REAL TIME INTELLIGENT TRAFFIC CONTROL OPTIMIZATION

Jayani Chathurangi Withanawasam

149156P

Degree of Master of Science in Artificial Intelligence

Department of Computational Mathematics

University of Moratuwa

Sri Lanka

May 2017

REAL TIME INTELLIGENT TRAFFIC CONTROL OPTIMIZATION

Jayani Chathurangi Withanawasam

149156P

Thesis submitted in partial fulfillment of the requirements for the degree of Master of Science in Artificial Intelligence

Department of Computational Mathematics

University of Moratuwa

Sri Lanka

May 2017

Declaration

I declare that this dissertation does not incorporate, without acknowledgment, any material previously submitted for a degree or a diploma in any university and to the best of my knowledge and belief, it does not contain any material previously published or written by another person or myself except where due reference is made in the text. I also hereby give consent for my dissertation, if accepted, to be made available for photocopying and for interlibrary loans, and for the title and summary to be made available to outside organization.

Name of student: Signature of student

Jayani Withanawasam

Date:

Supervised by

Name of supervisor: Signature of supervisor

Prof. Ashoka Karunananda

Date:

Acknowledgements

I would like to make this opportunity to express my sincere gratitude to my supervisor prof. Asoka Karunananda for his valuable guidance extended throughout the research. This research would not have been completed to success without his immense support and guidance. Further, I would like to thank ASP Mr. T.P. Gamlath, Director of the police CCTV division, Sri Lanka for agreeing to provide video surveillance data in Colombo roads to carry out this research as well as for motivating me by emphasizing the importance of this research. Also, I would like to convey special gratitude to Kaushalya Madhawa and my friends Abi Sachithanantham, Randima Hettiarachchi, Sameera Thilakasiri and Dinesh Asanka for their continuous assistance whenever needed. Last, but not least I would like to thank my parents for their invaluable support and guidance through out my life.

Abstract

Transportation has been a crucial aspect of human life for decades. Road traffic congestion is a critical socio-economic issue as it causes high fuel consumption, waste of time, increased environmental pollution, frustration and safety issues. Major cause for increased road traffic congestion is the sub-optimal use of available resources such as time and road space due to the fixed traffic signal plan. Traffic density at multiple adjacent intersections is not considered in current traffic signal plans.

The said problem is formulated as a distributed constraint optimization problem in this research. Optimized road traffic control mechanism, based on dynamic traffic conditions in multiple adjacent intersections is proposed. The goal is to attain the emerging effect of minimizing the passenger time loss due to traffic congestion of a selected area over time. This is achieved by permitting the optimal green time eligibility for each controlled link in a given intersection during traffic signal phase and cycle length design in a conflict free manner. Multi agent technologies are used to facilitate the communication and coordination between multiple adjacent intersections. Two types of agents namely CCTV agents who monitor the traffic density and traffic signal light agents who control the traffic flow are introduced. Two multi agent based algorithms namely individual lane based traffic signal control and cumulative intersection based traffic signal control are proposed to determine the optimal green time eligibility for a given intersection considering the traffic density of the incoming and outgoing lanes of multiple adjacent intersections. Convolutional Neural Networks (CNNs) is applied on video surveillance data to recognize the traffic density of adjacent intersections.

Real world video surveillance data in Sri Lankan roads are used to evaluate traffic density recognition. Simulation of Urban Mobility (SUMO) traffic simulation suite is used to assess the proposed dynamic traffic control optimization mechanism. Experimental results show that the proposed traffic signal plan effectively reduces the time loss due to traffic congestion for a given area by 34% compared to the fixed traffic signal plan. Accuracy of the traffic density recognition is 72%.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

1.1 Prolegomena

Recently, there is a significant interest in applying artificial intelligence (AI) technologies to solve many multifaceted real world problems that are difficult to solve with conventional techniques. Many AI techniques such as artificial neural network (ANN), multi agent systems (MAS), genetic algorithms, natural language processing (NLP), data mining, and ontology have become popular in terms of solving real world problems. Intelligent Transportation System (ITS) is an active research area that greatly uses artificial intelligence techniques to enable road safety and effective use of available resources. Road traffic congestion is a significant problem for people whom daily travel between different locations. This research is conducted to implement a real time, intelligent traffic control optimization system by effectively utilizing the available artificial intelligence technologies namely artificial neural network and multi agent systems. This chapter presents the importance of the problem domain, background and motivation, novel approach to traffic control, aim and objectives, problem in brief, resource requirements and structure of the thesis.

1.2 Background and motivation

Transportation has been an important aspect of human life for decades. Effective road traffic control system is a vital factor during urban development, as it facilitates efficient control of vehicles and enables road safety. Traffic density is an essential measure to determine the traffic flow. Traffic density is calculated using the number of vehicles per unit length of the road (e.g., vehicles per km). Traffic congestion occurs when the number of vehicles exceeds the available road capacity thus slackening the traffic flow rate and saturating the road space. Major cities all over the world in developing countries experience traffic congestion mainly due to the increased number of vehicles in the urban areas and population growth. According to the department of motor traffic Sri Lanka, the number of registered vehicles is increasing each year from 2011 – 2015 [1]. Further, poorly designed and managed roads and ineffective traffic signal plans increase the traffic congestion. Even though, it is evident that insufficient road infrastructure contributes traffic congestion to a great extent, it is unfeasible for urban

authorities to add more road infrastructure due to various limitations. Increased traffic congestion triggers innumerable socio-economic and environmental issues such as productivity loss due to delays, air pollution due to CO_2 emission of vehicles, accidents because of the limited distance between vehicles, increased fuel consumption and user frustration [2, 3].

In order to address the above issues, traffic signal light systems are deployed in different intersections ensuring regular traffic flow [4]. Further, variable speed limits and different pricing mechanisms for road infrastructure are introduced to regulate the traffic flow [5]. Intelligent transportation systems provide a promising direction to solve the problem of traffic congestion using different sensors such as loop detectors and surveillance cameras. However, the installation and maintenance cost of loop detectors is significantly high. Further, it assumes that vehicles follow a particular lane that doesn't always hold true in developing countries like Sri Lanka. Compared to loop detectors, surveillance cameras are more cost effective and easier to maintain. Further, it provides a wider coverage of the area that needs to be monitored. However, analyzing video surveillance data poses intrinsic challenges such as illumination and occlusion effects, different weather conditions and significantly high processing time [6]. Due to the previously mentioned limitations in the intelligent transportation systems based method for traffic control, respective authorities still use the static traffic signal plans during traffic control.

Traffic density of neighboring intersections that are closely located in urban areas influence each other’s traffic flow. Yet, traffic density information of neighboring intersections is not considered when regulating the traffic flow. Recently, deep learning techniques have proven significant advances in the area of computer vision tasks such as object detection and scene classification. Also, self-organization and emergent features of multi agent systems have enabled managing operations in complex distributed environments such as traffic congestion controlling [7]. Besides, this research is motivated by author’s personal experience as well. Author is currently travelling more than 60km on daily basis. Based on her previous experience, she is using different routes on different times of the day to avoid traffic congestion. Even though humans can achieve this based on their previous experience, current traffic signal system is not yet intelligent enough to coordinate among different traffic signal systems in multiple adjacent intersections, to optimally use the available infrastructure. Further, she has observed how traffic policemen coordinate with each other using walkie-

talkies and simulating similar behavior to automate traffic control is another inspiration for this project. Further, the proposed solution is inspired by a natural phenomenon as well. Water flow rate in channels such as rivers is the amount of liquid volume passed through an area per given time step. Water flow rate is proportional to the slope or gradient of the channel. In this solution the traffic flow is modeled in a similar manner such that traffic flow and traffic density difference between neighboring intersections correspond to water flow rate and the slope of the water channel respectively. Here, the goal is designing an algorithm, which can maximize the traffic flow considering traffic density differences in neighboring intersections. Accordingly, the research focuses on applying deep learning techniques and multi agent technologies to provide intelligent road traffic control optimization.

1.3 Road traffic congestion problem in brief

As stated earlier, fixed traffic signal plan which is currently operated in Sri Lankan roads has several drawbacks such as unnecessary waiting time due to extended red light duration, ineffective green light allocation for empty or less congested roads and unawareness of the traffic density of the destination intersection. In our work, the road traffic signal control optimization is formulated as a distributed constraint optimization problem. Formalization of distributed constraint optimization problem is presented as follows. Let us denote set of n agents as $A = \{a_1, a_2, ..., a_n\}$ and set of n variables that are assigned to each agent as $V = \{v_1, v_2, ..., v_n\}$. Values for the variables should be assigned from finite, discrete domains denoted as $D = \{d_1, d_2, ..., d_n\}$ in a distributed manner. Objective function F(A) that needs to be minimized or maximized is defined as an accumulation over set of constraints or restrictions f_{ij} on variables v_i, v_j , where $f_{ij} : D_i \times D_j \rightarrow N$. Objective function F(A) is formalized as given in Equation 1.1.

$$F(A) = \Sigma_{vi,\, vj \in V} f_{ij}(d_i, d_j), \text{ where } v_i \leftarrow d_i, v_j \leftarrow d_j \text{ in } A \qquad (1.1)$$

In this research, we have applied the above formalism in road traffic signal control optimization. Road traffic networks are inherently distributed. We model a road traffic network as a weighted directed graph G = (X, E), where vertices (X) correspond to intersection agents and edges (E) correspond to connections between intersections. $a_i \mid a_i \in A$ represents one such agent where $i \in X$. Weights correspond to the traffic density. In our context, near optimal traffic signal plans from the domain of all possible traffic signal plans

are given as variables (V). Time loss and number of conflicts between vehicles are introduced as constraints (f_{ij}) that need to be minimized over given variables. The goal is to solve the said problem in a decentralized manner with limited communications between agents.

1.4 Novel approach to real time intelligent traffic control optimization

Deep neural networks are introduced to address traffic density recognition problem using CCTV road traffic videos as input. Multi agent based approach along with swarm intelligence is proposed as traffic control optimization mechanism to simulate the behavior of human traffic policeman. The evaluation results show that the proposed approach is capable of minimizing the time loss for vehicles in a given area thus reducing the road traffic congestion.

1.5 Aim and objectives

The aim is to develop a system for addressing traffic congestion controlling problem with the use of machine learning and multi agent technologies.

1. Study of the available road traffic optimization solutions in intelligent transportation systems (ITS).
2. Study effective means of using machine learning (computer vision) and multi agent systems techniques to solve the road traffic congestion problem.
3. Design and develop intelligent road traffic control system for solving the problem.
4. Evaluation of the proposed solution using video surveillance data for road traffic density recognition along with a simulation approach for road traffic control optimization.
5. Preparation of final documentation.

1.6 Resource requirements

Hardware and software requirements for development of the project as well as execution are given in this section.

1.6.1 Hardware

Computer with at least 16 GB RAM and core i3 processor

1.6.2 Software

Software requirements to develop the project such as programming languages, software frameworks and APIs, software simulators and integrated development environment (IDE) are given below.

- Java 1.8
- Python 2.7
- Eclipse java EE IDE (Kepler service release 2)
- Java Agent Development Environment (JADE)
- Simulation of Urban Mobility (SUMO)
- Caffe deep learning framework (Convolutional Architecture for Fast Feature Embedding)

1.7 Structure of the thesis

Rest of the thesis is organized as follows.

Chapter 2 critically reviews the research area of real time intelligent traffic control optimization with background information. The existing approaches for traffic density recognition and traffic control optimization is discussed along with their limitations. Research gap or research problem and its importance are identified at the end of this chapter to define the research contribution of this project.

Chapter 3 describes the essentials of technologies applicable to solution such as artificial neural networks and multi agent systems along with a justification of choosing the said technologies. Relevance of using deep learning techniques to solve visual pattern recognition problems in intelligent transportation systems is discussed. Also, applicability of multi agent systems to traffic control optimization problem is reviewed.

Chapter 4 presents the novel approach to real time intelligent traffic control optimization. Proposed solution is explained in terms of input, output, process, users and features of the system.

Chapter 5 demonstrates the design of the system and the reasons behind the design decisions taken. Role of components or modules in the projects such as traffic density recognition and traffic control optimization is discussed along with the interactions among them.

Chapter 6 contains implementation of the components of modules given in the design stage. Software, hardware, and algorithms used are mentioned with justifications for selecting them. Pseudo code and important code segments are given whenever necessary.

Chapter 7 reports on evaluation strategy with respect to objectives of the project, experimental setup and experimental results of the new solution with real world simulation.

Chapter 8 interprets the evaluation results and concludes the overall success of the project along with the future work. Also, the limitations of the proposed solution and challenges faced are discussed.

1.8 Summary

This chapter describes the complete picture of the research project. Background of the research, motivation for conducting the research, research problem that is being solved, aim, objectives of the research and the proposed novel approach is explained briefly in this chapter. Resource requirements for this research (both software and hardware) are given. In addition to that, the structure of the thesis is discussed. Next chapter discusses about the state of the art traffic control optimization methods.

Chapter 2

Intelligent Traffic Control Optimization: State-of-the-art

2.1 Introduction

In the previous chapter we discussed the research problem in brief. To start with the research, it is important to find out the current state of the research in the world by reviewing other's work. Advantages and limitations of the current research that is conducted to address the identified research problem are analyzed. After identifying the research gap, the research problem that needs to be addressed is clearly defined. Literature survey was carried out in two aspects. They are,

- Traffic density recognition
- Traffic control optimization

2.2 Traffic density recognition

Different sensor technologies have been used to capture the real time traffic density on roads. Traffic-surveillance information acquired from a variety of sensors deployed across traffic networks is used collectively by exploiting a fusion mechanism in some approaches.

2.2.1 On road sensors

Sensors that are deployed on road itself to detect the number of passing vehicles as well as other traffic parameters such as the traffic flow rate such as loop detectors, geo magnetic detection, inter vehicular communication methods and multimedia sensors are discussed in this section.

2.2.1.1 Loop detectors

Inductive loop traffic detectors are used to detect the vehicles passing by or arriving at a certain point to identify traffic flow and velocity. Inductive loop is embedded in road surface using the magnetic fields of passing vehicles. Embedded inductive loop detector is used along with a speed sensor for traffic density estimation [8]. Lane changing effect of vehicles

during a traffic scenario is also modeled in this approach. It incorporates a markov chain into the state space model to describe the lane-change behavior. Traffic density estimation is achieved using the Kalman filter as signals from sensors are noisy. The advantage of loop detectors is that they can measure traffic condition on any weather or lighting condition. However, installation cost and maintenance cost of loop detectors are relatively high.

