Modeling and Generation of Soft Data in Kinematic Scenarios for Surveillance Applications

MODELING AND GENERATION OF SOFT DATA IN KINEMATIC SCENARIOS FOR SURVEILLANCE APPLICATIONS

BY

SAEID ROSTAMI, M.Sc.

A THESIS

SUBMITTED TO THE DEPARTMENT OF ELECTRICAL & COMPUTER ENGINEERING

AND THE SCHOOL OF GRADUATE STUDIES

OF MCMASTER UNIVERSITY

IN PARTIAL FULFILMENT OF THE REQUIREMENTS

FOR THE DEGREE OF

MASTER OF APPLIED SCIENCE

Master of Applied Science (2018) McMaster University
(Electrical & Computer Engineering) Hamilton, Ontario, Canada

TITLE:	Modeling and Generation of Soft Data in Kinematic Scenarios for Surveillance Applications
AUTHOR:	Saeid Rostami M.Sc., (Electrical Engineering) University, City, Country
SUPERVISOR:	Dr. T. Kirubarajan
NUMBER OF PAGES:	xi, 73

To my parents, for their endless love, support and encouragement

Abstract

Recently data generation has become an important research topic. Simulated data are not expensive and can be used immediately after being generated. Unlike simulated data, real data is expensive and time consuming to collect and in many cases real world data need to be cleaned before using. In this work we have developed a software that can generate soft data from events. This software generates output of NLP without using NLP complex technique, which can be used for testing fusion algorithms or using the generated data for testing data quality, as well as data mining algorithms. All the coding part has been done in C++ using Microsoft Visual Studio.

Acknowledgements

Being at Mcmaster university has been a period of intense learning for me, not only in the scientific area, but also on a personal level. I would like to mention few of the people who have supported and helped me so much throughout my study here.

First and foremost, I would like to extend special thanks to my academic supervisor, Dr. T. Kirubarajan, for his valuable advice, support and guidance, which helped me to excel in my work. I would also like to thank Dr. Tharmarasa for helping me in every step of my work. I am also thankful to the Department of Electrical & Computer Engineering, for giving me this opportunity to study at McMaster University. I want to thank the ECE department's graduate administrative assistant Cherly Gies for all the work she does in this department. I would also like to thank Dr. Chen and Dr. Jeremic for being members of my thesis defense committee.

Last, but not least, I would like to thank my parents, and all my friends for their support and encouragement during my study at Mcmaster university.

Notation and abbreviations

NLP	Natural Language Processing
POS	Part Of Speech
OSM	Open Street Map
ERC	Event Report Capability

Contents

List of Figures

Chapter 1

Introduction and Problem Statement

Data from simulation provides the opportunity to test models and algorithms before applying to real world data. Today, In both academia and industry researchers prefer to work with simulated data, since it is inexpensive and can be collected faster than real world data.

There are two kind of data: hard data and soft data. Hard data is the type of the data that come from devices like sensor, camera and computer. On the other hand, soft data come from human observation and decisions [1]. Data that come from devices are quantitative and can be measured and shown by numbers and graphs as opposed to soft data, which are qualitative and derived from human judgment [1].

Traditionally, researchers who work on multi-sensor data fusion collected their data from actual sensors. For example, in the problem of self-drive vehicles fusion of information gathered from multiple sources plays an important role in building a comprehensive situation picture for vehicles [2]. A self-drive vehicle, that can be used in real world driving environments, must be able to reliably detect and effectively track the nearby moving objects. In [3] the author presents a new, moving object detection and tracking system that is based on fusing their observations from active sensors (i.e., radars and LIDARs) with a vision sensor(i.e.,Optical or IR camera). In their system, the vision module detects pedestrians, bicycles, and vehicles to generate corresponding vision targets. Then, the system uses this visual recognition information for improving a tracking model selection, data association, and movement classification of their algorithm. Another critical application of multiple sensor information fusion is for extracting the road lines and borders. Radar sensor cannot provide any information for this purpose. This is when that observations from a camera can play an important role and fill the gaps. On the other hand, since the cameras are usually installed in the front window of vehicles they cannot cover the vehicle blind spots. Here it is the Radar or Lidar sensor that provide information.

However, there has been an increasing interest in using soft data for data fusion based applications that can yield more accurate fused information. Especially in the cases where physical sensors do not cover all the area and information from the field help to cover blind spot in fusion area. For example, what is the best path from your work to airport especially when you are in a hurry. From experience we know the shortest distance but we also know that the shortest path is not always the fastest; there maybe delays due to traffic or accident on the routes. In such scenario not only do we need hard data, which tell us our route options and distances but also need to know what is happening on the route that can come from people who are on the field at the time. Figure 1.1 shows how soft data and hard data can work together to get output.

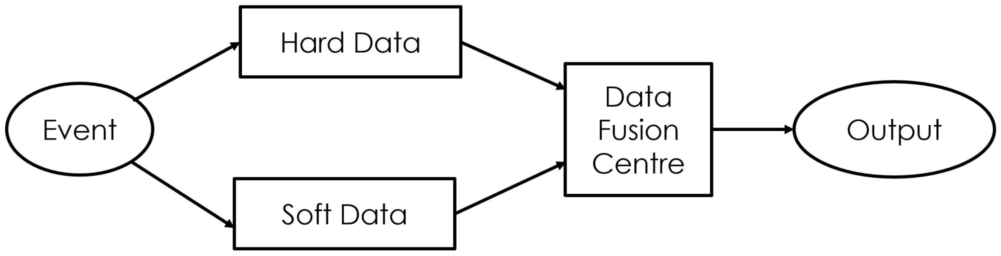

Figure 1.1: Hard Data/Soft Data Fusion Diagram.

There are several ways to get human generated data such as text message, phone call and social media platforms such as Facebook and Twitter. The problem is that most of the companies like Twitter and Facebook are reluctant to share their users' information and even when they do it tends to be very expensive. Furthermore, even after obtaining the data some complex algorithms like Natural language Processing(NLP) should be done on the data to extract the key words and get it ready for use. What NLP algorithms are doing here is converting human languages to quantitative form for further usage [4]. NLP demands efforts such as structuring the text, finding and extracting features, word segmentation, Part of Speech (POS) tagging and parsing [5].

In this work I have tried to create a software that can generate soft data from different events. Our software can generate the output of NLP without using NLP complex technique. Researchers can use this soft data in many areas like testing their fusion algorithms or for testing data quality, and for data mining algorithms.

1.1 Related Works

Simulated data are cheap and can be used immediately after being generated as opposed to real data, which is expensive and time consuming to collect and in many cases real data need to be cleaned before using.

Data generation is used in many fields from industry to health science and from military to academia. For example, data simulation software in human genetics was attempted in [6]. A novel software package for simulating of genomics data has been discussed in this paper [6]. A soft data simulation has been used for processing biomedical data in health science [7].

There is a huge demand for the applications that apply fusion technique on soft information sources [8]. In [9] the author used graphical methods for real time fusion with soft data. This study describes how to associate and use data for intelligence analysis purposes. Study in [10] shows the steps for generating data set for soft/hard information fusion. In [11] the authors had some research on soft and hard data collection. As we discussed before soft data are qualitative and it is difficult to show them as numbers and graphs.However, There are some studies to convert qualitative data to quantitative data. In [12] the authors used fuzzy set theory to convert soft data to quantitative data and then combined that with hard data. In [13] the author first converted all soft data into hard data and then used graphs to associate them.

1.2 Proposed Approach and Contribution

As discussed in previous sections the lack of soft data is a significant barrier to test fusion methods. In this work, a Software, which is able to generate soft data, the output of NLP algorithms, has been designed.

The challenge here is to convert qualitative data into the quantitative format to use it in our software. To the best of this author's knowledge, Fuzzy Set Theory is the the best approach for simulating qualitative data.

For having a realistic generated data, I tried to make it as real as possible. Consider a situation in which the person(sensor) report does not include the exact color of the cars involved in an accident. The word for color is ambiguous data. Our approach for this category of information are Confusion Matrixes. In each of these matrixes an inaccurate data can be an accurate one by a probability, which is based on real world situation. Using Bayesian Network, we introduced a method for soft data fusion. The software also supports both the first hand and second hand information. We also gave characteristics for sensors, which can be useful in situations where Machine Learning, data mining or data quality need to be applied on soft data.

1.3 Organization of the Thesis

This thesis is divided into four chapters. Chapter 2 discusses mathematical background in fuzzy data modeling. Chapter 3 presents the proposed approach to generating soft data. Chapter 4 presents results and simulation.

Chapter 2

Soft Data Modeling

In most of the real world applications, information and data are imperfect and uncertain [14], especially in soft data where information can be vague and imprecise. For example, the speed of the car was a little fast is a vague expression. Traditional data modeling systems , are incapable of modeling the uncertain information [15]. Since the 1980s, when fuzzy logic has been introduced by Zadeh [16], it has been used to represent and extend many data models. One of the main reasons for introducing fuzzy logic was to improve the traditional data modeling algorithms and being able to represent the imperfect and uncertain data in any database [15].

2.1 Imperfect Data and Fuzzy set Theory

We can classify imperfect data in different categories. As we can see from Table 1 the first category is inconsistency and it is semantic conflict [15]. It means that one data has been represented more than once and with different values in the database [15]. For example, the number of cars involved in the accident is saved twice of 5 and 7 in the database.

Table 2.1: Different Type of Imperfect Data

Imperfect Data					
	Inconsistency	**Imprecision**	**Vagueness**	**Uncertainty**	**Ambiguity**
Characteristics	Semantic Conflict	Relevant to Attribute Value	Linguistic Value	Degree of truth of Attribute value	Several interpretation

Imprecision data value should be chosen from a given range; the color of the car is {red, orange, yellow}, and vagueness data always have linguistic values [15]. For example, the speed of the red car was fast. Uncertainty depends on the amount of accuracy of an attribute value [15]. An example of that kind of data can be, I am 80% sure that the color of the car was red. The ambiguity means that we can have several interpretation from one value belonging to an attribute.

2.2 Fuzzy Sets and Fuzzy Data Representation

There has been some research on different type of imperfect information as shown in Tablel 1. Also there are many ongoing research on imprecision and uncertain data are based on fuzzy sets and probability distribution theories [17]. Zadeh in [16] and [17] explains the fuzzy sets as follows:

Let U be a universe of discourse and a fuzzy value of U can be obtained by a Fuzzy set F in U. The fuzzy set F in the universe of information U can be defined as a set of pairs and it can be shown as

$$F = \{(\mu_F(u), u) | u \in U\} \tag{2.1}$$

Here μ_F is membership function for F and can be defined as

$$\mu_F : U \to [0, 1] \tag{2.2}$$

here $\mu_F(u)$ is the membership degree of u in the fuzzy set F for each $u \in U$. Then we can define the fuzzy set F as

$$F = \{(\mu_F(u_1), u_1), (\mu_F(u_2), u_2), ..., (\mu_F(u_n), u_n)\} \tag{2.3}$$

When the membership degree $\mu_F(u)$ is supposed to be a measure of the possibility that a variable X has the value u, and fuzzy value is described by a possibility distribution π_x [17] as

$$F = \{\pi_X(u_1), u_1, \pi_X(u_2), u_2, ..., \pi_X(u_n), u_n\} \tag{2.4}$$

Here $\pi_X(u_i)$ is the possibility that u_i is true. We also know that $u_i \in U$ and π_X the possibility distribution representation.

2.3 Fuzzy Data Modeling

In Equation 2.4, π_X is the possibility distribution that shows the fuzzy value of variable X. Then we can conclude that the value of X is fuzzy and it can take one of the possible values of $(u_1, u_2, ..., u_n)$ [18].

For years, fuzzy sets and possibility theory have been used for modeling various type of data. In fuzzy database models, linguistic tags are used for showing fuzzy data [19], like, fast and slow for speed or young and old for age.

