

University of Nebraska - Lincoln

DigitalCommons@University of Nebraska - Lincoln

---

Computer Science and Engineering: Theses,  
Dissertations, and Student Research

Computer Science and Engineering, Department of

---


8-2019

# Analysis of Social Unrest Events using Spatio-Temporal Data Clustering and Agent-Based Modelling

Sudeep Basnet

University of Nebraska - Lincoln, [sbasnet@huskers.unl.edu](mailto:sbasnet@huskers.unl.edu)

Follow this and additional works at: <https://digitalcommons.unl.edu/computerscidiss>

 Part of the [Computer Sciences Commons](#), [Quantitative, Qualitative, Comparative, and Historical Methodologies Commons](#), and the [Social Control, Law, Crime, and Deviance Commons](#)

---

Basnet, Sudeep, "Analysis of Social Unrest Events using Spatio-Temporal Data Clustering and Agent-Based Modelling" (2019). *Computer Science and Engineering: Theses, Dissertations, and Student Research*. 176.  
<https://digitalcommons.unl.edu/computerscidiss/176>

This Article is brought to you for free and open access by the Computer Science and Engineering, Department of at DigitalCommons@University of Nebraska - Lincoln. It has been accepted for inclusion in Computer Science and Engineering: Theses, Dissertations, and Student Research by an authorized administrator of DigitalCommons@University of Nebraska - Lincoln.

ANALYSIS OF SOCIAL UNREST EVENTS USING SPATIO-TEMPORAL DATA  
CLUSTERING AND AGENT-BASED MODELLING

by

Sudeep Basnet

A THESIS

Presented to the Faculty of  
The Graduate College at the University of Nebraska  
In Partial Fulfilment of Requirements  
For the Degree of Master of Science

Major: Computer Science

Under the Supervision of Professors Leen-Kiat Soh and Ashok Samal

Lincoln, Nebraska

August, 2019

# ANALYSIS OF SOCIAL UNREST EVENTS USING SPATIO-TEMPORAL DATA CLUSTERING AND AGENT-BASED MODELLING

Sudeep Basnet, M.S.

University of Nebraska, 2019

Adviser: Leen-Kiat Soh and Ashok Samal

Social unrest such as appeals, protests, conflicts, fights and mass violence can result from a wide ranging of diverse factors making the analysis of causal relationships challenging, with high complexity and uncertainty. Unrest events can result in significant changes in a society ranging from new policies and regulations to regime change. Widespread unrest often arises through a process of feedback and cascading of a collection of past events over time, in regions that are close to each other. Understanding the dynamics of these social events and extrapolating their future growth will enable analysts to detect or forecast major societal events. The study and prediction of social unrest has primarily been done through case-studies and study of social media messaging using various natural language processing techniques. The grouping of related events is often done by subject matter experts that create profiles for countries or locations. We propose two approaches in understanding and modelling social unrest data: (1) spatio-temporal data clustering, and (2) agent-based modelling. We apply the clustering solution to real-world unrest events with socioeconomic and infrastructure factors. We also present a framework of an agent-based model where unrest events act as intelligent agents that continuously study their environment and perform actions. We run simulations of the agent-based model under varying conditions and evaluate the results in comparison to real-world data. Our results show the viability of our proposed solutions.

## ACKNOWLEDGMENTS

Firstly, I would like to express my sincere gratitude to my advisor Prof. Leen-Kiat Soh for the continuous support of my M.S. study and related research, for his patience and motivation, and guidance throughout my research and writing of this thesis. I could not have imagined having a better advisor and mentor for my M.S. study. I would also like to thank my co-advisor Dr. Ashok Samal, his office was always open whenever I ran into a trouble spot or had a question about my research or writing. I'd also like to thank Dr. Deepti Joshi and Dr. Stephen Scott for their insightful comments and encouragement, and for steering me in the right direction whenever I needed it.

I would also like to thank my former classmates Ian Howell, Yi Liu, and Tyler Bienhoff with whom I first conceived the idea for the agent-based model used in this thesis. Finally, I express my gratitude to my parents Anil K. Basnet and Kiran Basnet, and brother Sameer Basnet for providing me with unfailing support and continuous encouragement throughout my years of study and through the process of researching and writing this thesis.

The research presented in this thesis was conducted as part of project SURGE at the University of Nebraska - Lincoln, supported by the National Geospatial-Intelligence Agency. The work presented in the thesis was completed utilizing the Holland Computing Center of the University of Nebraska, which receives support from the Nebraska Research Initiative.

## Table of Contents

<b>List of Figures</b>	<b>viii</b>
<b>List of Tables</b>	<b>xiii</b>
<b>1 Introduction</b>	<b>1</b>
1.1 Problem Overview . . . . .	1
1.2 Proposed Solution . . . . .	3
1.3 Contributions . . . . .	6
1.4 Outline . . . . .	7
<b>2 Background and Related Work</b>	<b>8</b>
2.1 Spatio-Temporal Clustering Algorithms . . . . .	8
2.2 Agent-Based Modelling . . . . .	9
2.3 Related Projects . . . . .	10
2.3.1 EMBERS . . . . .	11
2.3.2 Activism via Attention . . . . .	18
2.3.3 ICEWS . . . . .	25
2.3.4 Shifting Sands . . . . .	28
2.3.5 Other Related Projects . . . . .	32
2.3.6 Summary - Comparison with SURGE . . . . .	33

<b>3</b>	<b>Multi-Factorial Distance Function</b>	<b>37</b>
3.1	Events and Attributes . . . . .	37
3.2	Distance Calculation . . . . .	39
3.2.1	Spatio-temporal Distance . . . . .	39
3.2.2	Socio-Economic Distance . . . . .	43
3.2.3	Infrastructural Distance . . . . .	43
3.2.4	Integrated, Multi-factorial Distance . . . . .	45
3.3	Implementation Details . . . . .	46
<b>4</b>	<b>Spatio-Temporal Data Clustering</b>	<b>48</b>
4.1	Spatio-Temporal $k$ -Dimensional Tree-based DBSCAN . . . . .	48
4.2	Implementation Details . . . . .	50
<b>5</b>	<b>Agent-Based Modelling</b>	<b>55</b>
5.1	Model Design . . . . .	55
5.2	Design Principles of Considering Influence of Neighboring Agents . . . . .	58
5.2.1	Decay Principle . . . . .	59
5.2.2	Distance Principle . . . . .	59
5.2.3	Intensity Principle . . . . .	59
5.2.4	Intensity Difference Principle . . . . .	59
5.2.5	Neighborhood Principle 1 . . . . .	60
5.2.6	Neighborhood Principle 2 . . . . .	60
5.3	Influence . . . . .	60
5.4	Implementation Details . . . . .	61
5.5	Evaluation Methodology . . . . .	63
5.5.1	Evaluation using synthetic data . . . . .	63
5.5.2	Evaluation with real data . . . . .	64

5.5.2.1	Total Intensity Value Error . . . . .	64
5.5.2.2	Average Intensity Value Error . . . . .	65
5.5.2.3	Trend Error . . . . .	66
5.5.2.4	Confidence Factor . . . . .	67
5.5.2.5	Time Lag . . . . .	69
<b>6</b>	<b>Experiments and Results</b>	<b>71</b>
6.1	Data Sources . . . . .	71
6.1.1	Events Data . . . . .	72
6.1.2	Socioeconomic Data . . . . .	73
6.1.3	Infrastructure Data . . . . .	74
6.2	Spatio-temporal Data Clustering . . . . .	75
6.3	ABM Simulation . . . . .	81
6.3.1	Synthetic Data . . . . .	83
6.3.1.1	Agents with no neighbors . . . . .	83
6.3.1.2	Agents with neighbors . . . . .	83
6.3.2	Real Data . . . . .	93
6.3.2.1	Location Selection for Analysis . . . . .	93
6.3.2.2	Model Evaluation using Total Intensity . . . . .	95
6.3.2.3	Model Evaluation using Average Intensity . . . . .	102
6.4	Summaries . . . . .	130
6.4.1	The Spatio-Temporal Clustering Approach . . . . .	130
6.4.2	The Agent-Based Modelling Approach . . . . .	131
<b>7</b>	<b>Conclusions and Future Work</b>	<b>133</b>
7.1	Conclusions . . . . .	133
7.2	Future Work . . . . .	135

**Bibliography**



## List of Figures

3.1	Schematic of the integrated, multi-factorial distance function . . . . .	46
4.1	Schematic of ST-KDT-DBSCAN describing the processes involved in clustering of events data. . . . .	49
4.2	3-d tree structure for event data to enable more efficient search of events within a bounding box . . . . .	51
4.3	Spatial bounding box formation for FRNN: (a) Calculating spatial coordinates in four cardinal directions. (b) Creating spatial bounding box using the coordinates. . . . .	52
5.1	(a) Curve fit for polynomial equation of degree 2, sum of the squares of the fit errors is 1.4. (b) Curve fit for polynomial equation of degree 3, sum of the squares of the fit errors is 0.7 . . . . .	58
6.1	Geographic plots of the 3 largest clusters when using spatiotemporal distances only: (a) largest cluster, 243 events, (b) Second-largest cluster, 162 events (c) Third-largest cluster, 122 events . . . . .	76
6.2	Geographic plots of the 3 largest clusters when using spatiotemporal and socioeconomic distances: (a) largest cluster, 412 events, (b) Second-largest cluster, 368 events (c) Third-largest cluster, 129 events. . . . .	77

6.3	Geographic plots of the 3 largest clusters when using spatiotemporal, socioeconomic and infrastructure distances: (a) largest cluster, 1341 events, (b) Second-largest cluster, 405 events (c) Third-largest cluster, 399 events.	78
6.4	Time evaluation of the existing algorithm against ST-KDT-DBSCAN . .	80
6.5	Agents with no neighbors, and their corresponding intensities: (a) Day-0: agent $e_1$ , $e_2$ , $e_3$ , $e_4$ and $e_5$ have intensities 8, 1, 7, 6 and 5 respectively. (b) Day-1, all agents have half of their starting intensity. (c) Day-3: agent $e_2$ has died as its intensity reaches 0.125 (d) Day-6: all agents have died	84
6.6	Agent behavior when multiple neighbors with same intensity but at different distances are available. (a) Intensities of agents during simulation, (b) Influence on $e_1$ . . . . .	86
6.7	Agent Behavior with a Recovery Rate ( $\lambda$ ) of 0.8 and Influence Rate ( $\gamma$ ) of 0.1 . . . . .	87
6.8	Agent Behavior with a Recovery Rate ( $\lambda$ ) of 0.8 and Intensity Rate ( $\gamma$ ) of 0.1. New agent added on day-15 with starting intensity of 1.0. . . . .	88
6.9	Agent behavior when multiple neighbors with different initial intensities at the same distance are available. (a) Intensities of agents during simulation, (b) Influence on $e_1$ . . . . .	90
6.10	Positions and starting intensity of events . . . . .	91
6.11	Agent Behavior with a Neighborhood Radius ( $R$ ) 0.4, Recovery Rate ( $\lambda$ ) of 0.9, and Influence Rate ( $\gamma$ ) of 0.1, agents with different start dates. . .	92
6.12	Agent Behavior with a Neighborhood Radius ( $R$ ) 0.4, Recovery Rate ( $\lambda$ ) of 0.8, and Influence Rate ( $\gamma$ ) of 0.1, agents with different start dates. . .	93
6.13	K-means clustering plots (201401–201406) (a) Category 1 clusters, (b) Category 2 clusters . . . . .	97

6.14 Total Intensity Value Error, Neighborhood Radius ( $R$ ) = 0.2. (a) Andhra Pradesh, (b) Karnataka, (c) Tamil Nadu . . . . . 99

6.15 Total Intensity Trend - *Precision*, Neighborhood Radius ( $R$ )=0.02 for (a) Andhra Pradesh, (b) Karnataka, (c) Tamil Nadu. Row 1: Increasing Trend; Row 2: Neutral Trend; Row 3: Decreasing Trend . . . . . 100

6.16 Total Intensity Trend - *Recall*, Neighborhood Radius ( $R$ )=0.02 for (a) Andhra Pradesh, (b) Karnataka, (c) Tamil Nadu. Row 1: Increasing Trend; Row 2: Neutral Trend; Row 3: Decreasing Trend . . . . . 101

6.17 Average Intensity Value Error for history window ( $w_H$ ) of 1 day, with Neighborhood Radius ( $R$ ) = 0.2, for states Andhra Pradesh, Karnataka, and Tamil Nadu . . . . . 105

6.18 Comparison of Estimated Average Intensity and Real Average Intensity comparison, history window = 1 day. Estimated Average Intensity for Recovery Rate and Influence Rate with the least error value in Figure 1. (a) Andhra Pradesh, Recovery Rate ( $\lambda$ ) = 0.4, Influence Rate( $\gamma$ ) = 0.4, (b) Karnataka, Recovery Rate ( $\lambda$ ) = 1.0, Influence Rate ( $\gamma$ ) = 0.0, (c) Tamil Nadu, Recovery Rate ( $\lambda$ ) = 0.6, Influence Rate ( $\gamma$ ) = 0.4 . . . . . 106

6.19 Average Intensity Value Error for history window ( $w_H$ ) of 7 days, Neighborhood Radius ( $R$ ) = 0.2: (a) Andhra Pradesh, (b) Karnataka, and (c) Tamil Nadu. Row 1: lag ( $t_{lag}$ ) = 1 day; Row 2: lag ( $t_{lag}$ ) = 4 days; Row 3: lag ( $t_{lag}$ ) = 7days . . . . . 109

- 6.20 Comparison of Estimated Average Intensity and Real Average Intensity comparison, history window ( $w_H$ )= 7 days, in **Andhra Pradesh**. Estimated Average Intensity for Recovery Rate and Influence Rate with the least error value in Figure 6.19; Column a. (a)  $t_{lag}$ = 1 day, Recovery Rate ( $\lambda$ ) = 0.6, Influence Rate ( $\gamma$ ) = 0.4, (b)  $t_{lag}$ = 4 days, Recovery Rate ( $\lambda$ ) = 0.8, Influence Rate ( $\gamma$ ) = 0.2, and (c)  $t_{lag}$ = 7 days, Recovery Rate ( $\lambda$ ) = 0.8, Influence Rate ( $\gamma$ ) = 0.2 . . . . . 110
- 6.21 Comparison of Estimated Average Intensity and Real Average Intensity comparison, history window ( $w_H$ )= 7 days, in **Karnataka**. Estimated Average Intensity for Recovery Rate ( $\lambda$ ) and Influence Rate ( $\gamma$ ) with the least error value in Figure 6.19; Column b. (a)  $t_{lag}$ = 1 day, Recovery Rate ( $\lambda$ ) = 0.8, Influence Rate ( $\gamma$ ) = 0.2, (b)  $t_{lag}$ = 4 days, Recovery Rate ( $\lambda$ ) = 0.8, Influence Rate ( $\gamma$ ) = 0.2, and (c)  $t_{lag}$ = 7 days, Recovery Rate ( $\lambda$ ) = 1.0, Influence Rate ( $\gamma$ ) = 0.0 . . . . . 111
- 6.22 Comparison of Estimated Average Intensity and Real Average Intensity comparison, history window ( $w_H$ ) = 7 days, in **Tamil Nadu**. Estimated Average Intensity for Recovery Rate ( $\lambda$ ) and Influence Rate with the least error value in Figure 6.19; Column c. (a)  $t_{lag}$ = 1 day, Recovery Rate ( $\lambda$ ) = 0.8, Influence Rate ( $\gamma$ ) = 0.2, (b)  $t_{lag}$ = 4 days, Recovery Rate ( $\lambda$ ) = 1.0, Influence Rate ( $\gamma$ ) = 0.0, and (c)  $t_{lag}$ = 7 days, Recovery Rate ( $\lambda$ ) = 1.0, Influence Rate ( $\gamma$ ) = 0.0 . . . . . 112
- 6.23 Confidence Factors greater than 60% and Percentage of True predictions,  $w_H$  = 7 days, lag ( $t_{lag}$ ) = 1 day for the states: Andhra Pradesh, Karnataka and Tamil Nadu. . . . . 114
- 6.24 Confidence Factors and Percentage of True predictions,  $w_H$  = 7 days, lag ( $t_{lag}$ ) = 4 days for the states: Andhra Pradesh, Karnataka and Tamil Nadu. 114

6.25	Confidence Factors and Percentage of True predictions, $w_H = 7$ days, lag $(t_{lag}) = 7$ days for the states: Andhra Pradesh, Karnataka and Tamil Nadu.	115
6.26	Average Intensity Value Error of focused events with 5-10 neighbors, history window $(w_H) = 7$ days, Neighborhood Radius $(R) = 0.1$ , lag $(t_{lag}) = 1$ day, for states: Andhra Pradesh, Karnataka, and Tamil Nadu. . . . .	122
6.27	Comparison of Estimated Average Intensity and Real Average Intensity comparison of focused set of events, history window $(w_H) = 7$ days and lag $(t_{lag}) = 1$ day. (a) Andhra Pradesh, Recovery Rate $(\lambda) = 0.0$ , Influence Rate $(\gamma) = 0.8$ , (b) Karnataka, Recovery Rate $(\lambda) = 0.0$ , Influence Rate $(\gamma) = 0.8$ , (c) Tamil Nadu, Recovery Rate $(\lambda) = 0.8$ , Influence Rate $(\gamma) = 0.0$ . . . . .	125
6.28	Comparison of Estimated Average Intensity and Real Average Intensity comparison of focused set of events, history window $(w_H) = 7$ days and lag $(t_{lag}) = 1$ day. (a) Andhra Pradesh, Recovery Rate $(\lambda) = 0.2$ , Influence Rate $(\gamma) = 0.8$ , (b) Karnataka, Recovery Rate $(\lambda) = 0.4$ , Influence Rate $(\gamma) = 0.6$ , (c) Tamil Nadu, Recovery Rate $(\lambda) = 0.8$ , Influence Rate $(\gamma) = 0.2$ . . . . .	127
6.29	Total Intensity Value Error of focused events with 5-10 neighbors, history window $(w_H) = 7$ days, Neighborhood Radius $(R) = 0.1$ , lag $(t_{lag}) = 1$ day: (a) Andhra Pradesh, (b) Karnataka, and (c) Tamil Nadu. . . . .	128
6.30	Real Total Intensity of focused set of events, history window $(w_H) = 7$ days and lag $(t_{lag}) = 1$ day. (a) Andhra Pradesh, (b) Karnataka, and (c) Tamil Nadu . . . . .	129

## List of Tables

2.1	Quality scores of EMBERS models by country [50] . . . . .	17
3.1	Default parameters used in distance calculations . . . . .	47
5.1	Average event counts per category for India states (201401–201406) . . .	57
5.2	Modified average event counts and intensities . . . . .	58
5.3	Error for true and false trend prediction ( $err_{trend,t}$ ) . . . . .	67
6.1	Categories of unrest events selected for analysis from GDELT . . . . .	73
6.2	Spatial and temporal spans of 3 largest clusters found when clustered using spatiotemporal, socioeconomic and infra-structure distances . . . . .	76
6.3	Spatial and temporal spans of 3 largest clusters found when clustered using spatiotemporal distances only . . . . .	76
6.4	Spatial and temporal spans of 3 largest clusters found when clustered using spatio-temporal and socioeconomic distances . . . . .	77
6.5	Spatial and temporal spans of 3 largest clusters found when clustered using spatio-temporal, socioeconomic and infra-structure distances . . . . .	78
6.6	Comparison of cluster and noise counts between ST-KDT-DBSCAN and regular DBSCAN . . . . .	80
6.7	Event start dates . . . . .	91

6.8	K-means clustering of event and locations for states in India between 201401 201406 . . . . .	96
6.9	Average number of neighbors for all events in each state . . . . .	98
6.10	Comparison of the predicted trend and real-world trends by number of predictions in <b>Andhra Pradesh</b> . (a) $t_{lag} = 1$ day, Recovery Rate ( $\lambda$ ) = 0.6, Influence Rate ( $\gamma$ ) = 0.4, (b) $t_{lag} = 4$ days, Recovery Rate ( $\lambda$ ) = 0.8, Influence Rate ( $\gamma$ ) = 0.2, (c) $t_{lag} = 7$ days, Recovery Rate ( $\lambda$ ) = 0.8, Influence Rate ( $\gamma$ ) = 0.2 . . . . .	116
6.11	Comparison of the predicted trend and real-world trends by number of predictions in <b>Karnataka</b> . (a) $t_{lag} = 1$ day, Recovery Rate ( $\lambda$ ) = 0.8, Influence Rate ( $\gamma$ ) = 0.2, (b) $t_{lag} = 4$ days, Recovery Rate ( $\lambda$ ) = 0.8, Influence Rate ( $\gamma$ ) = 0.2, (c) $t_{lag} = 7$ days, Recovery Rate ( $\lambda$ ) = 1.0, Influence Rate ( $\gamma$ ) = 0.0 . . . . .	117
6.12	Comparison of the predicted trend and real-world trends by number of predictions in <b>Tamil Nadu</b> . (a) $t_{lag} = 1$ day, Recovery Rate ( $\lambda$ ) = 0.8, Influence Rate ( $\gamma$ ) = 0.2, (b) $t_{lag} = 4$ days, Recovery Rate ( $\lambda$ ) = 1.0, Influence Rate ( $\gamma$ ) = 0.0, (c) $t_{lag} = 7$ days, Recovery Rate ( $\lambda$ ) = 1.0, Influence Rate ( $\gamma$ ) = 0.0 . . . . .	118
6.13	Comparison of the predicted trend and real-world trends by number of predictions in <b>Andhra Pradesh</b> with Recovery Rate ( $\lambda$ ) = 1.0 and Influence Rate ( $\gamma$ ) = 0.4. (a) $t_{lag} = 1$ day, (b) $t_{lag} = 4$ days, (c) $t_{lag} = 7$ days . . . . .	119
6.14	Comparison of the predicted trend and real-world trends by number of predictions in <b>Karnataka</b> with Recovery Rate ( $\lambda$ ) = 1.0 and Influence Rate ( $\gamma$ ) = 0.0. (a) $t_{lag} = 1$ day, (b) $t_{lag} = 4$ days, (c) $t_{lag} = 7$ days . . . . .	120

6.15 Comparison of the predicted trend and real-world trends by number of predictions in **Tamil Nadu** with Recovery Rate ( $\lambda$ ) = 1.0 and Influence Rate ( $\gamma$ ) = 0.0. (a)  $t_{lag}$ = 1 day, (b)  $t_{lag}$ = 4 days, (c)  $t_{lag}$ = 7 days . . . . 121

6.16 Comparison of the predicted trend and real-world trends by number of predictions in for simulations with the **least average intensity value error**,  $w_H$  = 7 days,  $t_{lag}$ = 1 day: (a) Andhra Pradesh, Recovery Rate ( $\lambda$ ) = 0.0, Influence Rate ( $\gamma$ ) = 0.8, (b) Karnataka, Recovery Rate ( $\lambda$ ) = 0.0, Influence Rate ( $\gamma$ ) = 0.8, (c) Tamil Nadu, Recovery Rate ( $\lambda$ ) = 0.8, Influence Rate ( $\gamma$ ) = 0.0 . . . . . 124

6.17 Comparison of the predicted trend and realworld trends by number of predictions in for simulations with the **least trend prediction error**,  $w_H$  = 7 days,  $t_{lag}$ = 1 day: (a) Andhra Pradesh, Recovery Rate ( $\lambda$ ) = 0.2, Influence Rate ( $\gamma$ ) = 0.8, (b) Karnataka, Recovery Rate ( $\lambda$ ) = 0.4, Influence Rate ( $\gamma$ ) = 0.6, (c) Tamil Nadu, Recovery Rate ( $\lambda$ ) = 0.8, Influence Rate ( $\gamma$ ) = 0.2 . . . . . 126

6.18 Simulation parameters with the least trend errors . . . . . 126



## Chapter 1

### Introduction

#### 1.1 Problem Overview

Social unrest can be defined as a public display of collective dissatisfaction or aggression by a significant human mass against a body of power such as the government, expressing a desire for change. Social unrest (such as appeals, protests, conflicts, fights or mass violence) can result from a wide ranging of diverse factors or motivations such as gun-control, civil rights, religious/ethnic control, etc. This makes the analysis of causal relationships challenging, with high complexity and uncertainty. The unrest events can result in significant changes in a society ranging from new policies and regulations to regime change, and ethnic cleansing. In the recent years, the availability of data and diffusion of new information has encouraged the broader science community including computer scientists to study the phenomenon of social unrest and the methodologies to anticipate such events, as these type of events often pose a risk of damage. Damages arising from such events can generally be described as physical or psychological harm to objects that humans value. This may be the loss of property, health or even life [30]. Identifying relationships between different events and how they evolve with respect to each other could allow us to predict future unrest. Predicting the occurrence of civil unrest events is in the interest of policy makers,

since local unrest can lead to regional instability [16]. It is easier to influence events in their earliest stages, before they become more threatening and less manageable [44].

While most social unrest events are initially intended for demonstration to the public or the government, they at times escalate into general chaos [42]. It is reasonable to assume that widespread unrest arises through a process of feedback and cascading of a collection of past events over time in regions that are close to each other. In our research, we call such a collection of events that form a network of closely related events in terms of spatial, temporal or conceptual similarities, an episode. Indeed, through the identification process of event episodes, we would be able to capture the dynamics of events across time, space and other underlying factors. This could allow us to see how events spread, evolve, merge or dissipate, and ultimately simulate and predict behaviors of events. As events progress, some become more violent, fueled by other social activities in the surrounding areas, and some might become less violent and decay or die out. The environment occupied by unrest events consists of various socioeconomic and infrastructural factors, and therefore can be very dynamic.

The study and prediction of social unrest has primarily been done through the study of social media messaging (example: Twitter and Facebook) using various natural language processing (NLP) techniques [47, 42, 35, 52]. While there has been some research that use macro-structural data such as political, social, economic and demographic factors [44, 55], grouping of related events are often done by human experts. In case of predictive models, it is often the subject matter experts that create profiles for a country or location. To the best of our knowledge, event-event interaction models have not been implemented or tested.

In this thesis, we focus on the following two main research problems:

1. Identification of event episodes from large datasets of events. Through the identification process of event episodes, we would be able to capture the dynamics of events across time, space and other underlying factors.
2. Investigate how events interact and impact each other to support predictions and what-if projection analyzes, and to understand how events evolve with respect to other events.

## 1.2 Proposed Solution

In addition to geospatial coordinates and time stamps, factors such as the socioeconomic and infrastructural attributes of the event-location must be considered. The addition of socioeconomic factors that fuel, say, the emergence of a social unrest event, and infrastructural factors that could facilitate, say, the propagation of an event to nearby regions allow events to be connected by more than just time and space. Therefore, we first develop a multi-factorial distance function that can combine different types of distances among events such as the spatio-temporal, socioeconomic and infrastructural distances into a single comprehensive distance value. This distance function allows weighting of various attributes and distance types based on domain knowledge associated with the data or application.

**Solution 1.** Our proposed solution to the first research problem is to use a spatio-temporal data clustering based approach with the multi-factorial distance function. Spatio-temporal data clustering allows grouping of objects based on their spatial and temporal similarity [38], and allow data analysts to interpret groups of data and find common patterns for each group and identify distinguishing patterns. For our work, we use the density-based clustering approach (DBSCAN) as the underlying clustering algorithm as it is efficient in finding clusters of similar densities in spatial database

with noise. The algorithm needs three inputs to execute: (1) *epsilon* (*eps*) the maximum distance between two points to be considered neighbors, (2) *minPts* the minimum number of points within the neighborhood of a point to be considered a core point, and (3) a distance function to measure the similarity between the points. One known weakness of this algorithm is that it is very sensitive to the input parameters *eps* and *minPts*. Thus the optimal values for these parameters is non-trivial to determine and often found empirically. Several methodologies for performing clustering on spatio-temporal data have been explored in [4, 58, 11]. While various methods of spatio-temporal clustering have been discussed in the past, they have not been applied in the analysis of social movements or social unrest networks. There has not been any significant effort towards classification or identification of episodes of unrest events based on clustering of multi-factorial spatio-temporal data. Some existing work on spatio-temporal pattern discovery include CSTP [23], where patterns in some form of spatio-temporal events are explored by defining a spatial and temporal threshold and then searching for events occurring within this defined constraint. However, the approach does not consider other environmental data such as the socioeconomic and infrastructural which are non-trivial to integrate with spatio-temporal attributes.

**Solution 2.** For the second research problem, due to the dynamic nature of the environment, we propose an agent-based solution to model social unrest events into intelligent agents. We then run multiagent simulations with the modelled agents. The agents continuously study their environment and perform actions which translates into the increase or decrease of their propensity towards violence or harm to property and lives. We represent this property as the intensity of an event, the higher the intensity of the event, the more risk it carries. On every simulation step, events will compare its intensity with that of the other agents inside its predefined neighborhood. Based on the intensity of its neighbor, each agent will then increase or decrease its

own intensity. This model is loosely based on the N-body gravitational model [1] where each object is trying to pull and push objects towards itself.

**Scalability Issue in Spatio-temporal Clustering.** We have further identified a sub-problem in the clustering approach. This sub-problem comes in the form of scalability while performing similarity checks between events on a large geospatial or temporal scale. Most popular conventional clustering methods in general do not scale well for very large data sets since they either need several passes over the data or they create data structures that do not scale linearly with the number of objects [13]. In most clustering techniques, the processing of large data in has been done through parallel processing or by reducing the data size. An example of data reduction is sampling which can be seen in CLARA (Clustering Large Application) [23], which first takes a sample of the data and then partitions it. Another example of such data reduction technique is the BIRCH [60] algorithm which first generates a more compact summary of the data that retains as much distribution information as possible, and then clusters the data summary instead of the original dataset. Since these approaches reduce the data size before the actual clustering, it is possible to lose some details. Different implementations of the DBSCAN [19] may use efficient data structures such as  $k$ -dimensional trees [7] to quickly scan through the data, but they only consider numeric data that allows standard metric calculation such as the Euclidean distances, and do not consider the data as geospatial or temporal. Additionally, the use of distance functions requires the creation of distance matrices, matrices that store precomputed distances between all pairs of events, this requires unnecessary calculations between events that may not be reachable from each other in the context of clustering.

To address the sub-problem of scalability, we propose a novel Spatio-Temporal  $k$ -Dimensional Tree-based DBSCAN (ST-KDT-DBSCAN) clustering approach that

restricts the search radius for each event during clustering by first organizing the dataset into a  $k$ -dimensional tree structure based on the longitude, latitude and time values, subsequently creating a Fixed-Radius Near Neighbor (FRNN) [34] object for each event, and then carrying out DBSCAN considering only each events FRNN object when computing reachability. Specifically, we show an implementation of a  $k$ -dimensional tree to organize spatio-temporal data for range searches that use both geospatial and temporal search radii. Since, the ST-KDT-DBSCAN algorithm restricts the search radius for each event, calculation of distances from events that are too far away in space or time is avoided resulting in faster computation time. Of course, the larger the search radius, the larger the FRNN object, and hence more calculations are required during clustering.

### 1.3 Contributions

A summary of our contributions are as follows:

1. We present a distance function designed for integrating both data-driven and model-driven approaches into measuring the difference between two events in general, and in particular, for analysis of social unrest events. The distance function mainly takes two events as input along with other parameters to determine the types of distances and thresholds, and outputs a distance metric between the events. Details of the distance function is given in Chapter 3.
2. We provide a method of using a  $k$ -dimensional tree data structure to organize spatio-temporal data for range searches that use both geospatial and temporal search radii. We then use this data-structure to create a modified DBSCAN clustering algorithm that is much more scalable than existing DBSCAN algo-

rithms. We then explore and report on our findings after applying the ST-KDT-DBSCAN algorithm on social unrest dataset.

3. We create a framework of an agent-based model that can simulate social unrest based on real world parameters to predict future unrest. We loosely apply the concept of  $n$ -body gravitational push-pull into our work as each agent can be imagined as an object with certain gravitational values represented by the intensity. We then show the viability of our model and explain, using synthetic data, how the intensities of events increase or decrease in a somewhat realistic manner under different circumstances. We then simulate real-world social unrest data and report on our findings in an attempt to investigate issues or needs to better implement this approach.

## 1.4 Outline

We discuss existing works related to our research in Chapter 2. Here, we first talk about the spatiotemporal clustering and the agent-based modelling approaches. In Section 2.3 we discuss some projects that have goals similar to our project. In Chapter 3 we describe our implementation of the multi-factorial distance function in detail. In Chapter 4, we describe the implementation of SD-KDT-DBSCAN algorithm. The agent-based modelling approach is then explained in Chapter 5. We also look at how the intensity variable for each event-type is defined in Section 5.1. In Section 5.2, we explain our design principles for the agent-based modelling. In Chapter 6, we present our results and discussions. Finally, we conclude the thesis and discuss future work in Chapter 7.

## Chapter 2

### Background and Related Work

#### 2.1 Spatio-Temporal Clustering Algorithms

Unsupervised learning via clustering is one of the most popular data mining techniques used to understand the dynamics of a dataset where known dependent variables or labels are not available. The clustering algorithms are divided into categories based on their approach. Partitioning algorithms tend to divide the data into a pre-defined number of clusters, for example k-Medoids [34] and CLARA. Hierarchical clustering algorithms build a hierarchy of clusters usually either following a bottom-up approach or a top-down approach, for example BIRCH and CURE [27]. Density-based clustering algorithms, on the other hand, *seek to find clusters by looking for dense regions of points within the dataset*. The clusters are separated by low density regions, for example DBSCAN and OPTICS [5]. Spatio-temporal (ST) clustering is a special form of clustering where the data points have spatial coordinates and time stamp as a subset of the attributes. The objective is to group points based on their spatial and temporal proximity. This is more challenging than traditional clustering since the inclusion of spatial and temporal attributes (ST-attributes) make the assumption of independence no longer valid. Properties such as spatial auto-correlation and temporal cycles play an important role in the clustering of points. The non-spatial and



non-temporal attributes also need to be incorporated, making the task of clustering even harder. Some of the ST clustering algorithms proposed in the literature include ST-DBSCAN [11] and STPC [31].

## 2.2 Agent-Based Modelling

Agent-based modelling (ABM) refers to computational models invoking dynamic actions, reactions and intercommunication protocols among the agents in a shared environment, in order to evaluate their design and performance and derive insights on their emerging behavior and properties [2]. Agent-based modelling was developed to support decision making and solving practical as well as theoretical problems. These models are complex systems built in a bottom-up approach, starting with the design of individual agents. The key entity in ABM is the agent. Agents can simply be described as goal-directed independent components. Agents have the following fundamental properties:

- They are autonomous components and can perform actions without user intervention.
- They are capable of communicating with other agents in their neighborhood.
- They can detect changes in the environment or take input from it, and then respond by taking some action.

Complex behaviors arise through independent actions of myopic agents without the need of a central control unit, the information that drives the decisions in agent is local to that agent. The big difference between the agent-based approach and the more aggregate, static conceptions and representations that they seek to complement, if not replace, is that they facilitate the exploration of system processes at the level of

their constituent elements [15]. In our project, we use Repast to build our multiagent simulation.

ABM has been utilized in various scientific disciplines such as computer science, biology, engineering, etc. The boids simulation [51] is an example of a ABM simulation built with simple rules that leads to organized emergent behavior replicating swarm behavior in fish and birds. Vytelingum, Voice, Ramchurn, Rogers & Jennings (2010) presented a novel agent-based micro-storage management technique that allows all (individually-owned) storage devices in the system to converge to profitable, efficient behavior. Berger (2001) presented a spatial multiagent programming model, which has been developed for assessing policy options in the diffusion of innovations and resource use changes. Some popular platforms for developing ABM and simulation are Swarm, MASON, Repast, StarLogo, NetLogo, OBEUS, AgentSheets and AnyLogic. A detailed review of the state-of-art software for ABM and simulation is provided by Abar, Theodoropoulos, Lemarinier & OHare (2017).