2.2.1.2 Geo magnetic detection

Distortion of earth's geomagnetic field due to vehicles' ferromagnetic components is measured in this method. Fusion mechanism is used by exploiting both geo magnetic detection data and floating car data (FCD) [9]. Traffic flow density is measured by considering average velocity (v) and traffic flow (f). Least square support vector regression (LS SVR) is used to enable sensor data fusion in this approach to model velocity, density and flow relation and estimate real time traffic flow density. In this method, the installation cost and maintenance cost is less than loop detectors. However, Sri Lankan urban roads do not have this facility in main intersections.

2.2.2 Inter-vehicular communication

Vehicle to vehicle communication methods based on either using sensors deployed on vehicles or using mobile devices are discussed in this section.

2.2.2.1 Vehicular ad-hoc networks (VANET)

Vehicle to vehicle (V-V) and vehicle to roadside communication is facilitated by VANETs using wireless communication methods (Wi-Fi). It is used as a traffic detection method in intelligent transportation systems [10-12]. Following two parameters are used for traffic density estimation [10].

(1) The number of beacons received by RSUs (Road Side Units)
(2) The roadmap topology/ sj ratio (i.e., streets/junctions)

Regression analysis is used to estimate the number of vehicles per km^2 in urban scenarios, according to the number of beacons received per Road Side Unit (RSU), and the sj ratio (i.e.,

streets/junctions) of the selected roadmap. Traffic density estimation formula is given in Equation 2.1.

$$f(x,y) = a + b \cdot ln(x) + \frac{c}{y} + d \cdot ln(x)^2 + \frac{f}{y^2} + \frac{g \cdot ln(x)}{y} \quad (2.1)$$

where,

X - average number of beacons received by each RSU

Y - sj ratio obtained from the roadmap

Simple count of the number of its neighboring vehicles in each hop in each hop is used to estimate the vehicle density [11]. This is a maximum likelihood estimator of the vehicle density. Vehicle density is modeled using the number of vehicles along the random direction that are directly connected to the chosen vehicle and maximum Euclidean distance from chosen vehicle. Although this provides great advantage in terms of "internet of things (IOT)" concept for intelligent transportation systems, vehicles and other roadside apparatus should be equipped with the required sensors. Uniform sampling of the road sections of interests is used to collect traffic density measurements [12]. Each vehicle is equipped with IEEE 802.11-like interface with nominal transmission range rc, identical for all nodes, and a global positioning system device, such as a GPS receiver thus forming a vehicular ad hoc network (VANET). Sampler is in charge of the following two tasks.

(1) Role switching

(2) Density estimation

During role switching the sampling kit is handed over to more suitable device. Density estimation formula in this approach is given in Equation 2.2.

$$\delta(t_i) = \frac{r+1}{L_s \cdot l} \cdot \quad (2.2)$$

where,

r – number of returned messages from vehicles in target area

Ls - the sampled lane length

l - the number of lanes

Comprehensive review on VANETs for traffic density estimation is available in [13] along with taxonomy graph for traffic density estimation as given in Figure 2.1.

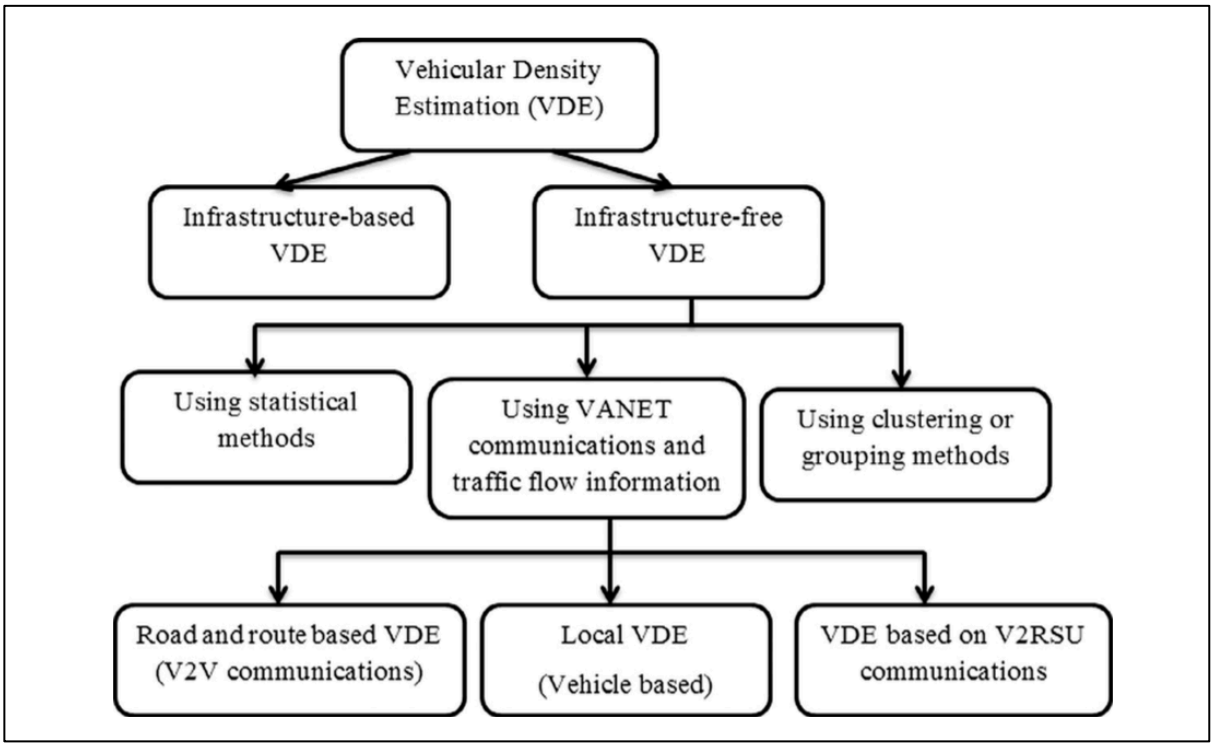

Figure 2.1: Taxonomy graph for traffic density estimation

2.2.2.2 Floating car data (FCD)

GPS data obtained from mobile devices in vehicles is used to detect vehicles' location, direction and velocity details using FCD method [9, 14]. Framework of FCD [9] is given in Figure 2.2.

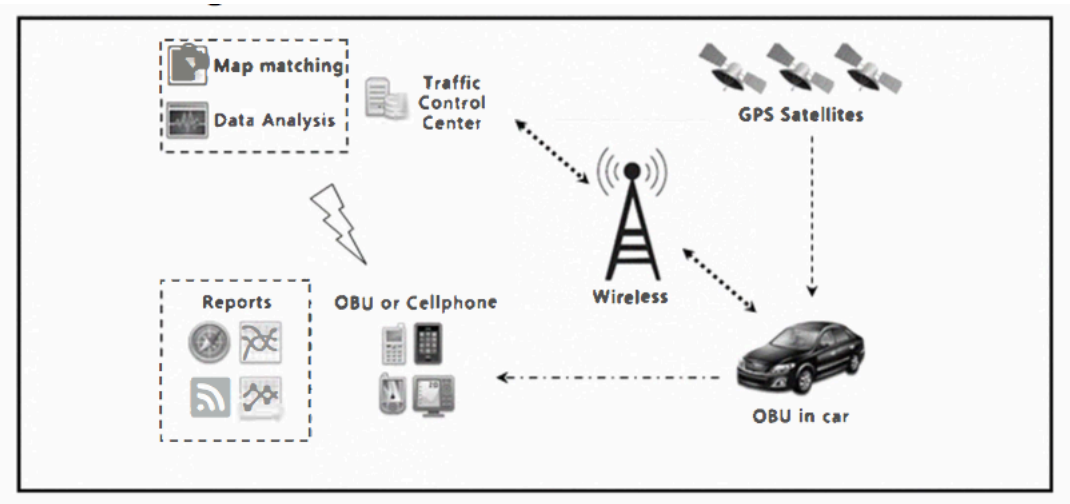

Figure 2.2: Framework of FCD

In [14], kernel-based density estimation method is utilized to extract the congestion spots in road networks based on collected position samples with time stamp from floating car data. Relatively accurate information about vehicle can be obtained using FCD. The drawback of this method is GPS enabled devices should be available for this method to be effective. Also this requires infrastructure-based communication and centralized data processing, which imply high costs, significant computational complexity, and non-negligible latencies.

2.2.3 Multimedia sensors

Multimedia sensors such as audio and video streams are used for traffic density recognition. The extracted features from the above sensors are classified for different traffic density states using a classification algorithm.

2.2.3.1 Audio based sensors

Acoustic signal assimilated from a roadside-installed single microphone is used to estimate traffic density. Tire noise, engine noise, engine-idling noise, occasional honks, and air turbulence noise of multiple vehicles are considered with these approaches. The acquired acoustic signal segments are classified using classification techniques such as Support Vector Machine (SVM) [15, 16] or Neuro-fuzzy classifier [17]. Passages of vehicles are detected using Hidden Markov Model (HMM) [18, 19]. Spectrogram of the different traffic state cumulative acoustic signals during low traffic condition (Figure 2.4) and heavy traffic condition (Figure 2.3) is given [17].

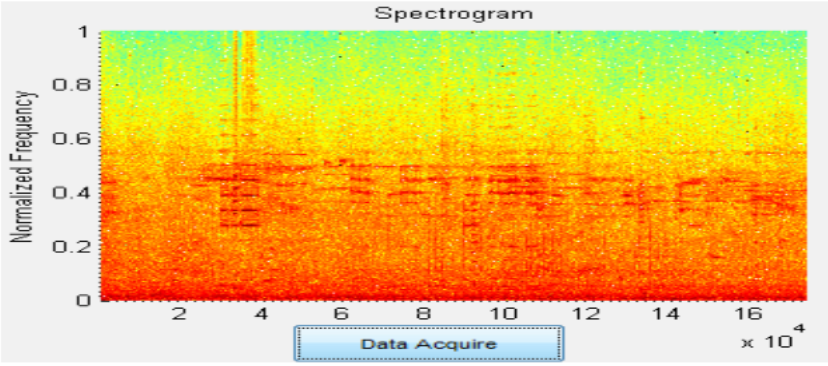

Figure 2.3: Spectrogram of heavy density traffic (0 - 20 km/h)

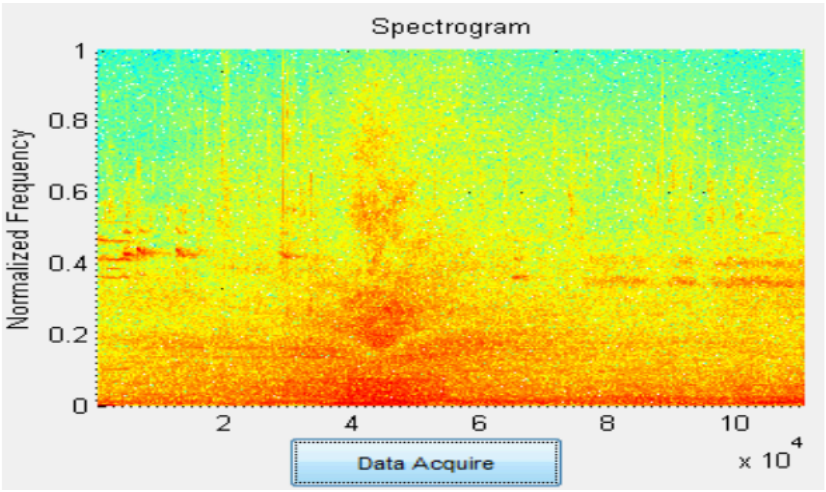

Figure 2.4: Spectrogram of low-density traffic (above 40 km/h)

Chaotic and non-lane-driven city traffic conditions with the extremely varied speed ranges is considered for traffic density estimation [15]. They have tested their classification approach with both Naïve Bayes classifier and Support Vector Machine (SVM). Support Vector Machine has given better accuracy compared to Naïve Bayes classifier when it comes to classifying acoustic signal segments. Mel-Frequency Cepstral Coefficients (MFCC) features are used for classification. SVM and ANN classifiers are used for classification of acoustic signal for traffic density estimation with Mel-Frequency Cepstral-Coefficient (MFCC) features [16]. Adaptive neuro-fuzzy classifier along with Mel-Frequency Cepstral-Coefficient (MFCC) is used for acoustic signal classification [17]. Neural networks algorithms are used to determine parameters of fuzzy system. This work has given better results under non-lane driven and chaotic traffic condition. Occlusion problem due to the overlying of the sounds of

moving vehicles are addressed by corresponding two sets of information from a stereo microphone [18]. This is achieved using recognition of the temporal variations that appear on the power signals when vehicles pass through an observation point. Advantage of using acoustic signals to determine traffic density is, it is independent of lighting condition. However, additional special purpose devices such as microphones need to be installed in required locations in Colombo to enable this approach, which incurs extra labor and installation cost. Also, a sensor covers a limited range in road and accordingly there can be degradation of accuracy.

2.2.3.2 Video based sensors

CCTV surveillance cameras are used to detect traffic density using video and image processing techniques. Figure 2.5 shows different traffic density states in a given Region of Interest (ROI) [20].

Figure 2.5: Different traffic density states from CCTV camera

Image features such as texture, edge, and color are extracted. Pre-processing techniques such as background subtraction, foreground segmentation [21], clustering [20] are used before classification. Support Vector Machine (SVM) [22–24] and K-Nearest Neighbor (KNN) [25] algorithms are used to classify traffic density state (e.g., full, high, low, empty).

Texture features can be used as an effective discriminative feature for traffic density estimation [21]. There, online SVM classifier is used instead of static SVM classifier. Novel background modeling approach is introduced to setup and update its kernel vectors. This is proven to give better results under changing static shadows. Hidden Markov Models (HMM) is combined with unsupervised clustering scheme called Autoclass to determine the traffic density state in a Region of Interest (ROI) of a road in a traffic video [20]. Support Vector Machine (SVM) is used with text features for traffic density recognition [22] . Real time virtual loop detector is proposed using booster SVM classifier with different combination of features such as edge, texture, color and motion [23]. Another SVM based traffic density status classifier is proposed with Invariant Features of Local Textures (IFLT) feature descriptors [24]. Extraction of microscopic features requires detecting and tracking vehicles in the scene individually. Macroscopic features are extracted by analyzing the video scene as a whole [25].