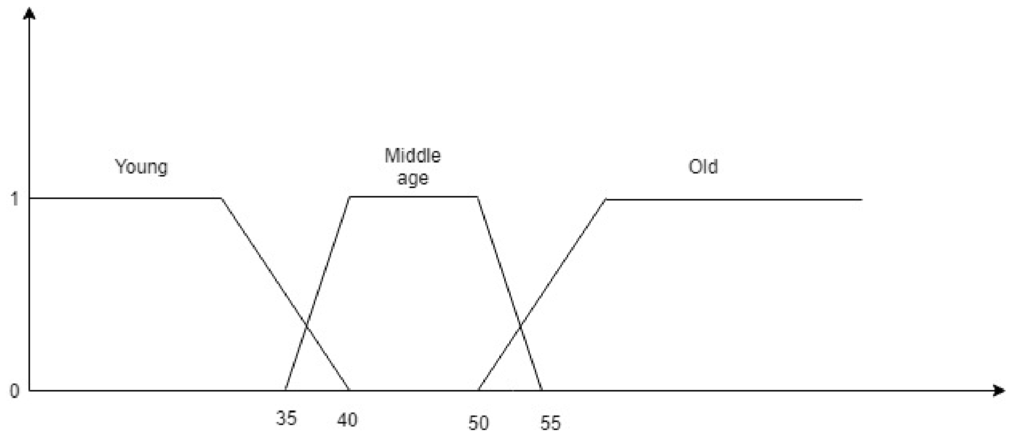

Figure 2.1: Representation of fuzzy data

As can be seen from Figure 2.1 and 2.2 three linguistic tags are young, middle age and old. The membership function has been defined over the domain $\{13, 14, , 15, .., 90\}$

and can be shown as $\{0, 35, 40\}$, $\{35, 40, 50, 55\}$ and $\{50, 55, 90\}$ respectively. Each of those values $\{u_1, u_2, ..., u_n\}$ is linked to an attribute in the database and XML file. In terms of database or XML each attribute has a data type the domain. Domain is the possible value that each attribute can obtain[18].

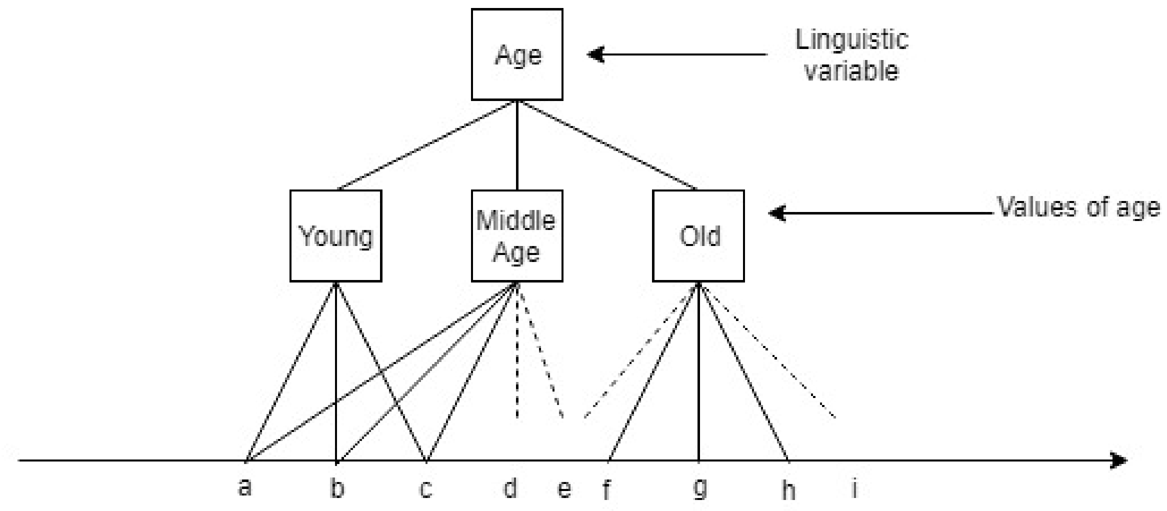

Figure 2.2: Linguistic Data Model

In traditional data model systems only some atomic data like an integer or real for each attribute are provided. However, in modern database systems, which may have complex values like collection and defined data type and fuzziness, can be in all data types [18].

2.3.1 Fuzzy Atomic Data Type

Atomic data is the basic data type defined by the Data Base Management System (DBMS)[18]. However, every system and database has its own data type like integer, string, float, boolean etc. Assume a university acceptance system based on an entrance exam where {90,Accepted}, {70,Conditional Accepted}, {60,Failed}. These fuzzy subset is a possibility distribution that is defined for Accepted, Conditional Accepted and Failed.

2.3.2 Fuzzy Collection Data Type

According to [18] Collection data type are used to represent a collection of data for an attribute. The subsets of collection data type can be an atomic data or another collection set. Assume that we have a set value {very slow, slow, fast, very fast} and the numerical value representing of the set has fuzzy numerical value. Here the linguistic tags are very slow, slow, fast and very fast and it is obvious that fast can be interpreted as modified fuzzy set of very fast. Then its membership function can be obtained through the membership function of very fast[18].

2.3.3 Fuzzy Defined Data Type

Based on information in [18] we have two kinds of defined data types: reduced defined data type and aggregative defined data type. In reduced defined data type we put some constrain on the underlying data type. For example we can say the speed of a car can be minimum zero and maximum 300 km/h. Aggregative defined data type tries to demonstrate the real object of the world and it contains attributes and each attribute has a value range and the fuzziness of aggregative data type come from those value ranges [18].

2.4 Mathematical Operations on Fuzzy Sets

Assume we have two fuzzy sets $\widetilde{A}$ and $\widetilde{B}$, the universe of information U and element u belongs to U.

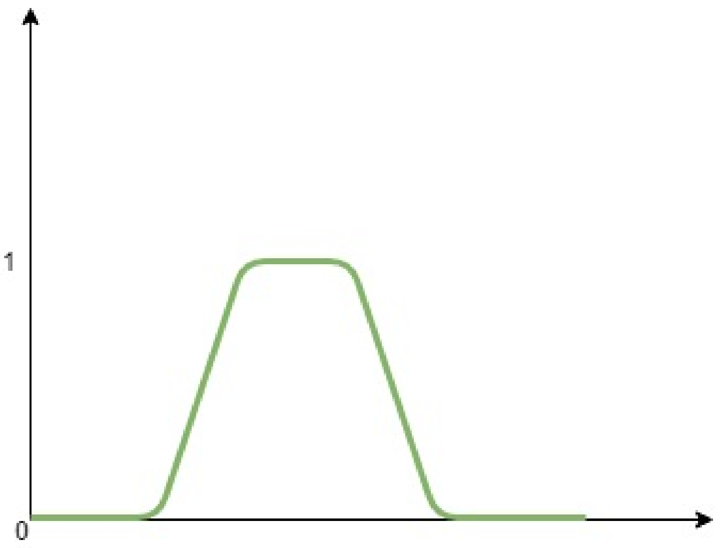

Figure 2.3: Fuzzy set $\widetilde{A}$

Union, exclusive OR, of two fuzzy sets $\widetilde{A}$ and $\widetilde{B}$ can be defined as follows [17]:

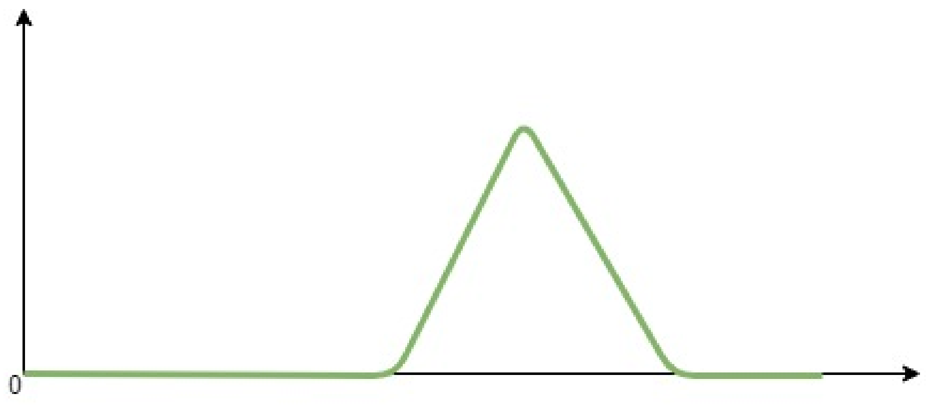

Figure 2.4: Fuzzy set $\widetilde{B}$

2.4.1 Union

$$\mu_{\widetilde{A}\cup\widetilde{B}}(U) = Max(\mu_{\widetilde{A}}(U), \mu_{\widetilde{B}}(U))\forall u \in U \tag{2.5}$$

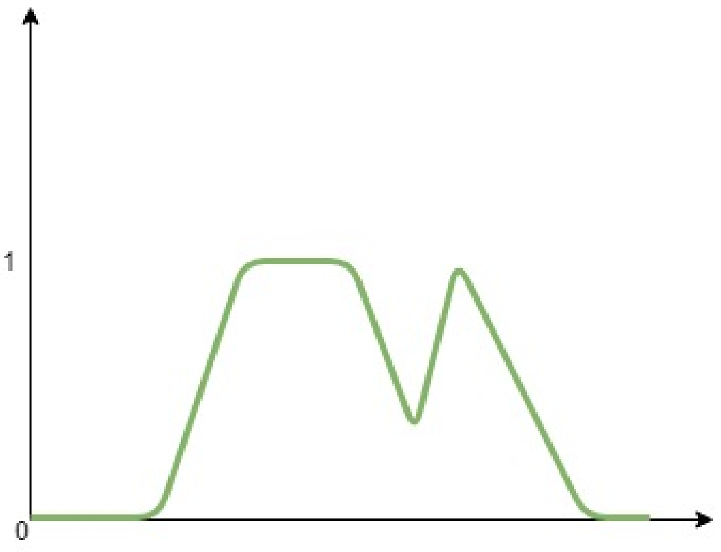

Figure 2.5: Union of fuzzy set $\widetilde{A}$ and fuzzy set $\widetilde{B}$

2.4.2 Intersection

Assume that we have two fuzzy sets $\widetilde{A}$ and $\widetilde{B}$ like above, then we can say that the intersection or logical And of two fuzzy sets $\widetilde{A}$ and $\widetilde{B}$ can be defined as follows [17]:

$$\mu_{\widetilde{A}\cap\widetilde{B}}(U) = Min(\mu_{\widetilde{A}}(U), \mu_{\widetilde{B}}(U))\forall u \in U \tag{2.6}$$

$$\mu_{\widetilde{A}\cap\widetilde{B}}(U) = Min(\mu_{\widetilde{A}}(U), \mu_{\widetilde{B}}(U))\forall u \in U \tag{2.7}$$

In the following figure it is shown graphically the intersection of two fuzzy sets.

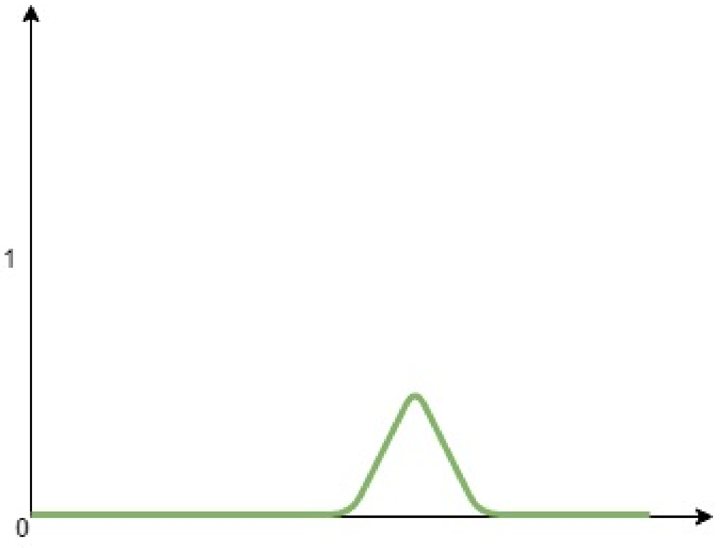

Figure 2.6: Intersection of fuzzy set $\widetilde{A}$ and fuzzy set $\widetilde{B}$

2.4.3 Complement

The membership function of complement can be defined as follows [17]:

$$\mu_{\overline{\widetilde{A}}}(U) = 1 - \mu_{\widetilde{A}}(U) \forall u \in U \tag{2.8}$$