## **2.3 Related Projects**

In this section, we present literature reviews of state-of-art projects and research methodologies dedicated to forecasting social unrest or similar social events using data-driven techniques. More specifically, we discuss data sources used, methodologies and results as reported by the authors in these projects. The Sections 2.3.1 and 2.3.2 are discussed in much more detail than other sections as we have identified them as the most relevant recent works in the field of unrest forecasting. In Section 2.3.5, we discuss more relevant projects but with less details. Then, in Section 2.3.6, we provide a summary and compare of our project (SURGE) with the discussed works. The research presented in this thesis was conducted as part of project SURGE at the

University of Nebraska - Lincoln, supported by the National Geospatial-Intelligence Agency.

### **2.3.1 EMBERS**

Early Model Based Event Recognition using Surrogates (EMBERS) [50] is an automated system that forecasts civil unrest across 10 countries of Latin America, viz. Argentina, Brazil, Chile, Colombia, Ecuador, El Salvador, Mexico, Paraguay, Uruguay, and Venezuela, using open source indicators such as tweets, news sources, blogs, economic indicators, and other data sources. The project defines civil unrest as population-level event wherein people protest against the government or other larger organizations about specific policies and issues, acts by criminals for private gain are not considered as civil unrest. EMBERS adopts a multi-model approach where different models use different data-sources to generate different predictions. These predictions are then combined to generate a final set of warnings. The EMBERS project is supported by IARPA (Intelligence Advanced Research Projects Activity) OSI (Open Source Indicators) program, the forecasts generated by the system is emailed to IARPA, which is then evaluated against a gold standard report (GSR). The GSR of protests used to evaluate the predictability of the EMBERS project is organized by MITRE, an independent group of human analysts who survey newspapers for reporting of civil unrest. The results presented and discussed in this section are based on the manually generated GSR report. According to Saraf & Ramakrishnan (2016), EMBERS is now using AutoGSR, an automated event coding system for civil unrest events, any performance gain due to AutoGSR has not been reported.

## Data

Latin America was chosen as the region for study in the EMBERS project because it experiences a large amount of civil unrest events on a daily basis, which makes it favorable for generating GSR reports and training machine learning models. These locations were found to be well covered by national and international news, and has a growing number of social network users, supporting the use of data mining algorithms. While a complete list of data sources is not provided by EMBERS, the following data sources are mentioned in their research: Twitter's public API, Datasift's processed Twitter feed, Facebook, Healthmap's alerts and reports, RSS news and blog feeds, Talkwalker alerts, NASA satellite meteorological data, Google Flu Trends, Bloomberg financial news, TOR usage data, ICEWS, GDELT, OpenTable's restaurant cancellation data, the PAHO health survey, and web-pages referenced tweets. One of the data sources discussed heavily is the Datasift's Twitter collection engine, this engine provides the ability to stream and query tweets in real time. The tweets from Datasift are augmented with meta-data such as the user's profile and geotags.

## Methodology

The methodology of the EMBERS project can be explained in four stages or modules- ingest, enrichment, prediction and delivery. During ingest, the module processes data from variety of sources as mentioned in Section 2.3.1. Each source is assigned a dedicated processor, the ingested data is packaged as JSON messages, assigned a unique identifier and published to either a database cache or to a source specific queue for archiving and subscription. One of the central ingest process utilizes the Datasift's Twitter collection engine.

In the enrichment stage, data with textual contents such as tweets and news feed

are subjected to a shallow linguistic process. This process includes identification of the language and identification and classification of named entities. Dates and geographic (city, state, state) information is then extracted from the text. The EMBERS system uses two geocoding systems, one for tweets and another for news and blog articles. To geocode tweets, the most reliable method is to use the Twitter geolocator, as it can determine the approximate geo-coordinates of a location a tweet might be referring to or was originated from. However, this information is only available for about 5% of the total tweets. Therefore, the system looks at Twitter places to geocode the tweet to a geographical coordinate. Finally, the place names in the user's profile is considered along with the mentions of places in the text of the tweet are considered. News and blog articles however, do not have user profiles and may contain several location names, such as the origin of the article and locations corresponding to any individual that may or may not be related to an event. EMBERS utilizes a probabilistic reasoning engine using probabilistic soft logic (PSL) to extract the locations that are the main focus of the article. This processed data is then utilized in the prediction models. EMBERS uses five different prediction models, each model has different underlying assumptions and uses different type of data sources.

**Planned Protest.** The tokenized output of the enrichment model is examined to find phrases indicating a call for action or plan to strike, indicating a potential planned protest. Messages that indicate a planned protest are then further examined for mentions of a future time/date. The spatial or location information is determined through the enrichment geocoders. This model reads three kinds of input messages: standard natural language text such as text from news or blog articles as well as content of webpages mentioned in tweets, microblogging text such as Twitter, and event pages such as the Facebook Events pages. A minimum number of retweets is required before a tweet is considered relevant. For Facebook, the public API is

used to search various event pages that may prove information regarding any planned protest, which generally includes the date and location.

**Volume-based Model.** This is a traditional machine learning model, it uses a logistic regression with LASSO (Least Absolute Shrinkage and Selection Operator) to select a sparse feature set that predicts the probability of occurrence of civil unrest on a country level. Tweets are the primary input for this model, these tweets are filtered using a keyword dictionary of unrest related words and words identifying country-specific actors such as public figures or political parties. The regression based model considers covariates that include counts of protests-related keywords, daily count of same keywords in news and blogs, country’s exchange rate, count and intensity of events identified by ICEWS (Integrated Conflict Early Warning System), count of events identified by GDELT (Global Data on Events, Locations and Tone), and the average tone of these events.

**Dynamic query expansion (DOE).** The dynamic query expansion model exclusively uses tweets to expand the vocabularies of interest. It is based on the idea that the causes of unrest could be quite varied and there might be several keywords that may indicate future unrest. A small set of unrest related keywords are first used to filter tweets, the words in these tweets are then weighted based on TF-IDF [49]. The higher ranked words are then used similarly in the second iteration to filter new tweets, and so on until the set of keywords and their weights become stable.

**Cascades Model.** This model is used to track recruitment of individual into a cause or a campaign. This is done primarily by tracking re-tweets of a certain message or hashtag within a certain time interval. If a followers re-tweets a message which is then re-tweeted by their follower, a follower-graph is created. This model considers re-tweets or messages by non-followers as well. These activity cascades are calculated on each day. The size of a cascade (number of participants) are used as input for a

generalized linear machine learning model to forecast the probability of occurrence of an unrest event.

**Baseline model.** This model calculates the maximum likelihood estimate (MLE) for the occurrence of future civil unrest based on the distribution of event schema frequency in the most recent part of GSR. An event schema is the combination of location, event type, a population and a day of the week.

Finally, a fusion and suppression engine is used to generate the final set of warnings and deliveries. The key operations of this engine includes: duplicate detection and warning update, filling of missing values such as event type, population or location based on the likelihood of appearance in the GSR, warning rewriting in case of a high probability of error in prediction, and balancing the recall-quality trade off.

### Evaluation Metrics

An alert contains the where/why/when/who of the protest, and confidence associated with the forecast. Similarly, the GSR also contains the where/why/when/who of a protest that has actually happened and a reported date of the event.

**Lead Time vs Accuracy of Forecast Date.** EMBERS defines four types of dates in a (alert, event) combinations: the date forecast is made (*forecast date*), the date event is predicted to happen (*predicted event date*), the date the event actually happens (*event date*), and the date the event is reported in a GSR source (*reported date*). For a valid prediction,  $forecast\ date < reported\ date$ . The *lead time* is given as:  $reporteddate - forecastdate$ . The difference between *event date* and *predicted event date* should be as low as possible, while the *lead date* should be as high as possible.

**Other Quality Aspects.** Other than the forecast dates, accuracy should also be calculated for the location, event type and population. Each of these aspects are

assigned scores and the overall quality of the prediction, called the quality score ( $QS$ ) is calculated as:

$$\text{Quality score } (QS) = DS + LS + ES + PS$$

where  $DS$ ,  $LS$ ,  $ES$ , and  $PS$  denote the date score, location score, event type score and population score, respectively.

**Inclusion Criteria.** This determines which warning-event pair is considered for scoring, based on the following inclusion criteria:

1. *Lead time*  $> 0$
2. Both warning and event are for the same country.
3. The *predicted event date* and *event date* must be within 7 days of each other.
4. locations are within 300 km of each other.

**Non Crossing Matching.** A non-cross matching is defined as a restrictive version of bipartite patching. Consider two warnings  $w_1$  and  $w_2$ , and two events  $e_1$  and  $e_2$  and assume  $w_1 < w_2$  and  $e_1 < e_2$ , then  $(w_1, e_2), (w_2, e_1)$  is defined as *cross matching*. Here, the earlier warning is paired with the later event (and vice versa).

## Results

The results presented in the EMBERS's report show that each model works differently based on the country or region of interest, and therefore a integration is useful to generate a warning with more accuracy and high quality score. For example; the DQE model was observed to be performing better for some countries like Brazil, Mexico and Venezuela due to the higher Twitter traffic in these countries. Table 2.1 shows the performance of different models in the countries selected in the EMBERS project.



The authors show that the fusion and suppression engine can be used to improve the quality of prediction by filtering out irrelevant warnings, which helps to balance recall-quality trade-off. One of the interesting results presented by EMBERS is *the ability to forecast ‘surprising’ events and significant uprisings, quite effectively*. The authors show that EMBERS was able to predict a sudden increase in protest numbers, when a significant rise in the number of protests were observed in Brazil, during the summer of 2013. The protests were triggered by rise in bus fares. The probability scores have a monotonic relationship with the likelihood of an event match. The quality scores are seen to be low for either very small (1 to 2 days) or very large (around 10 days and more) lead times, while the high quality for small lead time can be explained through high social media traffic immediately before an event, the high quality scores for a large lead time is achieved by tracking planned protests on Facebook or Twitter. Finally, the performance of EMBERS system is shown to have improved over time as the quality score rose from 2.0 in January of 2013 to 3.0 in January of 2014.

Table 2.1: Quality scores of EMBERS models by country [50]

<b>Model</b>	<b>AR</b>	<b>BR</b>	<b>CL</b>	<b>CO</b>	<b>EC</b>	<b>SV</b>	<b>MX</b>	<b>PY</b>	<b>UY</b>	<b>VE</b>	<b>All</b>
Dynamic Query Expansion	3.1	3.31	1.88	3.1	2.43	2.94	3.26	2.88	2.72	2.9	2.97
Volume-based Model	3.0	3.11	-	2.9	-	-	3.15	-	1.72	2.9	2.88
MLE	3.33	3.0	2.87	3.15	2.29	3.11	3.11	3.1	2.57	2.77	3.0
Planned Protest	2.59	2.64	2.4	2.85	1.92	-	3.0	2.89	2.85	2.66	2.76
Cascades Model	3.13	-	-	-	-	-	-	-	-	2.93	3.0

## Discussion

The EMBERS project has an automated system that utilizes massive amount of open source data to forecast future unrest events. The project recognizes that each location has unique properties and the same model might not perform well globally. Thus, weights are assigned to warnings issued from different models, and finally a fusion and suppression engine is used to further filter out irrelevant warnings, the details of this filtering is however not provided. The project has developed an effective vocabulary of unrest related keywords. Especially through their DQE model, the authors claim to have identified location specific keywords and hash-tags to identify unrest related social media posts. However, *majority of the accurate forecasts seem to be attributed by either an unusual amount of events in the immediate past of an upcoming event, or through dates and locations mentioned in Tweets or Facebook events.* One of the issues pointed out by the authors is that the project does not yet, utilize social science theory to develop models.

### 2.3.2 Activism via Attention

In this section we discuss a theory-motivated, spatio-temporal learning approach called *ActAttn* discussed in [18], that implements social movement theories and a deep learning framework to forecast future protests. In addition to the forecast, another key contribution according to the paper is the theory-relevant interpretations of the forecasts, such as presenting relevant features for different locations. This approach aims to analyze how different types of social movements develop. Some movements immediately garner mass media attention while others depend on local activists before being picked by the media. To illustrate such differences, the project uses three different social movements, each connected to a similar social issue but

different in progression. These events are: the Black Lives Matter (BLM) movement which facilitated unrest and protests in Ferguson, in August and November, and the marches followed by the white supremacist rally that took place in Charlottesville in August of 2017. These events left several online and offline activity traces, spatially and temporally. *The ActAttn approach is modelled on tracking online activities of a general population.* The presented model attempts to not only forecast future events, it aims to *explain* various spatio-temporal patterns and answer the following questions (as listed in the paper): In a movement, what social and activity features are associated with the subsequent events? To what extent are the local activities predictive of the subsequent events, compare to global activities? And what places' activities would have far-reaching predictive power in terms of signaling subsequent events in other places? In [18] the authors argue that the biggest contribution of this work is that the model builds upon existing social movement theories and studies regarding why people protests, and the geological and sociocultural factors contribute to the development of protests.

## Data

To choose the data for their experiments, the researchers first chose social movements with social significance so that the social features of the location could be applied. The selected movements: Black Lives Matter (BLM), and the counter-protests organized against the white-supremacist rally, were chosen, both movements have similar underlying issues. For BLM, two different waves of protests were chosen and are treated as separate movements in the research.

Tweets with specific keywords or hashtags were collected related to the Charlottesville rally, and the BLM related rallies. The *Charlottesville Dataset* were collected during the movement, through a streaming API using specific keywords and/or

hashtags that either mentioned white-supremacist groups (for example: KKK, Nazi, etc.), racism or the location Charlottesville. The *Ferguson I Dataset* and *Ferguson II Dataset* were collected using keyword such as #ferguson, #blacklivesmatter, black lives matter and the names of black people killed by police during 2014 and 2015. The location data for all the tweets are either collected from the geo-codes available in the tweet or inferred from the users profiles. In cases where only the city names are available, and if these city names are common among multiple states, the tweets are discarded.

Protest data was collected as ground truth from Elephrame, a website that provides information about social unrest in the United States. The data contains the start and end dates of the protest, state and city level location information, protest subjects, description and the source of the data as a link. In *ActAttn*, the number of events at the same location and on the same day are not considered, the researchers only consider if any event has occurred at a location or not, representing this data as binary features.

The following static features are mentioned in the paper: population of the state in which the location of unrest exists, population density, voting behavior of 2016, region of United States (Northeast, Midwest, South, West). The dynamic features are focused on four factors: emotion, identity, grievance, and social embeddedness. The emotion, identity and grievance are identified from the use of keywords from a predefined dictionary. The level of social embeddedness is determined by counting the number of tweets, number of reply tweets, and the number of tweets with URLs, as more tweets suggest more awareness of local issues and events.

## Methodology

Let  $L$  be the set of locations (states) of interest and each location  $l$  is denoted by a collection of static and dynamic variables. The static features either do change with time or change very slowly, examples of static features are population and political leanings. Dynamic features (e.g., percentage of tweets expressing anger) are updated for each time interval  $t$ . Let  $S_l$  denote the set of static features of the location  $l$ ,  $X_{t,l}$  be the set of dynamic features at time  $t$  for location  $l$ . The goal is to predict the value of binary variable  $Y_{t^*,l} \in 0, 1$ , which indicates the occurrence of a future protest for location  $l$  at time  $t^*$ . The *ActAttn* model considers the static and dynamic features of a location as well as the dynamic features of all other locations to make the prediction. The forecast can be written as the learning function:

$$F(S_d, X_{t-k+1:t}) \rightarrow Y_{t^*,d} \quad (2.1)$$

where  $X_{t-k+1:t} = X_{t-k+1}, \dots, X_t$ ,  $k$  represents the size of the time window considered, and  $d$  is the target location.

Since the *ActAttn* model considers the local and global dynamic features, the set of dynamic features used as input can be divided into intra-region and inter-region features. The model contains three primary components: temporal component ( $M^{tem}$ ) that models the contribution of intra-region features, spatial component ( $M^{sp}$ ) that models the contribution of inter-region features, and the static features ( $S_d$ ). In both  $M^{tem}$  and  $M^{sp}$ , LSTM [28] is used to capture temporal relationships among the dynamic features. To calculate the spatio-temporal relationship between different locations in  $M^{sp}$ , separate temporal components, similar to  $M^{tem}$ , are added for all the locations. The LSTM therefore, outputs  $h_d^{tem}$  and  $h_1^{sp}, h_2^{sp}, \dots, h_L^{sp}$  for  $M^{tem}$  and  $M^{sp}$ , respectively.

For spatial and spatiotemporal attention, a **hierarchical attention mechanism** is used. First, in  $M^{sp}$ , a single spatial attention layer is applied on top of  $h_1^{sp}, h_2^{sp}, \dots, h_L^{sp}$  to learn the importance or contribution of each location. Let  $v^{sp}$  be the spatial attention output that summarizes the aggregate contribution of all locations. Second, we introduce a spatiotemporal attention layer given by:

$$v^{st} = \alpha^{tem} h_d^{tem} + \alpha^{sp} v^{sp} \quad (2.2)$$

where  $\alpha^{tem}$  and  $\alpha^{sp}$  are the attention weights applied to the outputs of temporal and spatial components, respectively.

The forecasting is then given by:

$$\bar{Y}_{t^*,d} = \phi(W_c[S_d, v^{st}] + b_c) \quad (2.3)$$

where  $W_c$  and  $b_c$  are the weight matrix and bias vector to be learned, respectively.  $\phi$  is the activation function where we apply the *Softmax* [12] function to obtain probabilities of occurrence of events. Finally, to minimize the use of redundant features, **Group Lasso (GL)** regularization [40] is used to select the informative features.

## Evaluation Metrics

The *ActAttn* approach is compared with three sets of state-of-the-art approaches as the baseline methods. The first set includes logistic regression (LR) and support vector machine (SVM) classifiers. The second set includes more recently-developed models such as RNNs and LSTMs. The third set includes spatio-temporal event forecasting approaches using regularized multi-task feature learning (RMTFL), constrained multi-task feature learning I (CMTFL-1) and constrained multi-task feature

learning II (CMTFL-2).

The prediction is made on a state (geographical) level, and the time unit used is “day”. The window size ( $k$ ) is set to be 1, 2, 3, and the lead time ( $\tau = t^* - t$ ) is set to be 1, 2, 3. To evaluate the effectiveness of individual components of the *ActAttn*, several of its variations are compared. The variations are: *ActAttn (w/o GL)*, *ActAttn (w/o stAttn)* which doesn’t include spatiotemporal attention layer ( $h_d^{tem}$  and  $v^{sp}$  are concatenated), and *ActAttn (w/o spAttn)* which doesn’t include the spatial attention layer. In the experiments, ‘day’ is used as the time unit and state as the location unit.

## Results

**Performance Comparison.** *ActAttn* achieved the highest F-score [26] and AUC [20] values on the Charlottesville data (0.400 and 0.843), Ferguson I (0.462 and 0.822) and Ferguson II (0.471 and 0.853) data compared to other baseline methods such as Linear Regression, SVM, LSTM and CMTFL. The F1 scores are low for all methods due to imbalance in the class distribution. Furthermore, it was observed that combining inter-region features with static features and intra-region features increases the performance in all *ActAttn*-based methods other than *ActAttn(w/o stAttn)*. Testing the hierarchical attention mechanism, *ActAttn* was tested against its three variants. It was seen that *ActAttn* slightly outperforms *ActAttn(w/o GL)* but GL regularization provides sparsity and selection of a compact set of features. *ActAttn* was seen to outperform both *ActAttn(w/o stAttn)* and *ActAttn(w/o spAttn)*, which indicates that incorporating the spatio-temporal attention layer leads to the best performance of the model.

**Robustness to missing information.** Two types of missing data were tested in this set of experiments, (1) missing data in time and space, (2) missing data for a

certain region. As missing data could occur in any feature at of any location during any time, the existing data were randomly removed at different levels (20%, 40%, 60% and 80%) to create test sets. The missing data was filled by randomly assigning values from the non-missing values into the missing features. Using this technique, the *ActAttn* consistently performed better than any other method. Similarly, the data for certain regions were removed by different proportions (ranging from 20% to 80%), the *ActAttn* model performed better than other models in this set of experiments as well.

***Varying time lead.*** To examine how far in time the model could effectively forecast protest, the model was tested for different values of lead time ( $\tau$ ) = 1, 2, 3. The predictions were further tested against different values of history window size ( $k$ ). While the AUC and F1-score generally decreased as the value of  $\tau$  increased, the performance is still seen to be better than other methods.

**Impact of Features.** The most important features for predicting future protest were identified using the Group Lasso regularization. The most important features in the *Social Embeddedness* category was found to be *num\_tweets*, online activism is predictive of a future offline protest. In the *Emotion* category, different type of emotion was found to be predictive in different social movements. For example, *disgust* was predictive in Charlottesville; *hate* in Ferguson I; and *fear* in Ferguson II. Among other intra-region dynamic features, social identity pronouns such as *we*, *them*, *they*, *people* were found to be predictive in all models.

## Discussion

The authors create a novel deep-learning architecture to predict offline protest based on online activities. Through various experiments, the authors show the strength of the proposed model; compared to baseline models. Since the data was filled up



randomly, the authors could have gone in more details on what the types of features were that got removed. It is possible that replacing missing values in features that do not change or change very slowly over time, could be effective. At the same time, doing so to highly dynamic features should lead to worse performance. Therefore, the effectiveness of the method of simply assigning random value from non-missing data in place of the missing data should be explored further. The robustness in missing data comparison of *ActAttn* with other methods has less meaning, as the authors have already shown that *ActAttn* is superior to baseline methods for complete data. For Charlottesville, missing data reduces the F-score in significantly.

### 2.3.3 ICEWS

O'Brien (2010) explains the approaches taken by the US military in developing an integrated Crisis Early Warning System (ICEWS). The goal of the ICEWS program is to develop an integrated, comprehensive and automated system that can forecast crisis on a sub-national, national to international level, as well as to provide support for the decision makers to mitigate said crisis. This system relies heavily on social science principles theories, data and methods. The article identifies the strength and limitations of contemporary approaches in utilizing social science data in crisis early warning and concludes with a discussion of Computational Social Science Experimentation Proving Ground.

#### Data

The input data involves creating profiles of archetypical leaders by the subject matter experts (SMEs), essentially forming a model of a country based on various population elements to mirror the society that was being studied. In addition to the SME data, newspaper articles were collected with 6.5 million news stories about the countries

of interest, collected from international sources such as AP, UPI, BBC Monitor, and regional sources such as India Today, Jakarta Post, Pakistan Newswire, and Saigon Times. The news sources were then complimented by country-specific data, such as the data from Economist Intelligence Unit, Freedom House, International Monetary Fund (IMF), World Bank, Political Instability Task Force (PITF), and the Correlates Of War (COW) project. News stories are then coded using the TABARI [10] system, events data is created in four categories: verbal cooperation/conflict, material cooperation/conflict.

## **Methodology**

There are four conflict modelling systems discussed in the project. The first model is an agent-based design created using data collected from SMEs to evaluate the causal dynamics for potential futures. Second, a logistic regression model was developed that used commonly used data such as regime type, GDP per capita and degree of cooperation between the government and civil actors. The third model was built as a geo-spatial network that used factors such as event count, trade ties and social similarity profiles between different countries. The fourth model aggregated the forecasts from the three models described so far using Bayesian techniques. The main idea behind the fourth model is to ascribe more or less confidence to any individual model forecast according to the countries for which it has demonstrated high performance.

## **Evaluation Metrics**

The following performance metrics were used for testing the modeling solutions:

$$Accuracy = \frac{\# \text{ of correct predictions}}{\# \text{ of predictions made}}$$

$$Recall = \frac{\# \text{ of correctly predicted conflicts}}{\# \text{ of conflicts that occurred}}$$

$$Precision = \frac{\# \text{ of correctly predicted conflicts}}{\# \text{ of conflicts predicted to occur}}$$

In addition to the evaluations above, an instability index was predicted using the four models for each of the country (none/low, moderate, or high level of instability). The categories for which the models were tested as: rebellion, insurgency, domestic political crisis, ethnic/religious violence, and international crisis.

## Results

The evaluation was done by dividing a year worth of data into quarters and was tested on two years worth of data. There were 16 crisis events that occurred during the two years of analysis and the models were to see how many of these 16 crises the system was able to detect. The models were shown to surpass the previously set benchmarks for categories: rebellion, insurgency, and ethnic/religious violence but did not perform well for domestic political crisis and international crisis. The article points out that ethnic/religious violence often starts as domestic political crisis such that while a domestic crisis could be one of the indications of an upcoming ethnic/religious violence, the underlying factors that could identify an upcoming domestic crisis have not yet been identified. While the results of the models look promising in anticipating onset of crisis, the models were able to forecast the correct quarter during which the occurrence or cessation occurred, only 25% of the time. This indicates a critical need to improve the accuracy regarding the temporal aspect of the predictions.

## Discussion

The goal of ICEWS is to study and identify relevant factors that when combined with other factors, systematically precedes crisis in a probabilistic way. The models described in the article utilize different types of social data, yet there are many underlying factors that have not been identified. The agent-based models or the components in it were constructed by Subject Matter Experts. This process could be improved by forming these models through an automated analysis of available textual data. The forecasts are made spatially on a country level and temporally in quarters. While this is still useful, a significant amount of improvements is required before the models could be utilized on a lower level such as a state or a district within countries. The number of crisis events (sixteen) is also quite low and may not have given us a strong evaluation of the models described. The problem with a computational approach to anticipating crises or unrest using social theories is that there are many competing theories in social science, and even prominent theories fail from time to time. Therefore, an insight realized by the author is to identify other underlying factors and theories from psychology, history, religion and economics and incorporate them into a computational social science experimental proving ground.

### 2.3.4 Shifting Sands

Shellman, Levey and Young (2013) state that many studies of states and dissidents are only focused on structural factors that rarely change, but more recently there have been studies that focuses on dynamic interactions of these parties. Dissidents are described as a non-state group of people that use non-institutional means such as protests and violence to pursue political goals. The paper suggests that forecasting state-dissident interaction is most efficient when the forecast considers both the

dynamic and structural factors. The paper talks about the Mono Islamic Liberation Front (MILF) (an Islamic separatist group fighting for autonomy in the Philippines) and the Liberation Tigers of Tamil Eelam (LTTE) (a dissident group fighting for autonomy in Sri Lanka). These groups were chosen because they have different contexts and motivations.

## Data

Event data were extracted from newspapers into four pieces of information: ‘actors’ that take actions against ‘targets’ on a given date. The software used to extract data was built on TABARI, which is a system that performs pattern-matching based on established dictionaries, with natural language processing added on top of it. The verb dictionary is a modified version of the CAMEO coding scheme [24]. The dataset contains information from millions of news reports from over 587 different news sources such as the Philippines Daily Inquirer, BBC Monitoring and Associated Press. An actor dictionary was then built from the news reports. Other information collected includes dates and types of significant events were recorded, these include dates of regime changes, dissident and government leadership changes, and the entrance and exit methods of these new leaders. The data was finally aggregated for each group in each country for each month.

Structural data such as GDP and regime types do not change very often so these were ignored. The authors were however able to collect some structural variables such as consumer prices and unemployment rates that were reported monthly. We assume the unemployment rates were on the country level, it is not clear what kind of consumer prices were recorded.

Sentiment data was collected to explore the relationship between the dissidents, government and the masses. Masses or social actors refer to the general population

not in government leadership and not associated with the dissident groups. These sentiments were also coded using a dictionary and the scale used was -10 to +10. Average monthly aggregated measures of social actors expression towards government and towards dissidents were calculated. These form the profile of each political actor.

## **Model**

The two groups are modelled separately in their own time series. Since the forecast is binary, the model only predicts if the group is in a state of violence or not, a logistic regression model was used. The model performs maximum likelihood calculation that produces estimated parameters that have the highest probability of producing the observed dataset. The statistical significance and substantive impact of variables were assessed using standard modeling practices. Additionally, in-sample and out-of-sample forecasts were performed to assess how well the models can predict the violent phases of the groups. The independent variables in the model are lagged at least a month to avoid endogeneity and to forecast violent campaigns.

## **Evaluation Metrics**

A receiver operating characteristics (ROC) curve is used to determine how well the model fits the observed data. These ROC curves plots the true positive against the false positive rates. The area under the curve represents how well the model predicts positive (violent) and negative (non-violent) outcomes. The substantive impacts of various independent variables with the probability were tested.

## **Results**

While there were differences between the role of different variables across the models, some variables such as FDI, government repression, societal sentiment towards gov-

ernment, and societal sentiment towards the dissident groups were consistently stable estimators. The model of LTTE performed well for both in-sample and out-of-sample data. The area under the ROC curve for LTTE was 0.94, the true positive rates were classified correctly about 76% of the time, and the true negative rates were classified 93% of the time. In case of MILF, the area under the curve was 0.99. Out of 108 observations in MILF, the model only misclassified 3 observations. It was seen that government repression increases the probability of violence up to a certain point, after which it decreases the probability of violence. In terms of societal sentiment for LTTE, it was seen that the more the masses are in favor of the government, the probability of violence decreases. In case of the MILF, the societal sentiment had no significant effect on the probability of violence. Additionally, for LTTE, it was seen that higher food prices led to lower probability of violence, this variable was not available for MILF. Furthermore, another observation that we have is that none of the variables other than government repression can be seen to have very noticeable relationship with the probability of violence. This is quite strange as the true positive and true negative rates in MILF model were 0.95 and 0.98 respectively.

## **Discussion**

The goal of this paper was to show that dynamic interactions between groups (dissidents and government) are more useful indicators in forecasting violent events. The authors chose two groups of dissidents with different contexts and motivations and utilized separate models to examine the relationship of various independent dynamic variables with the probability of violence. The data collections methodologies, which seem to be one of the most important portions of this research, were not discussed in detail. From what is available, it seems that various existing dictionaries were used to identify events and actors. In the paper, the authors only explore two dissident

groups with limited number of actors, it will very easily become much more intensive to build profiles and dictionaries for all the available actors. The temporal data is not fine-grained, as the intervals are divided into months instead of days, even though the researchers have access to daily data. This makes it easier to predict violence as the model does not have to predict short-term nuances.

### **2.3.5 Other Related Projects**

Qiao, Li, Zhang, Ding, Cheng & Wang (2017) developed a hidden Markov model-based framework leveraging GDELT data. This model considers event development stages in forecasting unrest. The proposed framework utilizes temporal burst patterns in GDELT to identify the underlying mechanics of event development. The main contribution of the paper are: (1) identification of stages of event development leading to social unrest, (2) hidden Markov model framework for event prediction, (3) first research paper to utilize GDELT and create a practical HMM based pipeline, and (4) research on Southeast Asia with results outperforming logistic regression and baseline methods. The proposed framework has four major components: ground set extraction, burstiness modeling, HMM training and finally the prediction. A prediction period is defined, beforehand, the model then predicts if a social unrest will occur in that period or not. Due to lack of any other ground-truth dataset, GDELT has been treated as ground truth. The number of mentions of a certain type of event, identified as social unrest by the authors, was normalized by dividing it by the total number of mentions of the same type of event in the last 90 days. A threshold is then chosen so that only days with significant rises in the number of mentions are classified as days with social unrest. Since the model attempts to develop an evolutionary model of social unrest, other event types that are assumed to precede a significant social unrest are selected and normalized in the same way. These



additional types of events are: Demand, Disapprove, Reject and Threaten. Overall the model performs better than LogReg and baseline methods. The paper presents the idea of using HMM, specifically the idea that social unrest occurs in stages and given the current social state of a location, it might be possible to forecast unrest. The predictions are made on a country level, while GDELT does provide events data with longitude and latitude information which can be used to pinpoint locations.

Matsumoto, Hwang & Frank (2014) analyzed speeches of ideologically motivated groups at different points before an identified act of aggression (AoA) or act of resistance (AoR). The emotions and their levels in the speeches were found to be predictive of the groups committing acts of violence. Anger, contempt and disgust were found to be the strongest indicator hate and often resulted in acts of aggression. The authors use various sets of historical data, such as data from World War I and II, US Persian Gulf War, US invasion of Iraq, etc. The emotions were identified 12 months, 6 months and 3 months before the AoA or AoR. As expected, the AoA were preceded by an increase in anger, contempt and disgust. The same emotions were observed to have decreased for AoRs. The findings in this paper are meaningful, but there are some limitations in the study. The data sample used was quite small and could also be considered out-of-date since the speeches analyzed were made before the invention of online social media platform. It could be argued that ideological groups communicate differently and work faster now due to the availability of online platforms.

### **2.3.6 Summary - Comparison with SURGE**

In summary, a significant amount of research has been conducted for understanding the dynamics of social unrest where special attention has been given to the forecasting of such events. Among the discussed projects, EMBERS seems to be the most advanced and shows diverse methodologies that work in parallel to give the best per-

formance possible for its system. Aggregation and cleaning of events data has been identified as a major challenge in almost all the discussed projects.

**Data Used.** EMBERS (Section 2.3.1) and *ActAttn* (Section 2.3.2) heavily rely on data from social media platforms such as Twitter and Facebook. These projects use social platform datasets to either identify general emotions of the public or detect the intensity of certain unrest events by looking at the frequency of certain keywords use. These analyses are done through textual analysis of tweets or posts. EMBERS has a more diverse set of data-sources compared to other projects, which includes ICEWS. *ActAttn* relies mainly on tweets, but also uses some static demographic data. SURGE, currently, only uses event information extracted from news articles. These news articles also go through various textual analyses, this is handled by GDELT. In SURGE, we use socioeconomic data collected from census data and infrastructure data collected from OpenStreetMap. The only other product that uses data related to the demographics is [55], as explained in Section 2.3.4. All projects other than EMBERS use data that is either related to specific type of event, for example [18] only uses data related to two sets of events, both related to the Black Lives Matter movement, and [55] only discuss the activities of two dissident groups. SURGE on the other hand, uses data under eight unrest categories, identified by social scientists as relevant, and currently does not focus on the actors. Focusing on specific set of events or actors reduces the amount of data, which makes it easier to filter out noises in the data. Such information in our model will be handled by the multi-factorial distance function in the future. Currently SURGE does not have a way of identifying noise in the data, such as duplicates or incorrect tagging, and thus SURGE relies on our data-source to handle such cases.

**Methodologies.** Spatio-temporal data clustering has not been used in any of the discussed projects. In SURGE, we introduce this approach to identify groups of

related events. For the purpose of forecasting unrest events, we use an agent-based model with events as agents. While an agent-based model has been implemented in the ICEWS, the agents represent certain persons or groups such as the archetypical leaders and followers. These profiles were created by subject matter experts and each model is specific to a country. In SURGE, we use a more general model and it is based on the dynamics of events themselves rather than individuals. We do consider the socioeconomic and infrastructural factors as part of the distance function used to establish relationship among events (or agents), but currently our model is not location specific. As we start to incorporate more factors such as motivations or actors into our distance function, the model should be able to more systematically handle location-specific nuances such as the relationship between significant groups, political leanings and other social factors during the distance calculations. To predict the location and time of unrest, the EMBERS system scans for dates and place names in social media posts, such as tweets and planned events on Facebook. Therefore, accuracy of the forecasts based on planned unrest is quite high in EMBERS. This has proven to be an effective technique as reported by the team at EMBERS themselves. The *ActAttn* project considers global and local features in their model. In SURGE, we do not differentiate between global and local features. Our forecasting of an events intensity increasing, or decreasing is based on an events neighborhood, and the weighted, multi-dimensional distance between events decides the amount of influence any neighbor has on a given event.

Regardless of the methodologies, all projects have identified that using only one domain of data is not enough for understanding the dynamics of social unrest. The use of social media data has been seen as effective in unrest forecast. Some projects use very specific sets of data, and though the results from these projects are quite meaningful, their models may not work in a general setting. SURGE can therefore be

considered a general model with novel approaches in understanding and forecasting social unrest.

## Chapter 3

### Multi-Factorial Distance Function

The multi-factorial distance function combines different types of distances to give a composite conceptual distance between each pair of events. To compute the spatial distance, we use the haversine formula [3] on the geographic coordinates of two events as the points are on a sphere and this formula provides a realistic distance between geographic locations. The temporal distance is calculated differently and is explained in Section 3.1. Each of the distances is then normalized using feature scaling [33] so that all values for that distance function reside within the range of 0 and 1.