(1) Microscopic parameters: traffic velocity estimation, road occupancy rate and traffic flow
(2) Macroscopic parameters: analyzing the global motion in the video scene

Obtaining CCTV surveillance videos are cost effective and cost less in installation and maintenance. For this project CCTV surveillance camera videos are taken as input due to above mentioned reasons. However, the accuracy can be varying due to different illumination and weather (e.g., fog, rain) effects. Instead of hand engineering features, deep architecture is designed to learn the high level features to obtain structural information of the data in scene recognition [26]. Representations of data is learnt with multiple levels of abstraction [27]. Accordingly, deep learning algorithms, which has been vastly successful in various scene detection and object detection applications [26-28] will be used for traffic density recognition.

2.3 Traffic control optimization

Human policemen optimize traffic control in developing countries. This requires more human labor and inconvenient effort. Therefore, several automated traffic control optimization methods are introduced.

2.3.1 Fixed traffic signal control

Currently, fixed traffic signal control is used in signalized intersections in Colombo, Sri Lanka and many other areas of the world as well. Values for traffic signal parameters such as traffic phases and cycle time are predefined manually based on the historical data. Fixed traffic signal control can be effective if the traffic condition is steady over time. However, usually traffic condition in busy cities is heavily dynamic and uncertain due to situations such as different weather conditions, special events and road accidents.

2.3.2 Adaptive traffic signal control

Consequently, several technologies and approaches are adopted to solve the traffic signal control optimization in the literature.

Fuzzy logic based approach is suggested to optimize average waiting time of a given traffic intersection [29]. In this approach, quantity of the traffic on the arrival side and quantity of traffic on the queuing side are considered as fuzzy input variables. Green time allocation is given as the output. Further, portable fuzzy logic based traffic optimizer is introduced [30]. Number of vehicles accumulated and traffic flow is given as fuzzy input variables and green time allocation is the output of the above approach.

Genetic algorithms are used for traffic control [31, 32]. Traffic signal timing settings such as green time of traffic lights, traffic phase and cycle time are considered as chromosomes or possible solutions in the proposed approaches. Traffic signal control optimization is achieved using a fitness function that minimizes the number of vehicles in the queue or total network wide travel time.

Reinforcement learning (Q-learning) is proposed to yield the minimum possible vehicles in queue for each intersection [33, 34]. The objectives of q-learning algorithm are to gain experience and evolve to make better decisions using its impact on the environment incurred by the previous actions. Level of vehicles in the queue at each intersection is defined as states and increase or decrease of the green time is defined as possible actions. Goal of the reward function is to minimize the number of vehicles in each queue.

Swarm intelligence is another technique that is applied to solve traffic signal control problem. Ant Colony Optimization (ACO) [35, 36] is one such meta-heuristic algorithm that is used to solve computationally hard combinatorial optimization problems. Pheromone concentration in Pheromone trail is analogous to the vehicle waiting time in the proposed methods.

Swarm intelligence techniques such as Ant Colony Optimization (ACO) and Particle Swarm Optimization (PSO) have been applied in intelligent transportation system solutions such as optimal traffic signal timing plan [37], inter vehicular communication [35] and traffic lights scheduling [38]. So, this is identified as a promising technology to optimize traffic signal control.

Multi agent systems technology is also used in previous approaches to solve traffic control problem for multiple intersections [39-41]. Agents are used to effectively communicate and coordinate traffic control based on the traffic congestion levels of each area. However, the agent architecture is given in a hierarchical manner, thus creating a single point of failure as one central agent performs coordination between agents.

2.3.2.1 Limitations of current adaptive traffic control optimization approaches

During the critical review of the existing methods, we have identified the following open research problems that need to be addressed to enable practical use of adoptive traffic signal control methods. In a transportation network, traffic control decision of a given intersection influences the traffic density of multiple adjacent intersections. However, only an isolated intersection is considered in some of the proposed approaches such as fuzzy logic based methods. In such scenario, global optimization cannot be guaranteed based on the local optimizations achieved for the isolated intersections. Further, the solutions based on hierarchical architectures have proposed a centralized control mechanism where agents are not fully autonomous. There exist a single agent (E.g., district agent) that acts as a single point of failure thus degrading the robustness of the solution in a distributed setting. Also, the solutions based on learning components such as swarm intelligence and reinforcement learning require considerable amount of time to converge in to an optimal solution. The delay incurred is not acceptable in real time traffic control environment. Novel multi agent systems and swarm intelligence based approach will be used in this project to optimize the traffic control between multiple intersections.

2.4 Summary

In this chapter literature review was done by giving prominence to two areas, namely traffic density recognition and traffic control optimization. Limitations of the current approaches are identified. Typically, on-road sensors and inter-vehicular communication methods used for traffic density recognition have limitations such as high installation and maintenance cost.

Accordingly, multimedia sensors that are already deployed in urban roads are selected for traffic density recognition. However effectively analyzing data from multimedia sensors such as CCTV surveillance data is still a challenging problem due to illumination and weather effects. Current methods proposed for traffic control optimization such as fuzzy logic and Q-learning based methods do not consider the traffic density of multiple adjacent intersections when determining the traffic signal plan. Next chapter discusses about the technologies used for this research and their applicability to the problem domain in detail.

Chapter 3

Artificial Neural Network and Multi Agent Systems

3.1 Introduction

In the previous chapter, previous work on the research area was critically analyzed to clearly define the research problem. In this chapter, we discuss the technologies used to solve the said problem with justification of their applicability in the selected problem domain. Main artificial intelligence techniques that are being used in this research are discussed in this chapter. Definitions of each technology and features/ characteristics of each technology that make the selected technology favorable to address the identified research problem are also discussed. The selected technologies to solve the two main areas of the research problem are given below.

a. Artificial neural networks for traffic density recognition
b. Multi agent systems and swarm intelligence for traffic control optimization

3.2 Artificial Neural Networks (ANN)

ANN is an emerging AI technology that has shown significant breakthroughs in recent applications in areas such as computer vision and natural language processing. The goal of traffic density recognition is to estimate the traffic density state (i.e.: high or low) when surveillance data is given. Hence the problem of traffic density recognition can be treated as a pattern classification problem in computer vision. Artificial Neural Networks (ANN) is used as the classification algorithm for traffic density recognition. ANN models are inspired by biological neuron in central nervous system and they focus on applying algorithms to mimic neurons. In 1943, Mcculloch and Pitts came up with computational model for neuron as given in Figure 3.1 [42].

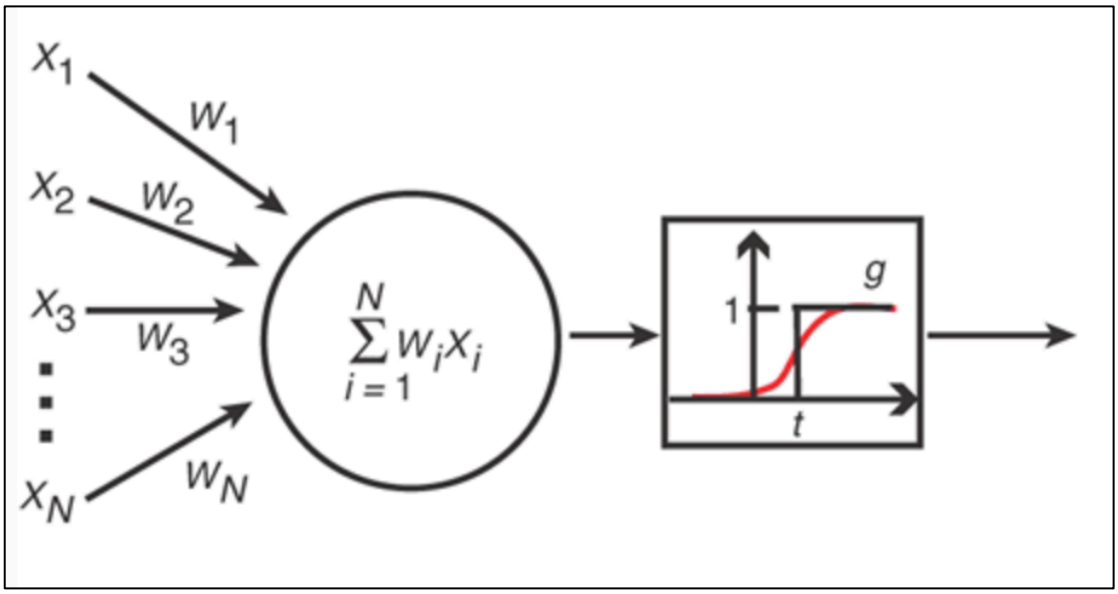

Figure 3.1: Computational model for neuron

The input to a neuron can be received from an external data source or from another neuron if it has another layer of neurons. Each input is weighted by the weights assigned to each input (ie: w_1, $w_2 \ldots w_N$) and added up. Then the output of the neuron is calculated using the activation function. The output of the unit is one, if the total output goes beyond certain threshold or otherwise the output is zero. Activation function defines an output of a neuron. Different activation functions are available as given below.

(1) Step function for classification
(2) Sigmoid function for regression
(3) ReLU (Rectifier Linear Units)

In 1958, Frank Rosenblatt came up with the concept of perceptron. In 1969, Marvin Minsky identified problems with perceptron.

(1) Perceptron can solve only a limited number of linearly separable problems
(2) Perceptron lacks sufficient processing power for its computations

However, many classification problems are non-linear. So, multi layer perceptron or feed forward neural network (Figure 3.2) is introduced with additional hidden layers to solve non-linear classification problems [42].

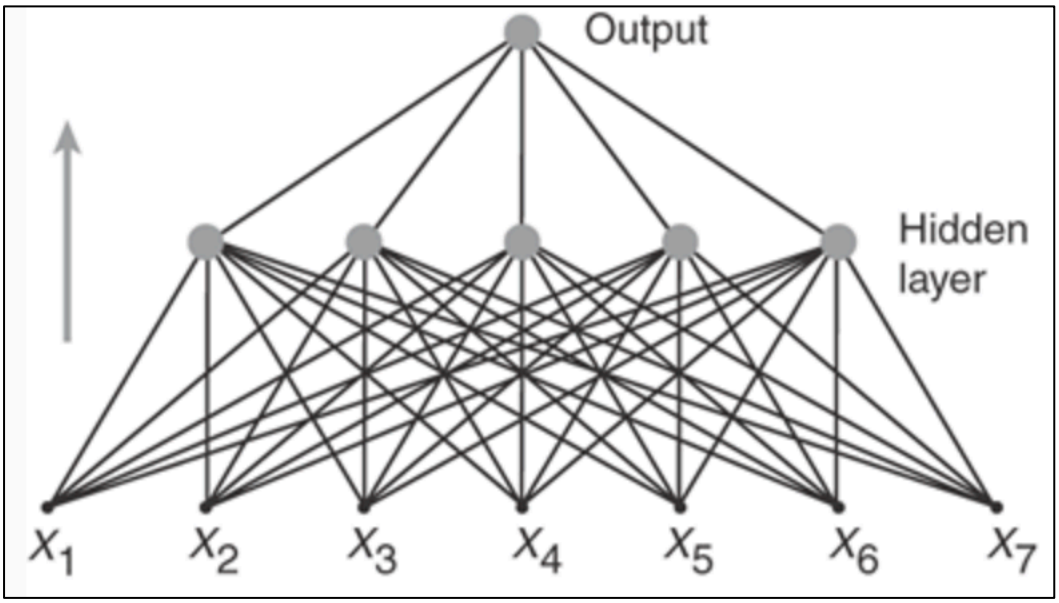

Figure 3.2: Multi layer perceptron

Error back-propagation is used to learn the feed forward neural network with gradient descent algorithm. During error back propagation, weights and thresholds are changed each time an example is presented, such that the error gradually becomes smaller. Different neural network architectures are available to solve different types of problems.

(1) Single layer feed forward networks
(2) Multi-layer feed forward networks
(3) Recurrent networks

Characteristics of ANN, which makes it a feasible technology for classification, are given below.

(1) Ability to learn and generalize using set of adaptive weights (parameter tuning during learning)
(2) Solve non-linear problems (using approximating non-linear functions of the input)

Conventional feed forward neural networks require features to be extracted and hand engineered by humans. This process can be time consuming, erroneous and incomplete. Also the learning complexity grows exponentially with linear increase in the dimensionality of data. As a solution, automatic data preprocessing and feature extraction techniques are introduced to reduce the dimensionality.

3.2.1 Deep learning

Deep learning models are inspired by information representation in mammalian brain [43, 44]. Neocortex is responsible for multiple cognitive abilities in mammalian brain. Deep learning is an emerging sub field in artificial neural networks, which computationally model the hierarchical information representation of neocortex. In deep learning, representation of data is learnt with multiple levels of abstractions. It computes representation of each layer by considering previous layer as input to current layer. Different types of deep learning architectures are available and few of them are given below.

(1) Supervised

a. Deep Convolutional Neural Networks (CNN)
b. Deep Recurrent Neural Networks (RNN)
c. Deep Belief Networks (DBN)

(2) Unsupervised

a. Auto encoders
b. Restricted Boltzmann Machines (RBM)

3.2.1.1 Deep Convolutional Neural Networks (CNN)

Deep convolutional neural networks focus on visual processing over hierarchical structure. This is inspired by the "simple to complex cell organization" in visual cortical cells of cats. CNN is a branch of multi layer perceptron that is specially designed for multi dimensional data models such as images and videos. CNNs have become prevailing approach in many recent recognition and detection tasks [27]. Convolution and sub-sampling as shown in Figure 3.3 are two major constructs of CNNs [45].

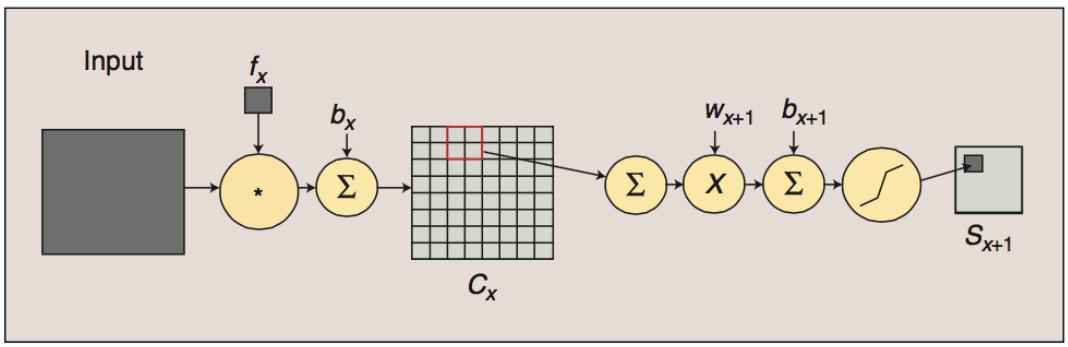

Figure 3.3: Convolution and sub-sampling process

During convolution, set of feature maps is derived from input image using set of filters. Local conjunctions of features from previous layer are detected. During the sub-sampling operation, semantically similar features are merged into one. This helps to further reduce the dimensionality, thus increasing the position invariance of the filters. The most commonly used sub-sampling method is “max pooling”. The above operations are applied to image of a Samoyed dog [27] as given in Figure 3.4.