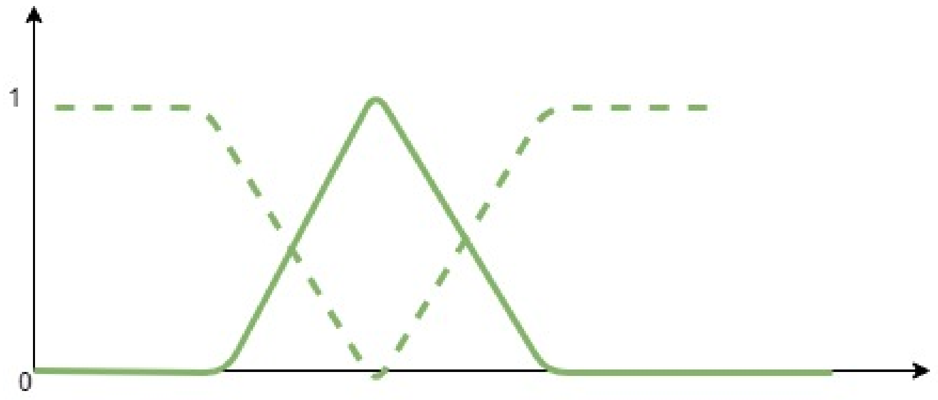

Figure 2.7: Complement of fuzzy set $\widetilde{C}$

2.5 Properties of Fuzzy Sets

Based on the research of [20] the fuzzy sets have the following properties:

2.5.1 Commutative Property

Assume that we have two fuzzy sets $\widetilde{A}$ and $\widetilde{B}$ then we can write following equations:

$$\widetilde{A} \cap \widetilde{B} = \widetilde{B} \cap \widetilde{A} \tag{2.9}$$

$$\widetilde{A} \cup \widetilde{B} = \widetilde{B} \cup \widetilde{A} \tag{2.10}$$

2.5.2 De Morgans Law

$$\overline{\widetilde{A} \cup \widetilde{B}} = \overline{\widetilde{B}} \cap \overline{\widetilde{A}} \tag{2.11}$$

$$\overline{\widetilde{A} \cap \widetilde{B}} = \overline{\widetilde{B}} \cup \overline{\widetilde{A}} \tag{2.12}$$

2.5.3 Associative Property

Having three fuzzy sets $\widetilde{A}$, $\widetilde{B}$ and $\widetilde{C}$ then we can write following equations:

$$\widetilde{A} \cap (\widetilde{B} \cap \widetilde{C}) = (\widetilde{A} \cap \widetilde{B}) \cap \widetilde{C} \tag{2.13}$$

$$\widetilde{A} \cup (\widetilde{B} \cup \widetilde{C}) = (\widetilde{A} \cup \widetilde{B}) \cup \widetilde{C} \tag{2.14}$$

2.5.4 Transitive Property

$$If \widetilde{A} \subseteq \widetilde{B} \subseteq \widetilde{C}, then \widetilde{A} \subseteq \widetilde{C} \tag{2.15}$$

2.5.5 Distributive Property

$$\widetilde{A} \cap (\widetilde{B} \cup \widetilde{C}) = (\widetilde{A} \cap \widetilde{B}) \cup (\widetilde{A} \cap \widetilde{C}) \tag{2.16}$$

$$\widetilde{A} \cup (\widetilde{B} \cap \widetilde{C}) = (\widetilde{A} \cup \widetilde{B}) \cap (\widetilde{A} \cup \widetilde{C}) \tag{2.17}$$

2.5.6 Identity Property

Assume that we have a fuzzy set $\widetilde{A}$ and universal set U.

$$\widetilde{A} \cup U = U \tag{2.18}$$

$$\widetilde{A} \cap U = \widetilde{A} \tag{2.19}$$

2.6 Dempster Shafer Theory

The Dempster-Shafer theory (DST) can be interpreted as the theory of belief functions and the generalization of Bayesian theory of subjective probability. And DST is a general framework for dealing with uncertainty [21]. It is a theory of evidence because it deals with weights of evidence. It is also a theory of plausible reasoning because it works with combination of evidence [22].

Assume that we have a set of n elemental propositions,$\{a_1, a_2, a_3\}$ (frame of discernment). A proposition can be $\{a_1 \cap a_2 or a_1 \cap a_2 \cap a_3\}$ [22]. Here we can say that $\Theta(\Theta = \{a_1, a_2, a_3\})$ is representing all possible facts of a system and 2^Θ is the subset of Θ including the empty set $\{\emptyset\}$, $2^\Theta = \{\{a_1, a_2, a_3\}, \{a_1, a_2\}, \{a_2, a_3\}, \{a_1, a_3$
$\}, \{a_1\}, \{a_2\}, \{a_3\}\{\emptyset\}\}$. The theory of evidence assigns a belief mass to each element of 2^Θ and a mapping $m : 2^\Theta \rightarrow [0, 1]$ is called a basic belief assignment (BBA) [23]. The Belief Mass Assignment (BMA) assigns values $m(\Lambda)$ for all subsets $\Lambda \in 2^\Lambda$ as

following functions [23]:

$$m(\Lambda) \geq 0 \tag{2.20}$$

$$m(\emptyset) = 0 \tag{2.21}$$

$$\sum_{\Lambda \in 2^{\Theta}} m(\Lambda) = 1 \tag{2.22}$$

As we can see from above equations the mass function should always be equal or bigger than zero and the mass function of an empty set is zero. Also it is obvious from Equation 2.24 that the sum of all masses of all subsets of 2^{Λ} should be one.

2.6.1 Belief and plausibility

The belief function (Bel) over $2^{\Lambda} \to [0, 1]$ can be calculated from the basic probability assignment m [24] as follows:

$$Bel(\Lambda) = \sum_{B|B \subseteq \Lambda} m(B) \tag{2.23}$$

Belief in a hypothesis is the sum of the masses of all subsets of the hypothesis set. Belief measures the strength of the evidence in favor of a proposition p. It ranges from 0 to 1 [25]. Also the plausibility function (PI) over $2^{\Lambda} \to [0, 1]$ is defined as follows:

$$PI(\Lambda) = 1 - Bel(\bar{\Lambda}) \tag{2.24}$$

$$PI(\Lambda) = \sum_{B|B\cap\Lambda\neq\emptyset} m(B) \tag{2.25}$$

As can be seen from above equations Plausibility is equal to one minus the sum of the masses of all sets. In other words plausibility is the sum of the masses of all sets whose intersection with the hypothesis is not empty. Like belief, the plausibility also ranges from 0 to 1. Plausibility (PI) is an upper bound and belief (Bel) is the lower bond on the possibility that the hypothesis could be true [25].

$$Bel(\Lambda) \leq P(\Lambda) \leq Pl(\Lambda) \tag{2.26}$$

If we have the Bel(B) for all subsets of B of Λ then the mass of $m(\Lambda)$ can be calculated as follows:

$$m(\Lambda) = \sum_{B|B\subseteq\Lambda} (-1)^{|\Lambda-B|} Bel(B) \tag{2.27}$$

For example, assume that we witnessed a car accident in the street and we have a belief of 0.4 and a plausibility of 0.6 that the driver of red car is dead. This means that we have strong facts that allow us to say the driver of red car is dead with the confidence of 0.4. However, there is a possibility of 0.3 that the driver of red car is alive. Also we have a mass of 0.3 (because the probability should be added up to one) that the driver could either be dead or alive.

Table 2.2: Dempster Shafer Rule Table

Hypothesis	**Mass**	**Belief**	**Plausibility**
Null (neither alive nor dead)	0	0	0
Alive	0.3	0.3	0.4
Dead	0.4	0.4	0.6
Either (alive or dead)	0.3	1	1

It can be seen from Table 2.2 that the sum of mass functions is equal to one while the sum of belief and plausibility do not have to be one and the null hypothesis is zero. Also the universal hypothesis Either is always one. Whereas the hypothesis of Alive and Dead have probabilities of 0.3 and 0.4, respectively.

2.6.2 Dempster's rule of combination

Assume that we have two independent sets of probability mass assignments for a specific problem and we want to combine those independent mass assignments. Different sensors can express their beliefs over specific event based on their interpretation and Dempster's rule of combination can be used as fusion algorithm. This rule focuses on common shared belief between multiple sources and ignores all the conflicting. The combination of two masses m_1 and m_2 is calculated as follows [26]:

$$m_{1,2}(\emptyset) = 0 \tag{2.28}$$

$$m_{1,2}(A) = (m_1 \oplus m_2)(A) = \frac{1}{1-K} \sum_{B \cap C = A = \emptyset} m_1(B) m_2(C) \tag{2.29}$$

Where,

$$K = \sum_{B \cap C = \emptyset} m_1(B) m_2(C) \tag{2.30}$$

As can be seen from Equation 2.30 the combination of two empty sets is equal to zero. In Equation 2.32 K is a measure of the amount of conflict between the two mass sets. For multiple independent evidences the combined version can be obtained as follows [26]:

$$m(A) = \frac{1}{1-K} \sum_{\cap A_i = A} \prod_{1 \leq i \leq n} m_i(A_i) for A \neq \emptyset \tag{2.31}$$

$$m(\emptyset) = 0 for A = \emptyset \tag{2.32}$$

Chapter 3

Soft Data Simulating

In this chapter different parts of soft data simulator, supported data types and reports from second hand information are discussed. The soft data simulator is composed of four major parts:

- Event Simulator
- Traffic Simulator
- Sensor Configuration Part
- Sensor Affiliation

As can be seen from Figure 3.1 each part functions independent of each other where the output of each part is the input of soft data simulator. Event simulator and sensor affiliation have their own configuration files. The configuration files and output of each component are in XML format.

There are some challenges in generating soft data including:

- Being realistic
- Working with linguistic base data
- Expressing information in XML format
- Soft data fusion

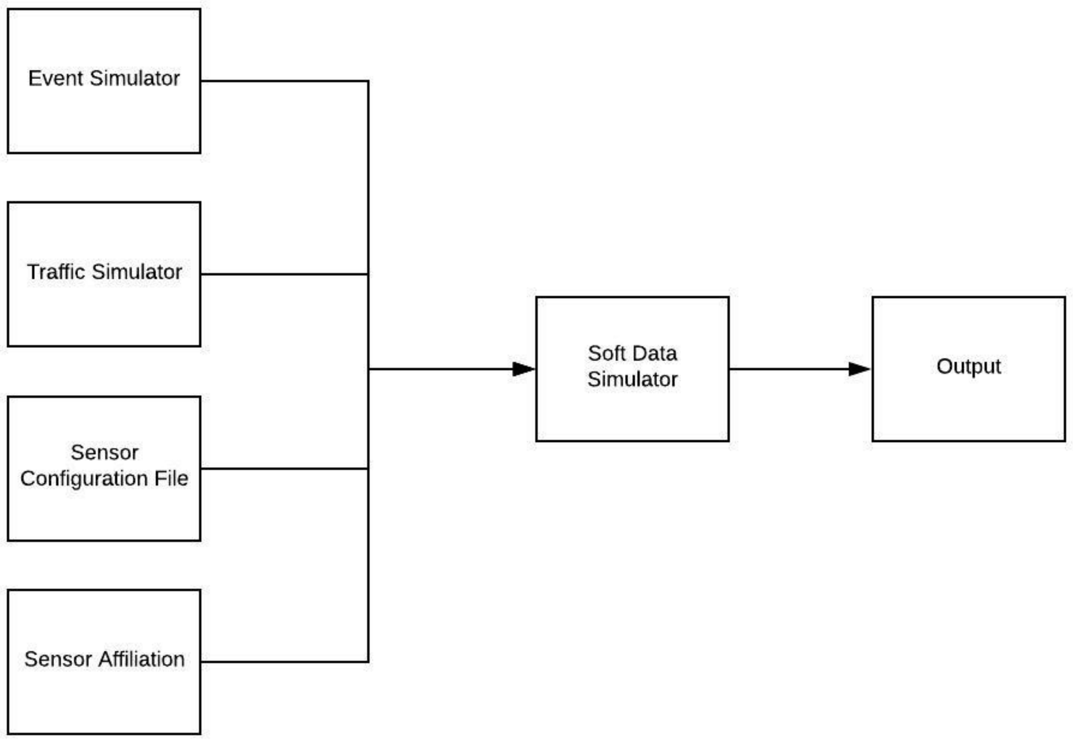

Figure 3.1: Input and output of soft data simulator

Since the generated soft data will be used in different areas like tracking, soft/hard data fusion, data accuracy and anomaly detection, the generated soft data should be as realistic as possible to test on different algorithms. Another challenge here is working with linguistic data type since it is difficult to interpret. For example, if there is an accident on the street, for one person the severity can be catastrophic but for another normal. Expressing the information in XML format is also challenging since the XML files should be as user friendly as possible as well as be able to cover all the information that the soft data simulator needs. Soft data fusion is one of the most challenging part of this application. Since the soft data is qualitative type data it is important to be able to express them in quantitative model. For example, if someone claims that the car is fast, then fast can be interpreted differently from person to person.