#### 3.1 Events and Attributes

We refer to any object with spatio-temporal attributes as an event. We will be working with historical social unrest data collected for the south Asian country India. Social unrest events are often political in nature and involve activities such as peaceful or violent demonstrations and strikes. These events can escalate into more violent outbreaks such as fights, assaults or mass violence. In our research, we consider a social unrest event as a single event with the following parameters:

1. Location ( $l$ ): A geographic location of the unrest event that can be expressed in geographic coordinate system (latitude, longitude).

2. Start-date ( $t_{start}$ ): the start-date in any unit of time that can be used in calculation to find differences between two events.
3. End-date ( $t_{end}$ ): the end-date of that episode of event such that  $t_{end} \geq t_{start}$  . An episode is the time period for which an event remains active.
4. Category ( $c$ ): unrest event needs to be classified as one of the eight unrest types identified by this project: Appeal, Demand, Threaten, Coerce, Protests, Assault, Fight and Engage in Unconventional Mass Violence.
5. Reported-source ( $url$ ): the source of information for this event.
6. Population ( $p$ ): Note that it is difficult to find the number representing human population at any specific geographic point. The value depends on how much area around the point is being considered. Since we will be looking at real-world data, we can find population data for countries and administrative divisions or further sub-divisions of areas inside that country. Therefore, we will use the population values of the administrative division inside which the event has occurred. For example, if any unrest event has occurred inside the administrative division of Lincoln, Nebraska in the United States, the population value  $p$  assigned to this event will be the population of Lincoln, Nebraska.

Therefore, an unrest event can be represented as  $e = \langle l_e, t_{e,start}, t_{e,end}, c_e, url_e \rangle$ . Notice that the spatial and temporal variables are the only necessary parameters required to define a general event, the other variables have been added specifically to define an event of type unrest. An unrest event can have more attributes than listed above but they are not fundamental to the definition of an unrest event, they can be referred to as acquired attributes. These acquired attributes are inherited from the locality of the unrest events geographic location. For example, if *populationdensity* is calculated

as number of persons in a certain area  $A$ , then this value can also be assigned to any unrest event that occurred inside the same area  $A$ . We will be using two categories of acquired attributes for this thesis, socioeconomic and infrastructural and will discuss the details further in Sections 3.2.2 and 3.2.3.

## 3.2 Distance Calculation

In this section we will discuss the various parameters and methodologies of calculating each type of distance separately and then explain how these distances are combined to give a composite distance value between two events.

### 3.2.1 Spatio-temporal Distance

To combine the spatial and temporal distances into a spatio-temporal distance, we use a weighted sum of these two distances. In order to do so, we introduce the notion of the primary event and the secondary event: a primary event occurred before a secondary event. In case of the two events occurring on the same date, the event that occurred inside a geographic region with higher population is tagged as the primary, creating an ordered temporal relationship between all pairs of events. We also consider population sizes in the design of the weights. We see that time has less value in loosely populated area than it has in a densely-populated one. This is based on the hypothesis that an unrest event would take longer to evolve or spread in a sparsely populated area than it would in an area with high population. For example, in a sparsely populated village, it is normal for information to take a longer time to travel, whereas in cities where the population is dense, it would take a much shorter time for information to travel. We therefore argue that a larger temporal difference between events in a village is equivalent to a much shorter temporal difference in a

city. At the same time, if we look at the spatial distance, an unrest event will most likely spread much faster in a very densely populated area than it does in a loosely populated area. For example, a 5-km radius in a city is a smaller distance compared to a 5-km radius in a village, and so a 5-km distance in a city might be equivalent to a 2-km distance in a village. Given this, we have the following weight assignment scheme for the temporal weight ( $w_t$ ) and spatial weight ( $w_s$ ):

$$w_t = \frac{\text{population}(\text{event\_primary})}{\text{population}(\text{event\_primary}) + \text{population}(\text{event\_second})} \quad (3.1)$$

$$w_s = 1 - w_t \quad (3.2)$$

The calculation of spatial distance is relatively straightforward, but since we are working with social events that have a time range instead of a single date value, we cannot simply check the difference in dates to calculate the temporal distances. We therefore look at two different temporal metrics to calculate the final temporal distance.

### Temporal Gap

The gap metric can be defined as the time period between any two events compared to the total temporal span created by the start of an event to the end of a later event. Overlapping events have negative gap metrics. The gap metric allows us take the event duration into account when looking at the temporal gap of between events. If  $e_1$  and  $e_2$  are two events with  $(t_{e_1,start}, t_{e_1,end})$  as the start and end dates of event  $e_1$  and  $(t_{e_2,start}, t_{e_2,end})$  as the start and end dates of event  $e_2$ , the temporal span of a pair of events is defined as the time period from the minimum start-date of the pair



of events, to the maximum end-date. The temporal span is calculated as:

$$span = days\_between[\max(t_{e_1,end}, t_{e_2,end}), \min(t_{e_1,start}, t_{e_2,start})] \quad (3.3)$$

and the gap between events is calculated as:

$$gap = days\_between[\max(t_{e_1,start}, t_{e_2,start}), \min(t_{e_1,end}, t_{e_2,end})] \quad (3.4)$$

The gap metric is then given by:

$$\Delta t_{gap}(e_1, e_2) = \frac{gap}{span} \quad (3.5)$$

### Temporal Coverage

While the gap metric gives us a good estimate of the distance between two events, it is solely based on the gap between events and the overall temporal span, it does not take the coverage or range of individual event into account. We therefore introduce a coverage metric, which is calculated as the relative difference between the two time periods using a Euclidian distance metric by treating the start and end dates of any events as the  $x$  and  $y$  coordinates. The value is then divided by the Manhattan distance, this is done simply to normalize the Euclidean metric. If  $e_1$  and  $e_2$  are two events with  $t_{e_1,start}$  and  $t_{e_2,start}$  as their start dates, and  $t_{e_1,end}$  and  $t_{e_2,end}$  as their end dates respectively, the separation metric is given by:

$$\Delta t_{Euclidean}(e_1, e_2) = \sqrt{(t_{e_1,end} - t_{e_2,end})^2 + (t_{e_1,start} - t_{e_2,start})^2} \quad (3.6)$$

$$\Delta t_{Manhattan}(e_1, e_2) = \max(|t_{e_1,end} - t_{e_2,end}| + |t_{e_1,start} - t_{e_2,start}|, 1) \quad (3.7)$$

$$\Delta t_{coverage}(e_1, e_2) = \begin{cases} \frac{\Delta t_{Euclidean}}{\Delta t_{Manhattan}} & \text{if } \Delta t_{Manhattan} > 0, \\ 1 & \text{if } t_{Manhattan} = 0 \end{cases} \quad (3.8)$$

The final temporal distance between any pair events  $e_1$  and  $e_2$  is then calculated as:

$$d_{temporal}(e_1, e_2) = w_{gap} \cdot \Delta t_{gap}(e_1, e_2) + w_{coverage} \cdot \Delta t_{coverage}(e_1, e_2) \quad (3.9)$$

where,  $w_{gap}$  and  $w_{coverage}$  are weights assigned to the temporal overlap metric and the temporal separation metric, respectively, such that:  $w_{overlap} + w_{coverage} = 1$ .

Finally, we combine the spatial and temporal distance together to give a spatio-temporal distance. Since the spatial and temporal distances are in different numeric ranges, we normalize the spatial distance. The normalization of spatial distance is done by dividing the spatial distance with a threshold value ( $d_{spatial\_threshold}$ ) beyond which the spatial distance is assumed to have the same meaning (too far apart). The normalized spatial distance is then calculated as:

$$d'_{spatial} = \min \left( \frac{d_{spatial}}{d_{spatial\_threshold}}, 1.0 \right) \quad (3.10)$$

The spatio-temporal distance between events  $e_1$  and  $e_2$  is given by:

$$d_{spatio-temporal}(e_1, e_2) = w_s \cdot d'_{spatial}(e_1, e_2) + w_t \cdot d_{temporal}(e_1, e_2) \quad (3.11)$$

Where,  $d'_{spatial}(e_1, e_2)$  is the normalized value of the spatial distance, calculated by  $w_s$  and  $w_t$ , which are the spatial and temporal weights, respectively.

### 3.2.2 Socio-Economic Distance

To calculate the socio-economic distance between two events, we first take the difference in the values of the same type of socio-economic variable assigned to both events. For any unrest event  $e$ , we compute a corresponding vector of socio-economic variables:  $\langle v_{1,e}, \dots, v_{n,e} \rangle$ , where  $v_{k,e}$  represents the  $k^{th}$  socio-economic variable of the event  $e$  that has been normalized so that all the variables are comparable. Let  $wt_k$  be the weight applied to  $k^{th}$  variable based on its relative importance compared to other socio-economic variables. The difference between the  $k^{th}$  socio-economic variable of events  $e_1$  and  $e_2$  is given by:

$$\Delta_{socio-economic,k}(e_1, e_2) = |v_{k,e_1} - v_{k,e_2}| \quad (3.12)$$

The socio-economic distance between two events  $e_1$  and  $e_2$  is then calculated as:

$$d_{socio-economic}(e_1, e_2) = \sum_{k=1}^n wt_k \cdot \Delta_{socio-economic,k}(e_1, e_2) \quad (3.13)$$

Since the weight is relative,  $wt_1 + wt_2 + \dots + wt_n = 1$ .

### 3.2.3 Infrastructural Distance

As alluded to earlier, incorporating geospatial objects such as infrastructural elements is a challenge. Now, let us first suppose we have data for  $n$  different types of infrastructure elements (schools, hospitals, police-stations, etc.). We can define a vector of normalized distances of the nearest infrastructure element from event  $e$  by  $i_{1,e}, \dots, i_{n,e}$ . The difference in *infrastructure proximity* between two events  $e_1$  and  $e_2$  for the  $k^{th}$

infrastructure type is given by:

$$\Delta_{infrastructure\_proximity,k}(e_1, e_2) = |i_{k,e_1} - i_{k,e_2}| \quad (3.14)$$

For two events  $e_1$  and  $e_2$ , the *infrastructure\_proximity* distance can then be calculated as:

$$d_{infrastructure\_proximity}(e_1, e_2) = \sum_{k=1}^n wi_k \cdot \Delta_{infrastructure\_proximity,k}(e_1, e_2) \quad (3.15)$$

We then collect the number of infrastructure elements in a pre-determined radius of an events location, separately for each infrastructure type. For any event  $e$ , the normalized count of infrastructure elements of  $n$  different types, within a pre-determined geospatial radius  $r$  can be denoted by  $c_{1,e}, \dots, c_{n,e}$ . The difference in *infrastructure\_density* between two events  $e_1$  and  $e_2$  for the  $k^{th}$  infrastructure type is given by:

$$\Delta_{infrastructure\_density,k}(e_1, e_2) = |c_{k,e_1} - c_{k,e_2}| \quad (3.16)$$

For two events  $e_1$  and  $e_2$ , the *infrastructure\_density* distances can then be calculated as:

$$d_{infrastructure\_density}(e_1, e_2) = \sum_{k=1}^n wi_k \cdot \Delta_{infrastructure\_density,k}(e_1, e_2) \quad (3.17)$$

where,  $wi_k$  is the relative weight assigned to each individual infrastructure-type based on its importance. Since  $wi_k$  is relative,  $wi_1 + \dots + wi_n = 1$ .

Similarly, we can define a vector of normalized distances of the nearest infrastructure element from event  $e$ , for  $n$  different types of infrastructures by  $i_{1,e}, \dots, i_{n,e}$ . The difference in *infrastructure\_proximity* between two events  $e_1$  and  $e_2$  for the  $k^{th}$

infrastructure type is given by:

$$\Delta_{infrastructure\_proximity,k}(e_1, e_2) = |i_{k,e_1} - i_{k,e_2}| \quad (3.18)$$

For two events  $e_1$  and  $e_2$ , the *infrastructure\_proximity* distance can then be calculated as:

$$d_{infrastructure\_proximity}(e_1, e_2) = w_{i_k} \cdot \Delta_{infrastructure\_proximity,k}(e_1, e_2) \quad (3.19)$$

We will then combine these distances to give the comprehensive infrastructure distance between two events by assigning weights to the distances calculated above.

$$d_{infrastructure}(e_1, e_2) = w_{density} \cdot d_{infrastructure\_density}(e_1, e_2) + w_{proximity} \cdot d_{infrastructure\_proximity}(e_1, e_2) \quad (3.20)$$

where,  $w_{density}$  and  $w_{proximity}$  are weights based on the relative importance of the types of distances, such that  $w_{density} + w_{proximity} = 1$ .

### 3.2.4 Integrated, Multi-factorial Distance

The final distance between events  $e_1$  and  $e_2$  is given by the weighted sum of spatio-temporal, socio-economic and infrastructure distances.

$$d(e_1, e_2) = d_{socio-economic}(e_1, e_2) \cdot w_{se} + d_{spatio-temporal}(e_1, e_2) \cdot w_{st} + d_{infrastructure}(e_1, e_2) \cdot w_{in} \quad (3.21)$$

where  $w_{se}$  is the socio-economic weight,  $w_{st}$  the spatio-temporal weight, and  $w_{in}$  the infrastructure weight. Figure 3.1 shows the schematic for the multi-factorial distance function.

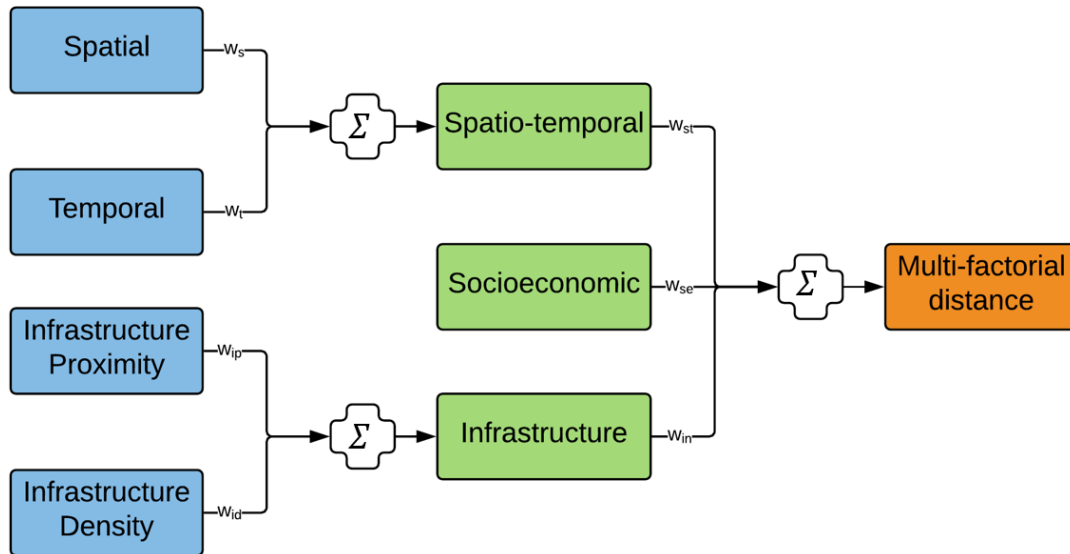


Figure 3.1: Schematic of the integrated, multi-factorial distance function

### 3.3 Implementation Details

The distance function takes two events ( $e_1, e_2$ ) as input, along with the weights for different distance types: spatio-temporal weight ( $w_{st}$ ), socioeconomic weight ( $w_{se}$ ), and infrastructural weight ( $w_{in}$ ). The events are objects that contain all the necessary variables within them (as explained in Section 3.1). If any of the weights is 0, the distance for that distance type is not calculated, otherwise we calculate the distance values for all three distance types as explained in Sections 3.2.1, 3.2.2 and 3.2.3. The code was first written in the R programming language for our clustering portion of the experiments. The code was then transferred to Java to use for the agent-based modeling part. To simplify the implementation, we have assigned default values to some of the parameters as listed in Table 3.1.

Table 3.1: Default parameters used in distance calculations

<b>Parameter</b>	<b>Default Value</b>	<b>Equation</b>
$w_{gap}$ and $w_{coverage}$	0.5	3.9
$d_{spatial\_threshold}$	100 km	3.10
$wt_k$	$1/n$	3.13
$wi_k$	$1/n$	3.17 and 3.19
$w_{density}$ and $w_{proximity}$	0.5	3.20

## Chapter 4

### Spatio-Temporal Data Clustering

The analysis of social events requires more than just the geo-spatial location and time stamps, as there are many other factors affecting emergence of such events arising from the collective behavior of a population. While it is possible to find patterns with only spatio-temporal data, in order to better capture the dynamics of social events, additional attributes must be considered that could connect events by more than time and space. Efficiency and scalability are always critical in data mining algorithms. Clustering algorithms generally do not scale well in terms of computation time as the size of the data increases. Additionally, spatio-temporal event data contains geospatial coordinates and time values so it cannot be directly plugged into standard clustering algorithms. It is therefore important to have a data structure and clustering algorithm that is particularly designed for spatio-temporal data, to efficiently perform clustering without losing the details in the data.

#### 4.1 Spatio-Temporal $k$ -Dimensional Tree-based DBSCAN

The main challenge is scalability of the clustering process using the aforementioned distance function. That is, a pre-computed distance matrix must be created in order to cluster the data. For very large data sizes, this becomes computationally expensive



as distance must be computed between every possible pair of events. Our strategy to addressing this challenge lies on the notion of a  $k$ -dimensional tree (or  $k$ -d tree) structure. More specifically, after pre-processing the dataset, we create a  $k$ -d tree data structure using the latitude, longitude and date values. Figure 4.1 shows the basic schematic of this process. After the creation of  $k$ -d tree, we pass the events dataset, the tree and a tuple search radius that contains the geospatial radius and a temporal radius to a Fixed Radius Near Neighbor (FRNN) algorithm [8]. The FRNN algorithm, for each event, first creates a bounding box by finding the maximum and minimum values possible for the longitude, latitude and date and then traverses the tree to find the points that fall within the bounding box. After the algorithm is finished, a FRNN object contains, for each event, the events id and a list of the other events in its valid neighborhood. We refer to this list as the FRNN neighborhood. The DBSCAN then uses the distance function to calculate comprehensive distances between an event and only its FRNN neighborhood.

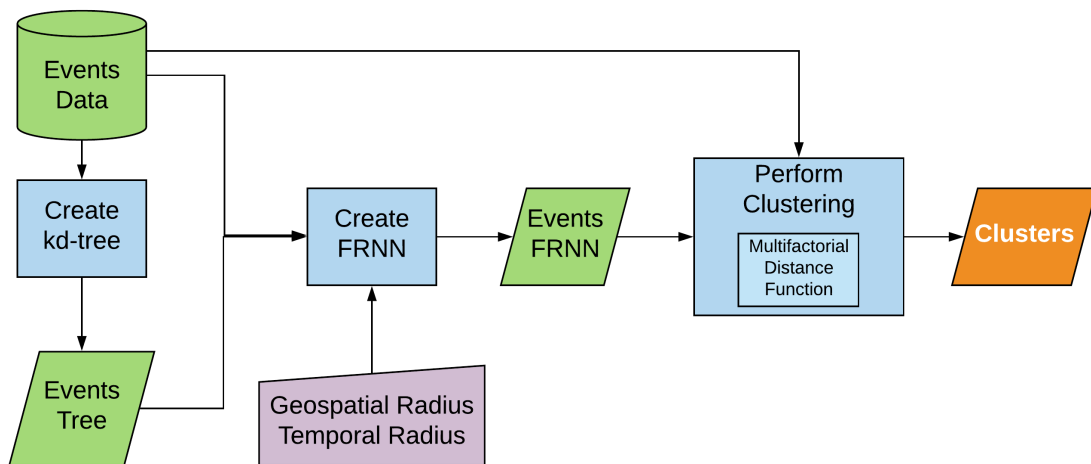


Figure 4.1: Schematic of ST-KDT-DBSCAN describing the processes involved in clustering of events data.

## 4.2 Implementation Details

Our implementation of  $k$ -d tree has three dimensions (i.e.,  $k = 3$ ): (1) longitude, (2) latitude, and (3) date with the date stored in the YYYYMMDD format to allow for numeric sorting. The dataset is first sorted based on the longitudes only, we then find the median longitude value in this sorted dataset and create a root node with it. Each data point with a longitude value higher than this median is passed to the left branch and that with a lower value is stored in the right branch. Then we carry out the process for the latitude and date dimensions for the second and third levels, respectively. The fourth level is again sorted based on longitude (first dimension), the fifth by latitude (second dimension) and the sixth by date (third dimension), and so on. The  $n^{th}$  level is sorted by the  $(((n - 1) \bmod k) + 1)^{th}$  dimension. This process is repeated until the whole dataset is organized.

Since we only store data point pairs that are within a spatial and temporal distance of one another, we use a Fixed-Radius Near Neighbors (FRNN) algorithm. Given a set of data points and a radius  $r > 0$ , the FRNN algorithm returns all pairs of points within a distance of  $r$  from each other. In our implementation, for each data point, we store the ID of all other data points that are within a given spatial and temporal radius from it. The radius is an ordered list (tuple) with two values: the first value is the spatial radius in kilometers (km) and the second value is the temporal radius (days). The algorithm then traverses through the  $k$ -d tree and finds all the data points inside this radius for each point. Figure 4.2 illustrates an example of a 3-d tree for a data point (i.e., an event) in our framework. Note that the tree branches are created based on the numeric values of all the variables (dimensions), not on any spatial or temporal distance calculation. Thus, implementation-wise, in order to computationally identify each neighboring events of an event specifically, we therefore create a bounding box

for each point to restrict the search area to find these neighboring events.

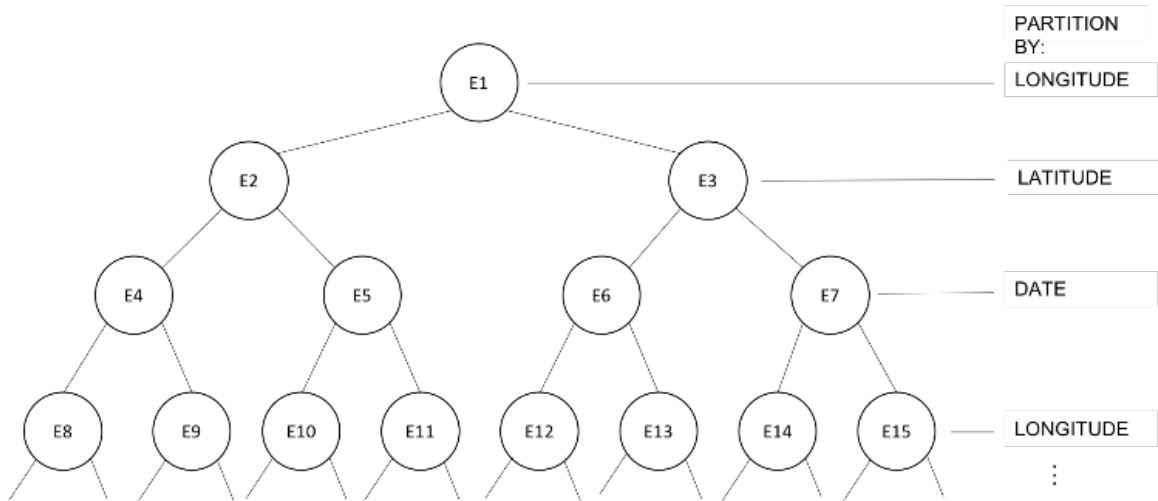


Figure 4.2: 3-d tree structure for event data to enable more efficient search of events within a bounding box

While the temporal bounding is simple to construct, the geospatial bounding requires geospatial distance calculation. For each data point, we first take the geospatial coordinates and create four new points at a spatial distance given by the radius and bearing angles of  $0^\circ$ ,  $90^\circ$ ,  $180^\circ$ , and  $270^\circ$ . (Bearing is the angle between a line connecting two points and a north-south line.) The distances are calculated using a variation of the haversine formula.

$$lat2 = \arcsin \left( \sin lat1 \cdot \cos \frac{d}{R} + \cos lat1 \cdot \sin \frac{d}{R} \cos B \right)$$

$$lon2 = lon1 + \arctan 2 \left( \sin B \cdot \sin \frac{d}{R} \cdot \cos lat1, \cos \frac{d}{R} - \sin lat1 \cdot \sin lat2 \right)$$

where  $lon1$  and  $lat1$  are the longitude and latitude of the data point, respectively,  $B$  the bearing angle,  $d$  the spatial radius (in kilometers or miles) value within which the search is to be restricted and  $R$  the approximate radius of the Earth (same unit as the spatial radius).  $lon2$  and  $lat2$  are the longitude and latitude of the new point,

respectively. Note that the longitude and latitude values must be in radians during calculation ( $\pi \text{ rad} = 180^\circ$ ) and the final results are then converted back to degrees. Using this identification and conversion technique, we first calculate points in four cardinal directions from each point, and then get the maximum and minimum values of longitude and latitude from these four points to form a spatial bounding box. Figure 4.3 shows the process for the creation of spatial bounding box. Once we have the maximum and minimum values for longitude, latitude, and date, we simply traverse down the  $k$ -d tree to find all neighbors within the spatial and temporal bounds of the bounding box.

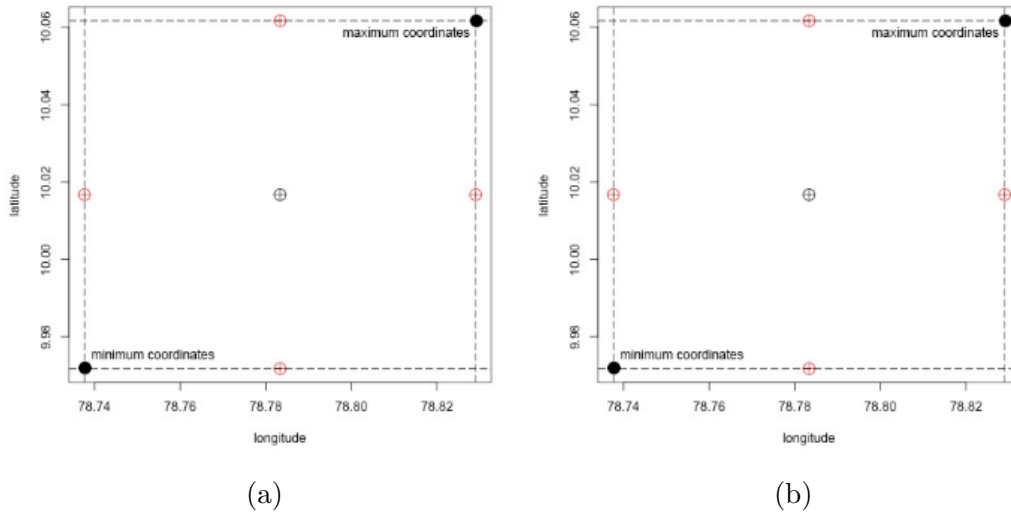


Figure 4.3: Spatial bounding box formation for FRNN: (a) Calculating spatial coordinates in four cardinal directions. (b) Creating spatial bounding box using the coordinates.

For clustering using DBSCAN, in order to incorporate the FRNN notion to reduce the computational complexity, a modified DBSCAN that checks only the neighbors established using our FRNN algorithm is needed as opposed to the standard implementation of DBSCAN. Algorithm 1 specifies the Spatio-Temporal  $k$ -Dimensional Tree-based DBSCAN (ST-KDT-DBSCAN). Note that DBSCAN uses a concept of

core points and neighbors to form clusters. Core points are points that have at least a *minPts* number of points, including itself, within a neighborhood radius of some *eps* value. The points inside the neighborhood of a core point are its neighbors. If any of the neighbors is a core point too, their neighbors are also added to the cluster. In our implementation, in addition to feeding the dataset, *eps* and *minPts* values into the modified DBSCAN function, we also create a *k*-d Tree (Step 2) and a FRNN object (Step 3). The FRNN object is a key-value object, for example, a dictionary in python language, that contains each events ID as the key and a list of neighborhood event ids as the item. In our design, the algorithm then checks the FRNN object for each event, and only calculates distances to its FRNN neighbors (Steps 13–19), to form a group of events. The algorithm, similar to the standard DBSCAN, then simply checks if this group of events is at least as large as specified by the *minPts* parameter. The output of this algorithm is a table with the same structure as the input data and additionally the cluster number assigned to each record under the column “cluster”. The ST\_KDT\_DBSCAN (Algorithm 1) is shown below with additional input parameters *frnn\_radius* and *distance\_types* the functions *KD\_TREE* and *FRNN* return a *k*-d tree data structure and a FRNN object, respectively.

---

**Algorithm 1** Spatio-Temporal k-Dimensional Tree-based DBSCAN
 

---

```

procedure ST_KDT_DBSCAN(events, frnn_radius, eps, minPts,
                          distance_types)

  events_tree  $\leftarrow$  KD_TREE(events)
  events_frnn  $\leftarrow$  FRNN(events, events_tree, frnn_radius)
  visited  $\leftarrow$   $\emptyset$ 
  cluster_number  $\leftarrow$  0
  for all events – id as key do
    cluster  $\leftarrow$   $\emptyset$ 
    q  $\leftarrow$  Queue()
    if (key not visited & LENGTH(events_frnn[key])  $\geq$  minPts) then
      PUSH(key) in q
      while q not empty do
        current  $\leftarrow$  POP(q)
        frnn_neighbors  $\leftarrow$  events_frnn[current] ▷ new
        neighbors  $\leftarrow$   $\phi$ 
        APPEND(current) in neighbors
        for all frnn_neighbors as frnn_n do
          d  $\leftarrow$  DISTANCE_FUNCTION(events[current], events[frnn_n],
                                       distance_weights)

          if d  $\leq$  eps then
            APPEND(frnn_n) into neighbors
          if LENGTH(neighbors)  $\geq$  minPts then
            APPEND(current) in visited
            APPEND(current) in cluster
            for all neighbors as n do
              if n not in cluster then
                APPEND(n) in cluster
                PUSH(n) in q

          if LENGTH(cluster)  $\geq$  minPts then
            cluster_number  $\leftarrow$  cluster_number + 1
            Assign cluster_number to all events in cluster
  return events

```

---

## Chapter 5

### Agent-Based Modelling

#### 5.1 Model Design

In our framework, each individual social unrest event acts as an agent that can communicate with other agents within its predefined neighborhood. An agent is considered a neighbor of another agent if the distance between the two agents is less than or equal to a predefined threshold radius. This distance is defined by a combination of spatio-temporal distance and environmental factors such as socioeconomic conditions and infrastructures. Since we use spatio-temporal data for our research, the environment of our simulation is a 2D map representing the locations on Earth. The simulation can begin at any point in time, each tick or step in the simulation is considered as a day. We will refer to a tick as a day and represent the tick-number as the day-number or a date value referred to as the simulation-date. The simulation model is implemented in Repast Symphony [43], developed by Jonathan Ozik, written in Java. Agents are defined as Java classes, the agents used in our project are the event-type agents and an observer-type agent. The observer-type agent is a passive agent whose only function is to observe the environment and record the data for analysis. The environment of each agent is comprising of various socioeconomic variables and infrastructural objects.

The dataset contains recorded occurrences of various past unrest events, over a large period of time. Whenever an event occurs, an agent is added to the environment during the simulation. The simulation begins at some starting point (day-1) with a certain number of events or agents. Each event has prior knowledge of its environment and will form a neighborhood of predefined radius for itself. Based on the intensities of its neighbors, events can either raise its intensity value or decrease it. Note that decreasing the intensity might result in the death of an agent, once an agent has died, it cannot come back to life and is excluded from all further calculations. An increase in intensity might result in an intensity much higher than the intensity that an event is born with. To retain practicality of the simulation, we define a maximum intensity (i.e., intensity-cap) that any agent may attain. If an agents intensity exceeds this cap, then its value will be set to this cap. Agents that survive the current day remain active on the next day. Additionally, as the simulation date moves forward by a day on each tick, any event whose date of occurrence is same as the simulation date is added to the simulation with a starting intensity based on its category. The assignment of starting intensity is explained in Section 5.1.

### **Intensity Profile**

Each agent has a starting intensity assigned to it based on its category. We first need to formulate a ranking order of the unrest categories and assign them intensity values that reflect the severity of unrest as well as a general order of development. The order of development refers to the category changes that events will generally go through as they become more violent or risky. For example; if an event starts as an appeal, the step for its evolution might be to change into an fight rather than UMW. Table 5.1 shows the categories we will be using for this project. These categories were predefined by the data-source as discussed in Section 6.1.1, the data-set contains



recorded occurrences of various past unrest events through newspapers and media sources. To determine a suitable value of starting intensity, we examine the event count for each category of events in India during the period of 201401 to 201406, as this is the period of analysis for our research. A total of 155,508 events under the eight aforementioned categories were recorded by GDELT. We first divide all the events by categories and by geographic states in India. Table 5.1 shows the average number of events in each state under each category. In our model, we assume that a high intensity event is formed through collaboration or fueling of low intensity events. We therefore expect events with high intensity to occur less frequently than events with low intensity. Based on this assumption, we assign the starting intensity to each event category based on their count distribution. We observe that the counts in Threaten, Demand and Protest are very close to each other, these event-types are quite similar to each other, we therefore combine these event-types into a single category.

Table 5.1: Average event counts per category for India states (201401 201406)

<b>Rank</b>	<b>Event Category</b>	<b>Average Event Count</b>
1	Engage in UMV	5
2	Threaten	207
3	Demand	326
4	Protest	352
5	Assault	450
6	Coerce	815
7	Fight	889
8	Appeal	1,399

These counts are scaled between 1-10 as we intend to assign an intensity level to the event categories, the scaled counts are then inverted since low count represents high intensity. We then fit this inverse count onto a polynomial equation of degree 3 using least squared error. The polynomial curve fitting is shown in Figure 5.1. Since the fit error is much smaller in the polynomial fit degree 3, we choose the values listed

in Table 5.2 as the final intensity values for each of the event-category.

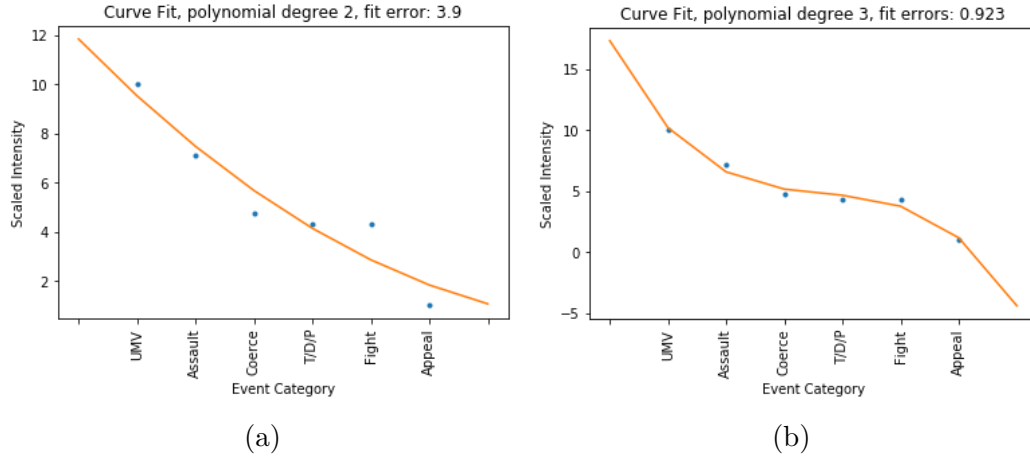


Figure 5.1: (a) Curve fit for polynomial equation of degree 2, sum of the squares of the fit errors is 1.4. (b) Curve fit for polynomial equation of degree 3, sum of the squares of the fit errors is 0.7

Table 5.2: Modified average event counts and intensities

Rank	Event Category	Average Event Count	Scaled Count	Inverse	Assigned Intensity
1	EngageInUMV	5	10		10.1840
2	Assault	450	7.1269		6.5696
3	Coerce	815	4.7704		5.1592
4	Threaten Demand Protest	885	4.3185		4.6555
5	Fight	889	4.2926		3.7612
6	Appeal	1,399	1		1.1788

## 5.2 Design Principles of Considering Influence of Neighboring Agents

Each agent has a neighborhood and the neighborhood will assert an influence on the agent. First, we assume that each agent can only assert a “positive” influence on another agent i.e., by adding a non-negative amount to the other agents intensity.