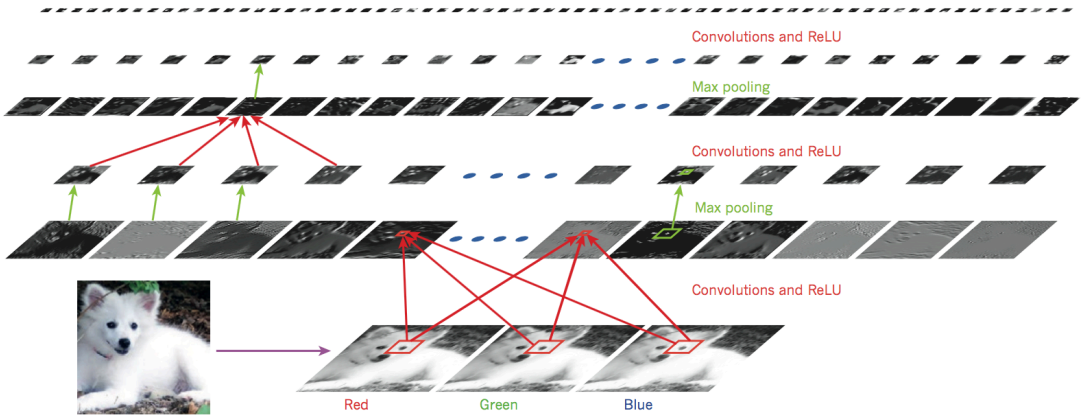

Figure 3.4: Inside convolutional neural network

Rectified linear units (ReLU) are used as the activation function in deep CNNs [46]. It solves one major problem of widely used sigmoid functions known as “vanishing gradient problem”, which hindered stacking of large number of neuron layers as the gradient decreases exponentially with the number of layers. ReLU made training of CNNs made of large number of layers practical. It has become the most used activation function as of 2015. Characteristics of CNNs that make them suitable algorithm for visual pattern recognition is given below.

(1) Classification invariance to a diverse range of transformations and distortions, including noise, scale, rotation, various lighting conditions, displacement
(2) Autonomous extraction of salient feature from images
(3) Better generalization when number of the examples is small
(4) Speed up processing time by GPU based parallelization

Caffe [47] is a deep learning framework developed by Berkeley vision and learning center (UC Berkley).

3.3 Multi Agent Systems (MAS)

Multi agent systems are used in intelligent traffic control optimization. Multi agent system is a collection of agents, which cooperates, negotiates and coordinates via message passing. It is a sub field of distributed artificial intelligence that facilitates global optimization by splitting the problem into multiple loosely coupled sub problems. In multi agent systems, those sub problems are considered as local optimization problems.

3.3.1 Intelligent agent

The computational entity that is allotted to solve a particular sub problem is referred as an "agent". An agent can be viewed as perceiving its environment through sensors and acting upon that environment through effectors [48] as given in Figure 3.5. An intelligent agent is capable of performing autonomous action by demonstrating reactive or proactive behavior when making decisions as well as interacting with other agents [49].

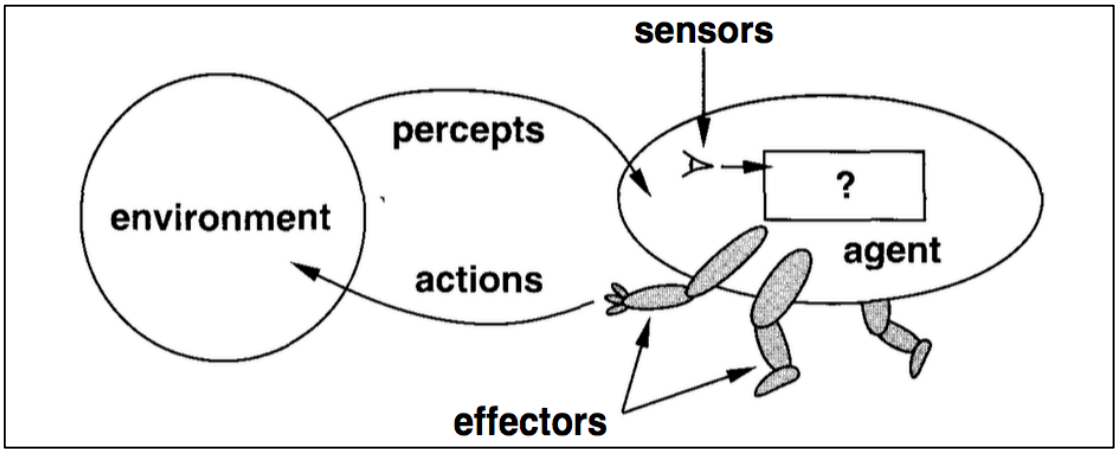

Figure 3.5: Intelligent agent

3.3.2 Characteristics of multi agent systems

Multi agent systems pursue certain characteristics that make them desirable to be applied in complex, distributed, dynamic and scalable environments.

- Autonomy and independence – Usually the agent decision making is at least partially independent from other agents and no external agent controls the decision making process.
- Decentralization – Centralized agent is not appointed to control the system. Decisions are taken at local levels.
- Local views – Agents use local information of the environment during decision making. Agents do not have access to global view of the system as having a global view of the system is not feasible due to its high complexity.
- Emergent behavior – Interactions between agents in a local level can form global level behaviors or structures that are not present initially using the self organizing capabilities [50].

Further, multi agent systems hold desirable characteristics such as parallel computation, asynchronous operation, robustness and reduced cost over centralized approaches. Consequently, multi agent systems fit well in solving real world problems in large number of application domains. Multi agent systems can be directly translated to solve the problem in hand, as road traffic networks possess certain inherent properties that are aligned with abstract multi agent concepts such as distributed operation and autonomous control.

3.3.3 Agent communication and interaction

Agents should cooperate to distribute tasks among each other and effectively coordinate to use the available limited resources. Agent coordination and cooperation tasks are achieved using agent communication. Agents communicate with each other using message passing [51]. Example MAS languages for message passing that can be used as agent communication protocol are Knowledge Query Manipulation Language (KQML) and Agent Communication Language (FIPA's ACL).

3.3.4 Swarm intelligence

Swarm intelligence is the collective behavior of decentralized, self-organized systems, natural or artificial. Agents do not follow any central control structure and they are functioning by following very simple set of rules built in-to them. However, interactions between these agents when following these rules lead in to emergence of intelligent global behavior. Individual agents are unaware of this newly emerged global behavior.

Some examples are given below.

(1) Ant colony optimization
(2) Artificial bee colony algorithm
(3) Particle Swarm Optimization (PSO)

3.4 Summary

Artificial neural networks and multi agent systems were discussed in this chapter as the main techniques for this research. Deep learning algorithms for computer vision were also discussed. Deep learning algorithms facilitate classification invariance to a diverse range of transformations and distortions in road traffic scenes due to different illumination and weather effects. Multi agent systems provide effective ways to communicate and coordinate between different entities in a complex system such as road traffic environment to reduce the traffic congestion. Approach used to solve the identified research problem using the selected technologies is discussed in the next chapter in detail.

Chapter 4

Novel Approach to Intelligent Traffic Control Optimization

4.1 Introduction

In previous two chapters, we identified the research problem and the technologies with preferable characteristics to solve the research problem. This chapter discusses about the approach which author has used to minimize the traffic control optimization problem using the selected technologies. System input, system output, process are discussed. Additionally, non-functional requirements of the system are also discussed in this chapter.

4.2 Hypothesis

Considering traffic density states in adjacent intersections helps to minimize road traffic congestion during traffic control.

4.3 Inputs

After considering several state of the art traffic density detection methods, it was decided to take live CCTV surveillance data from required intersections. A given intersection will have 4-6 cameras monitoring the traffic condition of different routes for 24/7. Some example images taken during daytime and nighttime are given in Appendix A. Input data is required for two modules namely traffic density state learning and traffic density state estimation. Past CCTV surveillance video data along with the time of the day as labels is used for traffic density state learning. Real time CCTV surveillance video data is used for traffic density state estimation. Also, adjacent intersection details of the selected geographical area considered for traffic control is required.

4.4 Output

Optimal traffic signal plan based on the current traffic state for the selected geographical area is given as output of the system. Traffic density state of each of the selected intersection can be also given as it helps for traffic information and route guidance systems.

4.5 Process

The process used to convert the given inputs to preferable outputs such as offline traffic density state learning, real time traffic density estimation, traffic light control optimization are discussed in this section. During offline traffic density state learning phase, system should be trained to recognize traffic density states such as high traffic and low traffic. CCTV surveillance video data for each intersection is automatically labeled using the heuristics of policemen on the already known traffic density states during the course of the day. Then, training is performed to create a model that classifies traffic density state in-to two states namely, high traffic and low traffic using machine learning techniques. During the real time traffic density state estimation phase, the model that is being created during traffic density state learning phase is used to recognize the real time traffic state for all the intersections. Once the traffic density states of all the junctions are available, optimal traffic signal plan is created to ensure that overall traffic congestion is minimized for the selected geographical area. This is achieved during traffic light control optimization phase.

4.6 Non functional requirements

Real time traffic control should take less computation time and system should always be responsive on time. Also, system should be fault tolerant as to function gracefully in the event of failure in acquiring traffic state data from a particular intersection. System should be functional in different weather and lighting conditions. Also, the output on traffic state should not impact by position or viewpoint of the camera.

4.7 Users of the system

Sri Lanka traffic police, police CCTV division and general public (anyone who is using transportation facilities by public transportation or private transportation) are users of the system.

4.8 Summary

This chapter focuses on the approach to design and implement traffic control optimization for Sri Lanka. Input and output are given along with the process, which will be carried upon. Users of the system and non-functional requirements are also discussed. Next chapter discusses the system design of the proposed system with significant focus on elaborating the process mentioned in this chapter.

Chapter 5

Analysis and Design of Intelligent Traffic Control Optimization system

5.1 Introduction

This chapter discusses about the system design aspect of real time intelligent traffic control optimization. Key components and interactions between those components are discussed in detail. Design of the system has adhered to the process identified in the approach that was discussed in the previous chapter. In this chapter, design of the system is divided into, high-level design, traffic density recognition and traffic control optimization.

5.2 High level design

System uses CCTV camera input and sends it to traffic density recognition module. Traffic density state is given as output from traffic density recognition module. Then, the current traffic density state is sent as input to traffic control optimization module to come up with optimal traffic control plan. Figure 5.1 depicts the high level design of the system.

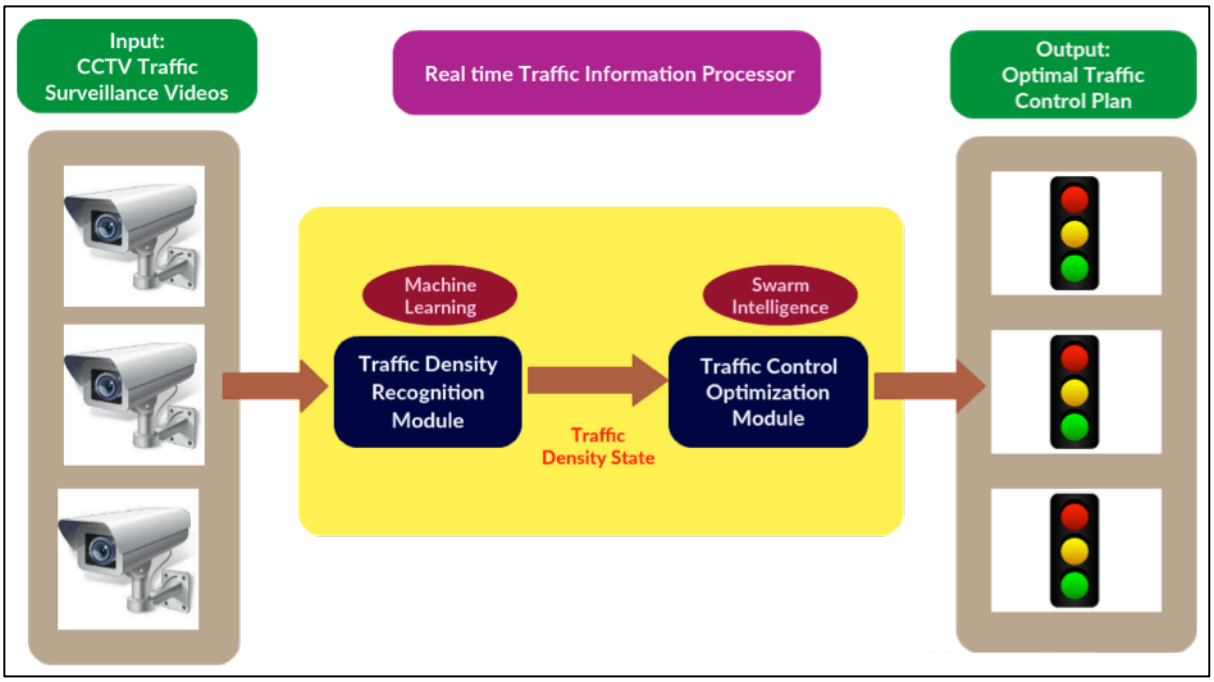

Figure 5.1: High level design

5.3 Traffic density recognition

CCTV camera input is acquired as video stream in traffic density recognition module. Then shot detection is performed. After that key frames are extracted for each shot as processing all the images in the image sequence of the video is computationally expensive. For each key frame, Region of Interest (ROI) is determined. Background is subtracted from the image. Region of Interest (ROI) is used for traffic density state classification task. CCTV camera input will be used in two main tasks as given below.

5.3.1 Training

During training, CNN model is learnt with pre-trained features using convolutional neural network (CNN). Labeled data acquired from the knowledge from policemen will be used here. For example, usually traffic congestion towards Colombo is high during 7 a.m. To 9 a.m. So, CCTV videos acquired during this time period will be automatically labeled as heavy traffic. This will save the time it takes for manual annotation of each traffic image. Graphical Processor Unit (GPU) based training mechanism can be used to speed up training the model with better accuracy.