3.1 Probabilistic Data in XML

In this section, how the data is stored in and read from XML file is discussed. Traditionally database management systems handle information with zero randomness data. However, recently in most applications there is a need to handle uncertain data too [27]. In this section it is shown how data can be represented and used from a XML file. According to [27], in many domains like scientific areas it is very difficult to have data with 100% of certainty, and errors can not be modeled by using traditional database system. Managing data in XML format has its own challenges. XML file has a structured base format, which sometimes makes it difficult to show probabilistic base data in structured format [27].

3.1.1 Data Representaion

As we talked above XML file format is structured and the structure is variable with a certain probability between 0 and 1 [27]. In this project we put the variables' names in attributes of each level of XMl file and assign a certain probability for each attribute. The nodes probabilities are not independent and it is obvious from the Figure 3.1 that each node probability is dependent to its parent. Each node probability exists if the parent node probability is available. Otherwise the probability of that node is zero [27].

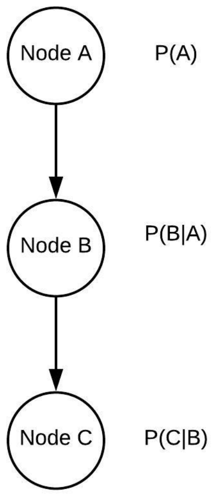

Figure 3.2: XML nodes probability

Here we assume that the main node, node A, has a probability of 1 and the other nodes have conditional probabilities of their parents. For calculating the probability of each node we can use the Bayes formula. For example, assume the chain of nodes as Figure 3.1, the probability of $P(A)$, $P(B)$ and $P(C)$ can be defined as follows:

$$\begin{aligned} &P(A) = 1 \\ &P(B|A) = \frac{p(A|B) \times P(B)}{p(A)} \\ &P(B) = p(B|A) \times P(A) \end{aligned} \tag{3.1}$$

As it can be seen from Equation 3.1 $P(A) = 1$ since it is the main node and $P(B)$

is the conditional probability of $P(B|a)$ times $P(A)$. From the same chain rule the probability of $P(C)$ can be calculated as follows:

$$P(C) = P(C|B) \times P(B|A) \times P(A) \tag{3.2}$$

The probability of any attribute in an XML file can be calculated by multiplying of all the conditional probabilities before that attribute.

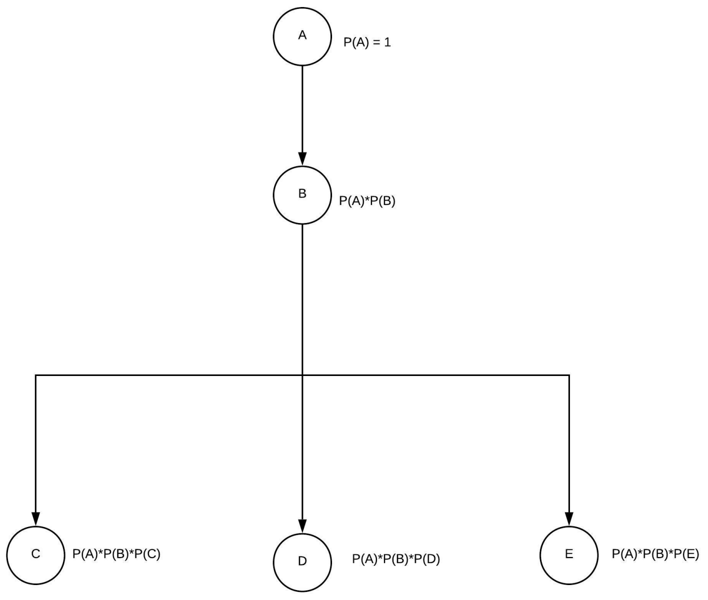

Figure 3.3: Nodes chain probability in XML file

3.2 Event Simulator

Event simulator is a platform for generating events. These events can be car accident, fire accident or any other type of events. The output of event simulator is a XML file with a specific format. The event Simulator uses Open Street Map (OSM) for generating location and time of each event. Event simulator output should contain following information:

- Event name
- Event severity
- Event ID
- Event time configuration
  - Simulation duration
  - Start time
- Event location configuration
  - Latitude
  - Longitude
  - street name
- Event details

  - Brand
  - Color
  - Number of passengers
  - Speed
  - Plate number

Each event has a name and ID. Assume that there are two car accidents in a street. Then we need an event ID for each event to recognize which event sensors are reporting about. Simulation time configuration is used to determine the time of accident and also the time that each person reports of that accident. Event latitude and longitude are used for gating, which is discussed later in this chapter. Event details are the actual measurements that people put in their reports.

3.3 Traffic Simulator

The main reason for using traffic simulator is to provide sensors IDs and trajectories. Usually if sensors are pedestrians or bots the IDs and trajectories are provided immediately otherwise IDs and trajectories first need to be generated for vehicle, drone or anything else that actual sensors are using and then obtain IDs and trajectories of sensors based on their movement. Traffic simulator output file should contain the following information:

- sensors ID
- Sensors latitude
- Sensors longitude
- Time step of sensors movement
- Speed
- Vehicle
  - Vehicle ID
  - Number of sensors in the vehicle
  - Sensors ID

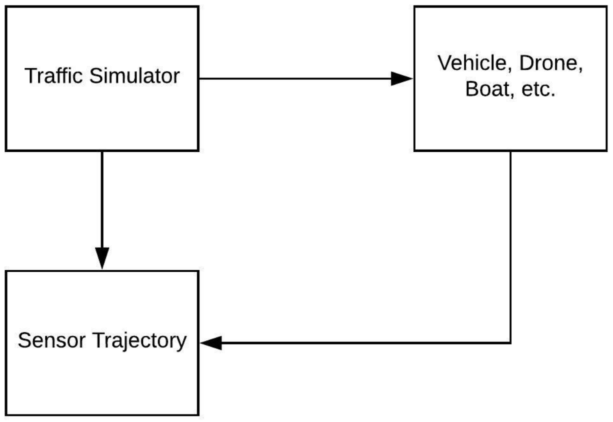

Figure 3.4: Traffic simulator to sensor trajectory block diagram

3.4 Sensor Characteristics

As mentioned earlier we want the simulation to be as realistic as possible. Since a big part of our sensors are people, as in the real world these sensors should also have characteristics like nationality, political view, sex and so on. These characteristics can also help with further statistical analysis over the soft data. For sensor characteristics we have two main XML files. The first one is sensor affiliation file, which acts as a database for all the characteristics that a person can have. The second is configuration file, which assigns some characteristics on sensors based on given probability. The following block of codes shows a sample configuration file. As we can see each affiliation and corresponding child have certain probabilities. It should be considered that the sum of child probabilities should be one. If the sum of probabilities is not one, then then software will normalize them to one.

```
<Sensor-Affiliation-Config>
        <Affiliation type = 'Nationality' prob = '0.9'>
                <Name name = 'Canadian' prob = '0.3'/>
                <Name name = 'Turkish' prob = '0.7'/>
        </Affiliation>
        <Affiliation type = 'Religion' prob = '0.8'>
                <Name name = 'Christianity' prob = '0.6'/>
                <Name name = 'Hinduism' prob = '0.3'/>
                <Name name = 'Buddhism' prob = '0.5'/>
        </Affiliation>
        <Affiliation type = 'Political␣View' prob = '0.9'/>
        <Affiliation type = 'Sex' prob = '0.9'>
                <Name name = 'Male' prob = '0.3'/>
        </Affiliation>
        <Affiliation type = 'Citizenships' prob = '0.8'>
                <Name name = 'Canada,USA' prob = '0.3'/>
                <Name name = 'Canada,France' prob = '0.1'/>
                <Name name = 'USA' prob = '0.1'/>
        </Affiliation>
        <Affiliation type = 'Job' prob = '0.7'/>
                <Name name = 'Photographer' prob = '0.2'/>
                <Name name = 'Engineer' prob = '0.4'/>
</Sensor-Affiliation-Config>
```

3.5 Gating and Timing

If we have N sensors,$\{N_1, N_2, N_3, ..., N_n\}$, and M events,$\{M_1, M_2, M_3, ..., M_m\}$, a certain portion of N sensors can see the event M_1 and certain portion the event M_2 and so on. It means that we need to have a threshold for each event and only sensors inside that threshold can see the event. It worth mentioning that some sensors can be in multiple threshold, which means that those sensors can react to more than one event. The exact location (latitude and longitude) of events can be obtained from the output of event simulator and the trajectories of sensors can be obtained from traffic simulator output file. Since we have all of these coordinates, then we can easily apply gating for each event. The size of threshold is defined by user in sensor configuration file. As we can see from Figure 3.5 the yellow circle indicates the gating circle for an event, where every sensor in the yellow area can see that specific event.

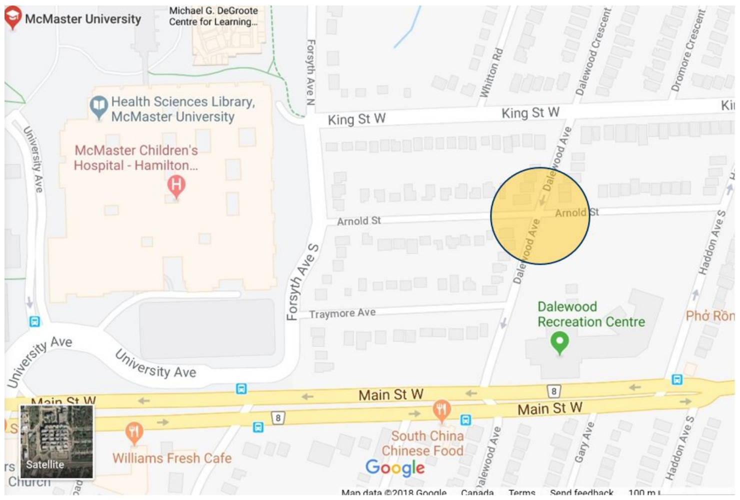

Figure 3.5: Gating area for a specific event

Presumably, not every person who witnessed the accident will report it. only a certain portion of people inside the gate will report the accident. Realistically, not every sensor that witnesses the accident will report it immediately. Usually people want to see the accident as close as possible before reporting it. Consequently, gating time and closest distance are calculated for each sensor inside the gate and report time is always calculated after witnessing the accident from the closest distance.

3.6 Sensor Configuration

Sensor configuration file is the main file for soft data simulator, which is composed of three main parts.

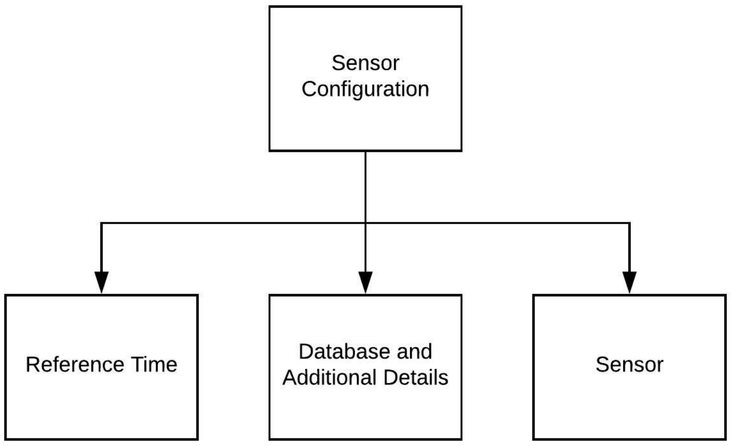

Figure 3.6: Sensor configuration block diagram

The first part is reference time, where the user sets a reference time for the simulator to start. Later, we use this reference time to generate the time of the accident and the report times of all sensors. The second part is database and additional details, which is used to represent the external files and their corresponding addresses. As we can see from Figure 3.6, the database can be color database, model database and so on. Confusion matrix is used for enum type data, which is discussed later.