This is a valid assumption since we only consider social unrest events. If we consider other types of events in the future, then it is plausible to have a “negative” influence where an agent reduces the intensity of another agent.

The following principles guide our design on how an agents intensity changes, including on how to consider a neighboring agents influence on an agent’s intensity.

### 5.2.1 Decay Principle

Each agent loses some intensity as the simulation moves from  $k^{th}$  day to  $(k+1)^{th}$  day, where  $k$  is any day during the simulation. We control this by applying a recovery rate ( $\lambda$ ) to the intensity of the agents. This parameter defines what percentage of the original intensity should an event retain on the next day.

### 5.2.2 Distance Principle

A neighbor that is closer to an agent has more influence on that agent than a neighbor that is further away. The distance is defined by the multi-factorial distance function discussed in Chapter 3 above.

### 5.2.3 Intensity Principle

A higher intensity neighbor has more influence on an agent than a lower intensity neighbor at the same distance. For example, if an event has two neighbors A and B, where A has an intensity of 4.0 and the neighbor B has intensity 5.0, then B has more influence on the event compared to A.

### 5.2.4 Intensity Difference Principle

The difference in intensities of two agents is also important. If  $e_1$  and  $e_2$  are two agents (events) where  $e_1$  has a higher intensity than  $e_2$ : (a)  $e_1$  influences  $e_2$  more

than  $e_2$  influences  $e_1$ , (b) the higher the difference in intensities of  $e_1$  and  $e_2$ , the more  $e_1$  influences  $e_2$ ; and the less  $e_2$  influences  $e_1$ . Therefore, the influence is asymmetric.

### 5.2.5 Neighborhood Principle 1

Number of neighbors also plays a role on the amount of influence an agent receives. In a scenario where two agents have neighbors with same intensities, the agent with more neighbors receives more influence than an agent with less neighbors.

### 5.2.6 Neighborhood Principle 2

It is often the case that the records of peace-keeping or anti-unrest events are not available. In the absence of these types of events that can neutralize unrest events, and the abundance of unrest events in some areas can lead to the total intensity increasing unrealistically. We handle this issue by normalizing the total impact of the neighborhood on any event, by the number of its neighbors.

## 5.3 Influence

The equation for calculating the intensity of agent  $e$  for day  $t_{n+1}$ , where  $n$  is the day-number in the simulation, is given by:

$$I(e, t_{n+1}) = \lambda I(e, t_n) + \frac{\gamma}{\beta} \sum_{\bar{e} \in N(e)} I(\bar{e}, t_n) [\alpha(e, \bar{e}) + (1 - d(e, \bar{e}))^2] \quad (5.1)$$

where  $N(e)$  is the neighborhood of the agent  $e$ ,  $D(e, \bar{e})$  is the composite distance between agents  $e$  and  $\bar{e}$ ,  $\lambda$  is the Recovery Rate,  $\gamma$  is the Influence Rate of the neighborhood,  $\alpha(e, \bar{e})$  is the gap factor given by:

$$\alpha(e, \bar{e}) = \begin{cases} \frac{I(\bar{e}, t_n) - I(e, t_n)}{maxgap} & \text{if } I(\bar{e}, t_n) > I(e, t_n) \\ \left(\frac{I(\bar{e}, t_n) - I(e, t_n)}{maxgap}\right)^2 & \text{otherwise} \end{cases} \quad (5.2)$$

where, *maxgap* is the difference between the intensity-cap and the minimum intensity below which an agent is assumed to be dead. The Influence Rate ( $\gamma$ ) can be defined as a weight of “contributions/impacts” from all the neighbors. The value of  $\beta$  represents which neighborhood principle is being applied, and is defined as:

$$\beta = \begin{cases} 1, & \text{for Neighborhood Principle 1} \\ |N(e)|, & \text{for Neighborhood Principle 2} \end{cases} \quad (5.3)$$

The equation for  $I(e, t_{n+1})$  satisfies all four principles: (1) the first component on the right-hand side (i.e.,  $\lambda I(e, t_n)$ ) meets the Decay Principle, (2) the second component (the summation) in itself meets the Neighborhood Principle; that is, the more neighbors an agent has in its neighborhood, the sum will be larger, given that everything else is equal; (3) the component  $I(\bar{e}, t_n)$  in the summation satisfies the Intensity Principle; (4) the component  $(1 - d(e, \bar{e}))^2$  in the summation meets the Distance Principle; and (5) the component  $\alpha(e, \bar{e})$  as detailed in Equation 5.2 meets the Intensity Difference Principle. The second part of the equation is the influence made by the neighborhood on each event.

## 5.4 Implementation Details

The simulations are implemented in Repast Symphony (<https://repast.github.io/>) developed by Jonathan Ozik, written in Java. Agents are defined as Java classes.

There are two types of agents defined in our project; (1) event-type agents and (2) observer-type agents. The simulation program is capable of producing outputs collected from each agent in the model. The observer-type agent is a passive agent whose only function is to observe the environment and record the data for analysis. We allow users to pass values into the simulation for the following parameters: spatio-temporal weight ( $w_{st}$ ), socioeconomic weight ( $w_{se}$ ), infrastructural weight ( $w_{in}$ ), recovery rate ( $\lambda$ ), influence rate ( $\gamma$ ), and history window ( $w_H$ ). The weights are provided with a default value of  $(1/3)$ , and the history window has the default value of 7 days. All other parameter values must be provided. An event-type agent, on the other hand, is an agent representing an event, such that it actively monitors its environment and changes its event intensity in response to the stimuli as perceived from its neighborhood, as described in Section 5.2.

To create these agents, the program first reads the events data from a file (.csv), and creates an agent for each record in the file. The information included in the data are: longitude, latitude, event-date, unique id, event category (appeal, threaten, fight, etc.), socio-economic feature values such as employment rate and literacy rates in the region where the event is located, and the infrastructure features. A numeric intensity value is then assigned to each of the event based on their event-category as described in Section 5.1. Note that all the socioeconomic and infrastructural parameter values are pre-calculated and available for each event. The simulation initializes at a starting date which is the same as the minimum event-date given in the file, the starting date is always 2014/01/01 for our experiments. At each step/tick in the simulation, the date of the simulation is increased by one day. If the date in the simulation is the same as any event's event-date parameter, the event is made "alive". This allows an event to interact with other events that are alive. At the end of the simulation, output files are created and saved in the set directory.

## 5.5 Evaluation Methodology

In this section, we discuss the evaluation methodologies we use to show the viability of our agent-based modelling approach in anticipating social unrest. In the experimentation portion of the project, we first use synthetic data to demonstrate the models functionality and flexibility. Real world historical data on social unrest will then be used to further explore and validate the model. We will evaluate the predictions made for an entire area rather than a single point. Due to the large number of events in the real data set, it becomes unfeasible to evaluate the prediction of each event. Furthermore, the real data only contains a single date representing the occurrence of an event, this makes the validation of a single event's progression (change in intensity) impossible as the same event does not exist on the next day. We therefore combine the individual predictions of all events in an area  $A$  to calculate the predicted intensity for the entire area on the next day. We then use this predicted intensity and compare it to the intensity calculated from the real data for the area  $A$ .

### 5.5.1 Evaluation using synthetic data

We first begin by simulating synthetic social unrest data to demonstrate the design principles of our simulation model. In our experiments, we use the term “observed” event to represent the primary agent whose behavior will be monitored as it reacts to other agents in its neighborhood. We explore the impact made by each individual neighbor as well as the change in intensity of the simulated agents. The scenarios explored will include:

- Agents with no neighbors, to evaluate the Decay Principle.
- Agents with neighbors, to evaluate the rest of the principles.

- Neighbors with the same intensity level, to evaluate Distance, Intensity, and Neighborhood principles.
- Neighbors with different intensity levels, to evaluate Distance, Intensity, Intensity Difference, and Neighborhood principles.

### 5.5.2 Evaluation with real data

We use several time series validation methodologies to validate and analyze or simulation data. Due to the lack of important data features such as negative intensity events and incomplete information on the causal analysis of each individual unrest event, it is challenging to obtain empirically optimal parameters for the simulation parameters: Recovery Rate, Influence Rate and Neighborhood size. We therefore perform simulation analysis using various combinations of these parameters to identify the combinations that result in the least error. We calculate the prediction error in two ways: *intensity value differences* and *intensity trend differences*.

#### 5.5.2.1 Total Intensity Value Error

As we run through simulations, we add new events that are available in the dataset. We then allow the simulation to run for any given number of days without new input, this is referred to as the *observation period* of the simulation. The individual intensities of all events are recorded on each day during the *simulation period*. Let  $E_t = \{e_{1,t}, \dots, e_{n,t}\}$  denote the set of all events and  $I_{e_{i,t}}$  denote the intensity of event  $e_{i,t}$  for day  $t$ , where  $n$  is the total number of active events. Note that the prediction made for the day  $t$  is based on the intensities of events on day  $(t - 1)$ . The total



predicted intensity of a geographical area  $A$  for day  $t$  is given as:

$$I^{simulation,total}(A, t) = \sum_{i=0}^n I_{e_i,t} \quad (5.4)$$

If  $I^{real}(A, t)$  is the sum of intensities of all events occurring on day  $t$  as observed in real data, then the total-intensity prediction error on day  $t$  is given as:

$$err_{total.intensity,t} = \frac{I^{simulation,total}(A, t) - I^{real,total}(A, t)}{I^{real,total}(A, t)} \quad (5.5)$$

The prediction error is calculated as the percentage difference of the observed value, a positive prediction value indicates overestimation and a negative value indicates underestimation. The *Total Intensity Value Error* is then calculated as the sum of squared errors. If  $t_{start}$  is the starting date of the observation period and  $t_{end}$  is the end of observation period, the *Total Intensity Value Error* for the simulation is given by:

$$Err_{total.intensity} = \sqrt{\sum_{t=t_{start}}^{t_{end}} err_{total.intensity,t}^2} \quad (5.6)$$

### 5.5.2.2 Average Intensity Value Error

In this methodology, we compare (1) the average of the estimated intensities of each event from simulations with (2) the average of the reported intensities of the events in the real world. With this approach, we intend to capture the general state or mood of any region and compare it with the simulated mood generated by our simulation model. The predicted average intensity of the area  $A$  for any day  $t$  is represented as:

$$I^{simulation,average}(A, t) = \frac{I^{simulation,total}(A, t)}{n} \quad (5.7)$$

where  $n$  is the number of alive events. If  $I^{real,average}(A, t)$  is the average intensity of all events occurring on day  $t$  in the area  $A$  as observed in real data, then the average prediction error on day  $t$  is given as:

$$err_{average.intensity,t} = \frac{I^{simulation,average}(A, t) - I^{real,average}(A, t)}{I^{real,average}(A, t)} \quad (5.8)$$

If  $t_{start}$  is the starting date of the observation period and  $t_{end}$  is the end of observation period, then the *Average Intensity Value Error* is then calculated as:

$$Err_{average.intensity} = \sqrt{\sum_{t=t_{start}}^{t_{end}} err_{average.intensity,t}^2} \quad (5.9)$$

### 5.5.2.3 Trend Error

As noted earlier, we lack some important features of the data that would give our models more complete knowledge of the relationship of various unrest events. We therefore attempt to predict if the intensity in the future is likely to increase, decrease or remain constant for an area. We define a minimum value  $\delta_{neutral}$  such that all predictions where the percentage change in intensity from day  $t - 1$  to  $t$  is under this minimum value, the prediction is neutral. The prediction value of *increase*, *decrease* and *neutral* is assigned as:

$$pred_{trend,t} = \begin{cases} increase, & \text{for } \frac{I^{simulation}(A, t) - I^{real}(A, t - 1)}{I^{real}(A, t - 1)} > \delta_{neutral} \\ decrease, & \text{for } \frac{I^{simulation}(A, t) - I^{real}(A, t - 1)}{I^{real}(A, t - 1)} < -\delta_{neutral} \\ neutral, & \text{otherwise} \end{cases} \quad (5.10)$$

where  $I^{simulation}(A, *)$  and  $I^{real}(A, *)$  could be either  $I^{simulation,total}(A, *)$  and  $I^{real,total}(A, *)$ , or  $I^{simulation,average}(A, *)$  and  $I^{real,average}(A, *)$  depending on which type of intensity value error we use. Note that the intensity in the real data is also assigned a label of *increase*, *decrease* and *neutral* based on comparison with the real intensity of the previous day. It is computed in a similar way:

$$real_{trend,t} = \begin{cases} increase, & \text{for } \frac{I^{real}(A, t) - I^{real}(A, t-1)}{I^{real}(A, t-1)} > \delta_{neutral} \\ decrease, & \text{for } \frac{I^{real}(A, t) - I^{real}(A, t-1)}{I^{real}(A, t-1)} < -\delta_{neutral} \\ neutral, & \text{otherwise} \end{cases} \quad (5.11)$$

We then calculate the prediction error by evaluating the predictions made during any set of simulations. Table 5.3 lists the error values assigned to correct and incorrect predictions of various types. If  $err_{trend,t}$  is the trend error observed for any prediction

Table 5.3: Error for true and false trend prediction ( $err_{trend,t}$ )

$pred_{trend,t} \backslash real_{trend,t}$	increase	neutral	decrease
<i>increase</i>	0	-0.5	1
<i>neutral</i>	-0.5	0	-0.5
<i>decrease</i>	1	-0.5	0

made for day  $t$  as given by Table 5.3, then the *Intensity Trend Error* of the model is calculated as:

$$Err_{trend} = \sum_{t=t_{start}}^{t_{end}} err_{trend,t} \quad (5.12)$$

#### 5.5.2.4 Confidence Factor

In this section, we describe our approach to assigning a confidence value to all the predictions made during the simulation. This confidence value of a prediction made

for event  $e$  is based on two things: (1) the density of neighborhood around event  $e$ , and (2) the variance in the intensities of neighbors as well as their distances from event  $e$ . The confidence value of each prediction is inversely related to the variation in distances and intensities, and the confidence value is directly related to the neighborhood density.

There are different ways to calculate the neighborhood density. For our model, the neighborhood density of an event  $e$ , denoted by  $\rho_e$ , is the number of its neighbors divided by the maximal number of neighbors in the domain  $R$ . If the number of neighbors is greater than the threshold value, we set the neighborhood density to 1.0. For any event  $e$ , we then calculate a weighted intensity value that is a combined representation of each neighbor's intensity and its distance from the main event  $e$ . Suppose  $[e'_1, \dots, e'_n]$  is the list of neighbors of an event  $e$ , and that  $[I'_1, \dots, I'_n]$  is the list of intensities of the neighbors, On any day  $t$ , we assign weights to each neighbor denoted as  $[w'_1, \dots, w'_n]$ . We then calculate the variance in the list of the weighted intensities  $[w'_1.I'_1, \dots, w'_n.I'_n]$  for each event. The weight is calculated based on each neighbor's distance to  $e$ , conforming to the Distance Principle of our model. The weight of the  $i^{th}$  neighbor is calculated as:

$$w'_i = \frac{R - d(e, e'_i)}{R} \quad (5.13)$$

To describe the variation in the weighted intensities, we use the coefficient of variation ( $C_V$ ) [14], because we are interested in the relative spread of the intensity and distance of neighbors compared to each event rather than the numeric values of the variance. If, for a list of neighbors, the standard deviation and mean of their weighted intensities

is denoted by  $S$  and  $M$  respectively, the coefficient of variation is computed as:

$$C_V = \frac{S}{M} \quad (5.14)$$

Note that  $C_V$  can take a value larger than unity, we therefore cap the value of  $C_V$  to 1.0 as it denotes maximum variability, this modified coefficient of variation is referred as  $C'_v$  and is calculated as:

$$C'_v = \min\left(\frac{S}{M}, 1\right) \quad (5.15)$$

Since both of our variables, neighborhood density and variance are in the range  $[0,1]$ , we calculate the confidence in a prediction made for event  $e$  is computed as:

$$confidence_{e,t} = wt_{var} \cdot (1 - C'_v) + wt_{\rho} \cdot \rho_e \quad (5.16)$$

In our validation methodologies, we examine the predictions made for an area rather than a single location. Therefore, we calculate the average confidence ( $confidence_{A,t}$ ) in the prediction made for area  $A$  and day  $t$  as the average of individual confidence values calculated for each event found in that area  $A$  for all  $N$  number of days prior to day  $t$  in the simulation.

### 5.5.2.5 Time Lag

It is possible that the predictions made by our model have less error when the prediction is made for more than one day in the future. We represent this concept as a lag, any prediction made for the next day is considered as a lag of 1 day, similarly any prediction made for two days into the future is said to have a lag of 2 days. Therefore, allowing the simulation to run for additional number of days, we calculate the simulated average intensity on any day  $t_{lag}$  in the future. We evaluate the average

intensity value error and the confidence factors for the same data with a lag of 4 days (i.e.,  $t_{lag} = 4$ ) and for 7 days (i.e.,  $t_{lag} = 7$ ).

## Chapter 6

### Experiments and Results

In this chapter we will first list the data sources used in our project, then we discuss the setup for our experiments. Our experiments and results are divided into two main sections: Section 6.2 contains the details for spatio-temporal data clustering and Section 6.3 contains the experiments and results for agent-based modelling and simulations. In Section 6.2, we first perform clustering by using only spatio-temporal distance and then we add the socioeconomic and infrastructural distances. For agent-based modelling and simulations in Section 6.3, we further divide our experiments into two sections: Section 5.5.1 using synthetic data, and Section 5.5.2 using real data.

#### 6.1 Data Sources

Our primary data for performing the experiments and analyses are social unrest event data. This data is extracted from newspapers and online articles by our data-sources. We additionally collected socioeconomic and infrastructure data for country of India, to be used in the multi-factorial distance function as explained in Chapter 3. In this section we list our data sources.

### 6.1.1 Events Data

We use the Global Database of Events, Language, and Tone (GDELT) [37] as the primary source for event data for our clustering work. The GDELT data comes from multiple sources, including local, regional, and national newspapers that report on events across the globe. The depth and breadth of the sources used minimize the possibility of publication bias, a concern when working with newspaper data [17]. While GDELT covers events for the whole world, our geographic scope for this paper only involves the country India for the year 2014. Since this was the year Indian Prime Minister Mr. Narendra Modi was elected, we expect a moderate number of events across the country. To simplify the clustering process, multiple events of the same category occurring in the exact same location on the same day, has been characterized as a single event. Aggregating multiple events affects the total and average intensity of any location as it changes the number of events, we therefore directly use the raw data for our agent-based modelling experiments. Furthermore, for the agent-based modelling simulations, due to the large amounts of data available, we only use the GDELT social unrest data between January 2014 and June 2014 for selective states to allow us to better manage our analysis and reviews. The methodologies used for the selection of these states is explained in Section 6.3.2.1.

Tracing events back to 1979, GDELT database utilizes 20 categories to define events including, but not limited to, protests, threats, and uses of unconventional mass violence. For our study, we use only 8 categories of unrest that were aimed at the state. The selected categories and their description is listed in Table 6.1. These categories are based on the Conflict and Mediation Event Observations (CAMEO) Event and Actor Codebook [54].



Table 6.1: Categories of unrest events selected for analysis from GDELT

<b>Event Category</b>	<b>Description</b>
Appeal	This category of unrest consists of different types of appeals that citizens can make regarding needs for certain items. This includes appealing for material cooperation, economic cooperation, military cooperation, and other types of cooperation from the state.
Demand	The public has requested a demand of the government or powers in the state. This can include the demand for economic cooperation, diplomatic cooperation, a policy change, or types of aid.
Threaten	This category is about the public threatening to boycott or even attack the state.
Coerce	These actions/events are about the destruction of items/places in order to get the outcomes that the people are interested in getting.
Protests	The people have engaged in some type of demonstration regarding an issue in which the public sees a problem. These demonstrations can be both violent and non-violent, but target the state/political powers.
Assault	The use of more hostile tactics, including abducting/hijacking, multiple forms of assault, bombings, and assassinations/attempts on ruling parties, by the people.
Fight	The general public has started to use non-violent tactics in order to fight back against the government. One example would be the use of small weapons or the occupation of a territory.
Engage in Unconventional Mass Violence	The country has started to experience mass killings, genocide, or other forms of mass violence.

### 6.1.2 Socioeconomic Data

The most reliable data-source for such variable can be obtained from surveys. We are primarily focusing on the data for India, and the most reliable source of survey data in India is the India census data. The census survey has been conducted in India every ten years beginning in 1871, the most recent census was conducted in 2011 and is the primary source of socio-economic data used in this project. The 2011 census

data was collected by the Office of the Registrar General & Census Commissioner, India (Ministry of Home Affairs, Government of India) [41]. All available extracts of the 2011- census can be downloaded from: [http://www.censusindia.gov.in/2011census/population\\_enumeration.html](http://www.censusindia.gov.in/2011census/population_enumeration.html).

Presently, we use two socioeconomic variables: literacy rate and main worker population for this analysis. The literacy rates are calculated for each district from the 2011 census data for India, as the total population of literate in a district divided by the total number of people that are in the age group of 7 years and above, living in the district. Similarly, the main worker population is calculated for every district as the main worker population in the district divided by the total population of the district for the age group of 7 years and above. The main worker population as described by the Indian census, is the number of total workers who have worked for at least 183 days in the preceding 12 months to the census taking. The socio-economic variables are then normalized using feature scaling. This selection of key drivers of unrest was a result of the analysis done by social scientists as part of SURGE [32] project, based on a significant body of literature examining the long-term socio-demographic and economic causes of unrest, as exemplified by the early work of [22].

### **6.1.3 Infrastructure Data**

The infrastructure data is collected in geospatial vector format that specifies the geographic positioning of any object or location on a map. Since geospatial vector data can be geometric objects of points, lines or areas (polygons) that represent spatial features. This kind of data is downloaded as shape (.shp) files, KML (.kml) files or geoJSON files. We have identified that the datasets from OpenStreetMap (OSM) are more suitable to our needs. OpenStreetMap is a free, editable map of the whole world that is being built by volunteers and users through a wiki-style process, it allows

free access to map images and underlying map data. When collecting data from a wiki-style source, the accuracy of data can never be guaranteed, but the collaborative effort from participants make OSM a good quality database. If any inaccurate data is added either maliciously or accidentally, other users can check it and either correct it or remove it. For the purposes of this thesis, we are using shape files of the OSM data extracted from Geofabrik [45]. Geofabrik has a free download server containing data extracts for all continents that are further divided into countries. We choose and download the data for India from the available extracts. The infrastructure data used in this paper include the point based locations of police stations, post-office or post box, hospitals, schools, colleges and universities in India.

## 6.2 Spatio-temporal Data Clustering

To start the DBSCAN clustering, we must first assign values for *eps* and *minPts* parameters, the optimal values for these parameters is difficult to determine. Here, after inspecting results of several sets of parameters, we used  $eps = 0.05$  and  $minPts = 5$  as the combination usually yielded the largest number of clusters and the largest average cluster size. The parameters used for our clustering experiments are listed in Table 6.2.

### Spatio-Temporal Distance Only

Running ST-KDT-DBSCAN on 29,371 events using only the spatio-temporal attributes identified 24,205 of the events as noise. Only 5,166 events were grouped into 376 clusters, while the other events were considered individual outliers (also known as noise points). It is possible that the majority of events identified are simply random events that either did not escalate or are simply isolated by spatial or temporal distance. Table 6.3 shows a summary of the 3 largest clusters found and Figure

Table 6.2: Spatial and temporal spans of 3 largest clusters found when clustered using spatiotemporal, socioeconomic and infra-structure distances

Function	Parameter	Value	Details
FRNN	frnn_radius (geospatial)	500 (km)	Radius, based on which the FRNN object is created.
	frnn_radius (temporal)	60 (days)	It should be greater than the normalization threshold.
ST-KDT-DBSCAN	<i>eps</i>	0.05	The combination usually yielded the largest number of clusters and the largest average cluster size.
	<i>minPts</i>	5	

6.1 shows their geographic plot.

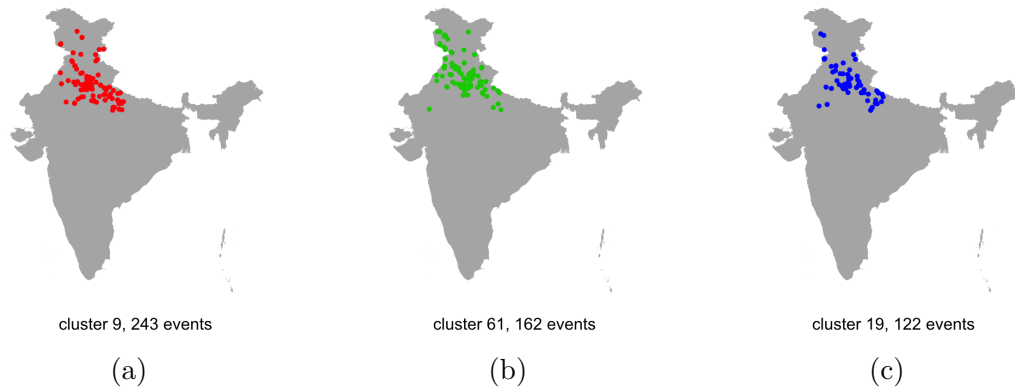


Figure 6.1: Geographic plots of the 3 largest clusters when using spatiotemporal distances only: (a) largest cluster, 243 events, (b) Second-largest cluster, 162 events (c) Third-largest cluster, 122 events

Table 6.3: Spatial and temporal spans of 3 largest clusters found when clustered using spatiotemporal distances only

Cluster ID	Events	Min.(date)	Max.(date)	Min.(km)	Avg.(km)	Max.(km)
9	243	20140930	20141105	0	247.8799	1051.6894
61	162	20140830	20140917	0	288.9725	1160.1203
19	122	20140817	20140828	0	272.228	1057.9207

### Spatio-Temporal + Socioeconomic Distance

Next, we look at the clusters formed by adding socioeconomic attributes to the exist-

ing events dataset. The socioeconomic and spatio-temporal distances are given equal weights (0.5) in the distance function. With the same *eps* and *minPts* used above, we obtained 20,992 noise points and 840 clusters. Notice that the number of noise points have reduced slightly. Since spatio-temporal attributes are region-specific attributes, neighboring events, even in neighboring cities or villages, will have same or similar attribute values. Therefore, events that may not be very close to each other with regards to the temporal values may still fall within the same cluster when considering socioeconomic attributes. A summary of the 3 largest clusters formed during clustering is shown in Table 6.4. A geographic plot of these clusters is presented in Figure 6.2.

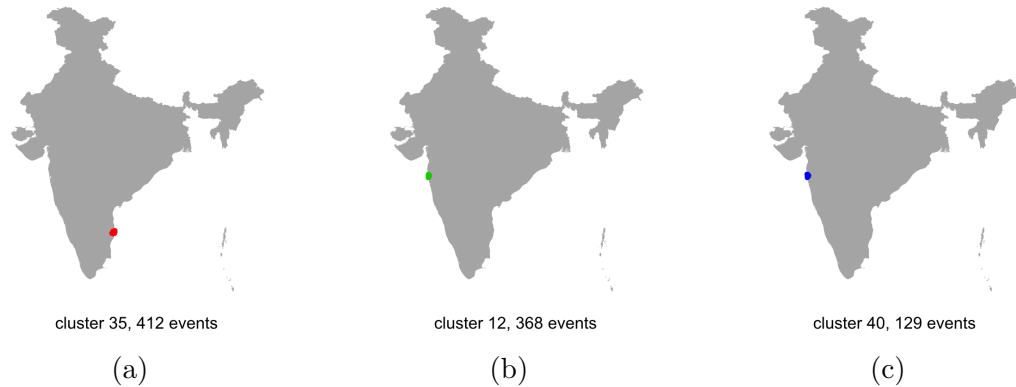


Figure 6.2: Geographic plots of the 3 largest clusters when using spatiotemporal and socioeconomic distances: (a) largest cluster, 412 events, (b) Second-largest cluster, 368 events (c) Third-largest cluster, 129 events.

Table 6.4: Spatial and temporal spans of 3 largest clusters found when clustered using spatio-temporal and socioeconomic distances

Cluster ID	Events	Min(date)	Max(date)	Min.(km)	Avg.(km)	Max.(km)
35	412	20140608	20141228	0	9.2603	51.2563
12	368	20140702	20141231	0	11.3066	36.5704
40	129	20140406	20140615	0	12.4272	38.3016

We can see that compared to Table 6.3, (1) the spatial range of the clusters

has significantly decreased, (2) the temporal range has slightly increased and (3) the cluster sizes have also increased. This is because the spatio-temporal distances have become relatively less important, and events must now also have socioeconomic similarities to be clustered.

### Spatio-Temporal + Socioeconomic + Infrastructural Distance

Next, we cluster the data by also including the `infrastructure_proximity` and `infrastructure_density` attributes as described in Section 3.2.3. Clustering the dataset with the same values of `eps` and `minPts` gives 16,280 noise points and 972 clusters. A summary of the 3 largest clusters is shown in Table 6.5. A geographic plot of these clusters is presented in Figure 6.3.

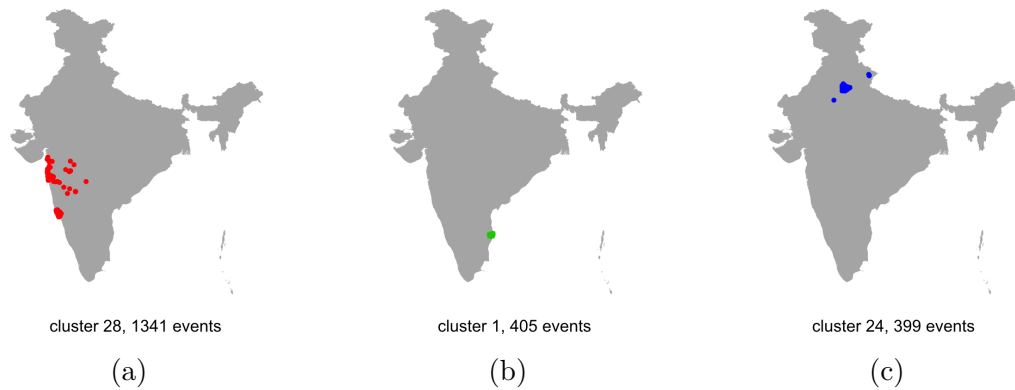


Figure 6.3: Geographic plots of the 3 largest clusters when using spatiotemporal, socioeconomic and infrastructure distances: (a) largest cluster, 1341 events, (b) Second-largest cluster, 405 events (c) Third-largest cluster, 399 events.

Table 6.5: Spatial and temporal spans of 3 largest clusters found when clustered using spatio-temporal, socioeconomic and infra-structure distances

Cluster ID	Events	Min(date)	Max(date)	Min.(km)	Avg.(km)	Max.(km)
28	1,341	20140101	20141231	0	175.558	717.9499
1	405	20140608	20141228	0	9.6117	51.2563
24	399	20140707	20141231	0	42.9988	493.6094

Due to addition of new infrastructure attributes, the distances between events

in general should decrease because similar to spatio-temporal attributes, the infrastructure attributes also depend on the geospatial locations of the infrastructures. Although infrastructures are point-based as opposed to the region-based socioeconomic attributes, event locations that are spatially close to each other tend to have similar infra-structural attributes. Since socioeconomic and infrastructure distances are adding constraints to the clustering process by bringing events in similar environments closer to each other and further separating events in dissimilar environments. We can see that the temporal attribute has become weaker in its role. The largest cluster now contains 1,341 events, spans 717.9 km and is connecting events during the whole year. The second largest cluster is still geographically limited, compared to the other clusters. This implies that with the addition of infrastructural factors, events can be even more connected spanning larger distances and lasting longer periods. This hints at the role of infrastructural factors in facilitating the spread of social unrest events.

### **Efficiency of ST-KDT-DBSCAN**

We begin by empirically comparing the computation times taken to perform the standard DBSCAN and the ST-KDT-DBSCAN on the same dataset using the same distance function. Note that to perform the standard DBSCAN using a customized distance function, we must create a distance matrix. While the clustering itself is quite efficient once the numeric distances are calculated for all pairs of events, in the standard DBSCAN, the creation of distance matrix is inefficient. Figure 6.4 shows the comparison between the two algorithms while performing clustering on spatio-temporal data (events data) along with additional socioeconomic and infrastructure attributes. For this experiment, we created 15 different datasets each containing 100 events more than the previous, starting from 100 to 1,500 events. Both algorithms were run on the datasets with same parameters. We also compare the clustering

results of the two algorithms for data-sizes 1000–1500 in Table 6.6, the clusters with lower data-sizes is exactly the same. We see from Figure 6.4 that the performance of the ST-KDT-DBSCAN is superior to the current method of clustering spatiotemporal data which includes creation of distance matrices for computation of geospatial and time values. The computation graph for the distance matrix does not follow a gradual line in some cases. Although the graph for the ST-KDT-DBSCAN is not exactly linear, the graph appears to be so because there isn't much variation in the computation times as we evenly increase the event count. Table 6.6 shows that there are minor differences between the results of the two algorithms.

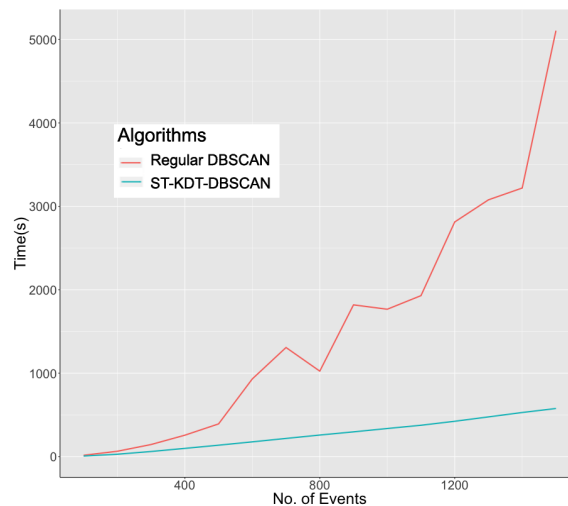


Figure 6.4: Time evaluation of the existing algorithm against ST-KDT-DBSCAN

Table 6.6: Comparison of cluster and noise counts between ST-KDT-DBSCAN and regular DBSCAN

Data size	ST-KDT-DBSCAN		Regular DBSCAN	
	Number of Clusters	noise points	Number of clusters	noise points
1,000	34	319	34	317
1,100	37	338	36	337
1,200	39	366	38	364
1,300	40	399	39	397
1,400	44	427	43	425
1,500	48	446	47	444



### 6.3 ABM Simulation

We will first begin by simulating synthetic social unrest data to demonstrate the design principles of our simulation model. In these experiments, we will use the term *observed* event as the primary agent whose behavior will be monitored as it reacts to other agents in its neighborhood. After demonstrating design principles and several possible behavior of the model, we use the real data to verify if the model is indeed capable of simulating social unrest events. In our experiments, we observe the intensity value of each day for all events alive at that point, we then calculate the expected intensity for the next day based on the alive events. We then compare this expected intensity for the next day with the actual observed intensity from the raw data.

Our prediction model only looks at the events and intensities on the current day to predict the intensities of the future. Our simulations are based on 6 months of unrest data. However, for each prediction of social unrest on a particular day  $d$ , a simulation only looks at the unrest events that occurred within a window of  $w_H$  days. The simulation models are run using different combinations of values for the simulation parameters: recovery rate, influence rate and neighborhood radius. The values used for our experimentation are:

1. Neighborhood Radius: 0.0, 0.2, 0.4, 0.6, 0.8, 0.1
2. Recovery Rate: 0.0, 0.2, 0.4, 0.6, 0.8, 1.0
3. Influence Rate: 0.0, 0.2, 0.4, 0.6, 0.8, 1.0

Using the above six values for each parameter, we run  $6 \times 6 \times 6 = 216$  sets of experiments. For each set of experiments, we run multiple simulations where we add real data for  $w_H$  number of days and predict the intensity for the next day. These

predicted intensities are then compared to the social unrest intensity observed in the real data to calculate error. Based on these error values, we evaluate our simulation models. Each model is further evaluated using two values of  $w_H$ :  $\{1, 7\}$  days. That is, a larger window means that a prediction is made by looking further back at the previous unrest events. Therefore, we run a total of  $2 \times 216 = 432$  simulations. Note that the value of  $t_{lag}$  is 1 day by default. From the experiments, we determine the value of  $w_H$  that gives us more accurate results. We further evaluate our results for two values of  $(t_{lag})$ :  $\{4, 7\}$  days. We evaluate our results to answer the following research questions:

1. For the same simulation parameters, do different states show different prediction performance patterns?
2. Do predictions improve for a larger history window ( $w_H$ )?
3. Are predictions made for more than one day in the future more accurate than predictions made for one day in the future?