5.3.2 Classification

Once the CNN model is trained, it is used for traffic density state recognition. Traffic density state is classified as two outcomes namely, high and low. Flow chart for training and classification operation is given in Figure 5.2.

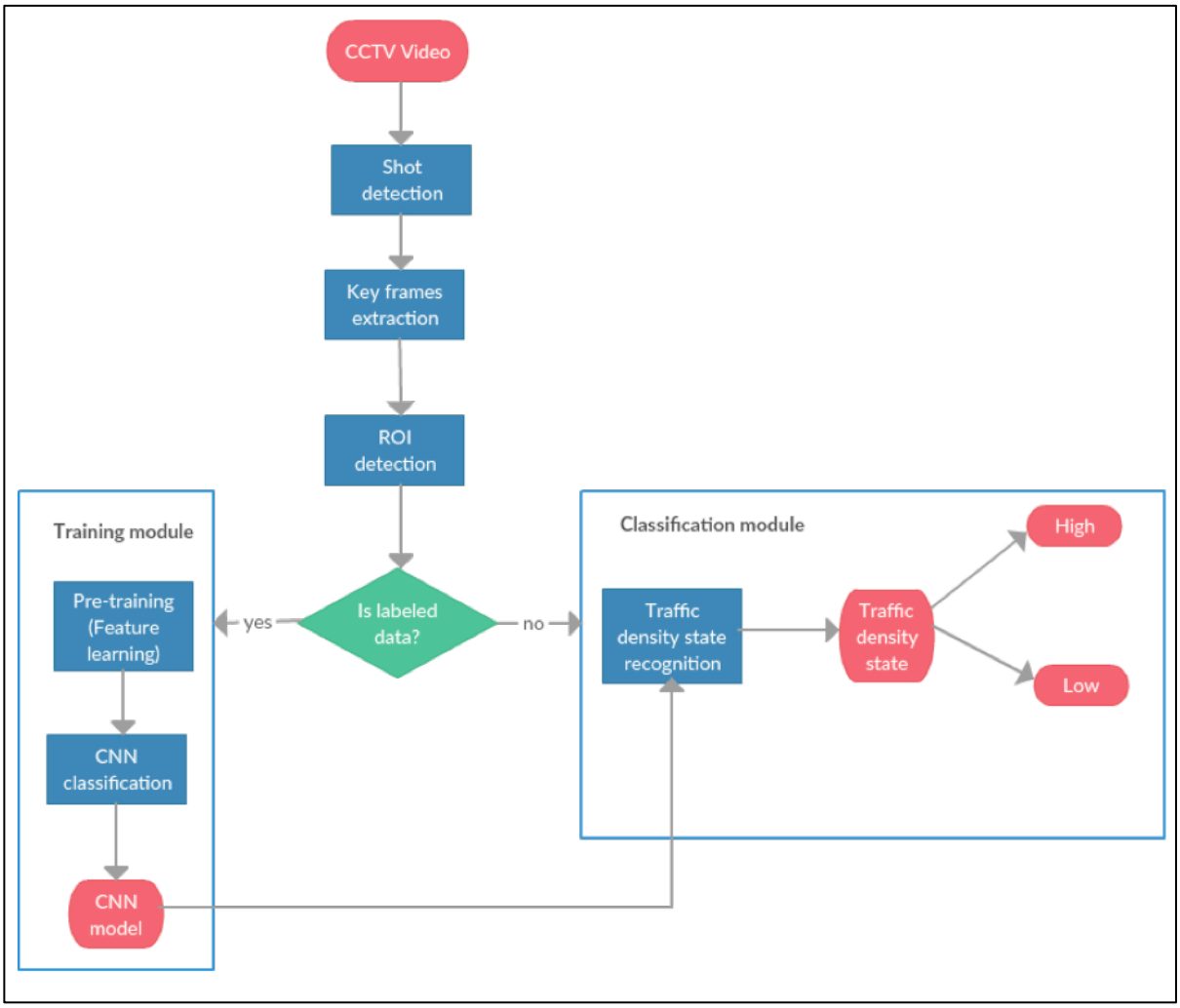

Figure 5.2: Design for traffic density recognition

5.4 Traffic control optimization

Multi agent based system is used for traffic control optimization. Behavior of a human traffic policeman, who is controlling traffic, is simulated during the design of this phase.

Agent architecture is given in Figure 5.5.

Two types of agents are introduced namely, traffic light agent and CCTV agent. Traffic light agent perceives traffic environment via communication with neighboring CCTV agents. Then, it uses reasoning module to determine the green light eligibility for different destinations as given in Equation 5.1.

Green light eligibility = source traffic density – destination traffic density (5.1)

Green light eligibility is calculated using two formulas based on two different approaches. Green light eligibility for individual controlled links that combines both incoming and outgoing lanes is calculated as given in Equation 5.2. Further, green light eligibility is calculated as cumulative traffic density of all incoming lanes to a given intersection as given in Equation 5.3.

$$g_c = d_{l1} - d_{l2} \mid \forall\ c, c \in C^i, d_{l1}, d_{l2} \in D^i \tag{5.2}$$

$$g_c = \Sigma^{n}_{i=1}(d_{l1}) - \Sigma^{m}_{i=1}(d_{l1}) \mid \forall\ c, c \in C^i, d_{l1} \in D^i \tag{5.3}$$

where,

G_c – Green light eligibility for a given controlled link c

d_{l1} – Traffic density for incoming lane

d_{l2} –Traffic density for outgoing lane

c – A given controlled link in an intersection

C^i – Set of all the controlled links in a given intersection i

D^i – Traffic density information for given intersection i

n - number of neighbors for intersection i

m – number of neighbors for intersection j

Example intersection with its neighbors is given in Figure 5.3. After that, it negotiates with its conflicting traffic light agents for optimal traffic signal control. Total green time eligibility is calculated for each conflict free controlled link combination in a given intersection as given in Equation 5.4. Example conflict free controlled link combination for intersection with three neighbors is given in Figure 5.4.

$$Z_f = \Sigma_{c \in f} g_c \mid \forall\ f, f \in F^i, g_c \in G^i \tag{5.4}$$

Z_f = Total green light eligibility for conflict free controlled link combination f

c – Controlled link in conflict free controlled link combination f

g_c – Green light eligibility for controlled link c

f – Conflict free controlled link combination in intersection i

F^i – Set of all the conflict free controlled link combinations for intersection I

G^i - Set of all the green light eligibilities for controlled links in intersection i

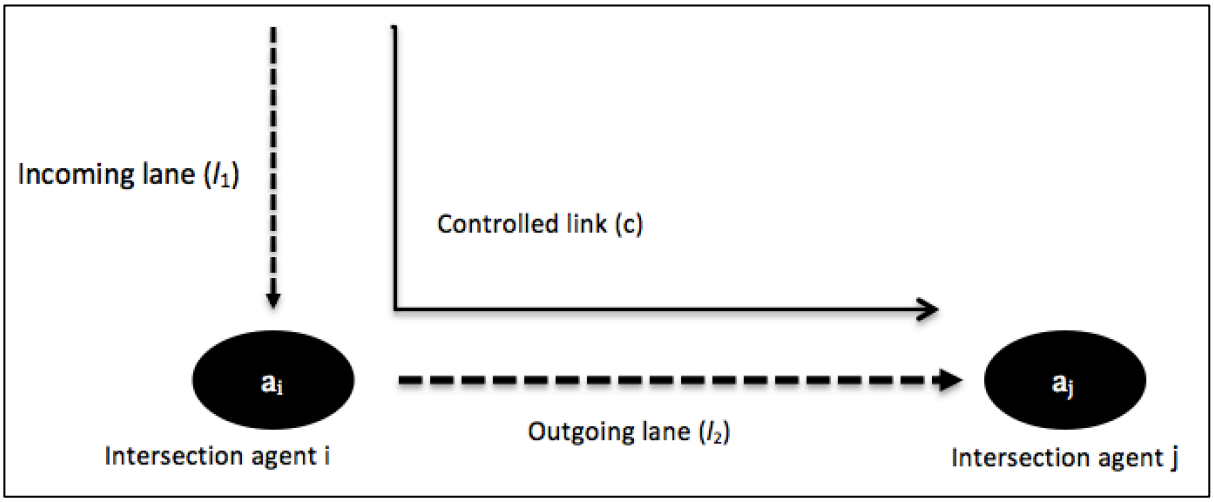

Figure 5.3: Example intersection with its neighbors

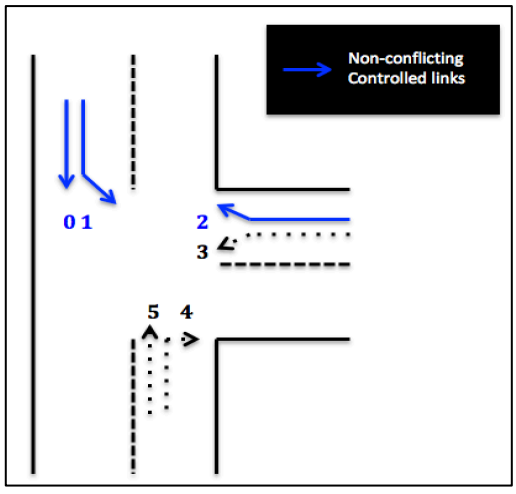

Figure 5.4: Example conflict free controlled link combination

We need to ensure minimum or less severe conflicts during conflict resolution and come up with an optimal traffic signal plan (phase design, cycle length design) accordingly.

- Phase design – during phase design, green light interval and its associated clearance time or amber interval is determined. Conflicting vehicle movements should be considered during this design stage
- Cycle length design – traffic signal rotations through all the phases are determined

Neighboring CCTV agent information and conflicting traffic light agent information are stored in local knowledge base for each agent.

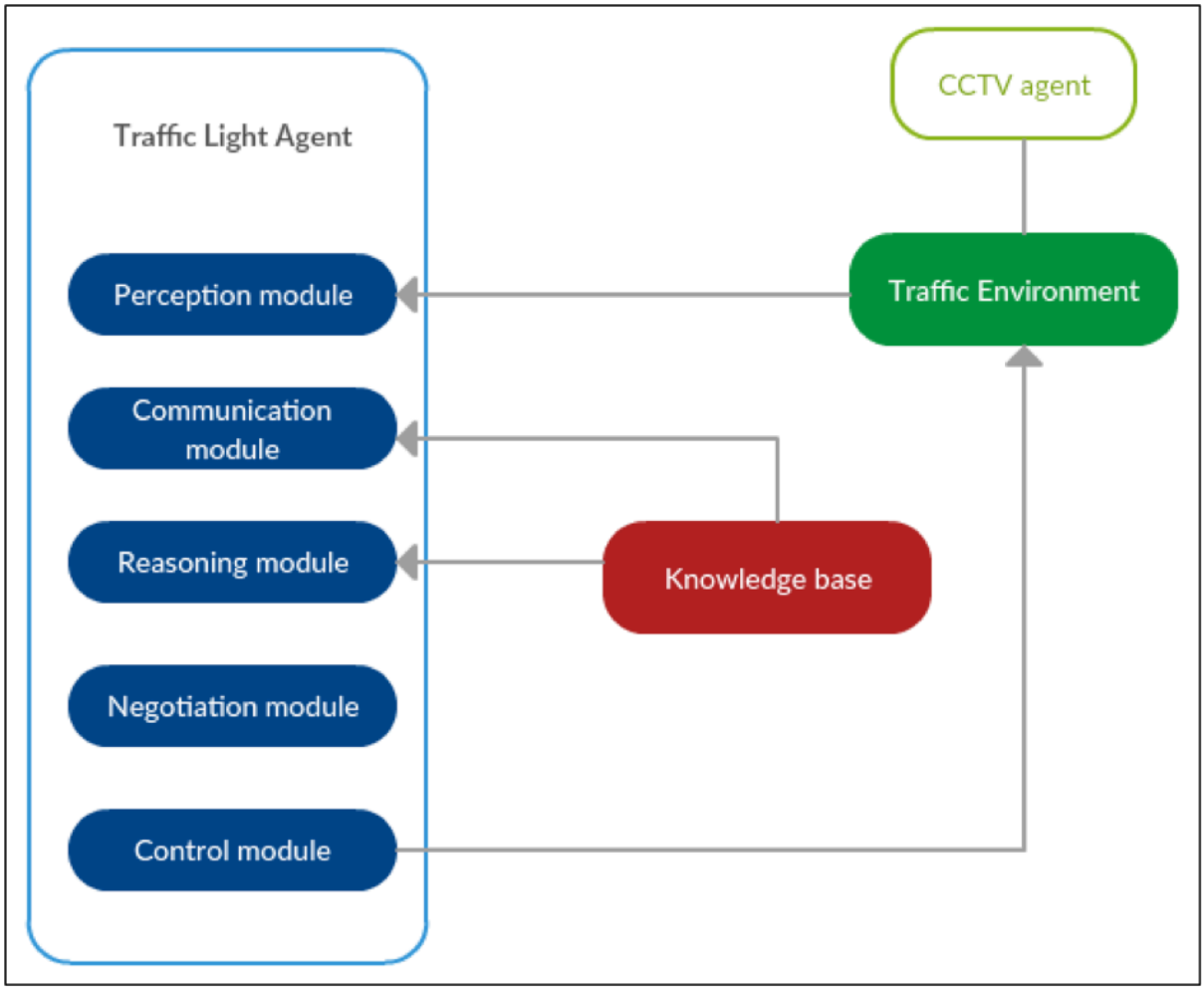

Figure 5.5: Agent architecture

Agent communication and coordination is illustrated in Figure 5.6.