The third part is divided in to two main part. Event Report Capability and Respond Type and each part has its own subclasses as follow:

- Event Report Capability (ERC)
  - Effective sensors probability
  - Gating distance
- Respond Type
  - Tweet
    - Measurements
    - Measurements type
  - Phone Call
    - Measurements
    - Measurements type
  - Text Message
    - Measurements
    - Measurements type
  - Facebook

∗ Measurements

∗ Measurements type

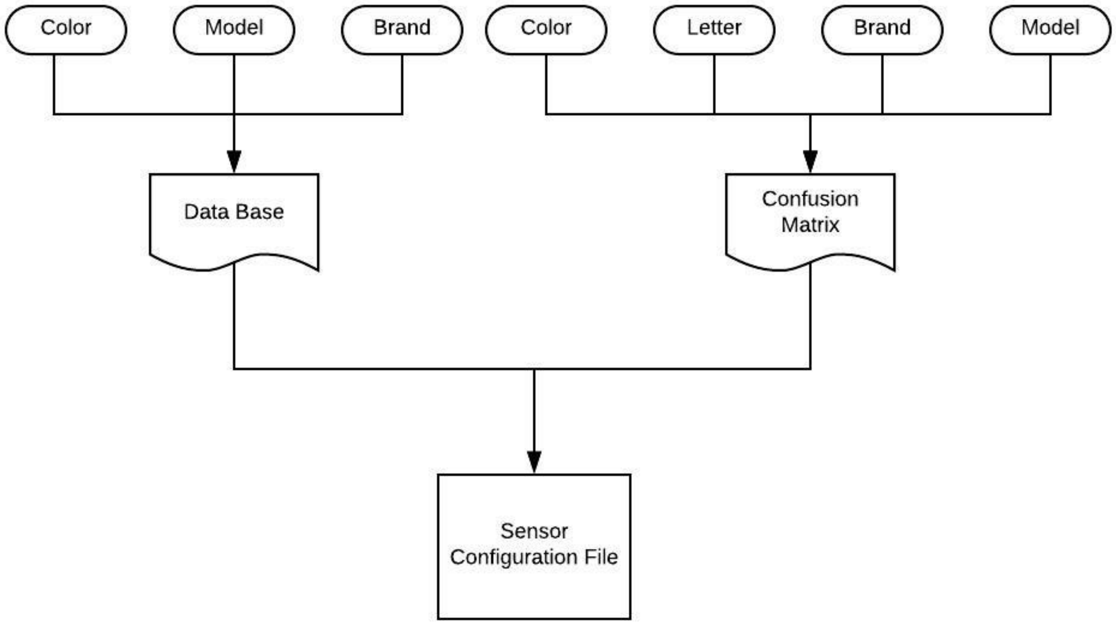

Figure 3.7: Data base and confusion matrix input for sensor configuration file

First, a probability is defined for selecting a portion of sensors that are willing to react to the event and called them Effective Sensors. Then the gating distance, which we talked about in previous sections is defined here for each type of events. Respond type is the way that a person can report an event in real life like tweet, text message and so on. For each response type sensors can report different measurements. For example if the event is car accident, then the sensors can report on brand and model of the cars, number of people involved in the accident or the location of accident. The user can define any measurements that are related to the accident.

3.6.1 Supporting Data Type

Figure 3.8 shows the types of data and the reasons for their use in this simulator.

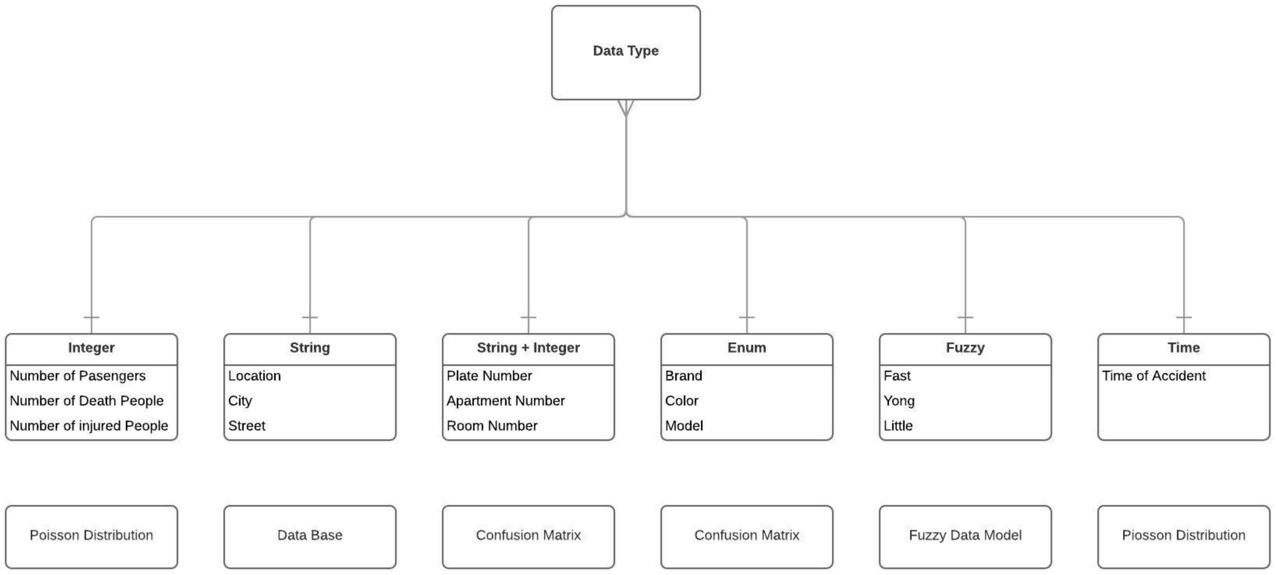

Figure 3.8: Supporting data type

Integer data type is used for reporting integer type measurements like number of passengers in cars, number of people involved in an accident, number of pedestrians or any integer measurement. Assume that there is a fire accident somewhere in the city and n people had died. Suppose that K people (sensors) reported that accident and 70% of sensors mentioned that n people died. The remaining 30% reported the wrong answer, intentionally or accidentally. The problem here is to determine what can be the error and how it can be measured. In terms of being realistic when n people died the wrong answers should probably be around n, which can be modeled by Poisson distribution. According to [28] The poisson distribution can be written as:

$$P = e^{-\lambda}\frac{\lambda^k}{k!} \tag{3.3}$$

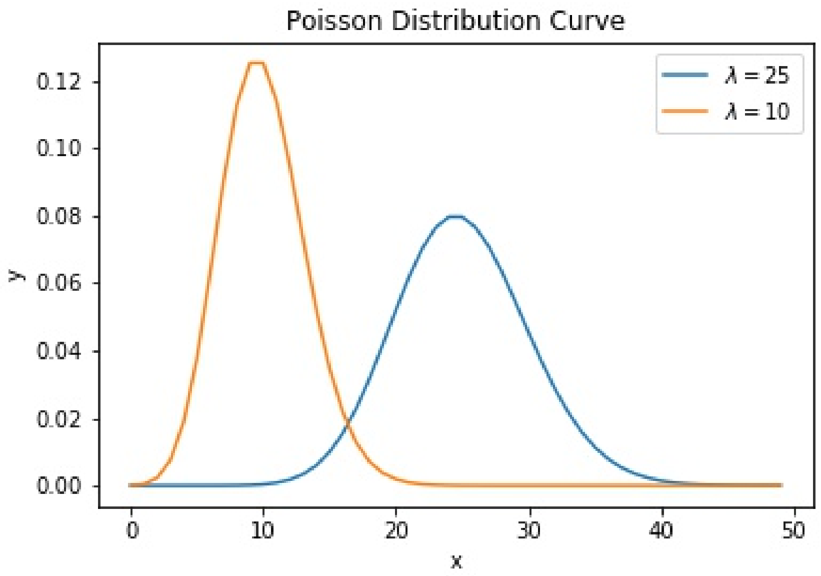

Figure 3.9: Poisson Distribution

Where λ is the average number of event and k = 1,2,3,...,n . Here we assume that λ is the true value of the measurement mentioned in ground truth. For example, If there are m deaths in a specific accident m will be given to the poisson distribution as λ.

In terms of measurement, the value of measurements should always be obtained from ground truth and the ground truth values can be found in the event file. For example, let's assume that there is a car accident and the sensor wants to report the number of passengers in each car. Then we can input the number of passengers as measurement in XML file as the following code. It can be seen from the following code that the measurement name is number of passengers with the probability of 90% and the type of the measurement is integer. Also it is obvious that some sensors, with probability of 60%, will use poisson distribution to generate the measurement value and the remaining 40% will tell the exact value of ground truth.

```
<measurement name="NumberofPassengers"
prob = "0.9" type = "Integer">
        <MeasurmentType Type= "Random" prob = "0.6">
                <Distribution name = "poisson_distribution">
                </Distribution>
        </MeasurmentType>
</measurement>
```

However, in some cases as it is shown in the next block of code that the value of measurement is dependent on the values of other measurements. For example, when a sensor wants to report the total number of passengers, yet the total number of passengers is not available in the event file. In this case the dependent measurements should also be considered in final report. From the code we can see that there is a flag named Sum. If this flag is true it means that there are some dependent measurements and their values should be added to each other for calculating the value of final report for the total number of passengers.

```
<measurement name="TotalNumberofPassengers" prob = "0.6"
 type="Integer" sum="true" SumAttributes="NumberofPassengers">
        <MeasurmentType Type= "Random" prob = "0.3">
                <Distribution name = "poisson_distribution">
                </Distribution>
        </MeasurmentType>
</measurement>
```

The time measurements also use poisson distribution for generating time measurements. We have a reference time in sensor configuration file and start time and duration in the event file. We can use these variables for generating time measurements like, time of accident, and for calculating the report time for each sensors.

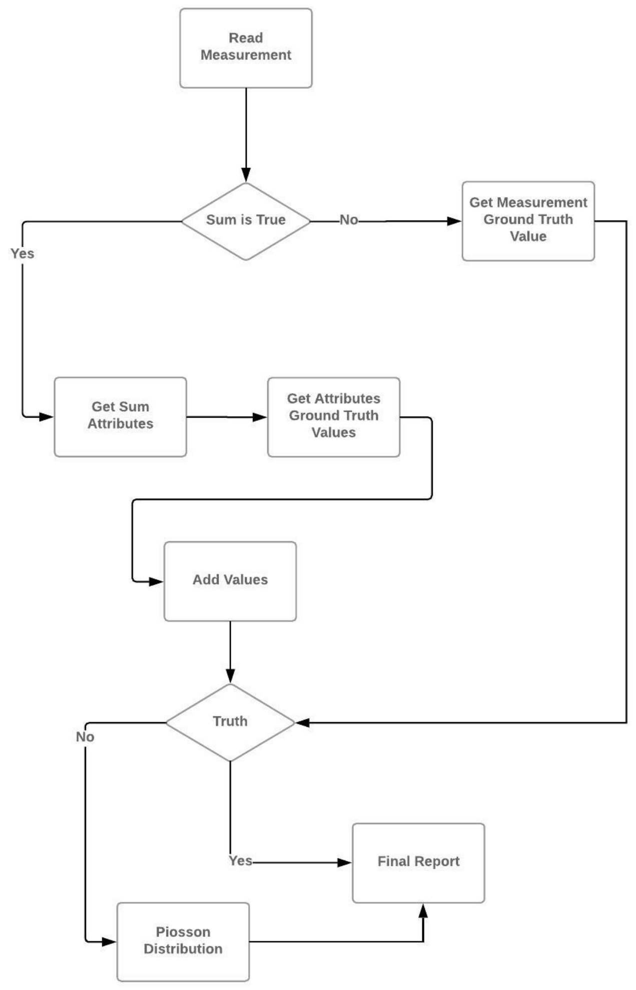

Figure 3.10: Integer type measurements reporting method algorithm

There are also fuzzy data type, which are linguistic but require mapping in the numeric data type. Linguistic data are qualitative and it can be interpreted differently from person to person. However, in terms of simulating we need to express the data as numbers. Although it is difficult to determine the exact value for linguistic data, we can still define some fuzzy interval for linguistic data. We used the fuzzy data method discussed in chapter two for modeling linguistic data. In terms of conflicting membership function we use probability to express the speed.