We use the prediction error values introduced in Section 5.5.2 to evaluate the performance of our model as we look at the research questions. We set  $\delta_{neutral} = 0.1$ , i.e., the increase predictions occur when the model predicts an increase in overall intensity of a location by at least 10% of the current intensity. Similarly, a decrease of at least 10% is marked as a decrease prediction. Any prediction with a change less than or equal to 10% of current intensity is marked as a neutral prediction. In our evaluations, the total intensity trend error is presented in terms of precision and recall. The precision and recall are calculated using the number of correct and incorrect predictions of different types. We define true positive ( $tp$ ) for a trend-type as the number of times the trend was correctly predicted. A false positive ( $fp$ ) for a trend-type is

defined as the number of times the trend was predicted incorrectly. A false negative ( $fn$ ) of any trend-type is the number of times the selected trend-type is correct but was not predicted by the model. The precision and recall is then calculated as:

$$Precision = \frac{tp}{tp + fp}$$

$$Recall = \frac{tp}{tp + fn}$$

### 6.3.1 Synthetic Data

In this section we create small datasets to test various situations that could occur in the real world. A smaller synthetic dataset allows us to observe the impacts made by neighbors on each other and track the change in intensities of events very clearly. It also makes it easier to present these observations.

#### 6.3.1.1 Agents with no neighbors

We begin by simulating simple agents, we generate an agent with some spatial and temporal parameters. The relative distribution of events and their intensities can be seen in Figure 6.5. Since the neighborhood radius is very small, none of the agents are able to form a neighborhood with other agents. Hence, the intensities can be seen to be gradually decreasing following the Decay Principle.

#### 6.3.1.2 Agents with neighbors

Now we look at some scenarios where the observed agent has neighbors distributed around it at different distances, for the sake of simplicity, we have allowed the neighbors to span on the same day. In this scenario, we must look at multiple sub-scenarios, each demonstrating a different simulation principle.

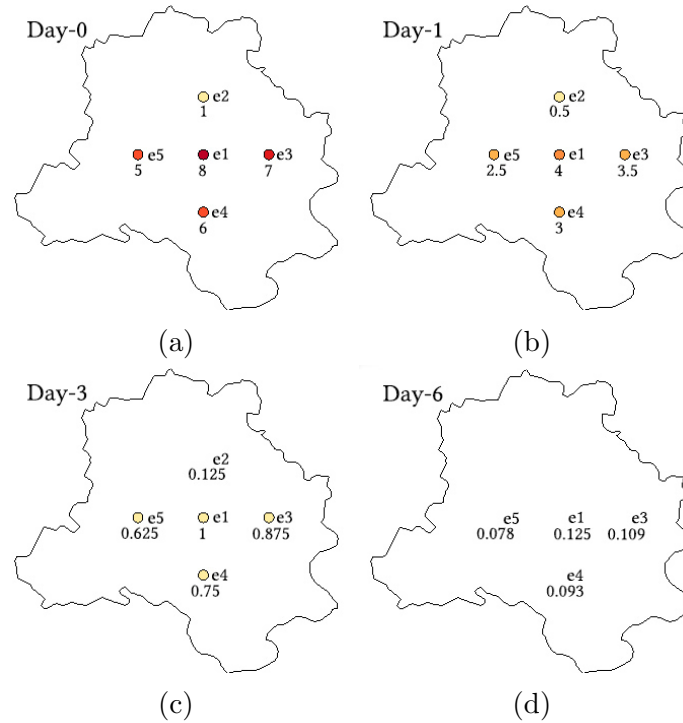


Figure 6.5: Agents with no neighbors, and their corresponding intensities: (a) Day-0: agent e1, e2, e3, e4 and e5 have intensities 8, 1, 7, 6 and 5 respectively. (b) Day-1, all agents have half of their starting intensity. (c) Day-3: agent e2 has died as its intensity reaches 0.125 (d) Day-6: all agents have died

### Neighbors with the same intensity level

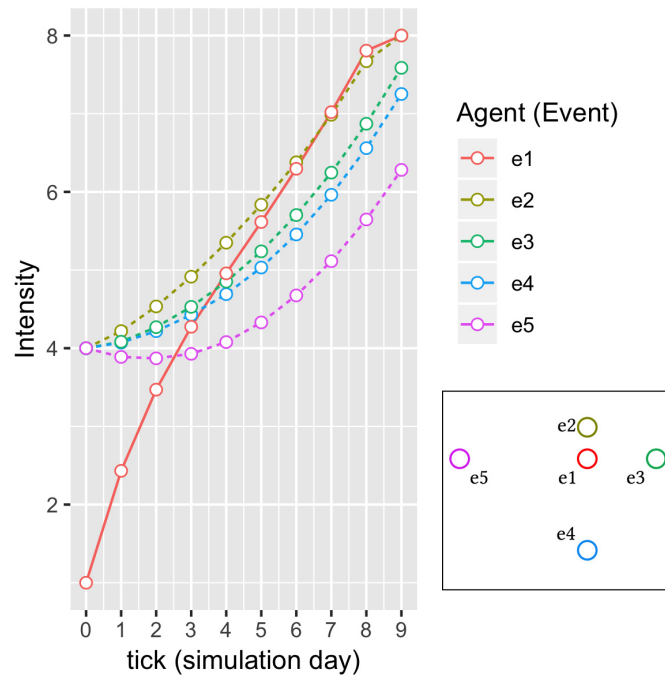
For this test, we create a random agent  $e_1$  with some spatial and temporal parameters. We then create four new agents ( $e_2, e_3, e_4$  and  $e_5$ ) at different spatial distances from  $e_1$ , such that  $d(e_1, e_2) < d(e_1, e_3) < d(e_1, e_4) < d(e_1, e_5)$ , where  $d(e_i, e_j)$  represents the distance between agents  $e_i$  and  $e_j$ . For this test, we only use the spatial distances as all agents begin on the same day, the spatial distance of the agent  $e_1$  from  $e_2, e_3, e_4$  and  $e_5$  are 10, 20, 30 and 40 kilometers (km) respectively.

**Behaviors given a high recovery rate and a low influence rate.** We assign a value of 0.9 to the Recovery Rate ( $\lambda$ ) and a value of 0.1 to the Influence Rate ( $\gamma$ ). We increase the size of the neighborhood radius  $R$ , this allows the agents to form a

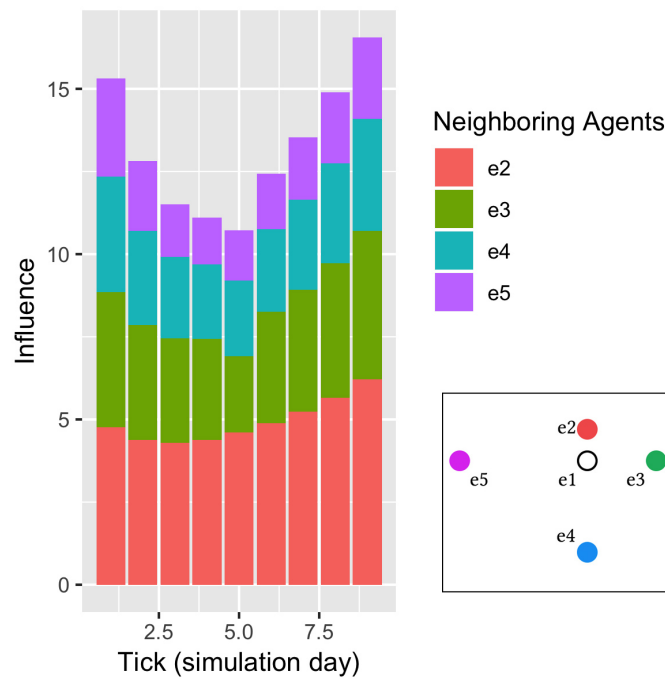
neighborhood among each other. While all the agents affect each other during the simulation, we only examine the influence on  $e_1$  (observed agent). Figure 6.6 shows the behavior of agents and influence on the observed agent during the simulation. From Figure 6.6(a), we can see that the intensity of the observed agent increases gradually on each tick as it is influenced by its neighboring agents. In Figure 6.6(b), we see the amount of influence that each neighboring agents exerts on the observed agent. As expected, the agent  $e_2$  has the highest influence on  $e_1$  as it is the closest to it.

Interestingly, we observe that the influence slightly decreases for a few ticks before growing again. Upon further investigation, we realize that this is because, as the observed agent increases its intensity, the difference between the intensities of the neighborhood and the observed agent becomes less which decreases the influence of the neighborhood on the observed agent. On tick-3, we see that the intensity of  $e_1$  is very close to the intensities of  $e_3$  and  $e_4$ , surpassing these agents on tick-4. After tick-4, the observed agent starts to more significantly influence its neighboring agents. This results in all the agents in the neighborhood influencing each other more and more towards the end of the simulation. This demonstrates that a relatively low-intensity agent could be “nurtured” by surrounding higher-intensity neighboring agents to a point that the low-intensity agent overtakes its neighboring agents in terms of level of intensity and in turn influences those neighboring agents, creating a situation where the agents help each other grow in intensity. Another emergent behavior here is that the relationship among the observed agent and its neighboring agents could reach a low point (e.g., around tick-3 or tick-4 in the above simulation) before rising back up.

**Behaviors given insufficient recovery rates and low initial intensity.** Since, if the Recovery and Influence rates are both too low, the intensities of the observed



(a)



(b)

Figure 6.6: Agent behavior when multiple neighbors with same intensity but at different distances are available. (a) Intensities of agents during simulation, (b) Influence on  $e_1$

agent may die or never attain the maximum intensity level. In Figure 6.7, we simulate the same agents and plot an intensity diagram for a Recovery Rate of 0.8 and keep all other variables same. The simulation has been run for 30 days to illustrate the behavior more clearly.

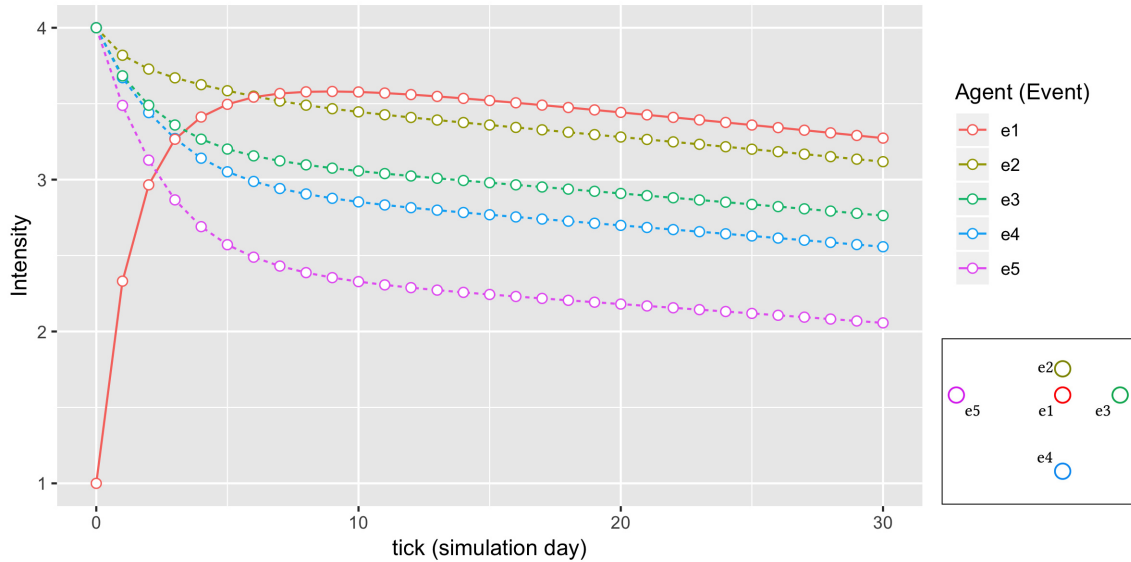


Figure 6.7: Agent Behavior with a Recovery Rate ( $\lambda$ ) of 0.8 and Influence Rate ( $\gamma$ ) of 0.1

We can see that a low value of Intensity Rate with an insufficient value for Recovery Rate results in a lower than initial intensity for the agents. We can observe that the intensity change becomes somewhat steady in this scenario after the gap in intensities of the agents becomes very small. We then expand the simulation and allow it run to the point where the observed agent and all other agents eventually die.

**Behaviors given low recovery rates and low initial intensity but with an injection of a new agent.** If we were to add another agent after the steady state of the simulation is reached, depending on the distance of this added agent to existing agents, the intensity levels of all agents should change. In Figure 6.8, we can see that the intensity levels of all agents increase when a new agent is introduced on day-15

with a starting intensity of 1.0, at the same location as the observed agent. Since this agent is at the center, it can influence all existing agents.

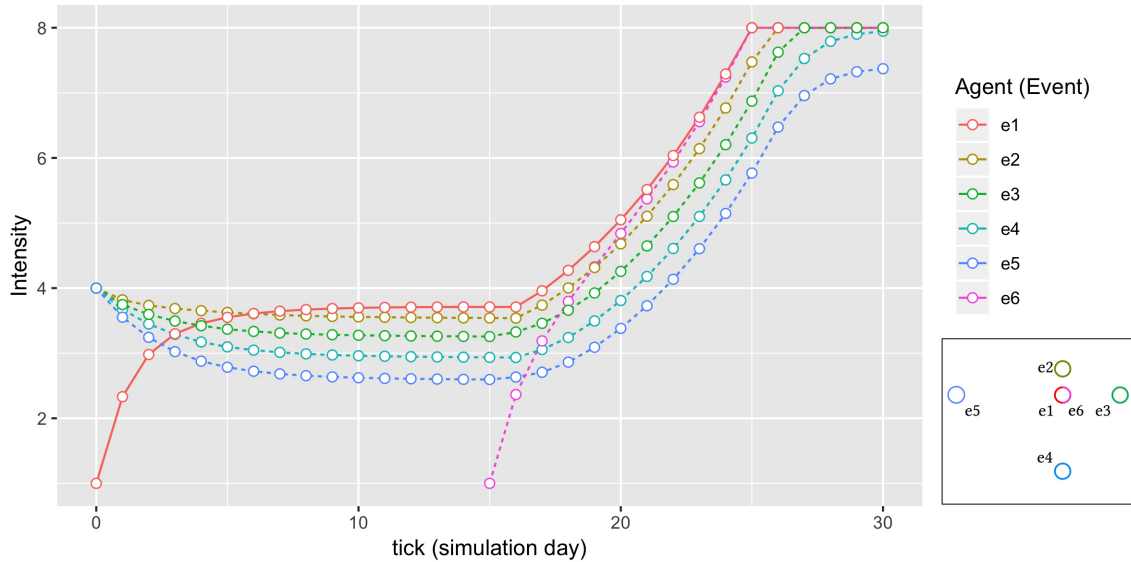


Figure 6.8: Agent Behavior with a Recovery Rate ( $\lambda$ ) of 0.8 and Intensity Rate ( $\gamma$ ) of 0.1. New agent added on day-15 with starting intensity of 1.0.

In summary, Figures 6.7 and 6.8 show the dynamic behavior of our simulation design and displays how different values of Recovery and Intensity Rates can affect the behavior of agents. These tests shown in this section, while mainly focusing on illustrating the Distance Principle, also illustrates the effect of the Intensity and Intensity Difference principles of the simulation model.

### Neighbors with the different intensity levels

For this test, we create a random agent  $e_1$  with some spatial and temporal parameters. We then create eight new agents ( $e_2$  to  $e_9$ ), at the same spatial distance of 50 km from  $e_1$ . The starting intensity of the observed agent  $e_1$  is 1.0 as we expect the central agent to gain intensity through the influence of its neighbors. The eight neighbor agents are each assigned a different starting intensity to represent the eight categories



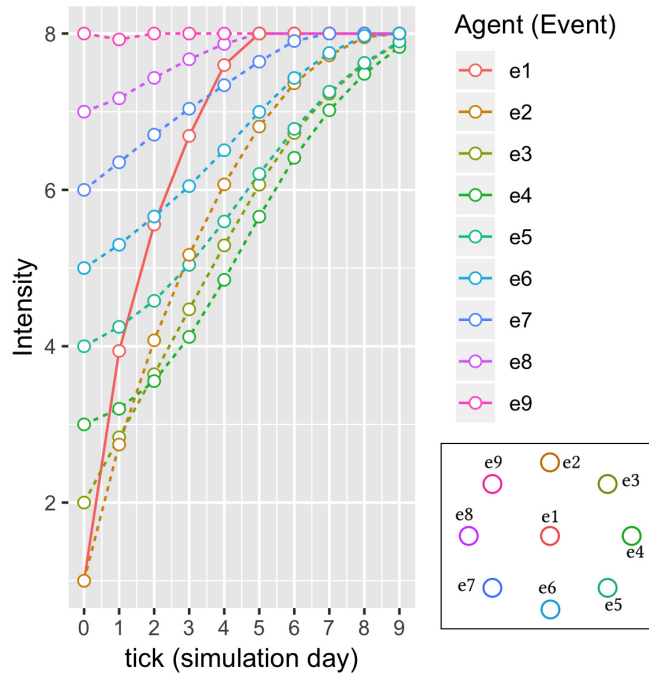
listed in Table 6.1. We look at the intensity plot for this simulation and examine the influence of the neighbors on the observed agent  $e_1$ . For our observation, we choose a Recovery Rate of 0.9 and an Influence Rate of 0.1 as this combination was observed to have the longest number of simulation days required for  $e_1$  to attain maximum intensity. The intensity diagram and neighborhood influence is shown in Figure 6.9.

From Figure 6.9(b), we can see that the largest influence on  $e_1$  is indeed made by the neighbor  $e_9$  which has the largest intensity among all the neighbors. As the simulation progresses, we observe that the intensities of all the neighbor agents rise along with the observed agent. This decreases the intensity gap between the observed agent and the neighbor agents, which results in a lower influence value during the later ticks. However, the neighbors with higher intensities continue to have the most influence on the observed agent.

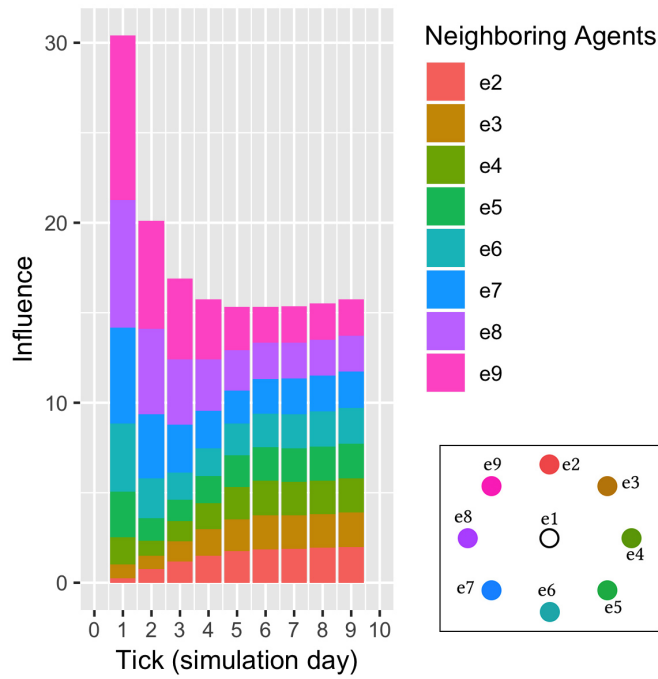
In summary, when the number of neighboring agents is larger the influence experienced by any event is much larger. This is evident from the Figure 6.9 that show the influence contributed by each of the neighboring agents in case of 8 neighbors and 4 neighbors respectively. These observations show that the model behaves as expected under the design principles.

### **Including Temporal Distances**

So far, we have only examined events that begin on the same day, we will therefore examine some events that also use the temporal distance calculations. For this scenario, we only use the spatio-temporal distances by not assigning any weight to the socioeconomic and infrastructural distances. To test this scenario, we create five events  $e_1$ ,  $e_2$ ,  $e_3$ ,  $e_4$  and  $e_5$ , the event  $e_1$  will be treated as the observed agent and will appear on the first day of the simulation. The observed agent is situated at the center and the rest of the four agents are located at a distance of 50 km around the



(a)



(b)

Figure 6.9: Agent behavior when multiple neighbors with different initial intensities at the same distance are available. (a) Intensities of agents during simulation, (b) Influence on  $e_1$

observed agent in the cardinal directions, with a starting intensity of 4.0. Figure 6.10 shows the geographic position of the agents with respect to each other along with their starting intensities, and Table 6.7 shows the start dates of each event.

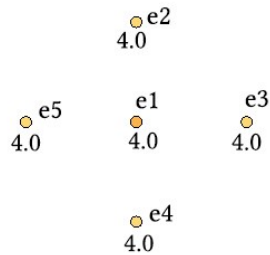


Figure 6.10: Positions and starting intensity of events

Event (Agent)	Start Date
e1	1/1/14
e2	1/3/14
e3	1/6/14
e4	1/10/14
e5	1/15/14

Table 6.7: Event start dates

We now run the simulations with different neighborhood sizes, and different recovery and influence rates. We first check for the neighborhood sizes that result in the observed agent reaching maximum intensity. Since the events start at different times, we have increased the simulation length to 100 days. The simulations show that a neighborhood radius ( $R$ ) of 0.3 or greater results in the observed agent reaching the maximum intensity. A larger neighborhood size means that agents that may be further away are able to affect the observed agent. Figure 6.11 shows an intensity distribution where neighborhood size ( $R$ ) is 0.4, with a recovery rate of 0.9 and a influence rate of 0.1.

We also look at a combination of recovery and influence rate that does not result in the observed agent attaining maximum intensity in Figure 6.12, of a simulation with a recovery rate of 0.9, influence rate of 0.1 and neighborhood radius 0.4. In Figure 6.11, we can see that the agents have gradually decreasing intensity levels as they start out, since they cannot find any neighbor agents that could influence them. As the simulation progresses, the distance between events become smaller since the temporal distance between them decrease due to the overlap between agents. When

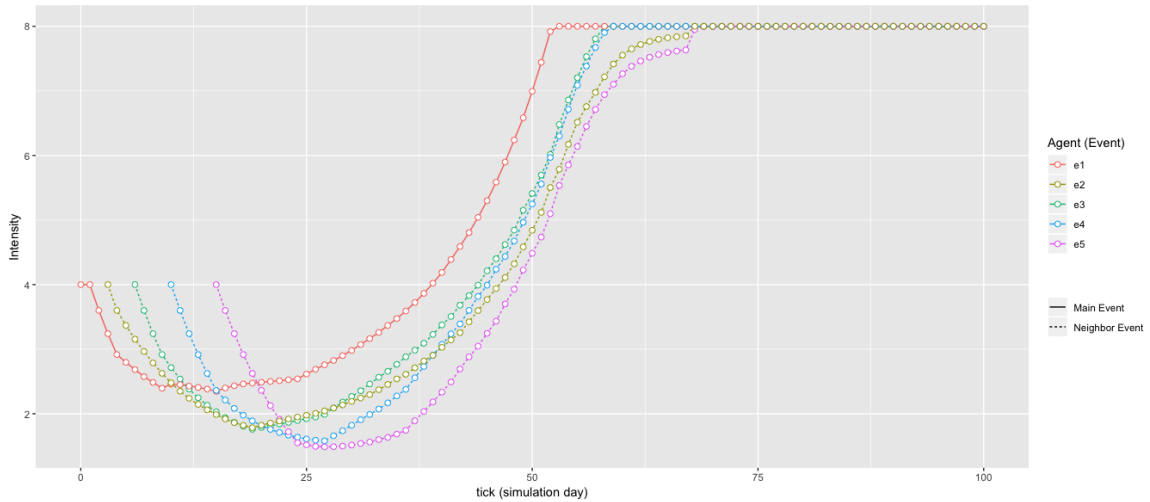


Figure 6.11: Agent Behavior with a Neighborhood Radius ( $R$ ) 0.4, Recovery Rate ( $\lambda$ ) of 0.9, and Influence Rate ( $\gamma$ ) of 0.1, agents with different start dates.

the distance between events  $e_1$  and  $e_2$  becomes small enough that they can form a neighborhood between each other, they start to influence each other, hence increasing their intensities. Similarly, event  $e_3$  follows the same pattern decreasing the recovery rate to 0.8 results in a completely different behavior. While the agents are able to form a neighborhood before dying, which can be observed in Figure 6.12 around tick 10, the influence rate is too small to sustain the intensities of any of the agents. The agent  $e_1$  however survives for a longer time because it keeps getting influenced from the neighboring agents in small amounts.

In conclusion, we can see how the temporal distances add another dimensionality to the distance function and consequently to the simulations. We can quickly notice that the effect of temporal distance is significantly different compared to the effect of spatial distance which is a constant between agents throughout a simulation.

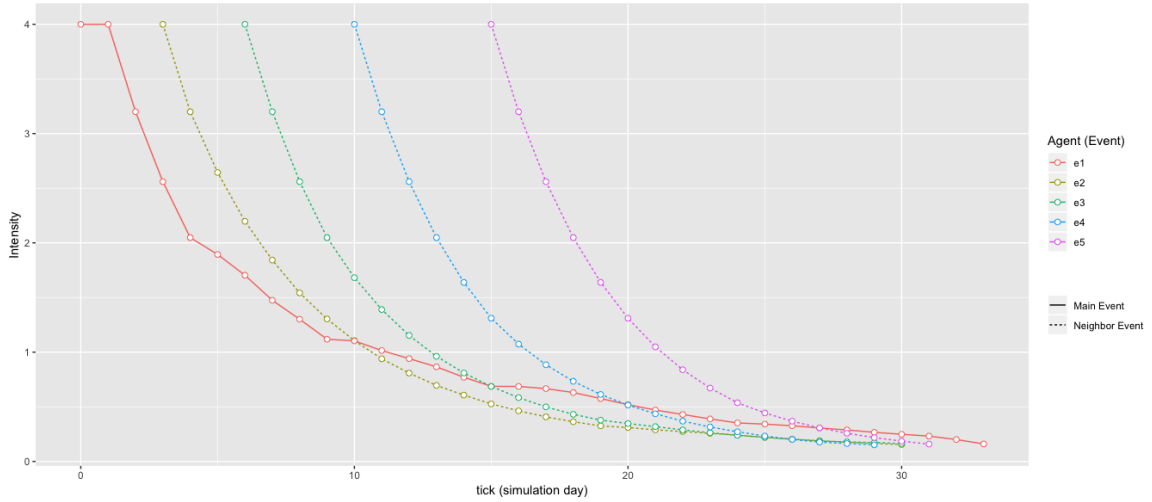


Figure 6.12: Agent Behavior with a Neighborhood Radius ( $R$ ) 0.4, Recovery Rate ( $\lambda$ ) of 0.8, and Influence Rate ( $\gamma$ ) of 0.1, agents with different start dates.

### 6.3.2 Real Data

In this section, we use real-world data downloaded from GDELT. We only use data between January, 2014 and June, 2014 (6 months), the total number of events in India during this period is 155,508. We perform our experiments on three states in India, the selection process is explained in 6.3.2.1. We then evaluate the results of our experiments using the total intensity value errors, average intensity value errors, and trend errors.

#### 6.3.2.1 Location Selection for Analysis

We assume that the optimal values of the simulation parameters recovery rate ( $\lambda$ ), influence rate ( $\gamma$ ) and neighborhood radius ( $R$ ) will change with the location properties, we will therefore not use the data for the entire country for analysis, instead we perform clustering on the events and locations data to classify various geographic states in India into states with suitable characteristics. We intend to find out clusters that have neither the highest nor the lowest number of total events. It is expected

that the states with all the major cities will have the highest number events disproportionately clustered together. We perform unsupervised learning through k-means clustering with a cluster count of 5, to separate out the states with insufficient and irregular data from the ones that contain relatively regular distribution of events and locations. The clustered dataset contains the count of individual events in each state and category, and the number of events in the top 50 locations in each state (top 50 refers to the locations with the largest number of events). The total number of variables for each state therefore is 58 (event count of 8 original categories, and event count at top 50 locations of each state). We attempt different methods to pre-process data before performing the clustering, we have categorized the clustering into the following:

1. Clustering on raw numbers (Category 1).

In this method, we simply cluster the states based on the raw number of events for each of the 58 variables. We expect to find states with similar number of events and identify the outlier states that vary from the rest of the country by a relatively large margin.

2. Data Scaling by total (Category 2).

In this method, we scale the event counts relative to the total event count for each state. The event counts of each category for a state are divided by the total number of events occurring in that particular state. This allows us to put states that have a different number of total events but similar distribution of the events under each category, into the same cluster.

The clustering results are summarized in Table 6.8, the ordering of the rows in the table is based on the Category 1. We can see that the states in cluster number 1 from Category 1 are generally very similar with regards to other clustering categories

as well. Similarly, we can see that the Maharashtra and NCT of Delhi are also very similar to each other due to high event counts, this is because two of the major cities in India are located inside these states. Figure 6.13(a) shows the PCA (Principle Component Analysis) and plots based on the two most significant variables that explain the clustering of Category 1. The figure does not show some eccentric states such as NCT of Delhi (cluster 1) and Maharashtra (cluster 5). Figure 6.13(b) shows the clustering results for Category 2.

We can see that there are overlaps in the clustering plot, since there are multiple parameters affecting the clustering results. We can observe in Table 6.8 that there are multiple states that fall under the same cluster for both category 1 and category 2. We observe that states in cluster number 3 of category 1 clustering has relatively medium size datasets and do not contain any eccentric states. We have therefore chosen the states *Karnataka* and *Andhra Pradesh*. Additionally, we also choose state *Tamil Nadu* for comparison as this state falls in a different cluster in Category 1 and is the closest to the two other states chosen for analysis.

### 6.3.2.2 Model Evaluation using Total Intensity

In this section, we investigate the prediction performance in three states (i.e., Karnataka, Andhra Pradesh, and Tamil Nadu). Here, the history window is 1 day, and the prediction lag is 0 day, which together is the basic configuration of our simulation.

First, we look at the average number of neighbors each event has during the simulation, for different neighborhood radius ( $R$ ), as shown in Table 6.9. In general, as the radius increases, we expect the average number of neighbors to also increase accordingly. Tamil Nadu has the highest number of events inside a neighborhood on average, we therefore, expect we expect a higher rate of increase (7–8%) when compared to the other two states (1–3%). This means that error rates in Tamil

Table 6.8: K-means clustering of event and locations for states in India between 201401 201406

State	Category 1 Cluster Number	Category 2 Cluster Number	Total Events
NCT of Delhi	1	4	32,013
Uttar Pradesh	2	3	16,927
Tamil Nadu	2	2	9,254
Kerala	3	4	3,031
Bihar	3	2	4,986
Punjab	3	2	3,107
Rajasthan	3	3	4,097
Haryana	3	3	3,901
Madhya Pradesh	3	3	3,776
Karnataka	3	2	6,782
West Bengal	3	2	5,346
Jammu & Kashmir	3	3	6,298
Andhra Pradesh	3	3	6,282
Gujarat	3	3	4,896
Lakshadweep	4	1	4
Chhattisgarh	4	2	1,684
Uttarakhand	4	2	1,257
Goa	4	2	804
Himachal Pradesh	4	2	765
Meghalaya	4	2	245
Sikkim	4	2	28
Arunanchal Pradesh	4	3	228
Andaman & Nicobar Island	4	3	142
Chandigarh	4	3	34
Nagaland	4	4	702
Tripura	4	4	488
Manipur	4	4	317
Puducherry	4	4	236
Mizoram	4	4	223
Dadara & Nagar Haveli	4	4	37
Daman & Diu	4	5	1
Odisha	4	2	2,239
Assam	4	3	2,335
Jharkhand	4	3	1,443
Maharashtra	5	2	31,600



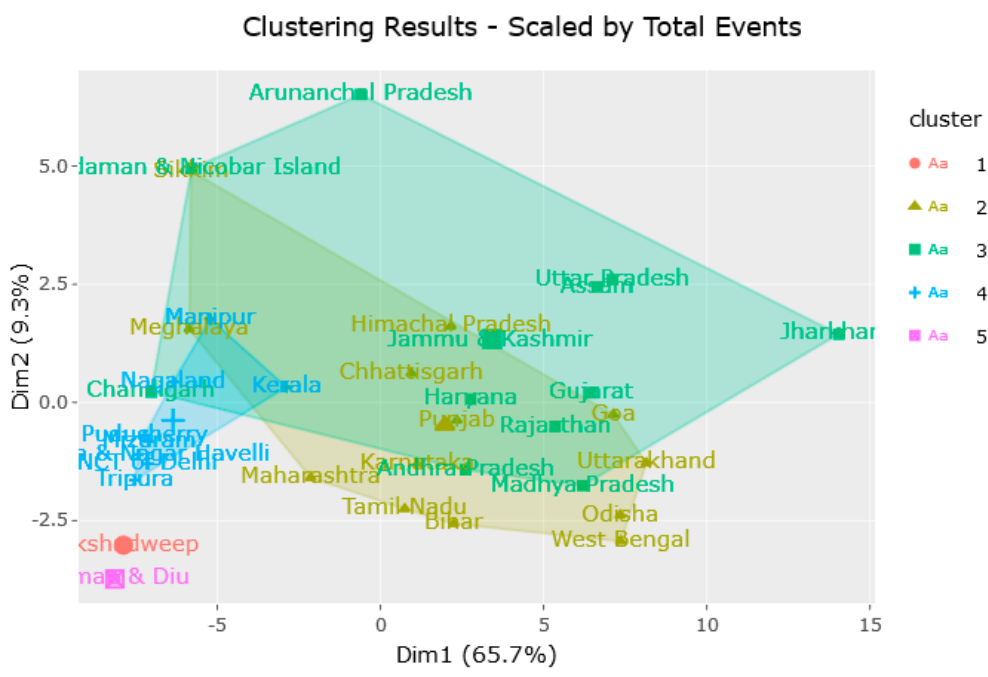
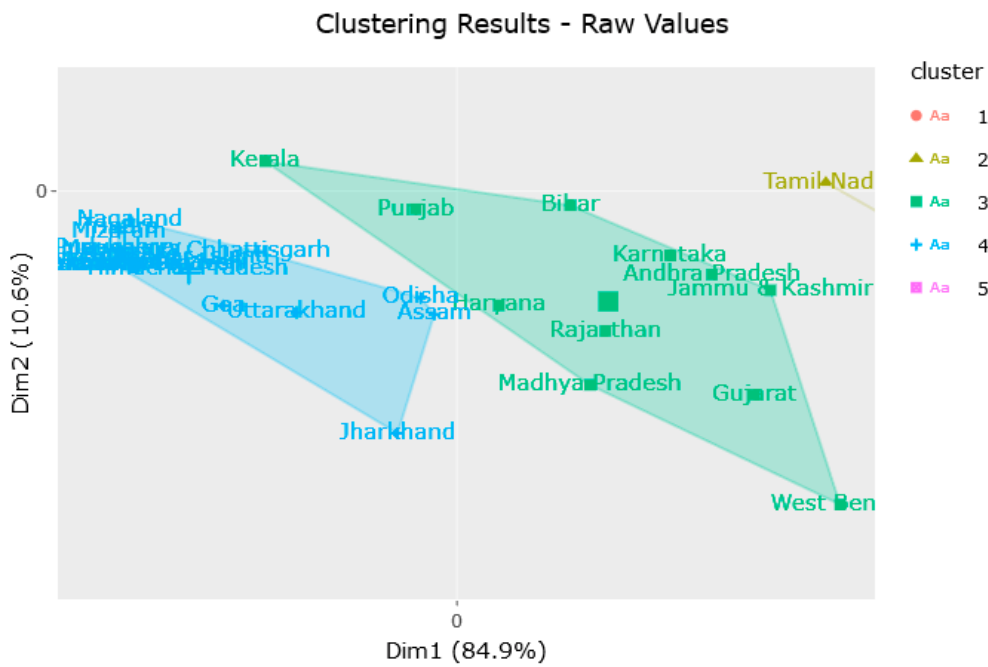


Figure 6.13: K-means clustering plots (201401 - 201406) (a) Category 1 clusters, (b) Category 2 clusters

Nadu are more likely to change as we increase the neighborhood radius, i.e., the neighborhood radius parameter has relatively more impact for the state of Tamil Nadu than for the other two states.