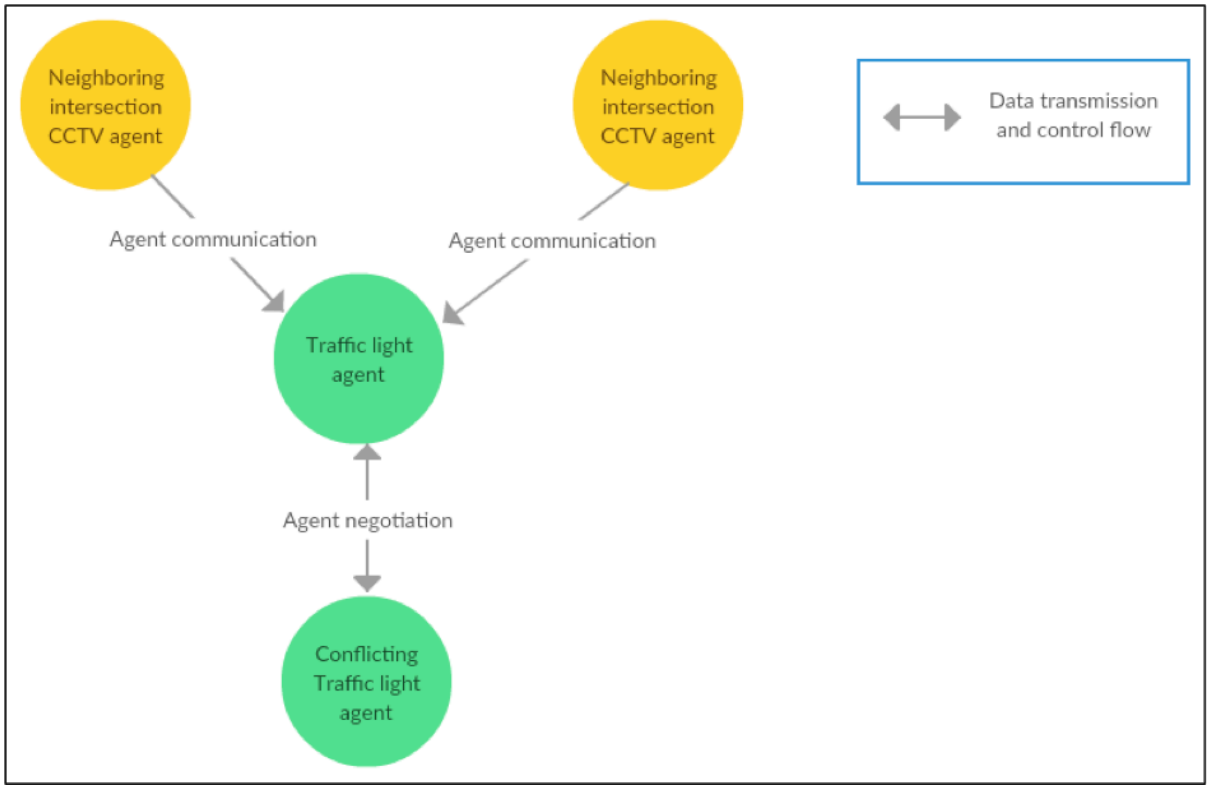

Figure 5.6: Agent communication and coordination

When designing agent interactions, multi agent systems characteristics are taken into consideration. Idea is to reduce road traffic density in a given area based on local interactions and actions of traffic light and CCTV agents as an emerging feature. Traffic light agents have their own knowledge base to make decisions and thus they are autonomous. No central agent is used to control road traffic. Traffic light agents only have their adjacent intersection traffic density states details as a local view. Class diagram of the given design for traffic control optimization module is given in Appendix B.

5.5 Summary

This chapter discussed the design of the system. Design for key components namely traffic density recognition and traffic control optimization is discussed. Main sub components for traffic density recognition are training and classification. Model that is capable of traffic density recognition is created during training stage using video surveillance data. The created model is used during classification stage to recognize traffic density for unseen traffic images. Traffic control optimization module uses agent-based approach to communicate and coordinate with both traffic light and CCTV cameras. Traffic signal control is performed

considering the number of vehicles in the adjacent, destination intersections. Implementation of the proposed design is discussed in the next chapter.

Chapter 6

Implementation of the Proposed Solution

6.1 Introduction

Implementation detail of each module of the system design is explained with the used algorithms, software and tools in this chapter. Out of the available software, most optimal software is selected for the required features in hand and the justifications for doing so are clearly stated in this chapter. Further, challenges faced and actions taken to overcome those challenges are mentioned.

6.2 Traffic density recognition

CCTV camera videos for monitoring traffic density vary in different aspects such as color, lighting conditions and different weather conditions. As discussed in chapter 3, deep neural networks are a suitable technique to overcome these challenges. Caffe is used for traffic density recognition using the input from the CCTV cameras.

6.2.1 Caffe: Convolutional Architecture for Fast Feature Embedding

Deep learning has proven results in the domain of computer vision, as opposed to hand-engineered features. Caffe [52] is a software framework for state of the art deep learning algorithms and a set of reference models mainly developed by Berkeley Vision and Learning Center (BVLC). Caffe is primarily developed in C++ language with high computational speed in mind. It provides interfaces using Python and Matlab bindings. Caffe is capable of training and deploying deep learning algorithms such as convolutional neural networks (CNN) effectively with minimal coding effort. To accelerate training, Caffe provides a convenient method to switch between CPU and GPU implementations. The steps to achieve traffic density recognition using CCTV camera videos as input are mentioned in the upcoming sections.

6.2.2 Data acquisition

As of 2015, CCTV cameras are deployed in 33 intersections in Colombo. Each intersection will have 0 to 2 Pan–Tilt–Zoom cameras (PTZ) and 0 to 5 fixed cameras. CCTV videos taken

from Narahenpita, Kanaththa and Ayurweda intersections during different times of the day are used for this research. Resolution of the obtained videos is 704*640 and frame rate is 8.33 (frames per second). Figure 6.1 and Figure 6.2 show the road scene captured by CCTV camera from the same intersection with heavy traffic density and low traffic density respectively.

Figure 6.1: Heavy traffic condition

Figure 6.2: Low traffic condition

6.2.3 Data preprocessing

From the CCTV video, image sequence is extracted using FFmpeg multimedia framework [53]. Then, region of interest is manually fixed for each lane using Scikit-image image processing library [54] to improve accuracy. The acquired images are resized using Caffe tools. Then, the input images are converted into Lightning Memory-mapped Database (LMDB) format. Caffe provides a command line tool to preprocess images and convert the images to the required LMDB format. First, we separate out the training and test images into two folders. Creating LMDB files for training images is explained here. We need to create a text file and each line should refer to a single training example in the training dataset with its label separated by space as given below. The label refers to the image category and in our case the image category refers to high (1) or low (0) traffic as we have introduced a binary classification scheme. If more than 30% road surface in region of interest is covered by vehicles traffic density is considered as high. If less than 30% road surface in region of interest is covered by vehicles traffic density is considered as low.

Traffic_img_0001.jpeg 1
Traffic_img_0002.jpeg 1
Traffic_img_0003.jpeg 0
Traffic_img_0004.jpeg 0

The ./convert_imageset can be found in $caffe_root/build/tools. We need to specify the path to training images folder and labels files as command line arguments for convert_imageset. Input images are resized to a similar size using --resize_height and --resize_width options. Order of the images can be randomly changed using --shuffle option. Also, we need to specify the output folder to store the generated LMDB files. An example command is given below.

./convert_imageset --resize_height=28 --resize_width=28 --shuffle
/caffe/traffic/training/images/ /caffe/traffic/training/labels.txt /caffe/traffic/training/lmdb

The prepared input is directly fed into artificial neural network without using any hand-engineered features for training using the data layer in Caffe.

6.2.4 Training artificial neural network

Prior to training, artificial neural network need to be constructed. Caffe uses Google protocol buffer files (.proto, .prototxt, .caffemodel) that are strongly typed binary json to represent models as they provide effective serialization and human readability. The network is defined in train.prototxt by setting layers such as convolution, pooling, loss and activation layers. During the training, input image data with its associate labels are passed through computer vision layers such as convolution and pooling to derive abstract representations. The final outcome is fed into loss layers. The complete network is trained using the loss values and the computed gradients. Stochastic gradient descent algorithm is used to train the model. Functionality of important layers in the network is explained below. Convolution layer and pooling layer are computer vision related layers.

- Convolution layer

Convolution layer is the most important layer in this network. Input images are directly fed in to convolution layers. In this layer, an image filter is convolved over the input image pixels. The values of this filter correspond to the weights of the artificial neural network. The weights of the filter initialized randomly and they are learned during back propagation based on the loss values. An image filter uniquely identifies a particular feature in the image. When number of filters is stacked together, it is called a "filter bank". Each filter produces an activation map. Activation map can be considered as an abstract representation of the input image or the activation map of the previous image. Important parameters for convolutional layer are given below.

I. Kernel size – number of filters
II. Stride – amount of the image that the filter is shifted at a time

Feature hierarchy is built from low-level features to high-level features.

- Pooling layer

Pooling layer reduces spatial size of the activation maps. "Max pooling" is the most widely used technique.

- Non-linearity layers

Activation function of the artificial neural network is defined here. Rectifier Linear Units (ReLU) have shown progress over other non-linear functions such as sigmoid functions.

Then, the training parameters such as momentum, base learning rate and number of iterations are defined in traffic_solver.prototxt.

I. Momentum – this parameter is used to avoid the local minima in loss space and reach the global minima
II. Learning rate – the speed that the weights are updated towards optimal weights, step size

Once the network is defined and training parameters are set, the model can be trained using either command line or Python interface. Command line execution is given below.

Caffe train -solver caffe/traffic/traffic_solver.prototxt

Further, Caffe provides methods to fine-tune the model using existing models as well. This

leads to better initialization of parameter values and faster execution during training. Example traffic_solver.prototxt and train.prototxt are given in Appendix C and Appendix D respectively.

6.2.5 Classification of traffic scenes

Python API (Pycaffe) is used for classification of traffic scenes. Code segment used for image classification is given in Appendix E.

6.3 Traffic control optimization

Jade is used to model traffic light agents and their adoptive behavior based on dynamic traffic conditions in traffic control optimization.

6.3.1 Jade: Java Agent Development Environment

Jade is a free and open source software framework developed in Java language that simplifies implementing multi agent systems by providing middleware that conforms to FIPA specification. Jade is distributed by Telecom Italia. Jade follows an Agent Oriented Programming (AOP) paradigm. Jade facilitates easy abstraction of agents and also provides methods to enable peer-to-peer agent communication using asynchronous message passing.

6.3.1.1 Agent modeling using Jade

Traffic light agents and CCTV agents are implemented using “Agent” class in Jade. Message passing between traffic light agents and CCTV agents is used to communicate about traffic density status, conflicting traffic lights and proposed optimized traffic signal plan. Agent communication in Jade is implemented by adhering to the foundation for intelligent, physical agents (FIPA) standards. In Jade, agent communication is achieved using ACL messages using asynchronous message passing. Type of an ACL message varies depending on its purpose. For example, when an intersection traffic light agent asks for traffic density information from another adjacent intersection CCTV agent the message type is “REQUEST”. When the adjacent intersection CCTV agent replies with its traffic density, the message type is “INFORM”. Each agent has a message queue to receive the messages sent by other agents. When a message is received to that queue, the respective agent is notified.
Agent task or jobs are modeled within “behaviors” in Jade. Behavior class is extended to implement actual behaviors in action() method.

6.3.2 SUMO: Simulation of Urban Mobility

SUMO is a free and open source road traffic simulation package that assists to design large road networks and dynamically simulate different traffic signal plan and vehicles. The employees of the institute of transportation systems at the German aerospace center develop SUMO. Sumo provides a fast openGL based graphical user interface. SUMO is developed using standard portable C++ libraries. Out of the number of features SUMO offers, different right-of-way rules, traffic lights, different vehicle types, interoperability with other application at run-time, microscopic routes (each vehicle has an own one) are used in this system.

6.3.2.1 Road traffic simulation using SUMO

Road traffic environment for a given area is simulated using SUMO XML description files. Road networks defined in xml description files can be imported using a command line tool named NETCONVERT.

Network definition

- Tl.nod.xml – intersections/ junctions are defined. When the intersection is controlled by a traffic light, the type of the node should be set as traffic light. Locations of the intersections are defined as x and y coordinates.
- Tl.edg.xml – roads and streets are modeled as edges. Source intersection and destination intersection are given as ‘from’ and ‘to’ values. Number of lanes for each street is also given.
- Tl.typ.xml – type of a lane based on the priority and number of lanes is defined.
- Tl.con.xml – connections between roads and streets are given with the source and destination lanes
- Tl.tll.xml – traffic light programs are defined with phase durations and traffic light state sequences for each phase
- Tl.sumocfg/ tl.netcfg – configuration details

Vehicle demand

- Tl.rou.xml - vehicles, vehicle types, and vehicle routes are defined here.

Example files for road traffic simulation using SUMO is given in Appendix F.

Traffic signal programs

Static traffic plans are defined as an additional file with the traffic phase durations and traffic light state information for a given intersection as given below.

```
<tllogic id="0/1" type="static" programid="0" offset="0">
      <phase duration="34" state="ggrrgg"/>
      <phase duration="5" state="ygrryy"/>
      <phase duration="8" state="rgrrrr"/>
      <phase duration="5" state="ryrrrr"/>
      <phase duration="34" state="rrgggr"/>
      <phase duration="3" state="rryyyr"/>
  </tllogic>
```

Characters in traffic signal state definitions are given below.

- R – red light for vehicles to stop
- Y – amber light for vehicles to slow down
- G – green light for vehicles to pass by

Tls_index represents a controlled link that is a connection between two edges (incoming lane and outgoing lane in an intersection). Individual traffic state signals (e.g., r, y or g) are defined for a tls_index. Traffic Control Interface (TRACI) is used to dynamically change the traffic signal states for intersections based on the real time traffic densities (number of vehicles in a lane). Green light eligibility is determined for each tls_index during both lane based and cumulative traffic density approaches. Traffic signal light assignment is done while ensuring minimum or severe conflicts between vehicles. Example traffic control simulation environment is given in Figure 6.3.