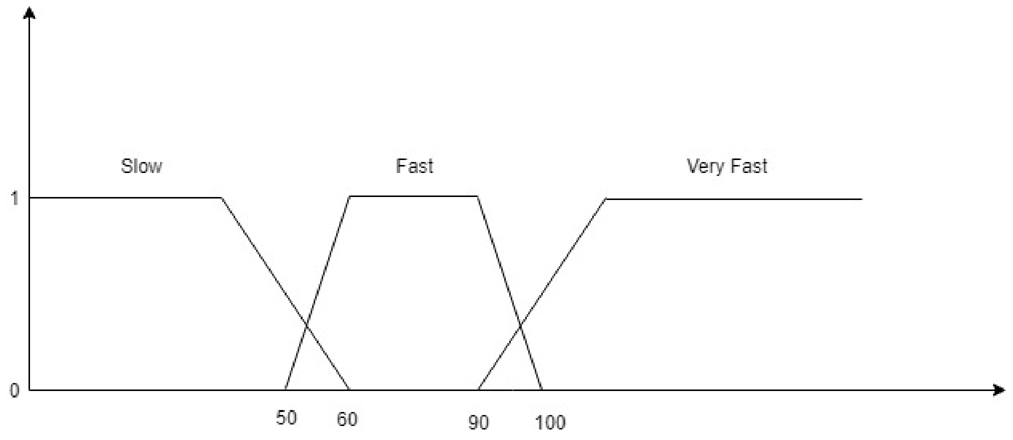

Figure 3.11: Fuzzy type data for speed of cars

There are some linguistic data that have no map in numeric world. We call that kind of data Enum data type and for modeling this type of data we use confusion matrix. We put all the possible values in confusion matrix with a certain probability. Assume that some sensors want to report the color of the cars. If the color of the car involved in the accident is red a portion of sensors will report red. However, not all of the sensors will report red and there is a probability that some sensors intentionally or accidentally report other colors.

```
<confusionMatrix default−diagonal−Values="0.5">
<row value = "Red">
        <column value ="Red" prob="0.5"/>
        <column value ="Reddish" prob="0.2"/>
        <column value ="Kind of red" prob="0.1"/>
        <column value ="Orange" prob="0.1"/>
        <column value ="Yellow" prob="0.1"/>
</row>
<row value = "Black">
        <column value ="Bluish" prob="0.2"/>
        <column value ="Blue" prob="0.1"/>
        <column value ="Red" prob="0.1"/>
        <column value ="Black" prob="0.4"/>
        <column value ="Green" prob="0.1"/>
</row>
<row value = "Blue">
        <column value ="Bluish" prob="0.1"/>
        <column value ="Blue" prob="0.1"/>
        <column value ="Near to blue" prob="0.3"/>
        <column value ="Kind of blue" prob="0.1"/>
        <column value ="Red" prob="0.1"/>
</row>
</confusionMatrix>
```

First, the ground truth value should be compared with the row value of confusion matrix. If there is a match then the confusion matrix columns value will be assigned to sensors with written probability. However, If there is no match then the probability of 50% will be assigned for the ground truth and the others will share the remaining 50%.

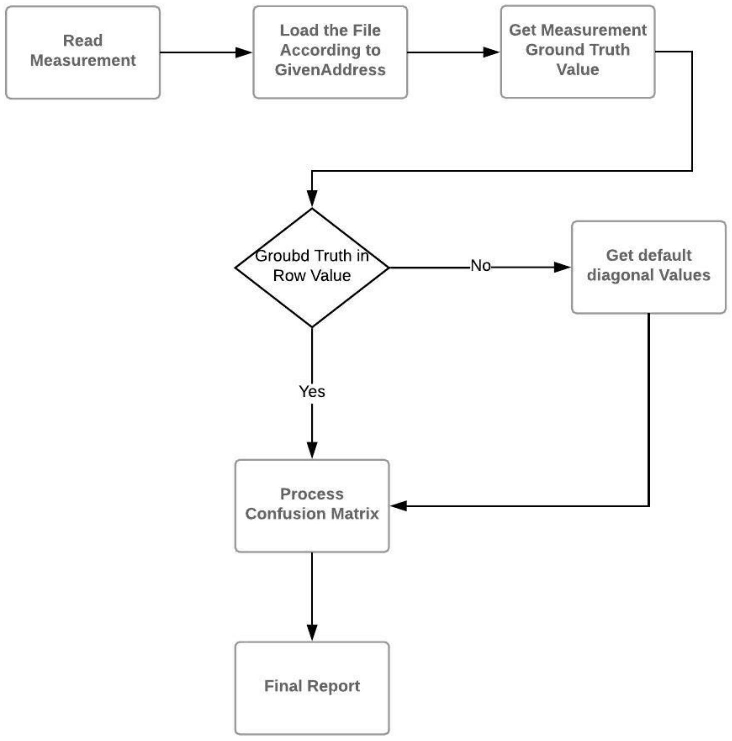
Read Measurement
Load the File According to GivenAddress
Get Measurement Ground Truth Value
Groubd Truth in Row Value
No
Get default diagonal Values
Yes
Process Confusion Matrix
Final Report

Figure 3.12: Proccessing Enum data in confusion matrix

We use string data type for pure string measurements like location of accident. We can get the ground truth of location from the output of event generator. As we can see from the following XML code, the location of the accident is Burnhamthorpe

Road East.

```
<Location>
        <lattitude>43.5016</lattitude>
        <longitude>−79.7383</longitude>
        <node_id>113079172</node_id>
        <way_id>33787169</way_id>
        <way_name>Burnhamthorpe Road East</way_name>
</Location>
```

We used String-Integer data type for expressing the measurements that contains both string and integer like plate number of cars, apartment or suite number. We used confusion matrix for modeling this kind of data. Unlike Enum data type, where we send the whole measurement value to the confusion matrix, in String-Integer we send every letter of measurement value to the confusion matrix. An example of part of the confusion matrix for String-Integer type data is shown blow.

```
<confusionMatrix>
        <row value = "A">
                <column value ="A" prob="1"/>
        </row>
        <row value = "B">
                <column value ="B" prob="0.5"/>
```

```
            <column value ="D" prob="0.2"/>
            <column value ="3" prob="0.3"/>
      </row>
      <row value = "C">
            <column value ="C" prob="0.5"/>
            <column value ="O" prob="0.3"/>
            <column value ="Q" prob="0.1"/>
            <column value ="0" prob="0.1"/>
      </row>
      <row value = "D">
            <column value ="D" prob="0.6"/>
            <column value ="O" prob="0.4"/>
      </row>
<confusionMatrix>
```

It is seen from the above block of code that for the letter A there is no room for error. But for the letter B there is a probability that a person can mistake it with letter D and number 3 with probability of 20% and 30% respectively. For the Letter C it is 50% for C, 30% for O, 10% for Q and 10% for 0. It is obvious from probability theory that the sum of probabilities for all columns should be one. In the XML file if a user put probabilities with the sum different from one, the program will normalize them to one.

3.7 Second Hand Information

So far we discussed sensors that actually see an event or events and try to report them. There is a situation when some sensors do not witness the actual event but still report based on information that they received from other sensors. The ground truth values for second hand sensor are the reports that they received from other sensors. The challenge here is to pick the right value to report. Here we use three different approach to solve this problem.

- Majority
- Accuracy
- Characteristic
- Confusion Matrix

Assume that sensor sh_1 read the reports of senor f_1, f_2, f_3. Report f_1 and f_3 reported that a red car was involved in the accident, while sensor f_3 reported yellow car. Here sensor sh_1 decided to report red car since the majority of sensors reported the red car.

For the accuracy we should have the value of quality of data for each sensor. Let's assume that we are given the quality of data reported by f_1, f_2, f_3 sensors, which are

0.3, 0.7 and 0.2 respectively. In this scenario although most of the sensors reported that the car in accident was red, sh_1 sensors decided to pick the yellow car because he/she trusts the sensor f_2 more than f_1 and f_3. Here the accuracy of report for sensor sh_1 can be calculated by using Dempster's rule of combination, which is discussed at chapter 2.

For the characteristic option assume that s_1 is a man from country A and he is doing job B. s_2 is a female from country C. Second hand sensor sh_2 is a man from country A and he is doing job D. Since second hand sensor sh_1 has more common characteristic with sensor s_1 than sensor s_2 than he trusts s_1 information more.

In the two options that we mentioned above the second hand sensor always chooses one value for each measurement based on other sensors' reports. But during fusion it is possible for a sensor to read from other sensors but report something completely different. For example, we have two sensors s_1 and s_2 as primary senors and sh_1 as second hand sensor. Table 3.1 and Table 3.2 show the confusion matrix for s_1 and s_2 respectively.

Table 3.1: Confusion matrix for s_1

	GroundT1	**GroundT2**	**GroundT3**	**GroundT4**
Report1	$P_{1_{s_1}}$	$P_{2_{s_1}}$	$P_{3_{s_1}}$	$P_{4_{s_1}}$
Report2	$P_{5_{s_1}}$	$P_{6_{s_1}}$	$P_{7_{s_1}}$	$P_{8_{s_1}}$
Report3	$P_{9_{s_1}}$	$P_{10_{s_1}}$	$P_{11_{s_1}}$	$P_{12_{s_1}}$
Report4	$P_{13_{s_1}}$	$P_{14_{s_1}}$	$P_{15_{s_1}}$	$P_{16_{s_1}}$

For doing the data fusion we used Basian Classifier, which has been investigated

Table 3.2: Confusion matrix for s_2

	GroundT1	**GroundT2**	**GroundT3**	**GroundT4**
Report1	$P_{1_{s_2}}$	$P_{2_{s_2}}$	$P_{3_{s_2}}$	$P_{4_{s_2}}$
Report2	$P_{5_{s_2}}$	$P_{6_{s_2}}$	$P_{7_{s_2}}$	$P_{8_{s_2}}$
Report3	$P_{9_{s_2}}$	$P_{10_{s_2}}$	$P_{11_{s_2}}$	$P_{12_{s_2}}$
Report4	$P_{13_{s_2}}$	$P_{14_{s_2}}$	$P_{15_{s_2}}$	$P_{16_{s_2}}$

in [29].

$$P(G|R) = max_c P(G)\Pi_{i=1}^{n} p(A_i|G) \tag{3.4}$$

Where G is ground truth, $A_i = \{Report1, Report2, Report3, Report4\}$ and $P(G) = 1$. Algorithm for soft data fusion can be written as:

- Read $P(Report|GroundTruth)$ value for every sensor from confusion matrix
- Calculate $P(G)\Pi_{i=1}^{n} p(A_i|G)$ for all possibilities
- Normalize the probabilities, which were calculated at previous step

Here Report(m) are the reports of primary sensors s_1 and s_2 and the probability of them are written in the confusion matrix, m = 1,2. $P(G)\Pi_{i=1}^{n} p(A_i|G)$ should be calculated from the confusion matrix too. Then the probabilities should be normalized in cases that the sum of probabilities is not equal to one. At the final step the measurement with the highest probability can be picked as the second hand sensor report or we can have a probabilistic output.

Chapter 4

Simulation and Results

We used C++ for coding the simulator. For this simulation we used three different events with three different severity level. The event output file is composed of three main parts, the first is severity and timing like the following block.

```
<EventOuput name = "CarAccident">
<Severity>Castastrophic</Severity>
<EventID>E1</EventID>
<Timing>
        <start_time>19</start_time>
        <event_duration>100</event_duration>
        <TweetingLatency>
                <distribution>exponential</distribution>
```

```
                    <mean>1</mean>
                    <variance>3</variance>
          </TweetingLatency>
</Timing>
```

The second part is giving information about the location of the accident.

```
<Location>
          <lattitude>43.5016</lattitude>
          <longitude>−79.7383</longitude>
          <node_id>113079172</node_id>
          <way_id>33787169</way_id>
          <way_name>Burnhamthorpe Road East</way_name>
</Location>
```

And the last part represents details about the accident like how many cars or people were involved in the accident, brand name, color, speed of the cars and so on.

```
<VehicleInvolved>
          <Brand>BMW</Brand>
          <Color>Red</Color>
          <Speed>113</Speed>
          <PlateNumber>ANDG1835</PlateNumber>
```

```
        <NumberofPassengers>1</NumberofPassengers>
</VehicleInvolved>
<VehicleInvolved>
        <Brand>SUZUKI</Brand>
        <Color>blue</Color>
        <Speed>111</Speed>
        <PlateNumber>QPTC194</PlateNumber>
        <NumberofPassengers>1</NumberofPassengers>
</VehicleInvolved>
<NumberOfPedestrians>17</NumberOfPedestrians>
```

For event one the severity level is catastrophic, the start time is 19 seconds after starting the simulation and four cars are involved in the accident. For the second event the severity level is normal and begins 7 seconds after starting simulation and six cars are involved in the accident. For the last event the severity level is minor and similar to the previous event six cars are involve in the accident where the start time of accident is 15 seconds after staring the simulation.

The reference time for the simulation is 12-04-2018 10:30:00, I also used 90% of the sensors for the events with the catastrophic severity of 60% and 80% for the normal and minor severity respectively. All of this information is given in the sensor configuration file. Sensor configuration file also shows that sensors use four ways of respond type: tweet, text message, phone call and facebook. All the measurements and probabilities are also given in this file.

