007	1	507.85	2007	1	340.30
2008	1	789.18	2008	1	528.16	2008	1	348.46
2004	3	436.34	2004	3	315.37	2004	3	220.32
2005	3	441.79	2005	3	309.17	2005	3	211.61
2006	3	464.29	2006	3	310.95	2006	3	216.33
2007	3	462.85	2007	3	312.84	2007	3	232.13
2008	3	463.76	2008	3	325.16	2008	3	223.69
2004	5	415.02	2004	5	285.50	2004	5	213.38
2005	5	419.98	2005	5	280.94	2005	5	205.84
2006	5	430.82	2006	5	284.43	2006	5	210.92
2007	5	433.25	2007	5	293.46	2007	5	228.22
2008	5	428.41	2008	5	300.61	2008	5	218.89

Predicting 10 Hours in the Future

We also conducted experiments for predicting the location of a storm after 10 hours. This is done using Equation (38) explained in Section 4.4. It should be noted that in order to predict 10 hours in future the point at the10th hour is required. In a given trajectory of length n we can evaluate only n-10 transitions. Hence the number of transitions evaluated for every year is reduced and is given in Table 5.11.

Table 5.11 Number of transitions evaluated for each year

TDY	NT
2004	7721
2005	6820
2006	6645
2007	7976
2008	6994

The results for predicting 10 hours in the future are provided in Table 5.12 for the Markov Model and Table 5.13 for the HMM.

Table 5.12 Mean Distance Accuracy of MM for predicting in 10 hour future

CGS=5		CGS=10		CGS=25	
TDY	MDA	TDY	MDA	TDY	MDA
2004	432.71	2004	402.69	2004	391.97
2005	412.34	2005	390.15	2005	377.34
2006	423.79	2006	375.49	2006	365.19
2007	448.39	2007	415.04	2007	396.20
2008	464.57	2008	408.60	2008	365.69

Table 5.13 Mean Distance Accuracy of HMM for predicting in 10 hour future

CGS=5			CGS=10			CGS=25		
TDY	INS	MDA	TDY	INS	MDA	TDY	INS	MDA
2004	1	392.61	2004	1	350.12	2004	1	327.97
2005	1	382.69	2005	1	337.99	2005	1	308.26
2006	1	374.31	2006	1	326.81	2006	1	306.22
2007	1	392.19	2007	1	353.18	2007	1	336.88
2008	1	406.36	2008	1	338.40	2008	1	316.16
2004	3	338.83	2004	3	330.14	2004	3	320.64
2005	3	321.11	2005	3	313.34	2005	3	301.92
2006	3	317.67	2006	3	305.40	2006	3	297.62
2007	3	347.39	2007	3	331.95	2007	3	328.91
2008	3	333.06	2008	3	314.89	2008	3	312.46
2004	5	328.14	2004	5	323.14	2004	5	317.90
2005	5	307.99	2005	5	304.80	2005	5	298.90
2006	5	304.77	2006	5	298.28	2006	5	297.57
2007	5	338.98	2007	5	333.39	2007	5	329.01
2008	5	316.8	2008	5	313.72	2008	5	311.61

From Table 5.12 and Table 5.13 we can still find that prediction improves significantly for larger values of cluster group size and INS values. The mean distance value for predicting 10 hour in the future is very high compared to the mean distance value of predicting 1 hour in the future, which means that predicting further in the future does give less precise results. This is because the uncertainty in predicting further hours in the future is higher than predicting the next hour.

Prediction Using Directional Attribute

The directional parameters were calculated as explained in Section 4.5. The directional parameters were included so that we can differentiate between storms that move in different directions and hence improve prediction. However, it was found from our experiments that the directional parameters were not improving prediction. We discuss more on the reasons about this problem in this section.

First, clusters were created using weights as W_S=0.7 and W_D=0.3 in Equation (5). Then the HMM was created assuming a 3D multivariate Gaussian distribution. This model predicts a 3D point that consists of both the directional and spatial parameters and the model is evaluated by finding the spatial distance for the spatial parameters and linear modular difference for the directional parameter. These experiments were conducted by assuming the directional attribute as a wrapped parameter as well as an unwrapped parameter.