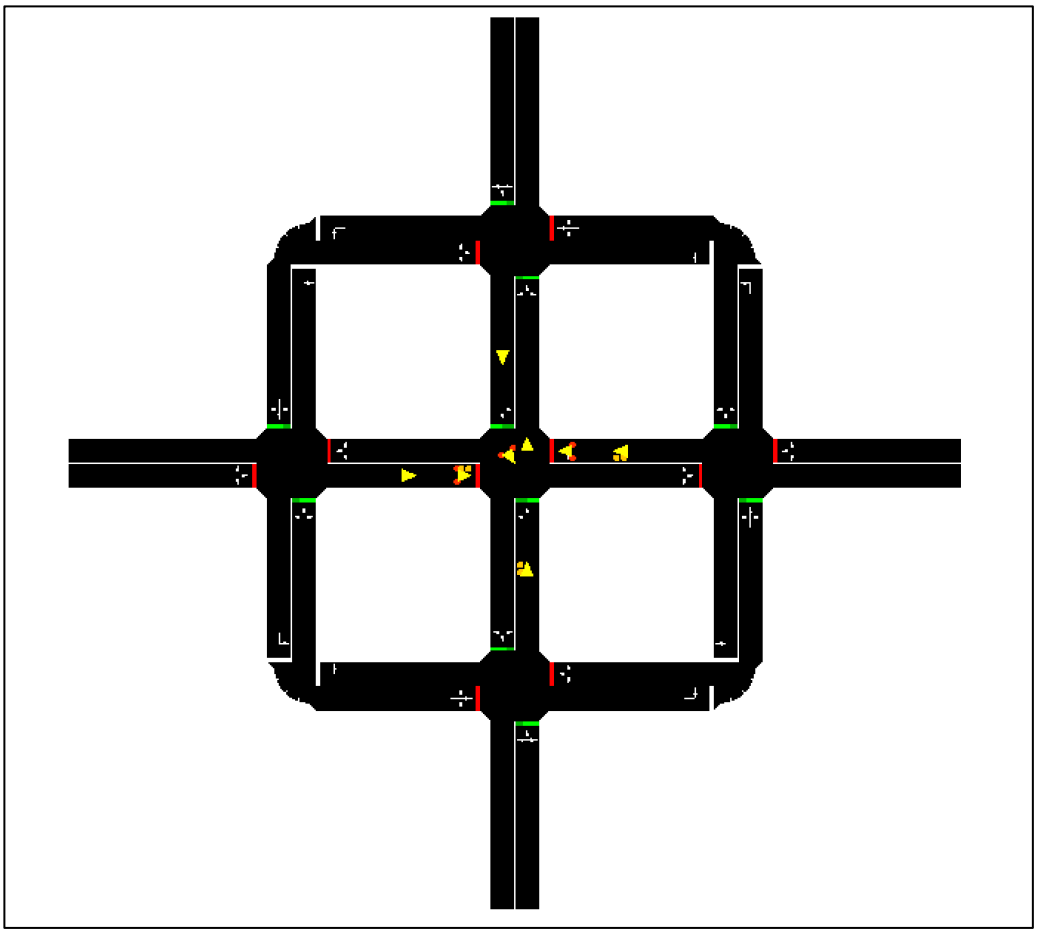

Figure 6.3: Simulation environment for traffic control optimization

6.3.2.2 Traffic control interface (TRACI)

TRACI is an interface developed to dynamically access a running SUMO simulation. Values of the simulation such as number of vehicles in a lane, current status of the traffic lights can be retrieved. Giving the access to a running road traffic simulation, it allows to retrieve values of simulated objects and to control their behavior dynamically.

6.3.3 Integration of SUMO with JADE

Traffic Simulation Manager Application Programming Interface (TRASMAPI) [55] is used to integrate Jade and SUMO by facilitating the communication between them.

TRASMAPI was developed in LIACC (Artificial Intelligence and Computer Science Laboratory), University of Porto. However, since TRASMAPI do not provide all the functionalities required, some functionalities were developed from the scratch by the author.

Implementation details along with the relevant code segments to retrieve controlled links information for a given traffic light is given in Appendix G.

6.4 Traffic control optimization using SUMO and Jade

Traffic light agents request traffic density of their adjacent intersections. The number of vehicles in the lane calculates traffic density of a given lane in the simulation. Green light eligibility of a particular traffic light controlling a link is determined by considering the traffic density of the source lane and the destination lane in a connection along with the conflicting traffic light details. Further, cumulative average traffic density of the intersection also considered and compared as another possible approach. Average waiting time for vehicle after specified time duration is calculated for evaluation.

6.5 Summary

In this chapter the implementation of the intelligent traffic control optimization system is discussed in detail. The system consists of two key modules namely traffic density recognition and traffic control optimization. First, traffic density recognition using Caffe is explained step by step. Then, traffic control optimization using Jade is discussed along with a real world simulation developed using SUMO. In the next chapter, evaluation of the proposed solution is given to justify if the proposed solution meets its end user needs.

Chapter 7

Evaluation of the Proposed Solution

7.1 Introduction

Evaluation strategy, experimental design and results are discussed in this chapter. Traffic density recognition is a subjective decision, so the techniques used to verify the accuracy is explained in detail. Experimental results for the two main components namely traffic density recognition and traffic control optimization are discussed.

7.2 Experimental design

Experiments designed to evaluate the performance of the two main components namely traffic density recognition and traffic control optimization are reported in this section.

7.2.1 Traffic scene datasets

Road traffic scenes from different intersections during different times are used for evaluation. In this study, traffic scenes are collected on following time durations.

- 6 a.m. – 6.15 a.m. (day time – less traffic)
- 8 a.m. – 10.30 a.m. (day time – heavy traffic)
- 5 p.m – 5.15 p.m. (day time – heavy traffic)
- 7 p.m. – 7.15 p.m. (night time – less traffic)

The adjacent intersections in Colombo that are considered in this experiment are as follows.

- Narahenpita
- Ayurveda junction
- Kanaththa junction

Details of one such example intersection are given in Table 7.1.

Location	Narahenpita junction
Sequence type	8 a.m. to 10.30 a.m
Resolution	704*640
Number of frames (8 fps)	72,000
Weather condition/ illumination	Sunny

Table 7.1: Details of an example intersection

Traffic density data related to different weather conditions such as rainy and foggy was unable to collect as the relevant past data was erased at the time of collection.

7.2.2 Evaluation strategy for traffic density recognition

In addition to the conventional validation where the available dataset is partitioned into two datasets, namely training dataset and test dataset, cross validation is performed to ensure that model provides better generalization over unseen examples. Experiments are conducted using traffic density information from both same and different intersections to create training dataset and test dataset interchangeably. The results from the cross validation experiments are averaged over all the rounds to get the final result. Further, combination of features from both same and different intersections is also evaluated during experiments. In addition, both global information and local information are considered and compared during feature extraction for accuracy. Also, traffic image frames scaled at different scales such as 80*80 and 128*128 are also tested to see if there any impact of image scale on the classification error. Quantitative accuracy calculation is conducted based on the said sampling strategies. Classifier accuracy is calculated as given in Equation 7.1.

$$\text{Accuracy} = \frac{\Sigma \text{ True positive examples} + \Sigma \text{ True negative examples}}{\Sigma \text{ Total number of examples}} \tag{7.1}$$

7.2.3 Traffic control optimization

Traffic control optimization using multi agent technology is evaluated using three different methods into consideration. Static/ fixed traffic control method is used as the base line method. Currently, fixed traffic control method is used in Sri Lankan roads/ intersections. In

the next method, traffic density is calculated by considering individual lanes (both source and destination for a given controlled link). Another method is proposed where cumulative average of a given intersection is considered when it comes to decide traffic density of adjacent intersections. Goal of a singular traveller is to make his or her own expedition as fast and economic as much as possible. Goal of the road traffic management authorities is to reduce the overall traffic congestion by optimally utilizing the available resources such as road space and vehicle waiting time. Dynamic evaluation metrics to measure the success of the application for this module are determined by considering both aforesaid aspects. Average vehicle waiting time and time loss are two important matrices to measure the delay caused due to traffic congestion. Average vehicle waiting time is given in Equation 7.2.

$$\text{Average vehicle waiting time} = \Sigma \text{ Vehicle halting time/ Number of vehicles} \quad (7.2)$$

However, vehicles that are slowing down due to traffic congestion and the associated time loss due to the decrease in speed are not considered in average vehicle waiting time metric. Time loss metric considers the vehicles delays due to both halting and slowing down. Time loss is given in Equation 7.3.

$$\text{Time loss} = \text{Time taken for vehicles to pass between two intersections during free flow} - \text{Actual time taken for vehicles to pass between two intersections with traffic conditions} \quad (7.3)$$

Accordingly, time loss is used to evaluate the success of the proposed solution in this research. Time taken for vehicles to pass between two intersections during free flow is calculated using the allowed speed for vehicles and the distance between intersections. Example road network is given in Figure 7.1.

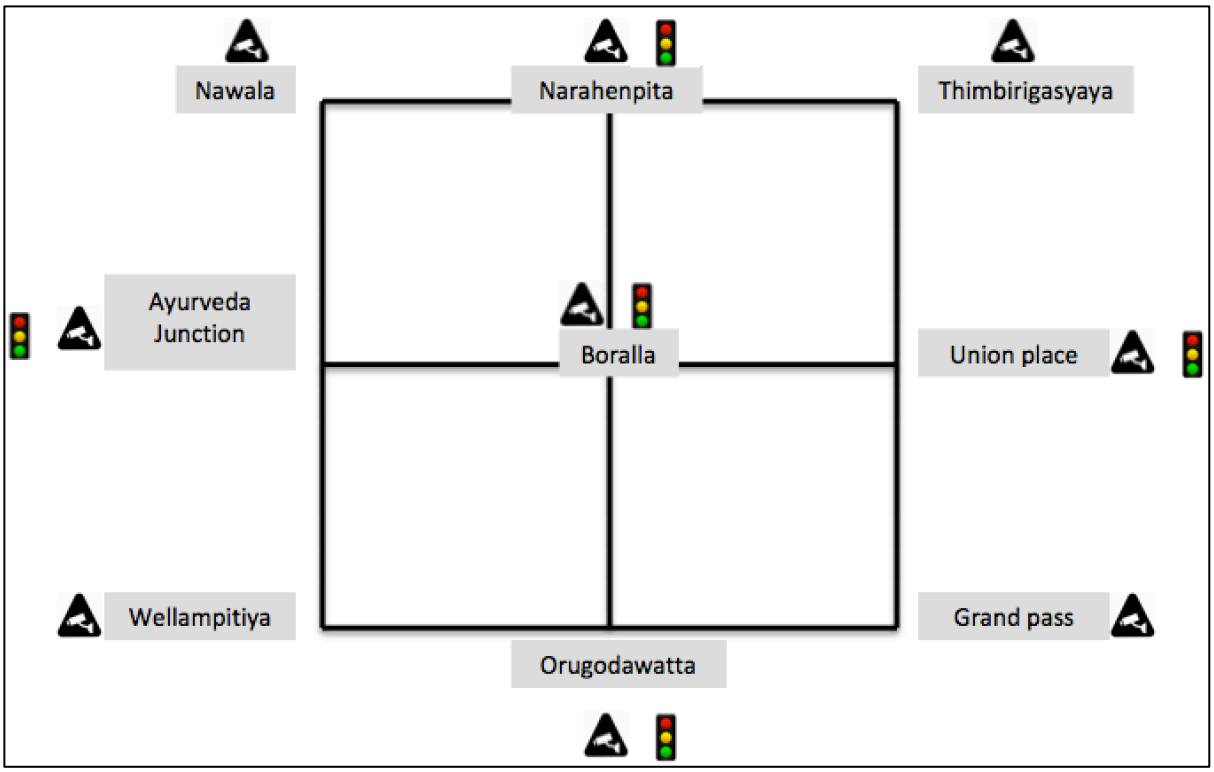

Figure 7.1: Example road network

7.3 Experimental results

7.3.1 Traffic scene classifier results

Convolutional Neural Networks (CNN) classifier is evaluated for accuracy in different scenarios. Accuracy for each of the scenarios mentioned in section 8.2.1 is given in Table 7.2.

Classifier/ time of the day	**Classifier accuracy in scale (80*80)**		**Classifier accuracy in scale (128*128)**	
Day time	**Global features**	**Local features**	**Global features**	**Local features**
	52%	70%	58.5%	72%
Night time	21% (local features)		23% (local features)	

Table 7.2: Traffic scene classifier results

Separate experiment is conducted to determine the type of features ideal for model creation. Local features that are extracted during daytime are used for this experiment. The results are given in Table 7.3.

Train dataset	Classifier accuracy
Same intersection	63%
Different intersection	51%
Combined (both same intersection + different intersection)	69%

Table 7.3: Experimental results for different training datasets

7.3.2 Traffic control optimization results

Traffic control optimization aims at minimizing the waiting time of vehicles. In this evaluation, time loss of vehicles in a given area is considered for static traffic control (control experiment) and the proposed methods namely for cumulative intersection based traffic control and lane based traffic control. Results are given in Figure 7.2.

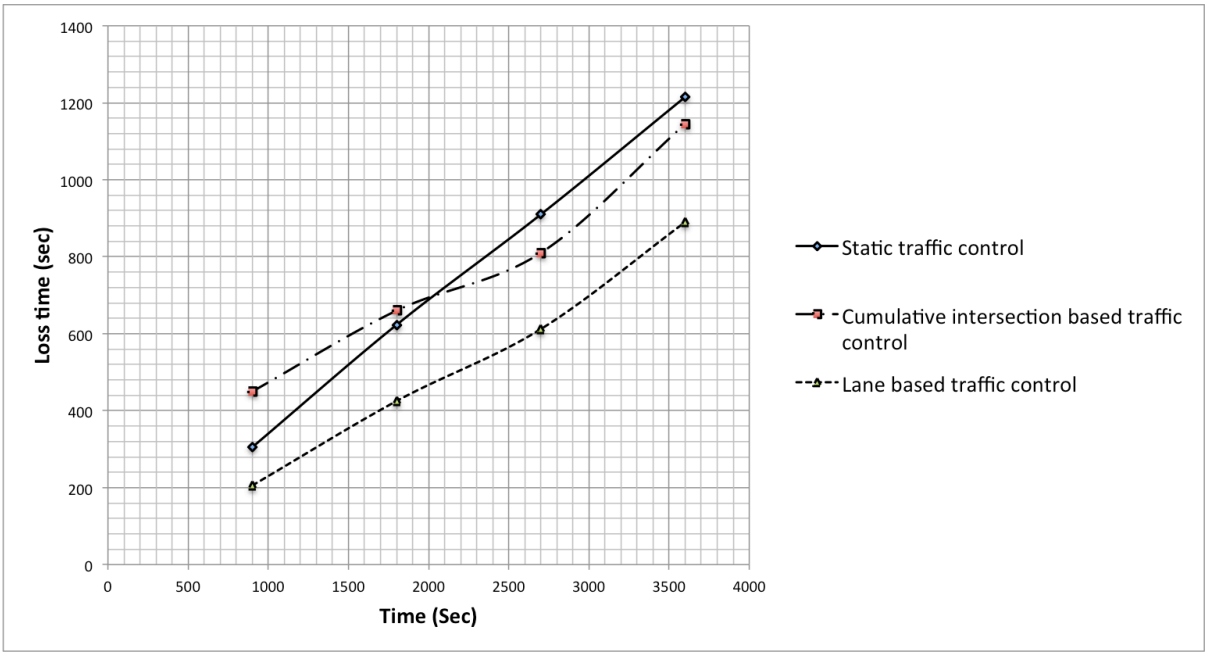

Figure 7.2: Results for traffic control optimization

7.4 Summary

In this chapter, evaluation strategy, experimental study, datasets used and the results are discussed. Two main components of the research namely, traffic density recognition and traffic control optimization are evaluated separately. Accuracy of the traffic density recognition is evaluated under different conditions such as different features, different scales and different illumination settings. Time loss in traffic control optimization module is evaluated to proposed multi agent based solutions with comparison to controlled experiment of static traffic control system. Next chapter discusses the interpretation of the experimental

results given in this chapter along with the conclusion and future work.