We also have sensors(people) trajectories file, which shows in each time step the location of the sensors. We use this file for gating the events and for deciding which sensors can see the events. For the sensor characteristic part we use different categories as follow:

- Nationality
- Religion
- Political View
- Sex
- Citizenships
- Job

However, this does not mean that all the sensors have all the characteristics. Some sensors can have part of them and some may not have them at all. We also have a configuration file that decide what portion of sensors should have which characteristics.

```
<Affiliation type = 'Nationality' prob = '0.9'>
```

```
</Affiliation>

<Affiliation type = 'Religion' prob = '0.8'>

        <Name name = 'Christianity' prob = '0.6'/>

</Affiliation>

<Affiliation type = 'Political View' prob = '0.9'/>

        <Affiliation type = 'Sex' prob = '0.9'>

        <Name name = 'Male' prob = '0.8'/>

</Affiliation>

<Affiliation type = 'Job' prob = '0.7'/>
```

As we can see from the code above the user defines that 90% of the sensors have nationality, 80% have religion, 90% have political view, 90% have sex and 70% have job characteristics. Since the nationality, political view and job are not specified in the configuration file all of them will be taken from the database.

After running the simulation the final output is given like following file. A sensor with ID number 25, used Tweet to report car accident with Catastrophic severity at 10:30:24. According to the tweet four cars were involved in the accident, reported plate numbers of the cars, speed of each car and number of passengers in each car. Total number of people in accident is 29, which includes number of passengers in each car plus number of pedestrians involved in the accident.

```
    <message Type="Tweet">
```

```
<reporter Type="People" ID="25" />
<EventType EventName="CarAccident"
 Severity="Castastrophic" />
<ReportTime>2018-04-12 10:30:24</ReportTime>
<infos>
<info Type="Brand">BMW;Nissan;Suzuki;
Ford</info>
<info Type="PlateNumber">AMOGI985;JCMS336;0PI0194;
HEBS696</info>
<info Type="NumberofPassengers">1;2;1;4</info>
<info Type="TotalPepopleNumberInAccident">29</info>
<info Type="start_time">2018-04-12 10:30:23</info>
<info Type="Speed">Very Fast;Very Fast;Very Fast;Very Fast</info>
</infos>
</message>
```

The report of total people number in accident for sensors 27, 12, 32 and 41 are shown as:

```
<message Type="Facebook">
<reporter Type="People" ID="27" />
<EventType EventName="CarAccident"
```

```
Severity="Normal" />
<ReportTime>2018−04−12 10:31:12</ReportTime>
<infos>
<info Type="TotalPepopleNumberInAccident">12</info>
</infos>
</message>

<message Type="TextMessage">
<reporter Type="People" ID="12" />
<EventType EventName="CarAccident"
Severity="Normal" />
<ReportTime>2018−04−12 10:30:02</ReportTime>
<infos>
<info Type="TotalPepopleNumberInAccident">12</info>
</infos>
</message>

<message Type="PhoneCall">
<reporter Type="People" ID="32" />
<EventType EventName="CarAccident"
Severity="Normal" />
```

```
<ReportTime>2018−04−12 10:32:13</ReportTime>
<infos>
<info Type="TotalPepopleNumberInAccident">16</info>
</infos>
</message>

<message Type="PhoneCall">
<reporter Type="People" ID="41" />
<EventType EventName="CarAccident"
 Severity="Normal" />
<ReportTime>2018−04−12 10:33:01</ReportTime>
<infos>
<info Type="TotalPepopleNumberInAccident">12</info>
</infos>
</message>
```

Second hand sensor sh_2 read above reports and decided to report 12 for total people number in accident because the majority of sensors reported 12.

```
<message Type="TextMessage">
<reporter Type="People" ID="sh_2" />
<infos>
```

```
<info Type="TotalPepopleNumberInAccident">12</info>
</infos>
</message>
```

Sensor sh_1 received information about color of the cars involved in accident from sensor 2_1 and 3_2 as follow:

```
<message Type="TextMessage">
<reporter Type="People" ID="2_1" />
<infos>
<info Type="Color">Red;Green;Red;Yellow</info>
</infos>
</message>

<message Type="PhoneCall">
<reporter Type="People" ID="3_2" />
<infos>
<info Type="Color">Yellow;Green;Green;Red</info>
</infos>
</message>
```

The confusion matrix for sensor 2_1 and 3_2 are given as:

Using the method, which is discussed in section 3.7, second hand sensor report

Table 4.1: Confusion matrix for sensor 2_1

	Red	**Blue**	**Green**	**Yellow**
Red	0.1	0.5	0.1	0.3
Blue	0.2	0.6	0.1	0.1
Green	0.1	0.3	0.4	0.2
Yellow	0.1	0.1	0.1	0.7

Table 4.2: Confusion matrix for sensor 3_2

	Red	**Blue**	**Green**	**Yellow**
Red	0.4	0.1	0.2	0.3
Blue	0.1	0.7	0.1	0.1
Green	0.2	0.1	0.6	0.1
Yellow	0.3	0.2	0.3	0.2

about the color of the cars in accident is given as:

```
<message Type="Facebook">

<reporter Type="People" ID="sh_1" />

<infos>

<info Type="Color">Blue;Green;Green;Yellow</info>

</infos>

</message>
```

4.1 Data Visualization

As we can see from Figure 4.1 most sensors responded to event one rather than other events.

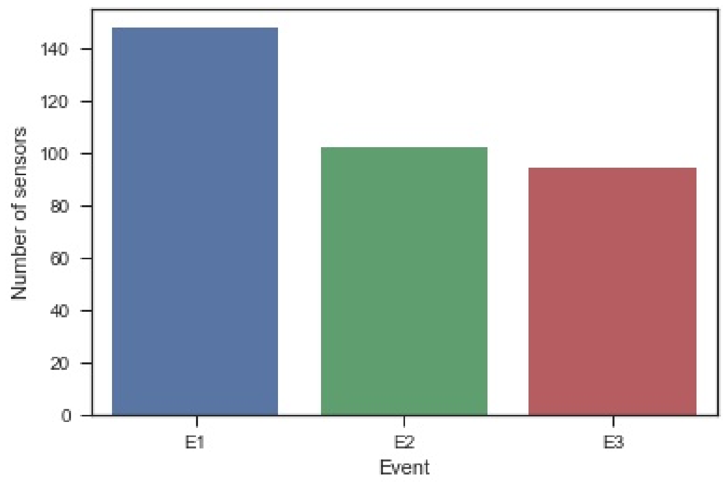

Figure 4.1: Number of sensors for each event

It is seen from Figure 4.2 that in total more men reported in all events than women.

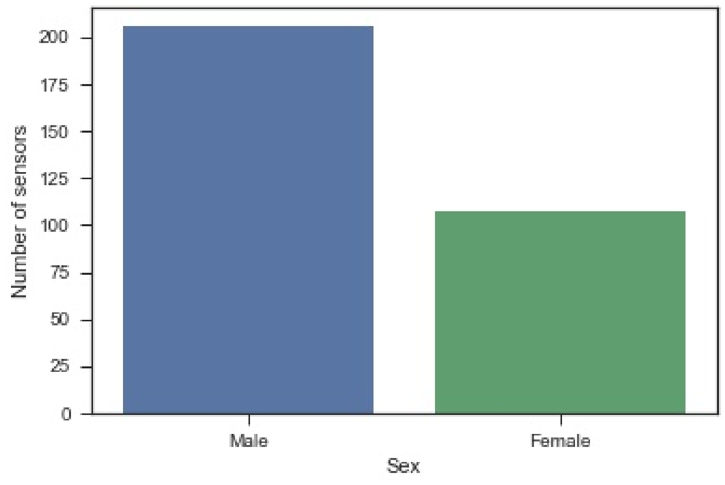

Figure 4.2: Total number of male and female

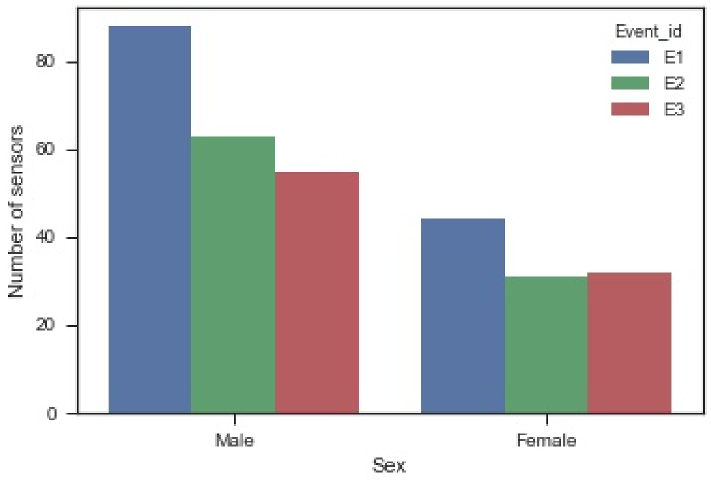

Figure 4.3: Total number of male and female in different events

Most people used Tweeter to report the events, after Tweeter, text message and phone call are the second most popular type of responds with almost same number of responds.

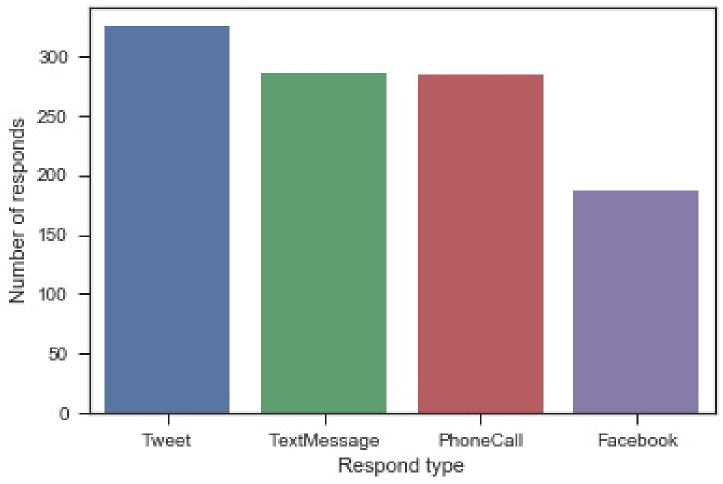

Figure 4.4: Different respond types

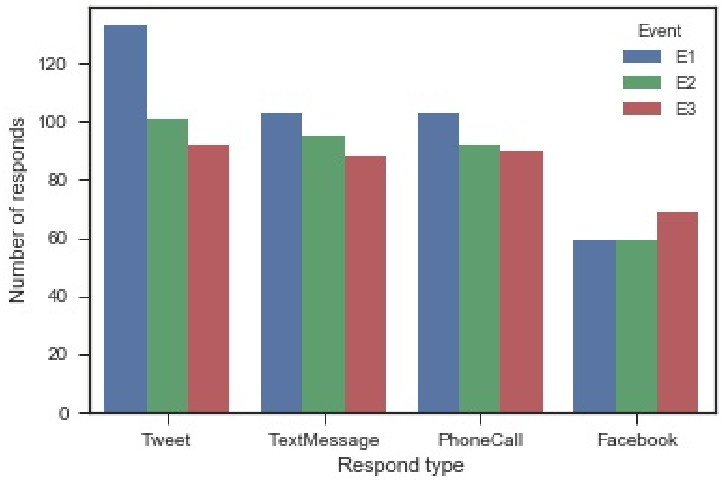

Figure 4.5: Different respond types for each event

Also we can see the people that reported accidents are from Brazil, France, America, Canada, Turkey and Ireland and they are supporting four different political parties, Green party, Liberal, Conservative and Democratic party.