By assuming the directional parameter as an unwrapped parameter the average spatial distance between the actual position and predicted position was found as 320-340 km and the average angular difference between the actual angle and predicted angle was found as 100-104 degrees for a HMM built with a cluster group of 5 clusters and 1 instance for all the years. Also, the average spatial distance was 220 km for a model with 5 clusters and 3 instances while the average angular difference remained almost the

same. Similarly, for different cluster groups and different numbers of instances there was no improvement in directional prediction while the spatial prediction improved. By assuming the directional parameter as a wrapped parameter, the average spatial distance improved while the angular difference remained the same for different cluster groups and numbers of instances. However the value of the directional difference lowered only to 74 degrees.

It should be noted that the clusters created using the above mentioned weights were spatially biased. Hence we gave more weights for the directional attribute by creating clusters using weights W_S=0.2 and W_D=0.8 in Equation (5) and then create a HMM using these clusters. Even with this more angular biased cluster the average directional distance did not improve for models with a larger number of clusters or more instances. One reason could have been the number of instances used was relatively small. This is because the range of the spatial attributes is very large compared to the angular value range of 0-360 degrees and the instances could represent the directional variations only after representing the spatial variations present in the cluster. However, we keep further investigation of this for future work.

The main aim of the research is to show the importance of learning the spatial and transitional information present in the clusters created using points in the storm trajectories, which is done by the Baulm-Welch algorithm.

Chapter 6

Conclusion and Future Work

In this research we propose a model for predicting rainfall storm movement using HMMs by considering states that are related to clusters obtained by clustering points in the overall storm trajectories. We then learn the spatial and transitional information present in these states using the Baum-Welch algorithm. First we build a simple Markov Model in which states correspond to clusters and then extend it to HMMs in which a cluster is represented using multiple state instances. From the results of clustering we understand that the clusters in cluster groups with larger numbers of clusters are more dependent on external information provided by transitions, as they have a smaller number of points than clusters in cluster groups with a smaller number of clusters. We compare the prediction results of models with different numbers of clusters. The prediction results are evaluated by finding the average distance between the actual point and the predicted point for all the transitions present in the training data. From the prediction results of the Markov Model we can see that the prediction gets better with a large number of clusters. The prediction results of HMMs shows that prediction improves when the number of state instances representing each cluster increases and also when the number of clusters increases. However, for a larger value of K, which decides the proportion of noise added to the observation distribution of each state, there was no improvement in prediction for HMMs built using a larger number of clusters. A comparison between prediction results of MMs and HMMs shows that the MM is similar to a HMM with one instance. The improvement in prediction with a larger number of instances and clusters is also found when predicting a point 10 hours in the future However, the accuracy is not as good as predicting 1 hour in the future as the uncertainty increases as we move further into the future. Also, the same improvement in prediction is

found for a model built using trajectories that only contain storms that move to another cluster but do not stay in the same cluster. However these models are not suited for performing prediction as most of the storms stay in the same cluster resulting in huge loss of data. The prediction results of HMM and MM show that a model built using one year of data can predict the storm movement in all the years with similar prediction accuracy.

Apart from predicting location we have also conducted experiments for predicting the direction of the storms. However the results of HMMs and MMs show that the direction prediction did not improve either with an increased number of instances or clusters

In addition to introducing a HMM-based prediction model we also propose an approach for deriving the overall storm trajectories form hourly storms that is much faster than the existing overall storm approach proposed by Jitkajornwanich et. al. [1].

In the future we would like to develop better methods for predicting directional parameters by building models from more than one year of data with a larger number of clusters and instances so that there are enough states to represent the directional variations. Also, we will perform experiments by considering a semi-wrapped distribution instead of an approximated wrapped Gaussian. We can also extend this approach for predicting non-spatial parameters such as precipitation, the number of sites, and so on, since we can get better prediction with more information.

References

[1] Kulsawasd Jitkajornwanich et al., "Complete Storm Identification Algorithms from Big Raw Rainfall Data," in *IEEE International Conference on Big Data*, 2013, pp. 13 - 20.

[2] NOAA's National Weather Service. (2011, December) The XMRG File Format and Sample Codes to Read XMRG Files. [Online]. http://www.nws.noaa.gov/oh/hrl/dmip/2/xmrgformat.html

[3] M. Minsky and S. Papert, *An Introduction to Computational Geometry*.: MIT Press, 1969.

[4] Lawrence R Rabiner , "A tutorial on hidden Markov Models and selected applications in speech recognition," *Proceedings of the IEEE* , vol. 77, no. 2, pp. 257 - 286, Feb 1989.