Table 6.9: Average number of neighbors for all events in each state

Neighborhood Radius ( $R$ )	Andhra Pradesh	Karnataka	Tamil Nadu
0.2	20.60	21.32	51.07
0.4	35.71	38.48	52.01
0.6	35.71	38.48	52.01
0.8	35.71	38.48	52.01
1	35.71	38.48	52.01

Now we look at the prediction performance, Figure 6.14 shows the total intensity value errors calculated for various combinations of neighborhood radius, recovery rates and influence rates. Figures 6.15 and 6.16 show the prediction trend errors in terms of precision and recall, respectively. Note that since the number of neighbors does not change significantly as we increase the neighborhood radius ( $R$ ) (as shown in Table 6.9), we only present the total error values for the neighborhood radius of 0.2.

**Total Intensity Value Error.** We can see from the plots in Figure 6.14 that, for any value of influence rate greater than 0, the error rates always increase as we move from a lower influence rate to a higher one, or a lower recovery rate to a larger one. The only exception to this observation is in Karnataka where the error drops slightly as we move from influence rate of 0.0 to 0.2, while the recovery rate is either 0.0 or 0.2. Therefore, the error values follow a similar pattern for Andhra Pradesh and Tamil Nadu, while the patterns in Karnataka is different for influence rate less than or equal to 0.2. This shows that the simulation can indeed behave differently for different locations for the same simulation parameters.

**Intensity Trend Error.** Next, we look at the intensity trend errors. In Figure 6.15

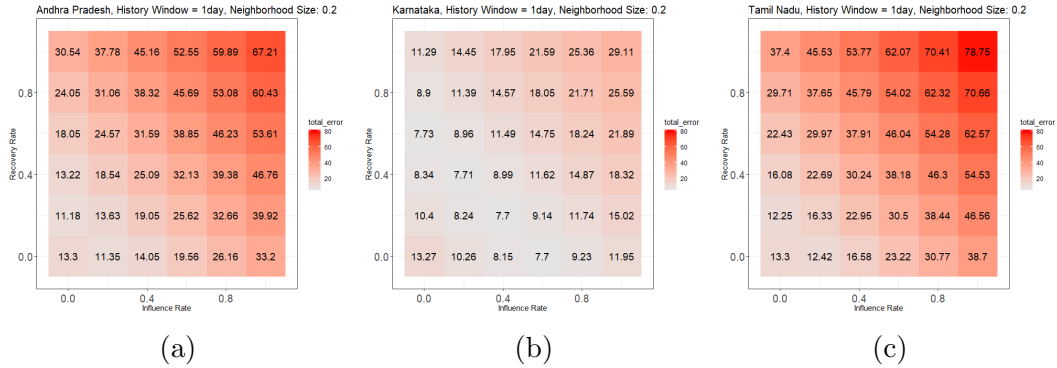


Figure 6.14: Total Intensity Value Error, Neighborhood Radius ( $R$ ) = 0.2. (a) Andhra Pradesh, (b) Karnataka, (c) Tamil Nadu

we observe that the behavior of precision error is different for Karnataka, compared to the other two states. At influence rate of 0.0, all trends are predicted as *decrease*, except when the recovery rate is 1.0, in which case all predictions are *neutral*. This is to be expected, as without any influence from the neighbors, all intensities will go down. Similarly, with a full recovery, the events will stay at the same intensity level throughout the simulation. Hence, all the predictions are *neutral* for a recovery rate of 1.0 when influence rate is 0.0. The recall values, as shown in Figure 6.16, of all the states follow a behavior pattern that is similar to the precision behavior for all the states.

**Summary.** In conclusion, we can say that due to inherent properties of different locations, the optimal simulation parameters are indeed different. Differences in behavior as noted in Karnataka could mean that different states or locations require different sets of simulation parameters to perform optimally. The predictions of our agent-based model, when evaluated based on total intensity error, seem to be quite unrealistic. There are multiple reasons for this observation. First, the data source contains a significant number of duplicate events. The number of duplicate events is not consistent throughout the dataset either, this causes very erratic changes in

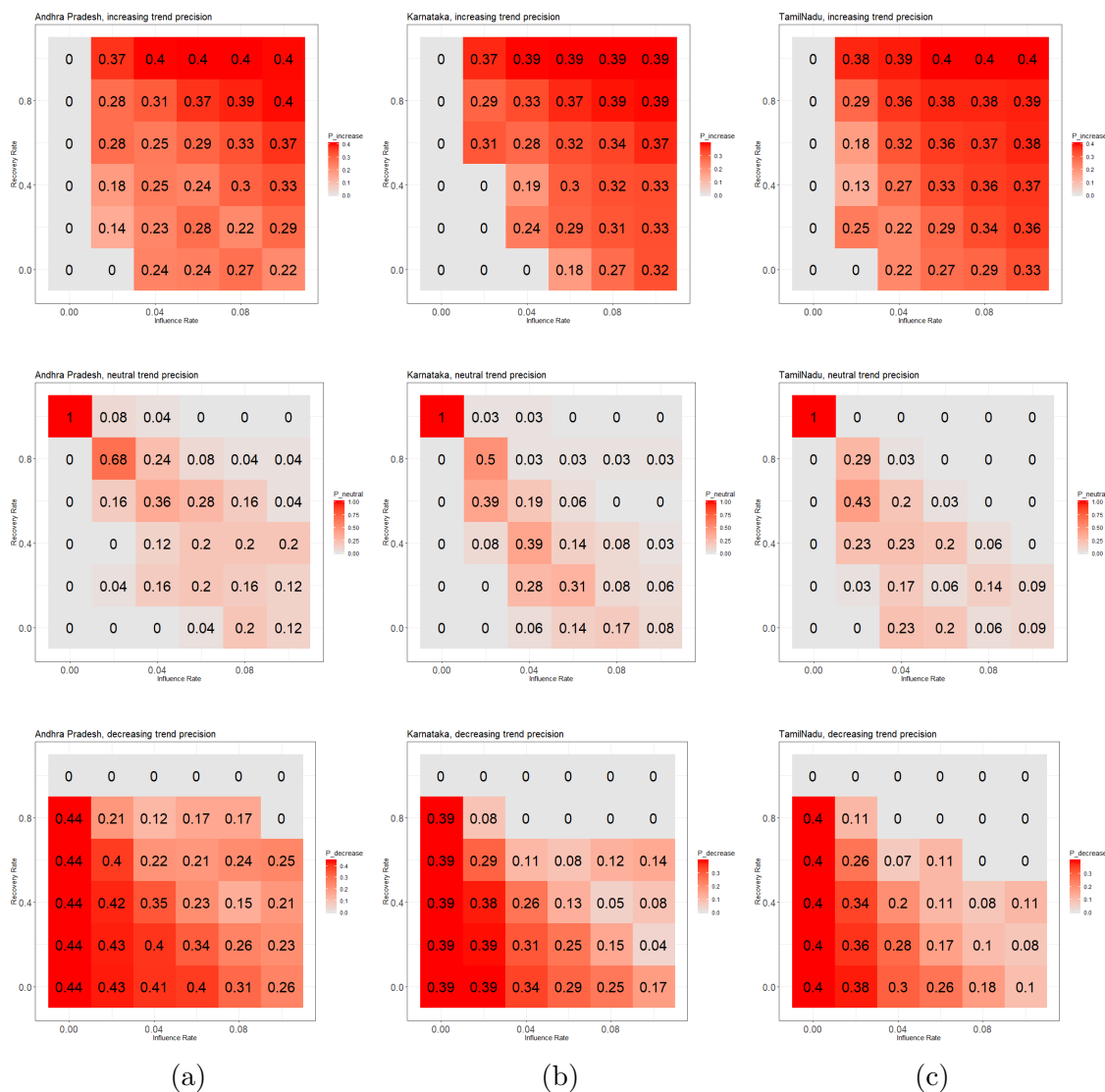


Figure 6.15: Total Intensity Trend - *Precision*, Neighborhood Radius ( $R$ )=0.02 for (a) Andhra Pradesh, (b) Karnataka, (c) Tamil Nadu. Row 1: Increasing Trend; Row 2: Neutral Trend; Row 3: Decreasing Trend

total intensity of the real-world dataset. Such erratic behaviors prohibit formation of any realistic or justifiable pattern of emergence of unrest in the real world. Second, we have seen in the dataset that some articles that do not seem to be about unrest events yet they have been tagged as unrest due to the use of specific keywords. While correctly tagging articles and duplicate detection is an ongoing research problem in

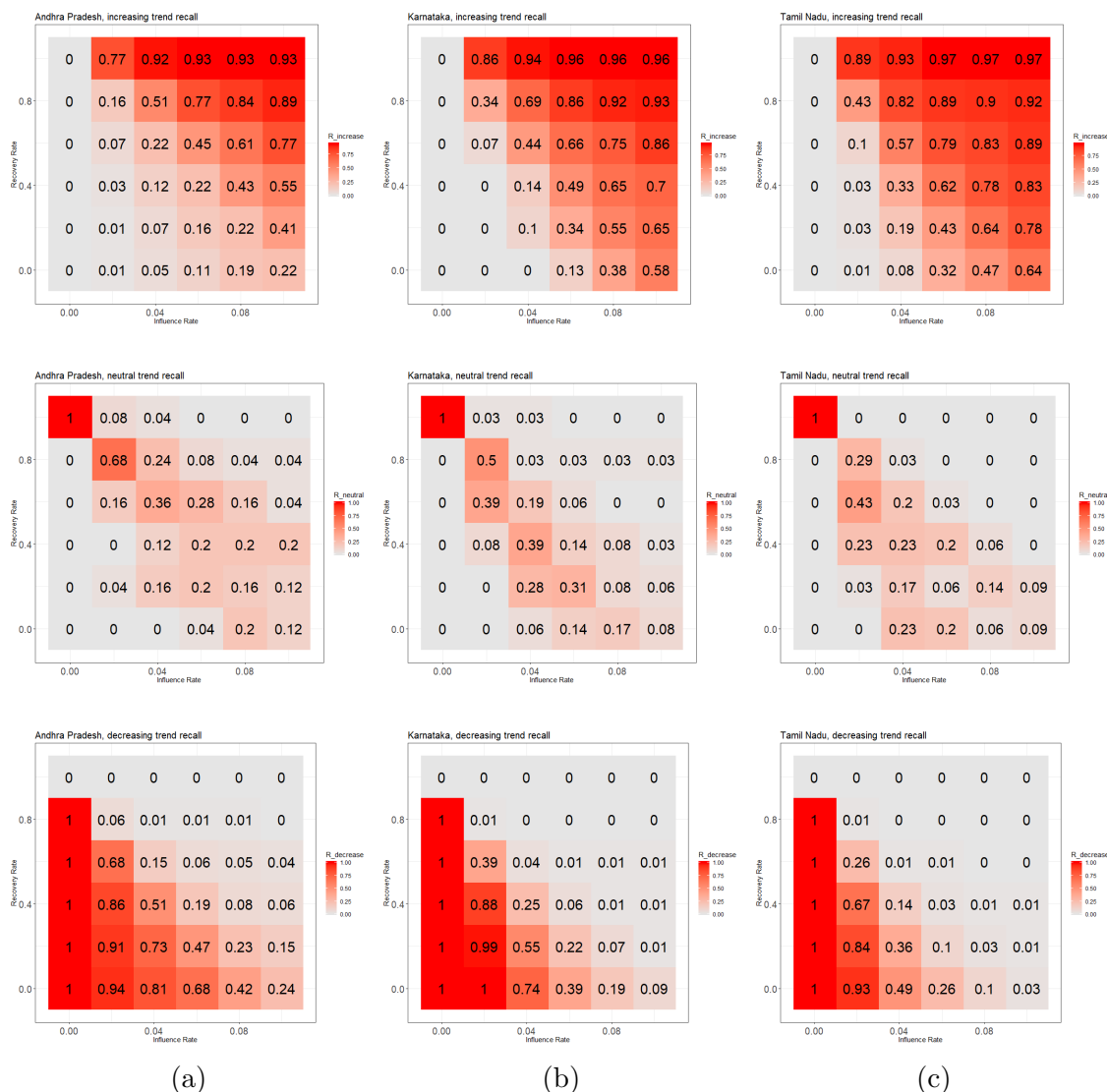


Figure 6.16: Total Intensity Trend - *Recall*, Neighborhood Radius ( $R$ )=0.02 for (a) Andhra Pradesh, (b) Karnataka, (c) Tamil Nadu. Row 1: Increasing Trend; Row 2: Neutral Trend; Row 3: Decreasing Trend

the natural language processing domain, it must be acknowledged that the ability to validate our agent-based modelling approach is affected by the lack of clean data. Third and finally, we also note that there are no negative-intensity events in the dataset. Examples of negative-intensity events are such as peace-keeping events or actions taken by the government (or body of power) to either appease the protesters

or to manage them at any level so as to decrease the harm to property and human lives. Due to the lack of such negative-intensity events, our model cannot be validated for the full cycle of any large unrest movement. While our model contains the recovery rate parameter which allows events in the simulation to decay with time, we have observed that GDELT lists an unrealistic number of unrest events on each day, due to the reasons discussed above, allowing most live events to continuously gain new neighbors and therefore remain alive despite the effects of the recovery-rate.

### 6.3.2.3 Model Evaluation using Average Intensity

Due to the problems with total intensity approach of model evaluation, we now compare (1) the average of the estimated intensities of each event from simulations with (2) the average of the reported intensities of the events in the real world. With this approach, we intend to capture the general state or mood of any region and compare it with the simulated mood generated by our simulation model. While we still see erratic behavior in the real-world data, taking the average of the intensities normalizes the additive effect of duplicate events in the GDELT dataset. In this section, we first present the average intensity value errors observed for different combinations of recovery rate, intensity rate and neighborhood radius, where each of these parameters have values between 0.0–1.0 with a step-size of 0.2. There are a total of 216 combinations of these simulation parameters. We run two sets of simulations with a history window of 1 and 7 days. The total number of simulations is  $216 \times 2 = 432$ . We also show plots of the estimated average intensity and the real average intensity for the three states Karnataka, Andhra Pradesh, and Tamil Nadu, for the simulation parameters with least average intensity value error.

Furthermore, we will use the trends values for the sets of simulations with the least average intensity value error to evaluate the confidence factors. The evaluation will

be done on simulations run with a 1-day history window (i.e.,  $w_H = 1$ ) as well as a 7-day window (i.e.,  $w_H = 7$ ). We also divide the confidence values in different ranges (for example;  $[0-0.1]$ ,  $[0.11-0.2]$ , etc.) and count the number of intensity predictions with confidence factor within each of these ranges. Next, we look at how accurately the confidence factors measure the numbers of increase, decrease and neutral trend predictions within each range. We have assigned  $\delta_{neutral}$  as 0.1. This means that if the predicted intensity for any day  $t$  is within 10% of the real intensity on day  $t - 1$ , the prediction is neutral. Similarly, if the predicted intensity is more than 10% of the real intensity on day  $t - 1$  then the prediction is positive, and if the predicted intensity is less than 10% of the real intensity on day  $t - 1$ , then the prediction is negative. If the percentage of correct predictions is close to the range of confidence factors, then the prediction is considered successful. For example, if there are 40 increase predictions made with confidence values ranging between 0.2 and 0.3, and if out of those 40 predictions, 10 predictions are correct, then the percentage is  $10/40 = 0.25$ , which is in the range of 0.2 and 0.3. In this example, these predictions will be considered to be accurate. This performance evaluation metric is similar to how the accuracy is evaluated in weather forecasting. In the next set of experiments, we add time lags to our predictions.

**Focus on events with 5–10 neighbors.** Additionally, due to the high number of noisy events in the GDELT datasets we perform a set of experiments that only focus on events that have 5 to 10 neighbors within a 7-day history window, to reduce the effects of noise in our analysis. These sets of experiments are different from other experiments and has a different neighborhood radius value of 0.1. The neighborhood radius was chosen empirically, as it resulted in a moderate number of events with 5–10 neighbors. The history window of 7 days was chosen to include temporal distances among events during the neighborhood calculation. The purpose of this set

of experiments is to observe how the model performs with what we assume would be a realistic number of neighbors for any event. We, therefore, try to create a more controlled environment based on real-world observations. To do this, we first ran the simulations with the complete dataset and store the number of neighbors for each event along with the corresponding neighborhood radius used for that simulation. The neighborhood radius used were in the range  $[0.0, 1.0]$  with a step size of 0.1, this was done to give the neighborhood values more range compared to the set of neighborhood radii used for earlier experimentation. We also determined empirically that a neighborhood of radius 0.1 gives us a moderate number of events with 5–10 neighbors. We then ran the simulations with only these selected events and their neighboring events. Next, we set up the real-world data for comparison with our predictions. To create a real-world dataset, we used the same events with 5–10 neighbors as center points. Assuming these events also exist on the next day (day for which predictions are made for), we collected all events that exist in the real-world dataset and are within a radius of 0.1 from the events selected as center points. We used the same distance formula in selection of events for simulation and real-world dataset preparation of this set of experiments, as described in Chapter 3. The experiments (simulations) were run using a neighborhood radius of 0.1 and the same sets of recovery rate and influence rate parameters as used in our previous experiments, i.e., in the range  $[0.0, 1.0]$  with a step size of 0.2. The simulations were run for the three states: Andhra Pradesh, Karnataka and Tamil Nadu. The total number of simulations in this section is  $36 \times 3 = 108$ .

#### **Average Intensity Value Error, History Window = 1 Day.**

In Figure 6.17, we show the average intensity error of each of the three states for a neighborhood radius of 0.2. These predictions are made for 1 day in the future. In



Figure 6.17, we observe that the error is minimum when we are near the diagonal line. These are the simulation parameter combinations where the changes in intensity are minimal, which means the predicted intensity values are similar to the intensities of previous day, we can see this behavior in all three states. In Figure 6.18 we show the average intensity predicted by the simulation compared against the real average intensity for the three states. These plots are generated with the neighborhood radius of 0.2, and the recovery rate and influence rate combination is the one with the least average intensity error value as shown in Figure 6.17.

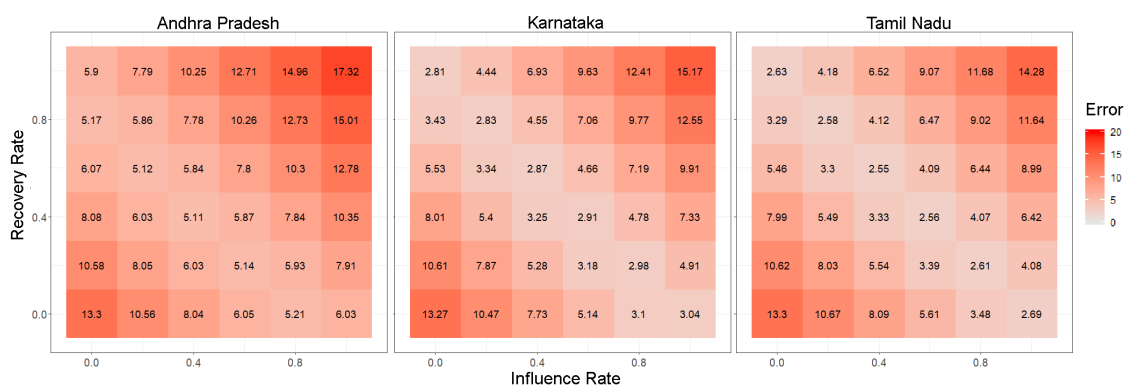
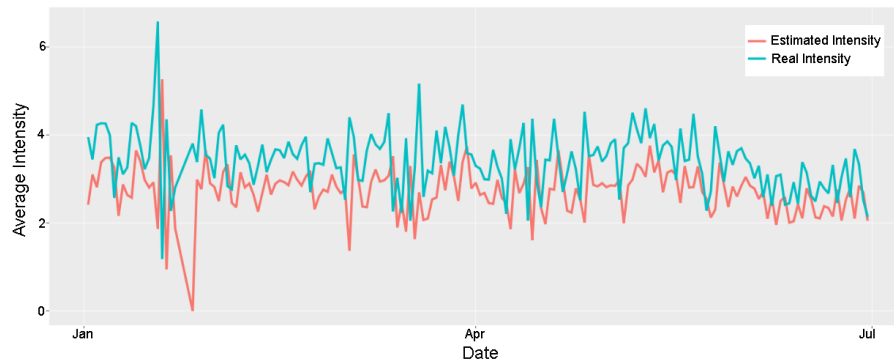


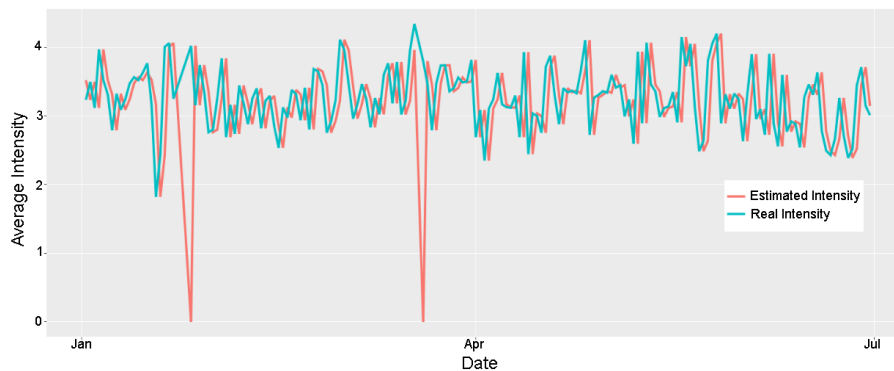
Figure 6.17: Average Intensity Value Error for history window ( $w_H$ ) of 1 day, with Neighborhood Radius ( $R$ ) = 0.2, for states Andhra Pradesh, Karnataka, and Tamil Nadu

In Figures 6.17 and 6.18(b), for Karnataka, we see that the least error occurs when the recovery rate is 1.0 and the influence rate is 0.0. This means the simulation is simply repeating the intensity observed in the past in order to predict the intensity of the next day. The plots for Andhra Pradesh and Karnataka in Figures 6.18(a) and 6.18(b) show similar behavior.

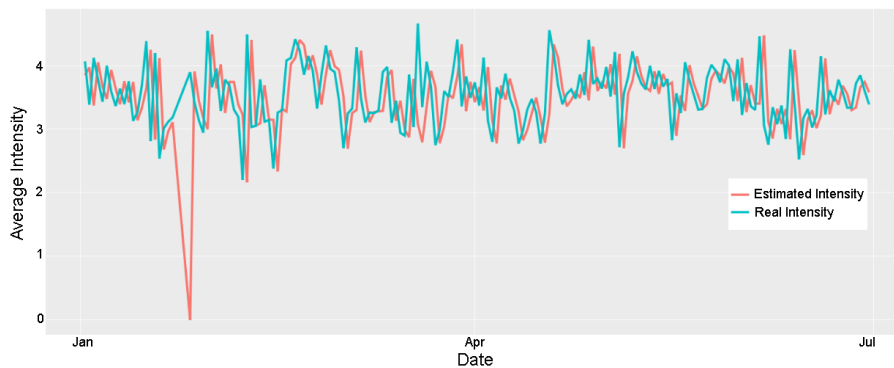
From these observations, we can conclude that given the erratic nature of the real-world data, a history window of one day is not sufficient to generate a realistic prediction. Predictions rely heavily on the number of live events and their intensities in the history window. Since the history window in this case is a single day, the live



(a)



(b)



(c)

Figure 6.18: Comparison of Estimated Average Intensity and Real Average Intensity comparison, history window = 1 day. Estimated Average Intensity for Recovery Rate and Influence Rate with the least error value in Figure 1. (a) Andhra Pradesh, Recovery Rate ( $\lambda$ ) = 0.4, Influence Rate ( $\gamma$ ) = 0.4, (b) Karnataka, Recovery Rate ( $\lambda$ ) = 1.0, Influence Rate ( $\gamma$ ) = 0.0, (c) Tamil Nadu, Recovery Rate ( $\lambda$ ) = 0.6, Influence Rate ( $\gamma$ ) = 0.4

events in the history window do not have enough time to stabilize. Therefore, no meaningful patterns are observed in this set of simulations. Increasing the history window should allow events to interact with other longer resulting in a more stable behavior, such as a relatively gradual ascent or descent in the average intensity, of the location.

### **Average Intensity Value Error, History Window = 7 Days.**

Next, we look at the results from simulations with a history window of 7 days, with predictions made for one day in the future ( $t_{lag} = 1$ ). The average intensity error plots are shown in Figure 6.19; Row 1 for neighborhood radius of 0.2. There is more of a distinction between the average intensity errors with small simulation parameters compared to large simulation parameters. The larger simulation parameters have larger error values compared to the plots with smaller history window. This is because *a larger history window allows more events to survive and therefore the average error values would be much higher* when the history window is 7 days compared to when it is 1 day. However, the least error is still consistently observed around the diagonal of the average intensity error plots. Figure 6.20, 6.21 and 6.22 show the comparison between the real average intensity and the average intensity predicted by the model with the least average intensity error, for Andhra Pradesh, Karnataka and Tamil Nadu, respectively. In Figures 6.20, 6.21 and 6.22, we observe that the changes in the average intensities simulated by the agent-based model is more gradual compared to changes with smaller history window. Due to the gradual change, we can see that *there are some points (days) where the predicted and real average intensity values are close to each other*. Furthermore, the general behavior of the average of the predicted intensities seem to follow the overall behavior of the average intensity in real data. This suggests that *it is possible for the model to predict the future intensities under*

*certain circumstances, given some historical data.*

As expected, the behavior of the predicted/estimated average intensity in the model follows a less erratic behavior compared to the behavior observed in Figure 6.19 (history window of 1 day). While this model does not predict sudden spikes in intensities, *it is able to follow the overall mood of any location.* Due to the 7-day window, our model does not lose or gain a large amount of intensity in an abrupt manner, but instead gradually fluctuates around the middle of the real dataset. Although the predicted trend, i.e., positive, negative, or neutral, does not match the observed in the real-world dataset accurately, there are several cases where the simulated intensities gradually converge to the observed real-world intensity.

#### **Average Intensity Value Error, History Window = 7 Days plus Lag = 4 and 7.**

Note that in the above investigations, the lag value is 1 day: prediction is compared to the next days observed real data. We are interested in finding out whether there is a lag in the prediction, meaning that the prediction might be more accurate if the simulation is predicting for more than one day into the future. Thus, we ran the same simulations with Lag = 4 and Lag = 7, to compare our prediction to the observed real data on the 4<sup>th</sup> day and also on the 7<sup>th</sup> day. Figure 6.19; Row 2 shows the average intensity value errors for all three states with  $t_{lag} = 4$  days, and Figure 6.19; Row 3 shows the same with a  $t_{lag}$  of 7 days. (Note that we only evaluate the lags for simulations with a history window of 7 days because our results, as presented in Figures 6.17 and 6.18, show that the model is severely under-performing when the history window is only 1 day.) In Figure 6.19, it can be observed that there are a few combinations, such as recovery rate = 0.8 and influence rate = 0.2 for Andhra Pradesh, where the error values are slightly lower for  $t_{lag}$  of 4 and 7 days compared

to the error with  $t_{lag}$  of 1 day. The average intensity value errors for the overall simulation, however, seems to be minimum when the prediction is made for a single day into the future. Each of the Figures 6.20, 6.21 and 6.22 show the comparison between the estimated intensity and the real intensity for Andhra Pradesh, Karnataka and Tamil Nadu respectively for the three values of  $t_{lag}$ : 1, 4 and 7 days.

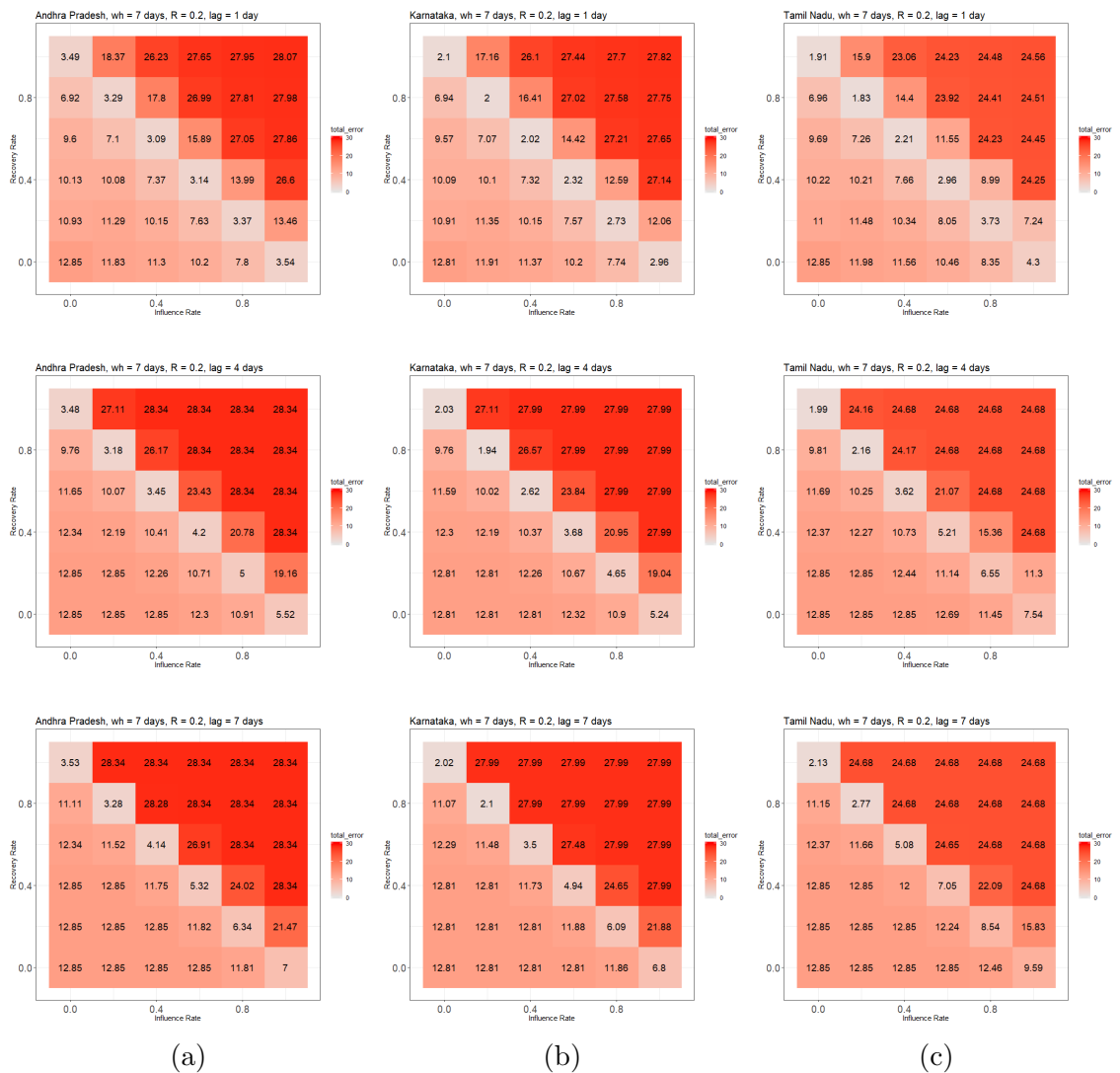
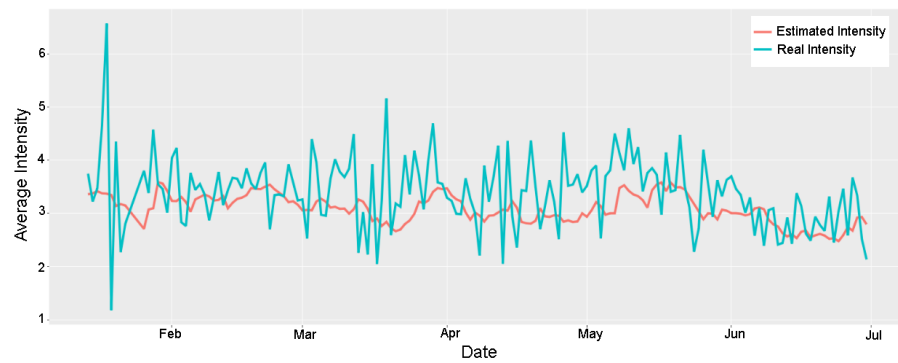
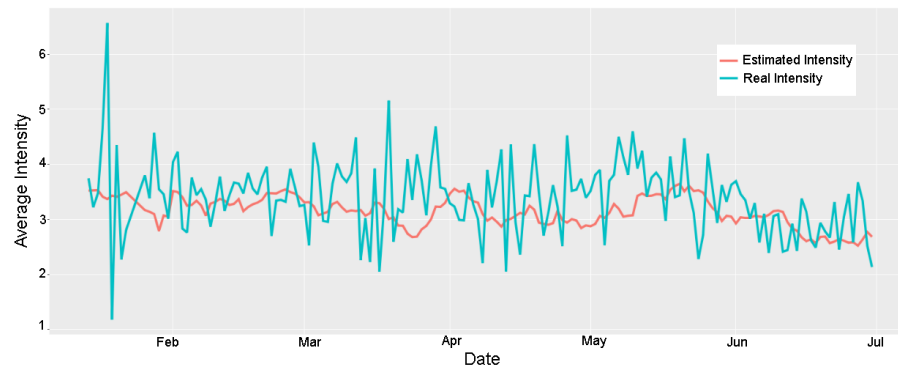


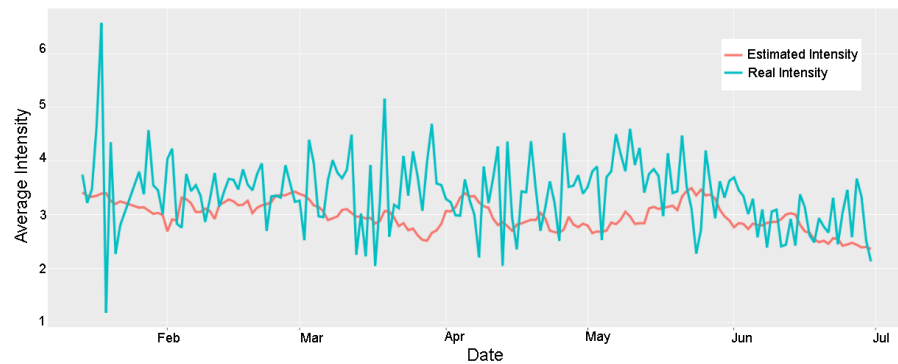
Figure 6.19: Average Intensity Value Error for history window ( $w_H$ ) of 7 days, Neighborhood Radius ( $R$ ) = 0.2: (a) Andhra Pradesh, (b) Karnataka, and (c) Tamil Nadu. Row 1: lag ( $t_{lag}$ ) = 1 day; Row 2: lag ( $t_{lag}$ ) = 4 days; Row 3: lag ( $t_{lag}$ ) = 7 days



(a)

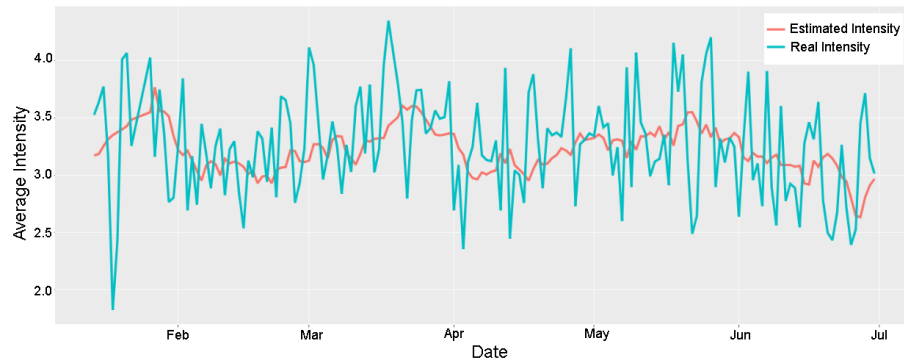


(b)