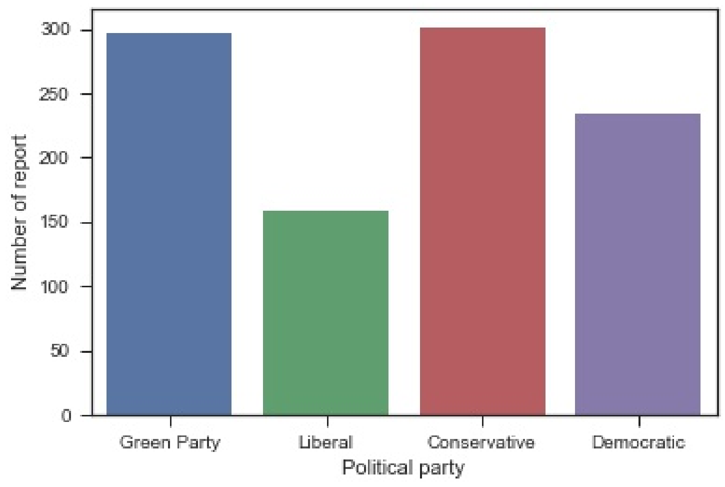

Figure 4.6: Total number of sensors based on political view

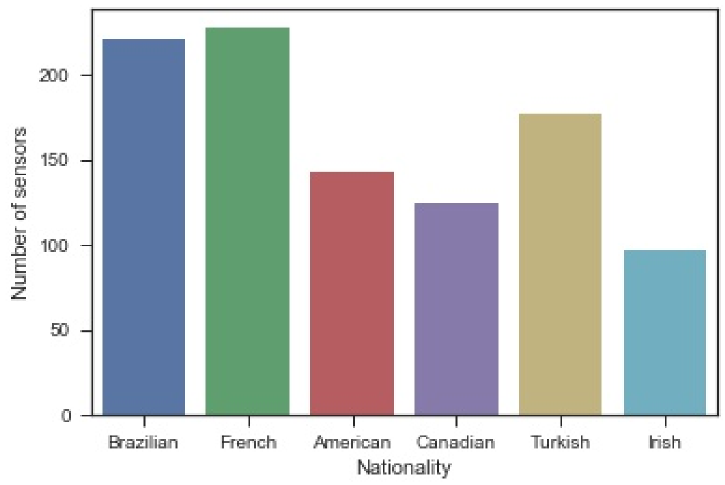

Figure 4.7: Total number of sensors based on nationality

We can also see from Figure 4.8 and 4.9 how people from different countries and different political views used different respond type to report the events. Figure 4.10 shows the relationship between different jobs and respond type. It appears most of the sensors were engineers and photographers.

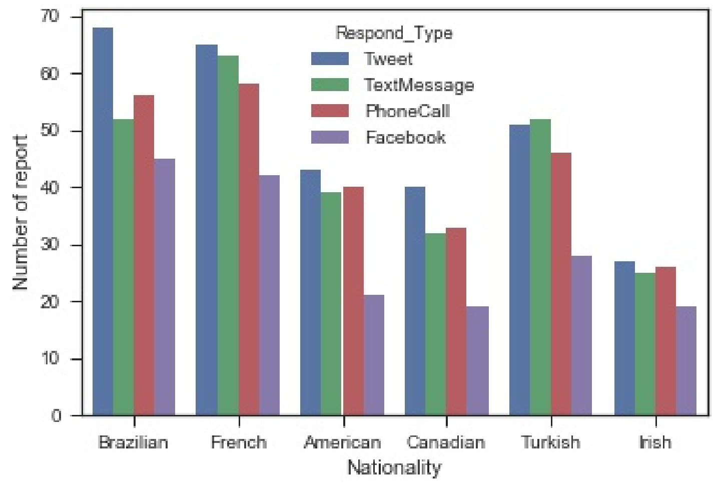

Figure 4.8: Total number of sensors based on nationality and respond type

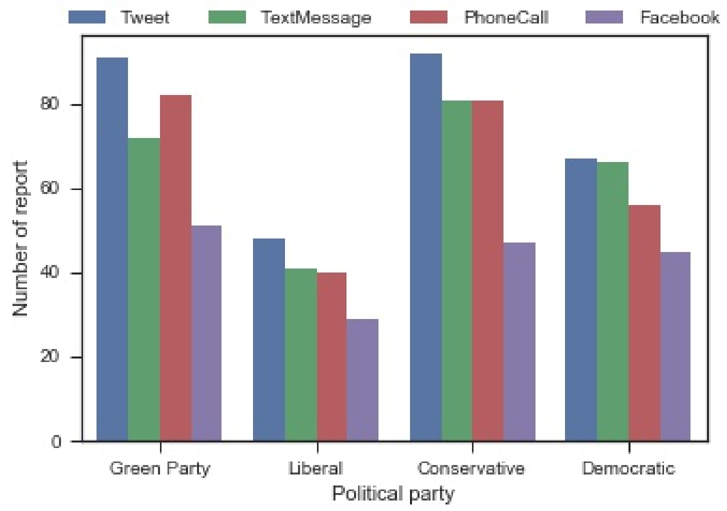

Figure 4.9: Total number of sensors based on political view and different respond type

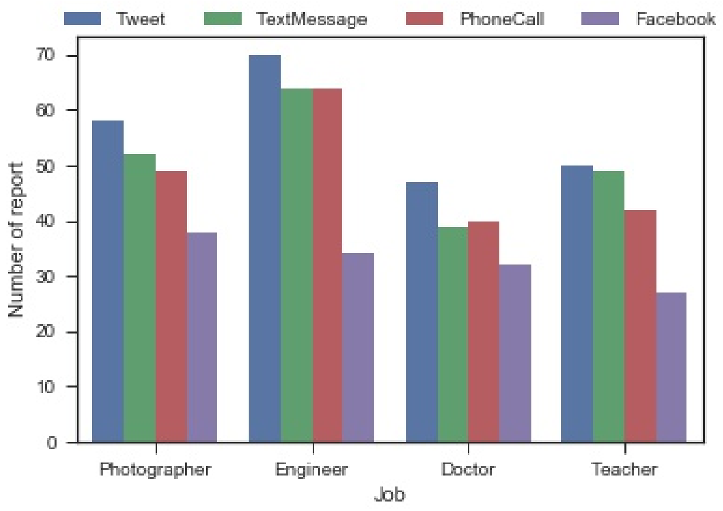

Figure 4.10: Total number of sensors based on different jobs and respond type

As can be seen from Figure 4.10 Tweet and text message are the most popular way of reporting events among the different jobs. Phone call is standing at the third place and Facebook is the less popular one among other type of responses.

4.2 Conclusions

This thesis illustrated a novel way for generating soft data. We supported first hand information, second hand information, sensor characteristics and soft data fusion. We supported the various data type in this project such as fuzzy, ambiguous, integer, string, strig-integer and enum. The simulator supports different kinds of soft sensors like pedestrians, bots, people in vehicle, people in boat and so on. We also tried to support different kind of events like car accident, fire accident etc. The output of the simulator can be used in different areas like soft and hard data fusion, anomaly detection, data quality and data mining.

4.3 Future Work

In this project we used OSM for generating event and trajectories for sensors. Online simulation from camera or picture can be considered for the next step. Generating hard data related to the soft data can be also very useful in terms of data fusion.

Bibliography

[1] Bahador Khaleghi, Alaa Khamis, Fakhreddine O Karray, and Saiedeh N Razavi. Multisensor data fusion: A review of the state-of-the-art. *Information Fusion*, 14(1):28–44, 2013.

[2] Gangqiang Zhao, Xuhong Xiao, Junsong Yuan, and Gee Wah Ng. Fusion of 3d-lidar and camera data for scene parsing. *Journal of Visual Communication and Image Representation*, 25(1):165–183, 2014.

[3] Hyunggi Cho, Young-Woo Seo, BVK Vijaya Kumar, and Ragunathan Raj Rajkumar. A multi-sensor fusion system for moving object detection and tracking in urban driving environments. In *Robotics and Automation (ICRA), 2014 IEEE International Conference on*, pages 1836–1843. IEEE, 2014.

[4] Ronan Collobert and Jason Weston. A unified architecture for natural language processing: Deep neural networks with multitask learning. In *Proceedings of the 25th international conference on Machine learning*, pages 160–167. ACM, 2008.

[5] Shiliang Sun, Chen Luo, and Junyu Chen. A review of natural language processing techniques for opinion mining systems. *Information Fusion*, 36:10–25, 2017.

[6] Scott M Dudek, Alison A Motsinger, Digna R Velez, Scott M Williams, and Marylyn D Ritchie. Data simulation software for whole-genome association and other studies in human genetics. In *Biocomputing 2006*, pages 499–510. World Scientific, 2006.

[7] Mathis Wagner, Malgorzata Zamelczyk-Pajewska, Constantin Landes, Holger Sudhoff, Joanna Kosmider, Tereza Richards, Ulrike-Marie Krause, Robin Stark, Andreas Groh, Frank Weichert, et al. Simulating soft data to make soft data applicable to simulation. *in vivo*, 20(1):49–54, 2006.

[8] Michael Prentice, Michael Kandefer, and Stuart C Shapiro. Tractor: A framework for soft information fusion. In *Information Fusion (FUSION), 2010 13th Conference on*, pages 1–8. IEEE, 2010.

[9] Kedar Sambhoos, James Llinas, and Eric Little. Graphical methods for real-time fusion and estimation with soft message data. In *Information Fusion, 2008 11th International Conference on*, pages 1–8. IEEE, 2008.

[10] Marco A Pravia, Ravi K Prasanth, Pablo O Arambel, Candy Sidner, and Chee-Yee Chong. Generation of a fundamental data set for hard/soft information

fusion. In *Information Fusion, 2008 11th International Conference on*, pages 1–8. IEEE, 2008.

[11] Marco A Pravia, Olga Babko-Malaya, Michael K Schneider, James V White, Chee-Yee Chong, and Alan S Willsky. Lessons learned in the creation of a data set for hard/soft information fusion. In *Information Fusion, 2009. FUSION'09. 12th International Conference on*, pages 2114–2121. IEEE, 2009.

[12] Yuan Long and Xiaolin Hu. Dynamic data driven simulation with soft data. In *Proceedings of the Symposium on Theory of Modeling & Simulation-DEVS Integrative*, page 16. Society for Computer Simulation International, 2014.

[13] Geoff A Gross, Rakesh Nagi, Kedar Sambhoos, Daniel R Schlegel, Stuart C Shapiro, and Gregory Tauer. Towards hard+ soft data fusion: Processing architecture and implementation for the joint fusion and analysis of hard and soft intelligence data. In *Information Fusion (FUSION), 2012 15th International Conference on*, pages 955–962. IEEE, 2012.

[14] Simon Parsons. Current approaches to handling imperfect information in data and knowledge bases. *IEEE Transactions on knowledge and data engineering*, 8(3):353–372, 1996.

[15] ZM Ma and Li Yan. A literature overview of fuzzy conceptual data modeling. *J. Inf. Sci. Eng.*, 26(2):427–441, 2010.

[16] Lotfi Asker Zadeh. Fuzzy logic. *Computer*, 21(4):83–93, 1988.

[17] Lotfi A Zadeh. Fuzzy sets as a basis for a theory of possibility. *Fuzzy sets and systems*, 100:9–34, 1999.

[18] Li Yan. Modeling fuzzy data with fuzzy data types in fuzzy database and xml models. *Int. Arab J. Inf. Technol.*, 10(6):610–615, 2013.

[19] Lotfi Asker Zadeh. The concept of a linguistic variable and its application to approximate reasoningi. *Information sciences*, 8(3):199–249, 1975.

[20] Takashi Mitsuishi, Katsumi Wasaki, and Yasunari Shidama. Basic properties of fuzzy set operation and membership function. *Formalized Mathematics*, 9(2):357–362, 2001.

[21] Glenn Shafer. Dempster-shafer theory. *Encyclopedia of artificial intelligence*, pages 330–331, 1992.

[22] Jeffrey A Barnett. Computational methods for a mathematical theory of evidence. In *IJCAI*, volume 81, pages 868–875, 1981.

[23] Javier Diaz, Maria Rifqi, and Bernadette Bouchon-Meunier. A similarity measure between basic belief assignments. In *Information Fusion, 2006 9th International Conference on*, pages 1–6. IEEE, 2006.

[24] Jeffrey A. Barnett. Calculating dempster-shafer plausibility. *IEEE Transactions on Pattern Analysis & Machine Intelligence*, (6):599–602, 1991.

[25] Glenn Shafer. *A mathematical theory of evidence*, volume 42. Princeton university press, 1976.

[26] Deqiang Han, Chongzhao Han, and Yi Yang. A modified evidence combination approach based on ambiguity measure. In *Information Fusion, 2008 11th International Conference on*, pages 1–6. IEEE, 2008.

[27] Andrew Nierman and HV Jagadish. Protdb: Probabilistic data in xml** work supported in part by nsf under grant iis-0002356. In *VLDB'02: Proceedings of the 28th International Conference on Very Large Databases*, pages 646–657. Elsevier, 2002.

[28] Joachim H Ahrens and Ulrich Dieter. Computer methods for sampling from gamma, beta, poisson and bionomial distributions. *Computing*, 12(3):223–246, 1974.

[29] V Mohan Patro and Manas Ranjan Patra. Augmenting weighted average with confusion matrix to enhance classification accuracy. *Transactions on Machine Learning and Artificial Intelligence*, 2(4):77–91, 2014.