[5] D. S. Poskit, "Autoregressive approximation in nonstandard situations: the fractionally integrated and non-invertible cases," *Annals of the Institute of Statistical Mathematics*, vol. 59, no. 4, pp. 697–725, 2007.

[6] Md. Rafiul Hassan and Baikunth Nath, "Stock Market Forecasting Using Hidden Markov Model:," in *International Conference on Intelligent Systems Design and Applications*, 2005.

[7] Sebastien Gambs and Marc-Olivier Killijian, "Next Place Prediction using Mobility Markov Chains," in *Proceedings of the First Workshop on Measurement, Privacy, and Mobility (MPM '12)*, 2012.

[8] S. Jung, C. Kim, and Y. Chung, "A Prediction Method of Network Traffic Using Time Series Models," in *Computational Science and Its Applications - ICCSA 2006*.: Springer Berlin Heidelberg, 2006, pp. 234-243.

[9] Kumar Abhishek, Abhay Kumar, and Sarthak Kumar, "A Rainfall Prediction Model using Artificial Neural Network ," in *IEEE Control and System Graduate Research Colloquium (ICSGRC 2012)*, 2012, pp. 82-87.

[10] Adi Nugroho and Bistok Hasiholan Simanjuntak, "ARMA (Autoregressive Moving Average) Model for Prediction of Rainfall in Re gency of Semarang - CentralJava - Republic of Indonesia," *IJCSI International Journal of Computer Science* , vol. 11, no. 3, pp. 27-32, May 2014.

[11] Dedetemo Kimilita Patrick, Phuku Phuati Edmond, Tshitenge Mbwebwe Jean-Marie, Efoto Eale Louis, and Koto-te-Nyiwa Ngbolua, "Prediction of rainfall using autoregressive integrated moving average model: Case of Kinshasa city (Democratic Republic of the Congo), from the period of 1970 to 2009," in *Journal of Computation In Biosciences And Engineering*, 2014.

[12] S.P. Meyn and R.L. Tweedie, *Markov Chains and Stochastic Stability*.: Springer 2005, 2005.

[13] Reid Andersen, Fan Chung, and Kevin Lang, "Local Partitioning for Directed Graphs Using PageRank," in *Algorithms and Models for the Web-Graph*, Anthony Bonato and Fan R. K. Chung, Eds.: Springer Berlin Heidelberg, 2007, pp. 166-178.

[14] Junji YAMATO, Jun OHYA, and Kenichiro ISHII, "Recognizing Human Action in Time-Sequent ial Images using Hidden Markov Model," in *Proceedings CVPR '92.*, 1992, pp. 379-385.

[15] Sergey Kirshner, "Modeling of Multivariate Time Series Using Hidden Markv Models," UNIVERSITY OF CALIFORNIA,IRVINE, Dissertation 2005.

[16] Hoyoung Jeung, Heng Tao Shen, and Xiaofang Zhou, "Mining Trajectory Patterns Using Hidden Markov Models," in *DaWaK 2007*, 2007, pp. 470-480.

[17] Pratap S Prasad and Prathima Agrawal, "Movement Prediction in Wireless Networks Using Mobility Traces," in *IEEE Consumer Communications and Networking Conference (CCNC), 2010*, 2010, pp. 1 - 5.

[18] Yen-Cheng Lu, Feng Chen, and Chang-Tien-Lu, "Autoregressive HMM Based Trajectory Modeling on Social Network," Virginia Tech, Research Presentation 2012.

[19] Wesley Mathew, Ruben Raposo, and Bruno Martins, "Predicting Future Locations with Hidden Markov Models," in *Ubiquitous Computing*, 2012.

[20] D.B. PAUL, "Speech Recognition Using Hidden Markov Models," in *The Lincoln Laboratory Journal*, 1990.

[21] Sovan Mitra and Paresh Date, "Regime switching volatility calibration by the Baum-Welch method," in *Journal of Computational and Applied Mathematics*, 2010, pp. 3243-3260.

[22] F., Falavigna, D. and Omologo, M. Brugnara, "Automatic segmentation and labeling of speech based on hidden Markov Models," in *Speech Communication*, 1993, pp. 357–370.

[23] D.T. Toledano, L.A.H. Gomez, and L.V Grande, "Automatic phoneme segmentation," in *IEEE Transactions Speech and Audio Proceedings*, 2003, pp. 617-625.