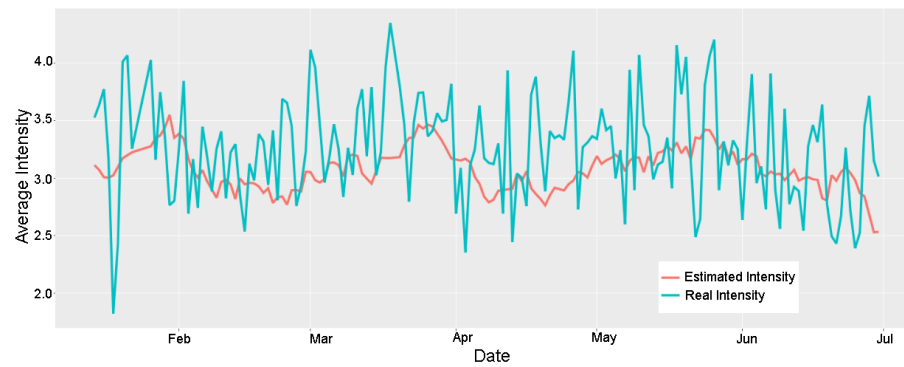


(c)

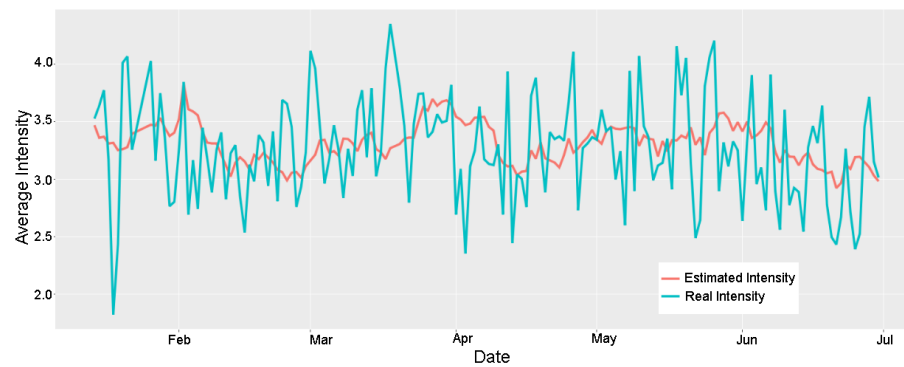
Figure 6.20: Comparison of Estimated Average Intensity and Real Average Intensity comparison, history window ( $w_H$ )= 7 days, in **Andhra Pradesh**. Estimated Average Intensity for Recovery Rate and Influence Rate with the least error value in Figure 6.19; Column a. (a)  $t_{lag}$ = 1 day, Recovery Rate ( $\lambda$ ) = 0.6, Influence Rate ( $\gamma$ ) = 0.4, (b)  $t_{lag}$ = 4 days, Recovery Rate ( $\lambda$ ) = 0.8, Influence Rate ( $\gamma$ ) = 0.2, and (c)  $t_{lag}$ = 7 days, Recovery Rate ( $\lambda$ ) = 0.8, Influence Rate ( $\gamma$ ) = 0.2



(a)

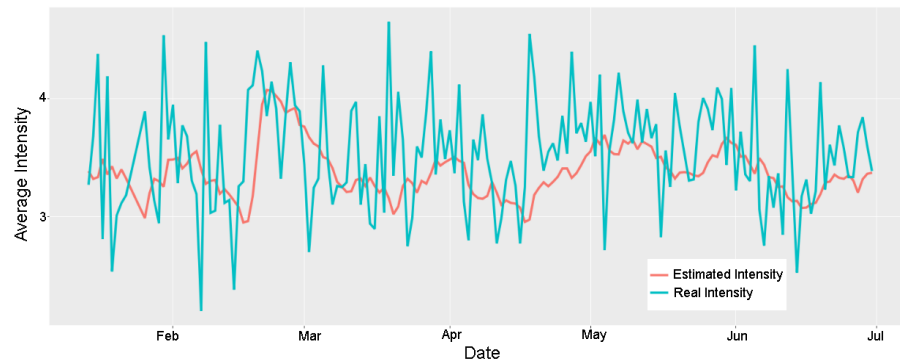


(b)

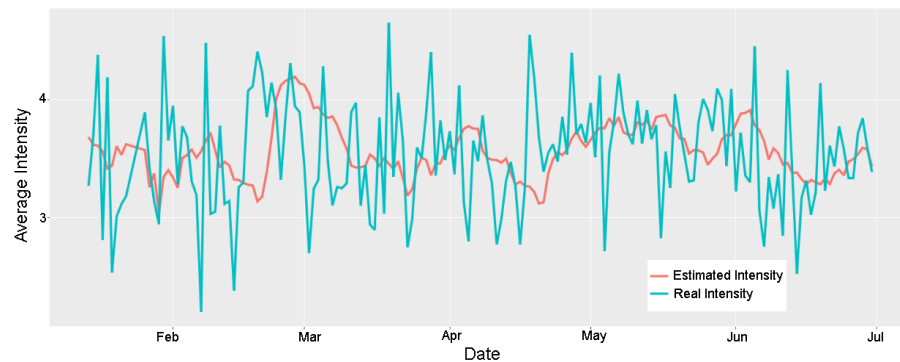


(c)

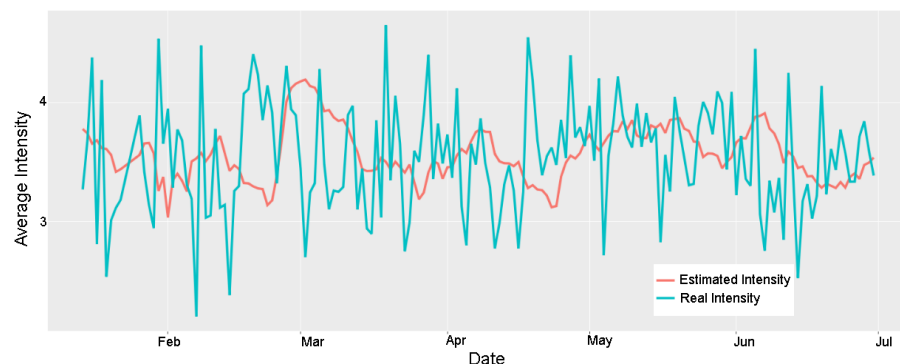
Figure 6.21: Comparison of Estimated Average Intensity and Real Average Intensity comparison, history window ( $w_H$ )= 7 days, in **Karnataka**. Estimated Average Intensity for Recovery Rate ( $\lambda$ ) and Influence Rate ( $\gamma$ ) with the least error value in Figure 6.19; Column b. (a)  $t_{lag}$ = 1 day, Recovery Rate ( $\lambda$ ) = 0.8, Influence Rate ( $\gamma$ ) = 0.2, (b)  $t_{lag}$ = 4 days, Recovery Rate ( $\lambda$ ) = 0.8, Influence Rate ( $\gamma$ ) = 0.2, and (c)  $t_{lag}$ = 7 days, Recovery Rate ( $\lambda$ ) = 1.0, Influence Rate ( $\gamma$ ) = 0.0



(a)



(b)



(c)

Figure 6.22: Comparison of Estimated Average Intensity and Real Average Intensity comparison, history window ( $w_H$ ) = 7 days, in **Tamil Nadu**. Estimated Average Intensity for Recovery Rate ( $\lambda$ ) and Influence Rate with the least error value in Figure 6.19; Column c. (a)  $t_{lag}$  = 1 day, Recovery Rate ( $\lambda$ ) = 0.8, Influence Rate ( $\gamma$ ) = 0.2, (b)  $t_{lag}$  = 4 days, Recovery Rate ( $\lambda$ ) = 1.0, Influence Rate ( $\gamma$ ) = 0.0, and (c)  $t_{lag}$  = 7 days, Recovery Rate ( $\lambda$ ) = 1.0, Influence Rate ( $\gamma$ ) = 0.0



The error did not decrease when lags of 4 and 7 days were added to the simulation, when compared to that with lag of 1. One of the reasons that we see higher error could be that the number of days that any prediction lags by, is not static, and depends on several other variables for which we have not accounted, such as the total number of live events or the spatio-temporal distribution of events inside the location, as well as the variation in the intensity level of events. For example, a small amount of very low intensity events is capable of decreasing the average intensity of a location with otherwise higher average intensity level. Therefore, further analysis of the predictions in reference to these variables could allow us to better understand on how to manage the lags better, and lead to more accurate predictions.

### **Confidence Factor Analysis, History Window of 7 Days.**

In this section, we look at the trends predicted by our model and compare the predictions against the average confidence factor for each day. Since the model with a history window of 1 day under-performs, we only include the results from simulations with a history window of 7 days.

**General Trend Predictions.** Figure 6.23 shows the confidence factors as percentage ranges on the x-axis and the number predictions on the y-axis, for the three states with a lag of 1 day. The y-axis further shows the number of True or False predictions, and the percentage of True predictions for each trend type (*increase, decrease or neutral*). Figures 6.24 and 6.25 show similar plots for a lag of 4 and 7 days, respectively. The figures are only plotted for the combination of recovery rate and influence rate that has the smallest average intensity value error (Figure 6.19). The confidence range in the figures is always larger than 60% because for the selected recovery rate and influence rate, we did not observe any cases with confidence factor within a 0–60% range.

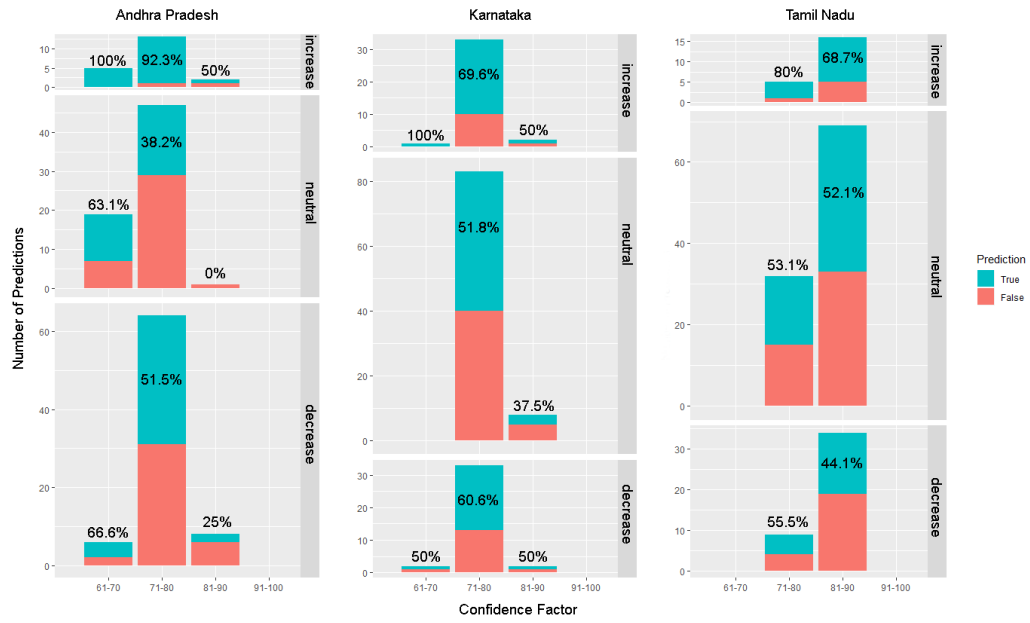


Figure 6.23: Confidence Factors greater than 60% and Percentage of True predictions,  $w_H = 7$  days,  $\text{lag}(t_{lag}) = 1$  day for the states: Andhra Pradesh, Karnataka and Tamil Nadu.

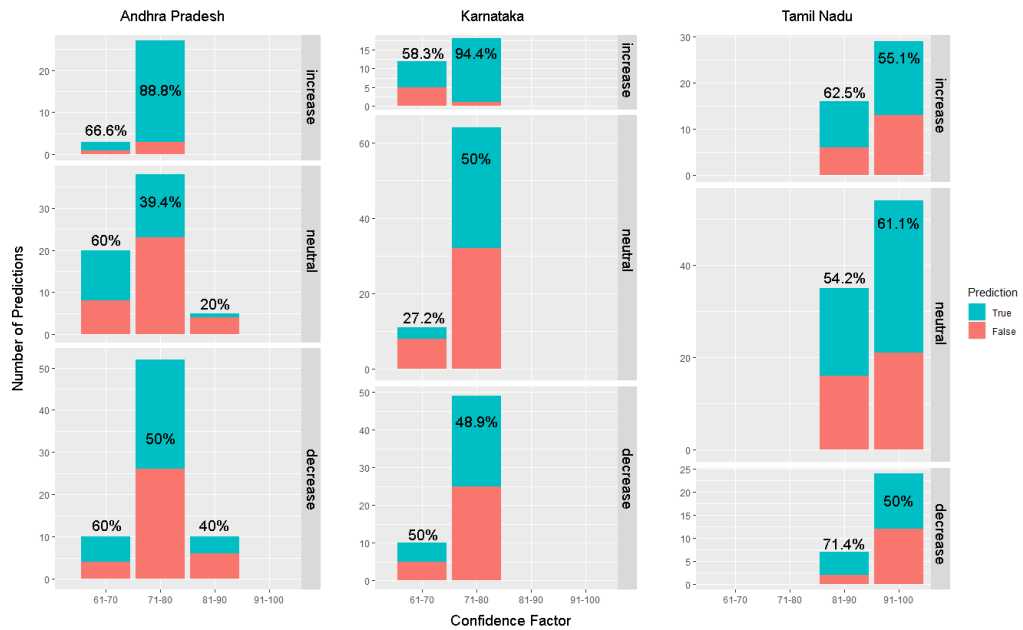


Figure 6.24: Confidence Factors and Percentage of True predictions,  $w_H = 7$  days,  $\text{lag}(t_{lag}) = 4$  days for the states: Andhra Pradesh, Karnataka and Tamil Nadu.



Figure 6.25: Confidence Factors and Percentage of True predictions,  $w_H = 7$  days, lag ( $t_{lag}$ ) = 7 days for the states: Andhra Pradesh, Karnataka and Tamil Nadu.

In Figure 6.23, we see that for all the three states, the number of *increase* predictions is noticeably lower than the number of *neutral* or *decrease* predictions. At the same time, the accuracy of *increase* predictions is fairly high (76% on average). Tables 6.10, 6.11, and 6.12 shows the trend predictions in more detail by showing all combinations of predicted and real, observed trend outcomes, for the selected simulation parameters. From Figures 6.24 and 6.25, we see that, for higher lag values of 4 and 7 days, respectively, the number of *increase* predictions rise. These *increase* predictions are in general more accurate than the predictions made for other trend outcomes.

**Confidence Factor Does Not Reflect Accuracy.** Another important observation that we can make from Figures 6.23–6.25, is that the calculated confidence factor does not reflect the accuracy of most predictions. All the predictions have confidence values greater than 60% while the accuracy (true trend-predictions) vary substantially.

Table 6.10: Comparison of the predicted trend and real-world trends by number of predictions in **Andhra Pradesh**. (a)  $t_{lag}=1$  day, Recovery Rate ( $\lambda$ ) = 0.6, Influence Rate ( $\gamma$ ) = 0.4, (b)  $t_{lag}=4$  days, Recovery Rate ( $\lambda$ ) = 0.8, Influence Rate ( $\gamma$ ) = 0.2, (c)  $t_{lag}=7$  days, Recovery Rate ( $\lambda$ ) = 0.8, Influence Rate ( $\gamma$ ) = 0.2

$pred_{trend,average,t}$ \ $real_{trend,average,t}$	<i>increase</i>	<i>neutral</i>	<i>decrease</i>
<i>increase</i>	18	1	1
<i>neutral</i>	24	30	13
<i>decrease</i>	12	27	39

(a)

$pred_{trend,average,t}$ \ $real_{trend,average,t}$	<i>increase</i>	<i>neutral</i>	<i>decrease</i>
<i>increase</i>	26	3	1
<i>neutral</i>	19	28	16
<i>decrease</i>	9	27	36

(b)

$pred_{trend,average,t}$ \ $real_{trend,average,t}$	<i>increase</i>	<i>neutral</i>	<i>decrease</i>
<i>increase</i>	19	3	0
<i>neutral</i>	21	16	13
<i>decrease</i>	14	39	40

(c)

Recall from Chapter 5 that the confidence factor for an individual event is derived by calculating the variation in intensity of each neighbor event and the distribution of events inside the neighborhood, such that the higher the neighborhood density, the more confident a prediction is, and the higher the variation, the lower the confidence. The overall confidence of a location is then derived as the average of the confidence of all the events present inside the location. The higher confidence of the simulations with lag of 7 days can be explained by a lower variation in the intensities of neighbors of each event. For a larger lag of 7 days, the events are allowed to co-exist longer in the simulation, the events with small intensities will die in the early days and allow the events with larger intensities to influence each other into increasing their

Table 6.11: Comparison of the predicted trend and real-world trends by number of predictions in **Karnataka**. (a)  $t_{lag}= 1$  day, Recovery Rate ( $\lambda$ ) = 0.8, Influence Rate ( $\gamma$ ) = 0.2, (b)  $t_{lag}= 4$  days, Recovery Rate ( $\lambda$ ) = 0.8, Influence Rate ( $\gamma$ ) = 0.2, (c)  $t_{lag}= 7$  days, Recovery Rate ( $\lambda$ ) = 1.0, Influence Rate ( $\gamma$ ) = 0.0

$pred_{trend,average,t}$ \ $real_{trend,average,t}$	<i>increase</i>	<i>neutral</i>	<i>decrease</i>
<i>increase</i>	25	9	2
<i>neutral</i>	18	46	27
<i>decrease</i>	1	14	22

(a)

$pred_{trend,average,t}$ \ $real_{trend,average,t}$	<i>increase</i>	<i>neutral</i>	<i>decrease</i>
<i>increase</i>	24	6	0
<i>neutral</i>	18	35	22
<i>decrease</i>	2	28	29

(b)

$pred_{trend,average,t}$ \ $real_{trend,average,t}$	<i>increase</i>	<i>neutral</i>	<i>decrease</i>
<i>increase</i>	30	9	7
<i>neutral</i>	13	49	24
<i>decrease</i>	1	11	20

(c)

intensities and therefore stabilizing at a higher than average and similar intensity level. This results in less variation of neighbors intensity and hence the higher overall confidence. The stabilization of events at high intensities also explains the rise in increase predictions as we move from a lag of 1 day to a lag of 7 days.

Table 6.12: Comparison of the predicted trend and real-world trends by number of predictions in **Tamil Nadu**. (a)  $t_{lag} = 1$  day, Recovery Rate ( $\lambda$ ) = 0.8, Influence Rate ( $\gamma$ ) = 0.2, (b)  $t_{lag} = 4$  days, Recovery Rate ( $\lambda$ ) = 1.0, Influence Rate ( $\gamma$ ) = 0.0, (c)  $t_{lag} = 7$  days, Recovery Rate ( $\lambda$ ) = 1.0, Influence Rate ( $\gamma$ ) = 0.0

$pred_{trend,average,t}$ \ $real_{trend,average,t}$	increase	neutral	decrease
increase	30	18	1
neutral	9	46	14
decrease	2	27	22

(a)

$pred_{trend,average,t}$ \ $real_{trend,average,t}$	increase	neutral	decrease
increase	31	13	0
neutral	14	43	12
decrease	6	23	22

(b)

$pred_{trend,average,t}$ \ $real_{trend,average,t}$	increase	neutral	decrease
increase	28	17	5
neutral	19	44	15
decrease	1	18	18

(c)

**Least Trend Errors.** Next, we evaluate the simulations with the least trend errors instead of least average intensity value errors by examining new intensity diagrams and percentage of current predictions by confidence factors for the three states. *The intensity diagrams follow the same patterns as the diagrams plotted for average intensity value errors in Figures 6.20(a), 6.21(a) and 6.22(a), i.e., the estimated average intensities fluctuate between the higher and lower extremes of the real-world average intensities. The confidence factors and percentage of true predictions also show the same type of relationship as seen from evaluating the confidence factors of the simulations with least average intensity value errors, i.e., all calculated confidence factors are above 60% and there is no clear relationship between the percentage of correct*

trend predictions and the confidence factors. The recovery rate and influence rate with the least trend error, however, is 1.0 and 0.0 respectively for all three states, and for all three lags. This indicates that the highest number of over-all correct trend predictions were made when the intensities of the events remained constant throughout the 7 days history period. Since the predicted intensity for any day is the average of the intensities of live events on that day (during simulation), the predicted intensity is in this case, becomes the average intensity of all the events that existed during the 7 days history period. Tables 6.13, 6.14 and 6.15 list the comparison of the predicted and real trend in Andhra Pradesh, Karnataka and Tamil Nadu, respectively for different lags.

Table 6.13: Comparison of the predicted trend and real-world trends by number of predictions in **Andhra Pradesh** with Recovery Rate ( $\lambda$ ) = 1.0 and Influence Rate ( $\gamma$ ) = 0.4. (a)  $t_{lag} = 1$  day, (b)  $t_{lag} = 4$  days, (c)  $t_{lag} = 7$  days

$pred_{trend,average,t}$ \ $real_{trend,average,t}$	<i>increase</i>	<i>neutral</i>	<i>decrease</i>
<i>increase</i>	32	8	4
<i>neutral</i>	21	46	24
<i>decrease</i>	1	4	25

(a)

$pred_{trend,average,t}$ \ $real_{trend,average,t}$	<i>increase</i>	<i>neutral</i>	<i>decrease</i>
<i>increase</i>	31	10	8
<i>neutral</i>	19	40	19
<i>decrease</i>	4	8	26

(b)

$pred_{trend,average,t}$ \ $real_{trend,average,t}$	<i>increase</i>	<i>neutral</i>	<i>decrease</i>
<i>increase</i>	32	11	9
<i>neutral</i>	20	36	19
<i>decrease</i>	2	11	25

(c)

Table 6.14: Comparison of the predicted trend and real-world trends by number of predictions in **Karnataka** with Recovery Rate ( $\lambda$ ) = 1.0 and Influence Rate ( $\gamma$ ) = 0.0. (a)  $t_{lag}$ = 1 day, (b)  $t_{lag}$ = 4 days, (c)  $t_{lag}$ = 7 days

$pred_{trend,average,t}$ \ $real_{trend,average,t}$	<i>increase</i>	<i>neutral</i>	<i>decrease</i>
<i>increase</i>	31	11	5
<i>neutral</i>	13	51	27
<i>decrease</i>	0	7	19

(a)

$pred_{trend,average,t}$ \ $real_{trend,average,t}$	<i>increase</i>	<i>neutral</i>	<i>decrease</i>
<i>increase</i>	31	14	6
<i>neutral</i>	13	43	23
<i>decrease</i>	0	12	22

(b)

$pred_{trend,average,t}$ \ $real_{trend,average,t}$	<i>increase</i>	<i>neutral</i>	<i>decrease</i>
<i>increase</i>	30	9	7
<i>neutral</i>	13	49	24
<i>decrease</i>	1	11	20

(c)

We notice that the number of correct decrease predictions is low compared to the predictions based on average intensity value errors, the number of correct increase and neutral predictions however are higher. Since the recovery rate is always 1.0 and influence rate is 0.0, all the events will retain their intensity and it is therefore more likely that the prediction made is *increase* or *neutral*.

In conclusion, we can say that the *computing meaningful confidence factors require more than just the density of neighborhood and variance in the intensities of neighbors*. Since the confidence factors shown in the plots are calculated exactly a day before the predictions, it is possible that the history window allows all events time to adapt to each others intensities such that the variation in intensities of all neighborhoods



Table 6.15: Comparison of the predicted trend and real-world trends by number of predictions in **Tamil Nadu** with Recovery Rate ( $\lambda$ ) = 1.0 and Influence Rate ( $\gamma$ ) = 0.0. (a)  $t_{lag}$ = 1 day, (b)  $t_{lag}$ = 4 days, (c)  $t_{lag}$ = 7 days

$pred_{trend,average,t}$ \ $real_{trend,average,t}$	<i>increase</i>	<i>neutral</i>	<i>decrease</i>
<i>increase</i>	23	10	4
<i>neutral</i>	25	63	19
<i>decrease</i>	0	6	15

(a)

$pred_{trend,average,t}$ \ $real_{trend,average,t}$	<i>increase</i>	<i>neutral</i>	<i>decrease</i>
<i>increase</i>	26	14	5
<i>neutral</i>	21	52	16
<i>decrease</i>	1	13	17

(b)

$pred_{trend,average,t}$ \ $real_{trend,average,t}$	<i>increase</i>	<i>neutral</i>	<i>decrease</i>
<i>increase</i>	28	17	5
<i>neutral</i>	19	44	15
<i>decrease</i>	1	18	18

(c)

become similar. From the comparison tables of average trend value error (Tables 6.10, 6.11 and 6.12) and the comparison tables for trend errors (Tables 6.13, 6.14 and 6.15), we can conclude that the simulation parameters with the least trend errors and that with the least average intensity value errors are not necessarily the same. *Additionally, if we are more concerned with the positive trend predictions and not interested in the negative trends, we could simply calculate the average of the intensities from all the events in the last week or so.* This could give a reasonable estimation of the *increase* or *neutral* trend in the coming days for the selected three states.

### Focused Locations (selected events with 5–10 neighbors)

**Average Intensity Value Errors.** After selecting events with 5–10 neighbors, we ended up with 902 (out of 6,230) events in Andhra Pradesh, 1,376 (out of 7,231) events in Karnataka and 1,177 (out of 9,727) events in Tamil Nadu. Comparing the results in Figure 6.26 with Figure 6.19; Row1, we can see that the error increases when we use the focused set of events.

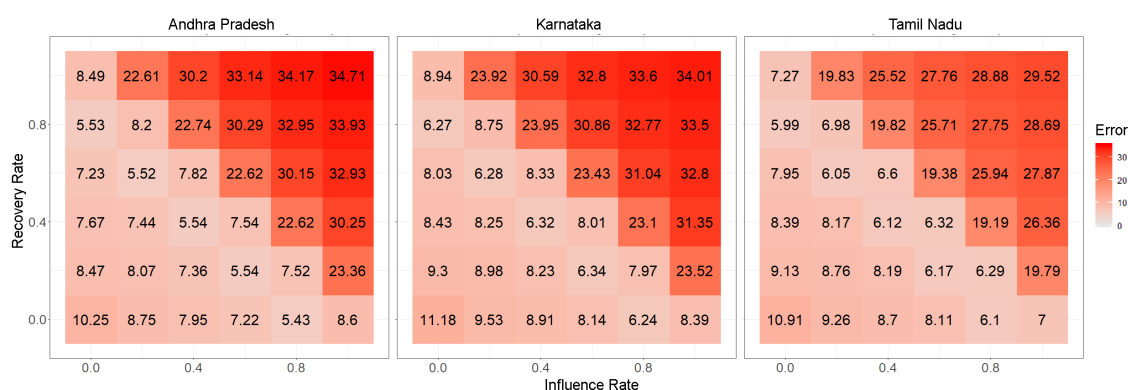


Figure 6.26: Average Intensity Value Error of focused events with 5-10 neighbors, history window ( $w_H$ ) = 7 days, Neighborhood Radius ( $R$ ) = 0.1, lag ( $t_{lag}$ ) = 1 day, for states: Andhra Pradesh, Karnataka, and Tamil Nadu.

From Figure 6.27, it can be seen that the average intensity values in the focused real-world data is more erratic than the average intensity values when complete data is used. In Andhra Pradesh, the standard deviation of the average intensity values per day in the complete dataset is 0.67, whereas the standard deviation of the average intensity values of the focused real-world data is 1.42. Similarly, the standard deviations of the average intensity per day of Karnataka and Andhra Pradesh are larger in the focused dataset. In Figure 6.27, the overall predicted average intensities of the simulations with least error are observed to be lower than the real-world average intensities in majority of the cases. Due to the erratic nature of the focused real-world data, we see that there is a significant disconnect between the simulated and

the real-world intensity values. Since the real-world dataset is now much smaller, the events with very low or very high intensities become more important and could more easily change the average intensity of any location. Due to this, the real-world data seems much more erratic compared to the complete dataset. This implies that the methodology of collecting real world data by treating selective events as center points may not be effective. We then look at the trend predictions for the set of simulations with the least average intensity value errors. The trend prediction compared with the real trends for the simulations with the least average intensity value errors is shown in Table 6.16. The trend predictions in Table 6.16 have a large number of incorrect decrease predictions and almost no prediction for neutral trend.

**Least Trend Errors.** Next, we look at the simulations with the least trend errors in the three states, the recovery rate and influence rate combinations with the least trend error observed for each of the states are listed in Table 6.18. Table 6.17 shows all combinations of predicted and real, observed trend outcomes, for the selected simulation parameters. From Table 6.17, we see that *the number of increase predictions are much higher than other trend predictions, and these predictions are more accurate than others as well.* The confidence factors, again, do not have any specific relationship with the percentage of correct prediction, although majority of the predictions have a confidence of 50% or higher.

The intensity diagrams for simulation parameters with the least trend errors is shown in Figure 6.28. If we look at the intensity diagrams in Figure 6.28 for the simulation parameters with least trend error, *we see that the estimated intensity levels are larger than the estimated intensity with the least average intensity value error from Figure 6.27.* The estimated intensity diagram is also more similar to the intensity diagrams plotted for the least average intensity value error of the complete dataset, i.e., the estimated intensity values do not lie at extreme ends and is rather fluctuating

Table 6.16: Comparison of the predicted trend and real-world trends by number of predictions in for simulations with the **least average intensity value error**,  $w_H = 7$  days,  $t_{lag} = 1$  day: (a) Andhra Pradesh, Recovery Rate ( $\lambda$ ) = 0.0, Influence Rate ( $\gamma$ ) = 0.8, (b) Karnataka, Recovery Rate ( $\lambda$ ) = 0.0, Influence Rate ( $\gamma$ ) = 0.8, (c) Tamil Nadu, Recovery Rate ( $\lambda$ ) = 0.8, Influence Rate ( $\gamma$ ) = 0.0

$pred_{trend,average,t}$ \ $real_{trend,average,t}$	<i>increase</i>	<i>neutral</i>	<i>decrease</i>
<i>increase</i>	33	0	0
<i>neutral</i>	4	2	1
<i>decrease</i>	26	18	21

(a)

$pred_{trend,average,t}$ \ $real_{trend,average,t}$	<i>increase</i>	<i>neutral</i>	<i>decrease</i>
<i>increase</i>	31	0	0
<i>neutral</i>	6	0	0
<i>decrease</i>	31	14	43

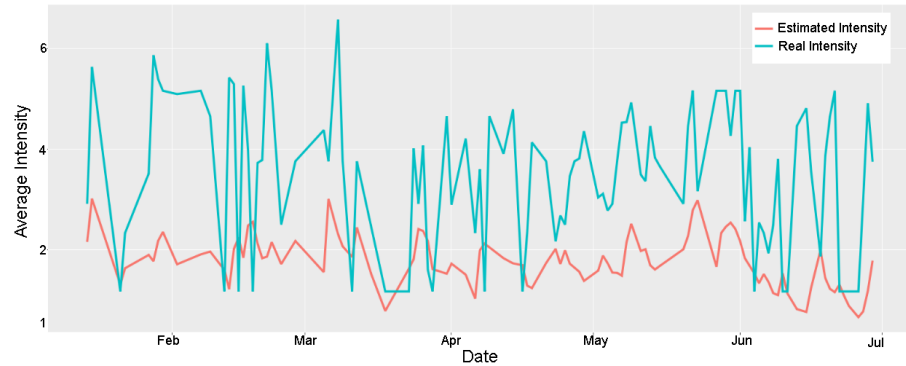
(b)

$pred_{trend,average,t}$ \ $real_{trend,average,t}$	<i>increase</i>	<i>neutral</i>	<i>decrease</i>
<i>increase</i>	29	2	0
<i>neutral</i>	2	0	0
<i>decrease</i>	33	12	41

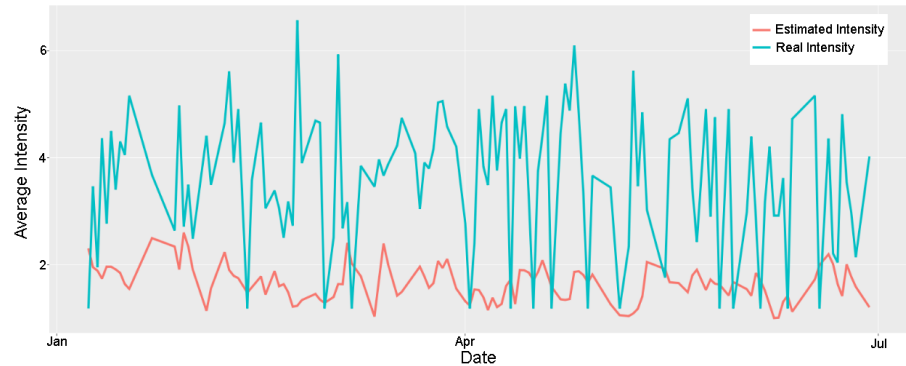
(c)

within the bounds of the focused real-world average-intensity levels. From Table 6.17, we can see that the number of *neutral* trend predictions are much lower than other prediction types, in fact, the number of *neutral* trends is lower than other trends in the focused real-world data.