[24] J. Yuan and M. Liberman, "Speaker identification on the SCOTUS corpus," in *Proceedings of Acoustics*, 2008, pp. 5687-5690.

[25] David Huggins-Daines and Alexander I. Rudnicky, "A Constrained Baum-Welch Algorithm for Improved Phoneme Segmentation and Efficient Training," in *INTERSPEECH, ISCA.*, 2006.

[26] Kulsawasd Jitkajornwanich, Upa Gupta, Ramez Elmasri, Leonidas Fegaras, and John McEnery, "Using MapReduce to Speed Up Storm Identification from Big Raw Rainfall Data," in *Proceedings of the 4th International Conference on Cloud Computing, GRIDs, and Virtualization (CLOUD COMPUTING'13)*, 2010, pp. 49-55.

[27] Kulsawasd Jitkajornwanich, Chengkai Li, Ramez Elmasri, and John McEnery, "Extracting storm-centric characteristics from raw rainfall data for storm analysis and mining," in *Proceedings of the 1st ACM SIGSPATIAL International Workshop on Analytics for Big Geospatial Data* , 2012, pp. 91-99.

[28] National Oceanic and Atmospheric Administration (NOAA). (2011, December) National Weather Service River Forecast Center: West Gulf RFC (NWS-WGRFC). [Online]. http://www.srh.noaa.gov/wgrfc/

[29] J. McEnery. (2011, DECEMBER) CUAHSI HIS: NWS-WGRFC Hourly Multi-sensor Precipitation Estimates. [Online]. http://hiscentral.cuahsi.org/pub_network.aspx?n=187

[30] J. Dean and S. Ghemawat, "MapReduce: Simplified Data Processing on Large Clusters," in *Proceedings of the 6th Symposium on Operating Systems Design and Implementation (OSDI'04)*, 2004.

[31] C. Lam, *Hadoop in Action*. New Delhi: Dreamtech Press, 2011.

[32] W. H. Asquith, M. C. Roussel, T. G. Cleveland, X. Fang, and D. B. Thompson, "Statistical Characteristics of Storm Interevent Time, Depth, and Duration for Eastern New Mexico, Oklahoma, and Texas," Professional Paper 1725. U.S. Geological Survey 2006.

[33] Trevor Hastie, Robert Tibshirani, and Jerome Friedman, "Hierarchical clustering," in *The Elements of Statistical Learning (2nd ed.).*: Springer, 2009, ch. 14, pp. 520–528.

[34] David J.C. MacKay, *The on-line textbook: Information Theory, Inference, and Learning Algorithms*, 4th ed.: Cambridge University Press, 2003.

[35] J. MacQueen, "Proc. Fifth Berkeley Symp. on Math. Statist. and Prob.," in *Proc. Fifth Berkeley Symp. on Math. Statist.*, 2008, pp. 281-297.

[36] R. J. G. B. Campello, D. Moulavi, and J. Sander, *Density-Based Clustering Based on Hierarchical Density Estimates*.: Springer Berlin Heidelberg, 2013.

[37] Mihael Ankerst, Markus M. Breunig, Hans-Peter Kriegel , and Jörg Sander, "OPTICS: ordering points to identify the clustering structure," in *Proceedings of the 1999 ACM SIGMOD international conference on Management of data* , 1999, pp. 49-60.

[38] Environmental Systems Research Institute ESRI. (2014) ArcGIS. [Online]. https://www.arcgis.com/features/

[39] Alan Genz and Frank Bretz, *Computation of Multivariate Normal and t Probabilities*.: Springer Berlin Heidelberg, 2009.

[40] Miguel Á. Carreira-Perpiñán, "Mode-Finding for Mixtures of Gaussian Distributions," in *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2000, pp. 1318-1323.

[41] Claus Bahlmann, "Directional features in online handwriting recognition," *Journal Pattern Recognition*, vol. 39, no. 1, pp. 115-125, January 2006.

[42] Simone Calderara, Andrea Prati, and Rita Cucchiara, "Learning People Trajectories using Semi-directional Statistics," in *Advanced Video and Signal Based Surveillance*, 2009, pp. 214-218.

Biographical Information

Sakthi Kumaran Shanmuganathan completed his Bachelor of Engineering in Computer Science and Engineering from Jeppiaar Engineering College which is affiliated with Anna University, Chennai, India in 2010. He joined University of Texas Arlington in 2012 to pursue his Master of Science in Computer Science. His research interest is in the fields of Machine Learning and Distributed Computation.