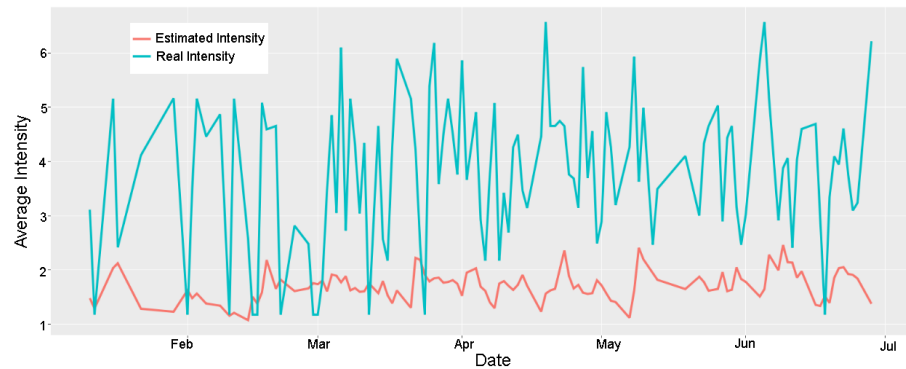
**Total Intensity Value Errors.** It is possible that in cases where we are looking at a focused set of events, we should also look at total intensity value errors. Since there are relatively fewer events in the focused dataset compared to the complete dataset, this could result in reasonable total intensity value errors. Therefore, we also look at the total intensity value error of the same set of simulations as before. Figure



(a)



(b)



(c)

Figure 6.27: Comparison of Estimated Average Intensity and Real Average Intensity comparison of focused set of events, history window ( $w_H$ ) = 7 days and lag ( $t_{lag}$ ) = 1 day. (a) Andhra Pradesh, Recovery Rate ( $\lambda$ ) = 0.0, Influence Rate ( $\gamma$ ) = 0.8, (b) Karnataka, Recovery Rate ( $\lambda$ ) = 0.0, Influence Rate ( $\gamma$ ) = 0.8, (c) Tamil Nadu, Recovery Rate ( $\lambda$ ) = 0.8, Influence Rate ( $\gamma$ ) = 0.0

Table 6.17: Comparison of the predicted trend and realworld trends by number of predictions in for simulations with the **least trend prediction error**,  $w_H = 7$  days,  $t_{lag} = 1$  day: (a) Andhra Pradesh, Recovery Rate ( $\lambda$ ) = 0.2, Influence Rate ( $\gamma$ ) = 0.8, (b) Karnataka, Recovery Rate ( $\lambda$ ) = 0.4, Influence Rate ( $\gamma$ ) = 0.6, (c) Tamil Nadu, Recovery Rate ( $\lambda$ ) = 0.8, Influence Rate ( $\gamma$ ) = 0.2

$pred_{trend,average,t}$ \ $real_{trend,average,t}$	<i>increase</i>	<i>neutral</i>	<i>decrease</i>
<i>increase</i>	52	5	6
<i>neutral</i>	7	4	9
<i>decrease</i>	2	4	16

(a)

$pred_{trend,average,t}$ \ $real_{trend,average,t}$	<i>increase</i>	<i>neutral</i>	<i>decrease</i>
<i>increase</i>	48	8	12
<i>neutral</i>	1	4	9
<i>decrease</i>	2	9	32

(b)

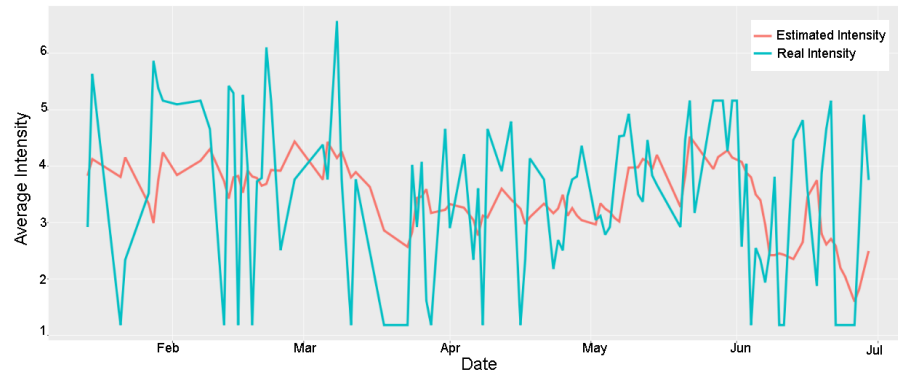
$pred_{trend,average,t}$ \ $real_{trend,average,t}$	<i>increase</i>	<i>neutral</i>	<i>decrease</i>
<i>increase</i>	49	7	8
<i>neutral</i>	4	2	8
<i>decrease</i>	6	8	27

(c)

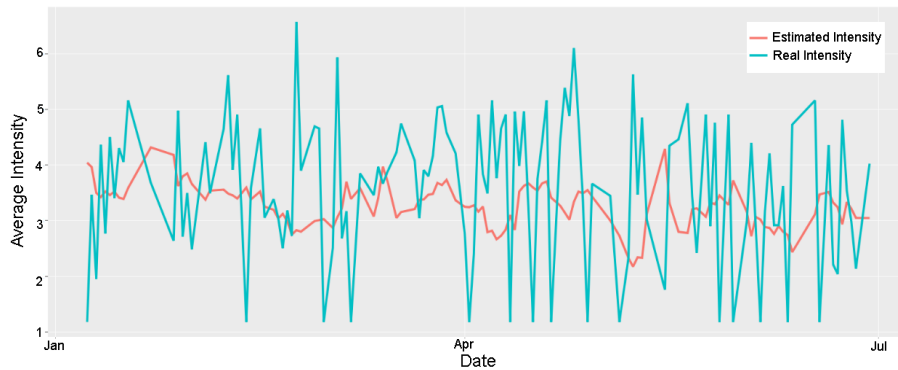
Table 6.18: Simulation parameters with the least trend errors

State	Recovery Rate	Influence Rate
Andhra Pradesh	0.2	0.8
Karnataka	0.4	0.6
Tamil Nadu	0.8	0.2

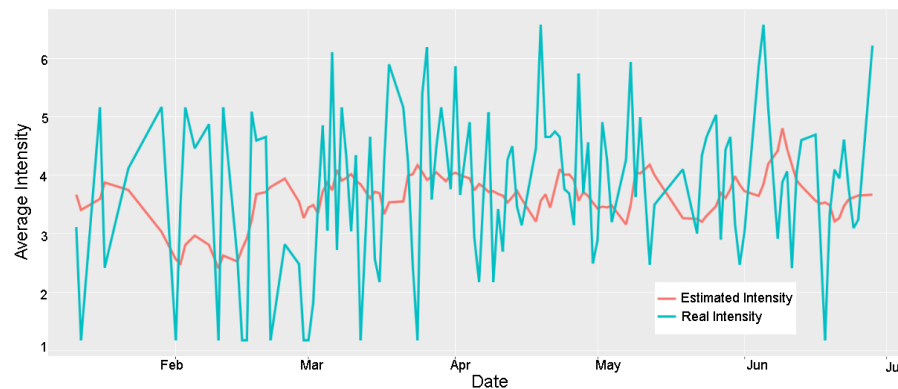
6.29 shows the total intensity value errors for a neighborhood radius of 0.2 and a history window of 7 days. Despite only having a smaller focused set of events to run our simulations, the error is high, and the least error is consistently seen when the recovery rate and influence rate are both 0. This implies that the total intensity in the real-world dataset fluctuates as well and that the total intensity is often very low (close



(a)



(b)



(c)

Figure 6.28: Comparison of Estimated Average Intensity and Real Average Intensity comparison of focused set of events, history window ( $w_H$ ) = 7 days and lag ( $t_{lag}$ ) = 1 day. (a) Andhra Pradesh, Recovery Rate ( $\lambda$ ) = 0.2, Influence Rate ( $\gamma$ ) = 0.8), (b) Karnataka, Recovery Rate ( $\lambda$ ) = 0.4, Influence Rate ( $\gamma$ ) = 0.6, (c) Tamil Nadu, Recovery Rate ( $\lambda$ ) = 0.8, Influence Rate ( $\gamma$ ) = 0.2

to 0). This means that our strategy of using a focused, smaller set of events backfires: the simulated result is even more erratic. Figure 6.30 shows the total intensity of the focused set of events in each of the three states. We can see from Figure 6.30 that for most days, the intensities are very low with some spikes in intensities in between. The low intensity represents a lower number of events compared to the spikes in intensities where we suddenly get relatively high number of events. This suggests that the distribution of events may not be realistic: in some cases there are too many events, and in other there are too few. As noted earlier in Section 5.5.2 (evaluation using total intensity), there is an additive effect on the intensity due to the lack of any negative intensity events and thus the total intensities in the simulations tend to increase.

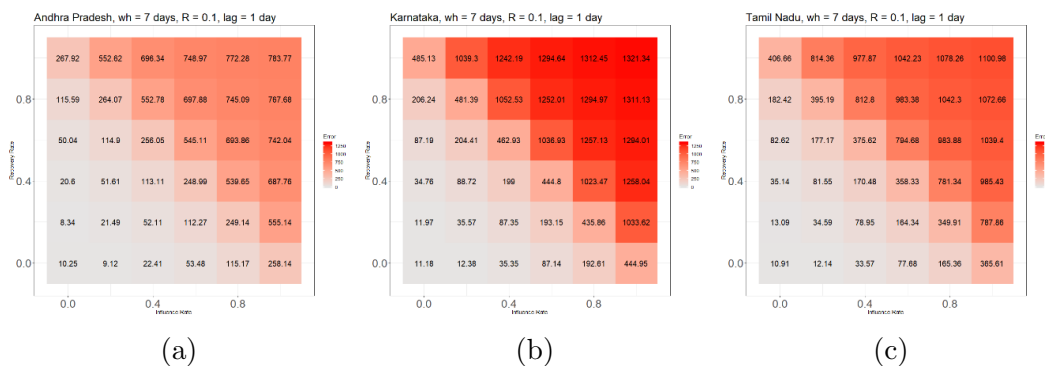
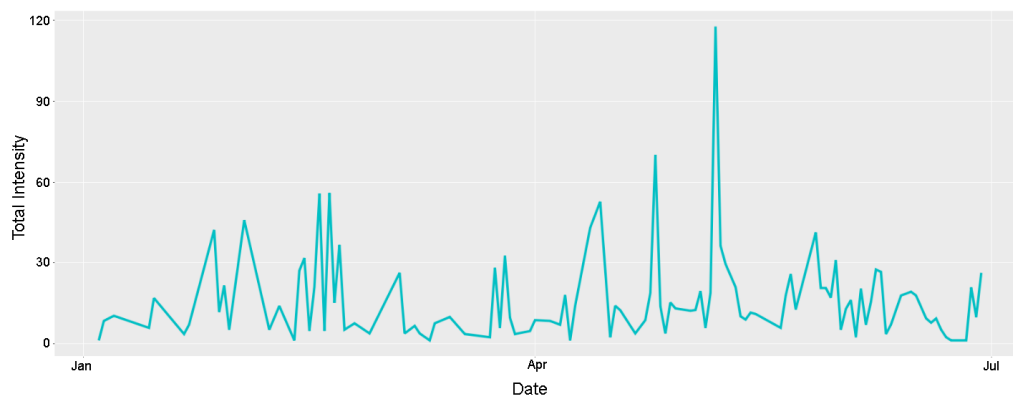


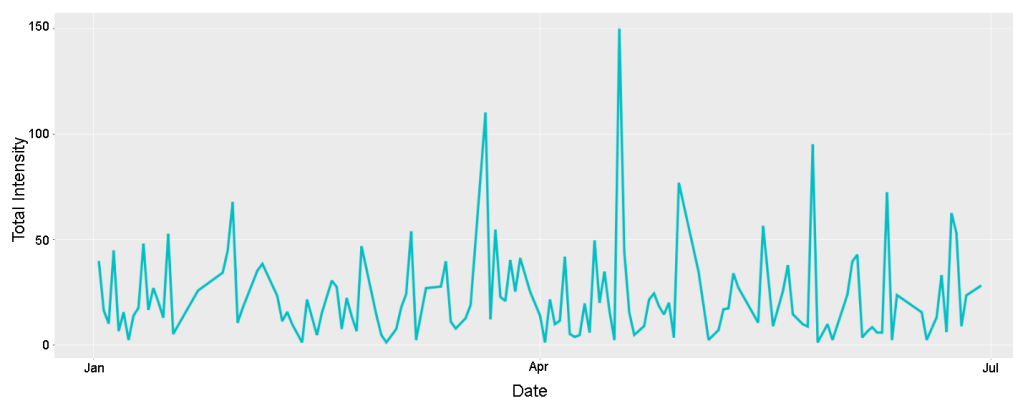
Figure 6.29: Total Intensity Value Error of focused events with 5-10 neighbors, history window ( $w_H$ ) = 7 days, Neighborhood Radius ( $R$ ) = 0.1, lag ( $t_{lag}$ ) = 1 day: (a) Andhra Pradesh, (b) Karnataka, and (c) Tamil Nadu.

In conclusion, there are some fundamental limitations in our events data such as duplicates and incorrect mappings, this is carried over to the experiments using focused set of events. We attempted to isolate a small group of events to mitigate some amount noise in the data. On the contrary, the limitations of the complete dataset were seen to be amplified in the focused dataset. The experiments conducted using average intensity yielded better accuracy compared to using total intensity.

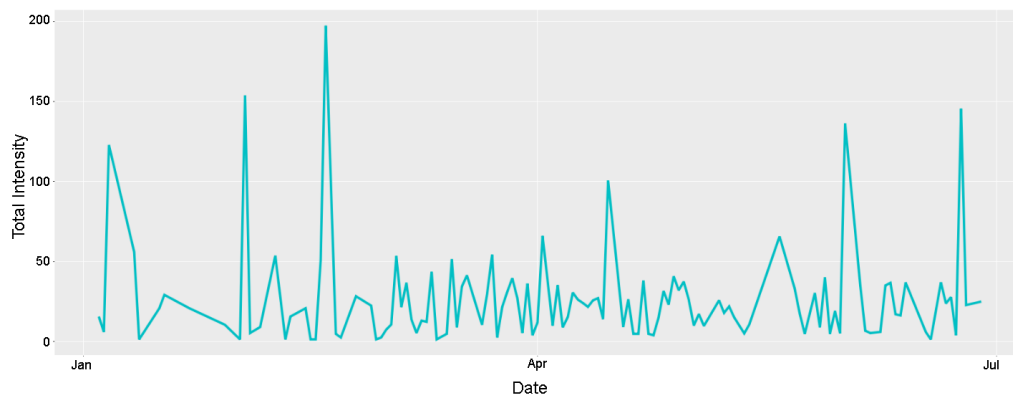




(a)



(b)



(c)

Figure 6.30: Real Total Intensity of focused set of events, history window ( $w_H$ ) = 7 days and lag ( $t_{lag}$ ) = 1 day. (a) Andhra Pradesh, (b) Karnataka, and (c) Tamil Nadu

## 6.4 Summaries

### 6.4.1 The Spatio-Temporal Clustering Approach

To summarize, we have investigated the spatio-temporal clustering approach using the events dataset combined with socioeconomic and infrastructural data. In the clustering results, we see that it is indeed possible to cluster events based on features other than just spatial locations. In the first clustering results shown in Figure 6.1 and Table 6.3, we see that temporal interval of the largest cluster spans throughout our analysis period. However, since we weight the spatial and temporal distances, it is possible that even within the same period and close spatial proximity, events can be grouped into different clusters. We see this behavior again in the clustering results using socioeconomic variables in addition to the spatio-temporal variables in Figure 6.2 and Table 6.4. It is important to note that since we now add a new distance type, spatial distance loses some importance and therefore it becomes difficult for events separated by a large distance, compared to the distances in Table 6.1, to be in the same cluster. While this is to be expected in this case, our model is able to cluster elements that are farther away in spatial distance as long as they are, for example, very close in temporal proximity and have similar socioeconomic variables. Thus, by adjusting the weights used in the distance function, we could form different types of event episodes through the clustering process, which in turn could then provide insights in the evolutionary patterns of social unrest events.

As more distance types or new variables are added to the distance function, the function could become less manageable. It is therefore important these values either be assigned after thorough analysis of the variables with regards to presence of social unrest events or be assigned by domain experts. In our experiments, we assigned equal weight to all the variables because the purpose of our current experimentation

is to develop a robust and configurable way of establishing a meaningful relationship among social unrest events.

Finally, we address the issue of scalability in DBSCAN clustering algorithm with regards to clustering of spatio-temporal data by implementing the ST-KDT-DBSCAN function. Our algorithm (ST-KDT-DBSCAN) allows the use of custom distance function. In Figure 6.4, we can see that the time taken for a regular DBSCAN function is quadratic, while the time efficiency is close to linear. It is possible for the ST-KDT-DBSCAN to also reach quadratic time complexity if the values of frnn radius is very large, but practically the time complexity of the ST-KDT-DBSCAN will always be lower than regular DBSCAN.

#### 6.4.2 The Agent-Based Modelling Approach

To summarize, we have developed a multi-agent simulation framework for social unrest events and have performed various simulations of our agent-based model on various sets of synthetic and real data, with varying parameters. It is easy to interpret the results of synthetic data as it is more controlled and has fewer events which makes it more manageable. The results from the experiments using synthetic data show that it is indeed possible for our model to generate several realistic behaviors, adhering to the design principles listed in Section 5.2.

The biggest challenges we have faced for our experiments is the validity of the real-world data, and identification of optimal simulation parameters ( $R$ ,  $\lambda$  and  $\gamma$ ). The dataset doesn't reflect our theoretical expectations. That is, the events or intensities of events do not seem to follow any specific patterns. Theoretically, total intensity value errors should be more meaningful as it would represent the effect of each event inside a neighborhood. But, due to the lack of negative events and an over-abundance of unrest events, our model generally predicts much higher intensities than observed in

the real-world dataset. This makes it difficult to assess the validity of our model. We therefore continue the evaluation of our model using average intensity value error, as the average intensity could allow us to capture the general mood of any location. This is important because a location with a high average intensity could be considered as a volatile location in relation to the propensity to violence. The problem with using the average intensity in our experiments is that the total count of events is ignored. For example, one event with an intensity value 5.0 becomes equivalent to 3 events with intensity values 2.0, 5.0 and 8.0, since the average in both cases is the same. At the same time, additive effects of the total intensity values are reduced through this change. We can see from Figures 6.20, 6.21 and 6.22, that the predicted intensities do follow the overall pattern in the real-world intensities.

We have used additional measures to evaluate our simulation results, by looking at trend errors and evaluated confidence factors in our predictions. We can see in Tables 6.10, 6.11 and 6.12 that the trend predictions in the three states are quite reasonable with a higher accuracy for increase and neutral trends compared to negative trends. This is due to the same problem as noted above, i.e., no negative intensity events. The confidence factors have been found to be inaccurate as well and we should consider new factors while making predictions.

Overall, the project shows some promise in being able to predict trends and average intensity of any locations. In the future, the model needs to address the noted problems such as duplicate events data, incorrect labels, and lack of negative events (e.g., peace-keeping events). A more robust method perhaps with domain expertise and from the model-driven perspective, in addition to a data-driven one should be developed for estimating optimal simulation parameters.

## Chapter 7

### Conclusions and Future Work

#### 7.1 Conclusions

In this thesis, we first introduced a multi-factorial distance function to help us determine a conceptual distance between any pair of social unrest events based on their spatio-temporal, socioeconomic, and infrastructural differences. We then provided a method of organizing spatio-temporal data in a k-dimensional tree that uses spatial and temporal radii to search events. We used this data-structure and the distance function to create a modified DBSCAN clustering algorithm called ST-KDT-DBSCAN. We showed that this modified DBSCAN (ST-KDT-DBSCAN) is more scalable than the existing DBSCAN algorithm. We applied this algorithm to a social unrest dataset and presented our results. We presented an agent-based model which allows us to simulate and predict social unrest events. We loosely based this model on the n-body gravitational push-pull principle. We then ran experiments on synthetic as well as real-world dataset and presented our results. We used these experiments to evaluate the viability of our model, and to identify issues in order to better implement it.

Through the clustering results, we showed that the multi-factorial distance function allows us to connect events by not just space or time, but also by socioeconomic

and infrastructural properties of their locations. Thus, by controlling the weights assigned to the different distance types (spatio-temporal, socioeconomic and infrastructural), we can form different types of event clusters (episodes). Since we use weighing of different types of distances, as well as the parameters, it quickly becomes difficult to manage. Note that we supply the threshold values for spatial distances and scale the available dataset. Hence, the distance values from two experiments with different datasets cannot be directly compared. Furthermore, since the resultant distance is always a single value, it becomes difficult to understand conceptually what the distance value means in terms of the different distance types. For example, we cannot estimate what a distance of 0.2 means in terms of spatial distance alone. It is hence very important to estimate and assign correct weights to the different distance types in the distance function. Therefore, a more dynamic method of assigning weights must be identified which could be based on either data-driven approaches or social theories.

In our agent-based modelling experiments performed on synthetic data, we can see that the behavior of events is somewhat realistic. However, we do not see significant resemblance in the experiments performed with real data. While it is possible that there are problems in the ABM design, we pointed out that the data collected from GDELT does not have a time-range for events, making it very difficult to verify the actual decay or rise of intensities for different events. Moreover, we identified that there are several instances where the data was incorrectly reported which can cause abrupt rise or fall in intensity levels of the real-world data throwing off the comparisons of expected intensity during simulation and actual observed intensity from raw data. We also identified that there are no negative events available to us, in the real world it is often the case that government groups or the group in power will attempt to appease the protesters by listening to their demands or in other cases

deploy forces to control them. In our model, we refer to these events as peace-keeping events. Such peace-keeping events will result in either resolution of the unrest event or suppression of the protesters such that the unrest event ends. Due to the lack of such unrest-events, our model is unable to simulate immediate decay of intensity in the model.

## 7.2 Future Work

In the clustering approach, we identify several clusters (episodes) of social unrest but a detailed analysis is necessary to validate the results. Thus, one future work item will focus on leveraging the approach towards identifying episodes of social unrest events. We plan to work on addressing challenges such as identification of unrest episode type and identifying uncertainty due to quality of data. The distance function can be further developed so that optimal values for the attribute weights are auto determined based on either social theories or scope and scale of data and used in computation without intervention of the researcher.

We used a modified version of the DBSCAN algorithm (ST-KDT-DBSCAN) in our clustering approach, a locality-sensitive hashing (LSH) approach could also be taken. The basic idea behind LSH is to project the data into a low-dimensional binary (Hamming) space; that is, each data point is mapped to a  $b$ -bit vector, called the hash key [36]. If this projection is performed appropriately by adapting it to the multi-factorial distance function, we can find clusters faster than the regular clustering algorithm. Thus the challenge in a LSH based approach is in the methodology used for the projection. Another approach could be to train a neural network to learn an embedded representation of the data points and then use a simple distance metric such as a Euclidean distance paired with LSH to pre-process and cluster the data.

Depending on how we implement these algorithms, the results could be different from the ST-KDT-DBSCAN algorithm. Points with insufficient similar points could be regarded as noise points, similar to the density based clustering.

We will also be working on several modifications to the simulation model such as collecting and using data with negative intensity which are events of the peace-keeping nature. This should add more realism to the model and allow us to predict the decay in intensities of any region. It is possible that the simulation parameters such as recovery-rate could be dynamic. The rate at which any event dies naturally could be different based on the socioeconomic properties of that location. Additionally, the more an event survives without enough neighbors, the more likely it could be that it decreases to zero intensity (or, “it dies”). Hence, based on these points, it is possible that the simulation parameters such as recovery rate should be dynamic. Methodologies should be developed to auto-assign simulation parameters based on social theories. The confidence factor of unrest prediction is an important outcome and we should therefore identify necessary variables in estimating the confidence factor for social unrest predictions. Other future development includes the ability to run “what-if” scenarios. In “what-if” scenarios, the user will be able to add new events at any desired location and observe the impact of said event in the simulation. This could allow decision makers to form plans on how to influence social unrest and avoid human and property damage.



## Bibliography

- [1] S.J. Aarseth and S.J. Aarseth. *Gravitational N-Body Simulations: Tools and Algorithms*. Cambridge Monographs on Mathematical Physics. Cambridge University Press, 2003.
- [2] Sameera Abar, Georgios K. Theodoropoulos, Pierre Lemarinier, and Gregory M.P. OHare. Agent based modelling and simulation tools: A review of the state-of-art software. *Computer Science Review*, 24:13 – 33, 2017.
- [3] C. N. Alam, K. Manaf, A. R. Atmadja, and D. K. Aurum. Implementation of haversine formula for counting event visitor in the radius based on android application. In *2016 4th International Conference on Cyber and IT Service Management*, pages 1–6, April 2016.
- [4] Gennady Andrienko and Natalia Andrienko. Interactive cluster analysis of diverse types of spatiotemporal data. *SIGKDD Explor. Newsl.*, 11(2):19–28, May 2010.
- [5] Mihael Ankerst, Markus M. Breunig, Hans-Peter Kriegel, and Jörg Sander. Optics: Ordering points to identify the clustering structure. *SIGMOD Rec.*, 28(2):49–60, June 1999.

- [6] Jon L. Bentley, Donald F. Stanat, and E.Hollins Williams. The complexity of finding fixed-radius near neighbors. *Information Processing Letters*, 6(6):209 – 212, 1977.
- [7] Jon Louis Bentley. Multidimensional binary search trees used for associative searching. *Commun. ACM*, 18(9):509–517, September 1975.
- [8] Jon Louis Bentley. K-d trees for semidynamic point sets. In *Proceedings of the Sixth Annual Symposium on Computational Geometry, SCG '90*, pages 187–197, New York, NY, USA, 1990. ACM.
- [9] Thomas Berger. Agent-based spatial models applied to agriculture: a simulation tool for technology diffusion, resource use changes and policy analysis. *Agricultural Economics*, 25(23):245–260, 2001.
- [10] Rebecca H. Best, Christine Carpino, and Mark J.C. Crescenzi. An analysis of the tabari coding system. *Conflict Management and Peace Science*, 30(4):335–348, 2013.
- [11] Derya Birant and Alp Kut. St-dbscan: An algorithm for clustering spatial-temporal data. *Data Knowl. Eng.*, 60(1):208–221, January 2007.
- [12] Christopher M. Bishop. *Pattern Recognition and Machine Learning (Information Science and Statistics)*. Springer-Verlag, Berlin, Heidelberg, 2006.
- [13] Matthew Bolaos, John Forrest, and Michael Hahsler. *Clustering Large Datasets Using Data Stream Clustering Techniques*, pages 135–143. 10 2014.
- [14] Charles E. Brown. Coefficient of variation. In *Applied Multivariate Statistics in Geohydrology and Related Sciences*, pages 155–157. Springer Berlin Heidelberg, Berlin, Heidelberg, 1998.

- [15] Andrew Crooks, Christian Castle, and Michael Batty. Key challenges in agent-based modelling for geo-spatial simulation. *Computers, Environment and Urban Systems*, 32(6):417 – 430, 2008. GeoComputation: Modeling with spatial agents.
- [16] Hakim Darbouche. *The Implications of the Arab Uprisings for Oil and Gas Markets*. 2011.
- [17] Jennifer Earl, Andrew Martin, John D. McCarthy, and Sarah A. Soule. The use of newspaper data in the study of collective action. *Annual Review of Sociology*, 30:65–80, 2004.
- [18] Ali Mert Ertugrul, Yu-Ru Lin, Wen-Ting Chung, Muheng Yan, and Ang Li. Activism via attention: interpretable spatiotemporal learning to forecast protest activities. *EPJ Data Science*, 8(1):5, Feb 2019.
- [19] Martin Ester, Hans-Peter Kriegel, Jörg Sander, and Xiaowei Xu. A density-based algorithm for discovering clusters a density-based algorithm for discovering clusters in large spatial databases with noise. pages 226–231, 1996.
- [20] James Fogarty, Ryan S. Baker, and Scott E. Hudson. Case studies in the use of roc curve analysis for sensor-based estimates in human computer interaction. In *Proceedings of Graphics Interface 2005*, GI '05, pages 129–136, School of Computer Science, University of Waterloo, Waterloo, Ontario, Canada, 2005. Canadian Human-Computer Communications Society.
- [21] Deen Freelon, Charlton D. McIlwain, and Meredith Clark. Beyond the Hashtags: #Ferguson, #Blacklivesmatter, and the Online Struggle for Offline Justice. 2016.
- [22] William Gamson. *Review: The Strategy of Social Protest*, volume 90. [Academy of Political Science, Wiley], 1975.

- [23] J. E. Gentle, L. Kaufman, and P. J. Rousseeuw. Finding Groups in Data: An Introduction to Cluster Analysis. *Biometrics*, 2006.
- [24] Deborah J. Gerner, Rajaa Abu-Jabr, Philip A. Schrodtt, and mr Yilmaz. Conflict and mediation event observations (CAMEO): A new event data framework for the analysis of foreign policy interactions. In *of Foreign Policy Interactions. Paper presented at the International Studies Association*, 2002.
- [25] Brian J. Goode, Siddharth Krishnan, Michael Roan, and Naren Ramakrishnan. Pricing a protest: Forecasting the dynamics of civil unrest activity in social media. *PLOS ONE*, 10(10):1–25, 10 2015.
- [26] Cyril Goutte and Eric Gaussier. A probabilistic interpretation of precision, recall and f-score, with implication for evaluation. In David E. Losada and Juan M. Fernández-Luna, editors, *Advances in Information Retrieval*, pages 345–359, Berlin, Heidelberg, 2005. Springer Berlin Heidelberg.
- [27] Sudipto Guha, Rajeev Rastogi, and Kyuseok Shim. Cure: An efficient clustering algorithm for large databases. *SIGMOD Rec.*, 27(2):73–84, June 1998.
- [28] Sepp Hochreiter and Jrgen Schmidhuber. Long short-term memory. *Neural Computation*, 9(8):1735–1780, 1997.
- [29] T. Hua, C. Lu, N. Ramakrishnan, F. Chen, J. Arredondo, D. Mares, and K. Summers. Analyzing civil unrest through social media. *Computer*, 46(12):80–84, Dec 2013.
- [30] Stephanie Huerre, Jehee Lee, Ming Lin, and Carol O’Sullivan. *Simulating Believable Crowd and Group Behaviors*. SA ’10. ACM, New York, NY, USA, 2010.

- [31] Deepti Joshi, Ashok Samal, and Leen-Kiat Soh. Spatio-temporal polygonal clustering with space and time as first-class citizens. *GeoInformatica*, 17(2):387–412, Apr 2013.
- [32] Deepti Joshi, Leen-Kiat Soh, Sudeep Basnet, Ashok K Samal, Regina E Werum, Hariharan Arunachalam, and Shawn Ratcliff. SURGE Social Unrest Reconnaissance GazEteer (Demo Paper). In *GIS*, volume 2017-November. Association for Computing Machinery, 11 2017.
- [33] P Juszczak, D M J Tax, and Robert P W Duin. Feature scaling in support vector data description. In *Proc. ASCI*, 2002.
- [34] L. Kaufman and P.J. Rousseeuw. *Clustering by Means of Medoids*. Delft University of Technology : reports of the Faculty of Technical Mathematics and Informatics. Faculty of Mathematics and Informatics, 1987.
- [35] Rostyslav Korolov, Di Lu, Jingjing Wang, Guangyu Zhou, Claire Bonial, Clare Voss, Lance Kaplan, William Wallace, Jiawei Han, and Heng Ji. On predicting social unrest using social media. In Ravi Kumar, James Caverlee, and Hanghang Tong, editors, *Proceedings of the 2016 IEEE/ACM International Conference on Advances in Social Networks Analysis and Mining, ASONAM 2016*, Proceedings of the 2016 IEEE/ACM International Conference on Advances in Social Networks Analysis and Mining, ASONAM 2016, pages 89–95, United States, 11 2016. Institute of Electrical and Electronics Engineers Inc.
- [36] B. Kulis and K. Grauman. Kernelized locality-sensitive hashing for scalable image search. In *2009 IEEE 12th International Conference on Computer Vision*, pages 2130–2137, Sep. 2009.

- [37] Kalev Leetaru and Philip A Schrodtt. GDELT: Global Data on Events, Location and Tone. *International Studies Association*, 2012.
- [38] Oded Maimon and Lior Rokach. *Data Mining and Knowledge Discovery Handbook*. 2nd edition, 2010.
- [39] David Matsumoto, Hyisung C. Hwang, and Mark G. Frank. Emotions expressed in speeches by leaders of ideologically motivated groups predict aggression. *Behavioral Sciences of Terrorism and Political Aggression*, 6(1):1–18, 2014.
- [40] Lukas Meier, Sara Van De Geer, and Peter Bhlmann. The group lasso for logistic regression. *Journal of the Royal Statistical Society: Series B (Statistical Methodology)*, 70(1):53–71, 2008.
- [41] GoI Ministry of Home Affairs. Office of the Registrar General & Census Commissioner, India, 2011.
- [42] Sathappan Muthiah, Bert Huang, Jaime Arredondo, David Mares, Lise Getoor, Graham Katz, and Naren Ramakrishnan. Planned protest modeling in news and social media. 2015.
- [43] Michael J. North, Nicholson T. Collier, Jonathan Ozik, Eric R. Tatara, Charles M. Macal, Mark Bragen, and Pam Sydelko. Complex adaptive systems modeling with repast simphony. *Complex Adaptive Systems Modeling*, 1(1):3, Mar 2013.
- [44] Sean P. O’Brien. Crisis early warning and decision support: Contemporary approaches and thoughts on future research. *International Studies Review*, 12(1):87–104, 2010.

- [45] OpenStreetMap contributors. Planet dump retrieved from <https://planet.osm.org> . <https://www.openstreetmap.org>, 2017.
- [46] Aditya Parameswaran, Ming Han Teh, Hector Garcia-Molina, and Jennifer Widom. Datasift: A crowd-powered search toolkit. In *Proceedings of the 2014 ACM SIGMOD International Conference on Management of Data*, SIGMOD '14, pages 885–888, New York, NY, USA, 2014. ACM.
- [47] Fengcai Qiao, Pei Li, Xin Zhang, Zhaoyun Ding, Jiajun Cheng, and Hui Wang. Predicting Social Unrest Events with Hidden Markov Models Using GDELT. *Discrete Dynamics in Nature and Society*, 2017.
- [48] Fengcai Qiao, Pei Li, Xin Zhang, Zhaoyun Ding, Jiajun Cheng, and Hui Wang. Predicting social unrest events with hidden markov models using GDELT. *Discrete Dynamics in Nature and Society*, 2017:1–13, 2017.
- [49] Anand Rajaraman and Jeffrey David Ullman. *Data Mining*, page 117. Cambridge University Press, 2011.
- [50] Naren Ramakrishnan, Patrick Butler, Sathappan Muthiah, Nathan Self, Rupinder Khandpur, Parang Saraf, Wei Wang, Jose Cadena, Anil Vullikanti, Gizem Korkmaz, Chris Kuhlman, Achla Marathe, Liang Zhao, Ting Hua, Feng Chen, Chang Tien Lu, Bert Huang, Aravind Srinivasan, Khoa Trinh, Lise Getoor, Graham Katz, Andy Doyle, Chris Ackermann, Ilya Zavorin, Jim Ford, Kristen Summers, Youssef Fayed, Jaime Arredondo, Dipak Gupta, and David Mares. 'beating the news' with embers: Forecasting civil unrest using open source indicators. In *Proceedings of the 20th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, KDD '14, pages 1799–1808, New York, NY, USA, 2014. ACM.

- [51] Craig W. Reynolds. Flocks, herds and schools: A distributed behavioral model. *SIGGRAPH Comput. Graph.*, 21(4):25–34, August 1987.
- [52] Ruchika Parmarth Ganar and S. Ardhapurkar. Prediction of civil unrest by analysing social network using keyword filtering: A survey. In *2016 Online International Conference on Green Engineering and Technologies (IC-GET)*, pages 1–4, Nov 2016.
- [53] Parang Saraf and Naren Ramakrishnan. EMBERS AutoGSR: Automated Coding of Civil Unrest Events. In *Proceedings of the 22Nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, KDD '16*, pages 599–608, New York, NY, USA, 2016. ACM.
- [54] Philip A. PA Schrodtt, Event Data Project, and Pond Laboratory. CAMEO : Conflict and Mediation Event Observations Event and Actor Codebook. 2012.
- [55] Stephen M Shellman, Brian P Levey, and Joseph K Young. Shifting sands: Explaining and predicting phase shifts by dissident organizations. *Journal of Peace Research*, 50(3):319–336, 2013.
- [56] Marina Sokolova and Guy Lapalme. A systematic analysis of performance measures for classification tasks. *Information Processing & Management*, 45(4):427 – 437, 2009.
- [57] Dataset Title, World Bank, Esoc Princeton, Wits Data, Esoc Princeton, Memoria Hist, Basta Ya Datasets, Violence Hrdag, Rights Data, Analysis Group, Project Esoc, Civil War, Esoc Princeton, Bfrs Political, Tracker Website, Major Victor H Sundquist, Giulia Sandri, Mario Telò, Luca Tomini, Nancy Rivenburgh, Clionadh Raleigh, Development Practitioners, Donatella DELLA PORTA, Sidney TARROW, Donatella Della Porta, Typology Overview September, Giulia



- Mascagni, Mona Lena Krook, Harry W. Fischer, Joseph R A Ayee, W. Eubank, L. Weinberg, A J Bebbington, Capt Kenneth, R Langford, South Africa, Data Comparison, and ACLED. Armed Conflict Location & Event Data Project ( ACLED ) User Guide. *Terrorism and Political Violence*, 2015.
- [58] Hadi Fanaee Tork. Spatio-Temporal Clustering Methods Classification. *Doctoral Symposium on Informatics Engineering (DSIE'2012)*, 2017.
- [59] Perukrishnen Vytelingum, Thomas D. Voice, Sarvapali D. Ramchurn, Alex Rogers, and Nicholas R. Jennings. Agent-based micro-storage management for the smart grid. In *Proceedings of the 9th International Conference on Autonomous Agents and Multiagent Systems: Volume 1 - Volume 1*, AAMAS '10, pages 39–46, Richland, SC, 2010. International Foundation for Autonomous Agents and Multiagent Systems.
- [60] Tian Zhang, Raghuram Ramakrishnan, and Miron Livny. Birch: An efficient data clustering method for very large databases. volume 25, pages 103–114, New York, NY, USA, June 1996. ACM.
- [61] L. Zhao, F. Chen, J. Dai, T. Hua, C.-T. Lu, and N. Ramakrishnan. Unsupervised Spatial Event Detection in Targeted Domains with Applications to Civil Unrest Modeling. *PLoS ONE*, 9(11), October 2014.