Chapter 7

Conclusion and Future Work

8.1 Introduction

In the previous chapter we discussed about the evaluation strategy along with the obtained results of the proposed solution. This chapter focuses on interpreting the results given in evaluation, discussing the limitations and future work of the solution.

8.2 Concluding remarks

The aim of this research is to develop a system for addressing traffic congestion controlling problem with the use of machine learning and multi agent technologies. As mentioned in the previous chapters, the aim along with its associated objectives is achieved successfully in this project. Traffic congestion is reduced by dynamically allocating the traffic signal plans using the analyzed road traffic density information with CCTV surveillance cameras. According to the results shown in the evaluation, it is clearly evident that the said aim is achieved to a great extent. The research is conducted as two key modules namely traffic density recognition and traffic control optimizations. Traffic density recognition module was able to gain 72% of accuracy, whereas traffic signal control optimization module was able to improve the traffic flow by 34% compared to static traffic control plan that is operated in Sri Lankan roads. During the experiments conducted to evaluate the traffic density recognition, it was found that local features that were extracted using the pre-defined Region of Interest (ROI) are more expressive than the global features extracted from the whole traffic scene. Further, it was uncovered that using combination of traffic frames from both same intersections as well as from different intersections provides better generalization ability for the classifier. Image scale used as training input doesn't have much impact on the final outcome of the classifier. During the evaluation of the traffic control optimization it was found that both individual lane based traffic control and cumulative intersection based traffic control methods that are based on multi agent systems perform better than static traffic control method. Out of the proposed two method, individual lane based traffic control method performs better than the cumulative intersection based traffic control method after a given time period.

8.3 Limitations and future work

This research is conducted with limited resources such as time, cost and access to hardware. Hence, this research provides further research avenues to be explored in order to solve the defined traffic congestion problem.

- Internet of Things (IOT) for smart cities

In this research, traffic control optimization is performed in a simulated environment. The concept of Internet of Things (IOT) can be explored to apply this solution in a real world setting. Heterogeneous hardware sensors both in roads and vehicles can be used to monitor and coordinate traffic congestion.

- GPU based acceleration for Convolutional Neural Networks (CNN)

In this research, traffic density recognition is done using convolutional neural networks in Caffe only in CPU mode. However, training the dataset in GPU mode will rapidly improve the required computation time. Accordingly, the analysis can be performed to a larger dataset. Large amount of traffic information are available as CCTV surveillance systems are operating 24/7 in main intersections. So, the acquired information can be effectively used to further improve model accuracy.

8.4 Summary

In chapter 1, aim and objectives were defined for this research project. First, we need to get a comprehensive background knowledge on the selected research area namely, traffic control optimization in intelligent transportation systems. Once the importance of the research problem is identified in chapter 1, study is performed on the current approaches for traffic density recognition and traffic control optimization to address the traffic congestion problem in chapter 2. Intelligent traffic control optimization system is designed and developed using technologies such as artificial neural networks and multi agent systems as given in chapter 3, chapter 4 and chapter 5. Simulated environment is used to evaluate the success of the research using Simulated Urban Mobility (SUMO). Extensive evaluation that is conducted for the proposed solution is given along with the experimental setup and the obtained results

in chapter 6. Accordingly, it is evident that all the objectives defined at the beginning of the project are successfully met in this research.

References

[1] "Statistical Pocket Book 2016," Department of Census and Statistics - Sri Lanka, Survey.

[2] C. Xiao-Feng, S. Zhong-ke, and Z. Kai, "Research on an intelligent traffic signal controller," in *Intelligent Transportation Systems, 2003. Proceedings. 2003 IEEE*, 2003, vol. 1, pp. 884–887.

[3] C. Li and S. Shimamoto, "A real time traffic light control scheme for reducing vehicles CO 2 emissions," in *2011 IEEE Consumer Communications and Networking Conference (CCNC)*, 2011, pp. 855–859.

[4] N. J. Garber and L. A. Hoel, *Traffic and highway engineering*. Cengage Learning, 2014.

[5] C. Chen, Z. Jia, and P. Varaiya, "Causes and cures of highway congestion," *IEEE Control Syst. Mag.*, vol. 21, no. 6, pp. 26–32, 2001.

[6] M. Litzenberger *et al.*, "Vehicle counting with an embedded traffic data system using an optical transient sensor," in *2007 IEEE Intelligent Transportation Systems Conference*, 2007, pp. 36–40.

[7] Bo Chen and H. H. Cheng, "A Review of the Applications of Agent Technology in Traffic and Transportation Systems," *IEEE Trans. Intell. Transp. Syst.*, vol. 11, no. 2, pp. 485–497, Jun. 2010.

[8] K. Singh and B. Li, "Estimation of Traffic Densities for Multilane Roadways Using a Markov Model Approach," *IEEE Trans. Ind. Electron.*, vol. 59, no. 11, pp. 4369–4376, Nov. 2012.

[9]Z. Tehao and C. Feng, "Acquisition of traffic flow density using multi-source data fusion," in *Measurement, Information and Control (MIC), 2012 International Conference on*, 2012, vol. 2, pp. 595–599.

[10] J. Barrachina *et al.*, "A V2I-Based Real-Time Traffic Density Estimation System in Urban Scenarios," *Wirel. Pers. Commun.*, vol. 83, no. 1, pp. 259–280, Jul. 2015.

[11] R. Mao and G. Mao, "Road traffic density estimation in vehicular networks," in *Wireless Communications and Networking Conference (WCNC), 2013 IEEE*, 2013, pp. 4653–4658.

[12] L. Garelli, C. Casetti, C.-F. Chiasserini, and M. Fiore, "Mobsampling: V2V communications for traffic density estimation," in *Vehicular Technology Conference (VTC Spring), 2011 IEEE 73rd*, 2011, pp. 1–5.

[13] T. Darwish and K. Abu Bakar, "Traffic density estimation in vehicular ad hoc networks: A review," *Ad Hoc Netw.*, vol. 24, pp. 337–351, Jan. 2015.

[14] A. Tabibiazar and O. Basir, "Kernel-based modeling and optimization for density estimation in transportation systems using floating car data," in *Intelligent Transportation Systems (ITSC), 2011 14th International IEEE Conference on*, 2011, pp. 576–581.

[15] V. Tyagi, S. Kalyanaraman, and R. Krishnapuram, "Vehicular Traffic Density State Estimation Based on Cumulative Road Acoustics," *IEEE Trans. Intell. Transp. Syst.*, vol. 13, no. 3, pp. 1156–1166, Sep. 2012.

[16] M. M. nal Bhandarkar and M. T. Waykole, "Traffic Density Estimation and Mechanical Condition Determination of Vehicles using Acoustic Signals."

[17] P. Borkar and L. G. Malik, "Acoustic signal based traffic density state estimation using adaptive Neuro-Fuzzy classifier," in *Fuzzy Systems (FUZZ), 2013 IEEE International Conference on*, 2013, pp. 1–8.

[18] J. Kato, Y. Hiramatsu, and T. Watanabe, "Estimating traffic density using sounds of moving vehicles," in *Proceedings of the Seventh Eurographics conference on Multimedia*, 2004, pp. 21–29.

[19] J. Kato, “An attempt to acquire traffic density by using road traffic sound,” in *Active Media Technology, 2005.(AMT 2005). Proceedings of the 2005 International Conference on*, 2005, pp. 353–358.
[20] E. Tan and J. Chen, “Vehicular traffic density estimation via statistical methods with automated state learning,” in *Advanced Video and Signal Based Surveillance, 2007. AVSS 2007. IEEE Conference on*, 2007, pp. 164–169.
[21] T. Wassantachat, Z. Li, J. Chen, Y. Wang, and E. Tan, “Traffic Density Estimation with On-line SVM Classifier,” 2009, pp. 13–18.
[22] S. B. Purusothaman and K. Parasuraman, “Vehicular traffic density state estimation using Support Vector Machine,” in *Emerging Trends in Computing, Communication and Nanotechnology (ICE-CCN), 2013 International Conference on*, 2013, pp. 782–785.
[23] Z. Li, E. Tan, J. Chen, and T. Wassantachat, “On Traffic Density Estimation with a Boosted SVM Classifier,” 2008, pp. 117–123.
[24] P. Janney and G. Geers, “Advanced framework for illumination invariant traffic density estimation,” in *Intelligent Transportation Systems, 2009. ITSC’09. 12th International IEEE Conference on*, 2009, pp. 1–6.
[25] O. Asmaa, K. Mokhtar, and O. Abdelaziz, “Road traffic density estimation using microscopic and macroscopic parameters,” *Image Vis. Comput.*, vol. 31, no. 11, pp. 887–894, Nov. 2013.
[26] Y. Yuan, L. Mou, and X. Lu, “Scene Recognition by Manifold Regularized Deep Learning Architecture,” *IEEE Trans. Neural Netw. Learn. Syst.*, pp. 1–1, 2015.
[27] Y. LeCun, Y. Bengio, and G. Hinton, “Deep learning,” *Nature*, vol. 521, no. 7553, pp. 436–444, May 2015.
[28] B. Zhou, A. Lapedriza, J. Xiao, A. Torralba, and A. Oliva, “Learning deep features for scene recognition using places database,” in *Advances in Neural Information Processing Systems*, 2014, pp. 487–495.
[29] S. Mehan and V. Sharma, “Development of traffic light control system based on fuzzy logic,” in *Proceedings of the International Conference on Advances in Computing and Artificial Intelligence*, 2011, pp. 162–165.
[30] K. T. K. Teo, K. B. Yeo, S. E. Tan, Z. W. Siew, and K. G. Lim, “Design and development of portable Fuzzy Logic based traffic optimizer,” in *ICCE-China Workshop (ICCE-China), 2013 IEEE*, 2013, pp. 7–12.
[31] K. T. K. Teo, W. Y. Kow, and Y. K. Chin, “Optimization of Traffic Flow within an Urban Traffic Light Intersection with Genetic Algorithm,” 2010, pp. 172–177.
[32] F. Teklu, A. Sumalee, and D. Watling, “A genetic algorithm approach for optimizing traffic control signals considering routing,” *Comput.-Aided Civ. Infrastruct. Eng.*, vol. 22, no. 1, pp. 31–43, 2007.
[33] Y. K. Chin, W. Y. Kow, W. L. Khong, M. K. Tan, and K. T. K. Teo, “Q-Learning Traffic Signal Optimization within Multiple Intersections Traffic Network,” 2012, pp. 343–348.
[34] A. W. Addison, T. N. Rao, J. Reedijk, J. van Rijn, and G. C. Verschoor, “Synthesis, structure, and spectroscopic properties of copper (II) compounds containing nitrogen–sulphur donor ligands; the crystal and molecular structure of aqua [1, 7-bis (N-methylbenzimidazol-2′-yl)-2, 6-dithiaheptane] copper (II) perchlorate,” *J. Chem. Soc. Dalton Trans.*, no. 7, pp. 1349–1356, 1984.
[35] W. Narzt, U. Wilflingseder, G. Pomberger, D. Kolb, and H. Hörtner, “Self-organising congestion evasion strategies using ant-based pheromones,” *IET Intell. Transp. Syst.*, vol. 4, no. 1, p. 93, 2010.

[36] D. Renfrew and X.-H. Yu, “Traffic Signal Control with Swarm Intelligence,” in *Proceedings of the 2009 Fifth International Conference on Natural Computation - Volume 03*, Washington, DC, USA, 2009, pp. 79–83.
[37] D. Renfrew and X.-H. Yu, “Traffic Signal Control with Swarm Intelligence,” 2009, pp. 79–83.
[38] J. García-Nieto, E. Alba, and A. C. Olivera, “Enhancing the urban road traffic with Swarm Intelligence: A case study of Córdoba city downtown,” in *Intelligent Systems Design and Applications (ISDA), 2011 11th International Conference on*, 2011, pp. 368–373.
[39] J. Zhao and D. Tang, “Coordination traffic control under the framework of multi-agent technology,” in *Computer Science and Engineering, 2009. WCSE’09. Second International Workshop on*, 2009, vol. 1, pp. 219–222.
[40] J. Zhang, L. Yan, Y. Han, G. Song, and X.-L. Fang, “Research on the Method and Simulation of Intersection Signal Control Based on Multi-Agent,” in *Management and Service Science, 2009. MASS’09. International Conference on*, 2009, pp. 1–4.
[41] W. Wu, G. Haifei, and J. An, “A Multi-agent Traffic Signal Control System Using Reinforcement Learning,” in *2009 Fifth International Conference on Natural Computation*, 2009, vol. 4, pp. 553–557.
[42] A. Krogh, “What are artificial neural networks?,” *Nat. Biotechnol.*, vol. 26, no. 2, pp. 195–197, 2008.
[43] T. Dean, “A computational model of the cerebral cortex,” in *Proceedings of the National Conference on Artificial Intelligence*, 2005, vol. 20, p. 938.
[44] T. Serre, L. Wolf, S. Bileschi, M. Riesenhuber, and T. Poggio, “Robust object recognition with cortex-like mechanisms,” *Pattern Anal. Mach. Intell. IEEE Trans. On*, vol. 29, no. 3, pp. 411–426, 2007.
[45] I. Arel, D. C. Rose, and T. P. Karnowski, “Deep Machine Learning - A New Frontier in Artificial Intelligence Research [Research Frontier],” *IEEE Comput. Intell. Mag.*, vol. 5, no. 4, pp. 13–18, Nov. 2010.
[46] X. Glorot, A. Bordes, and Y. Bengio, “Deep Sparse Rectifier Neural Networks.,” in *Aistats*, 2011, vol. 15, p. 275.
[47] “http://caffe.berkeleyvision.org/.” .
[48] S. J. Russell and P. Norvig, *Artificial intelligence: a modern approach*. Englewood Cliffs, N.J: Prentice Hall, 1995.
[49] G. Weiss, Ed., *Multiagent systems: a modern approach to distributed artificial intelligence*. Cambridge, Mass: MIT Press, 1999.
[50] V. I. Gorodetskii, “Self-organization and multiagent systems: I. Models of multiagent self-organization,” *J. Comput. Syst. Sci. Int.*, vol. 51, no. 2, pp. 256–281, Apr. 2012.
[51] M.-P. Huget, *Communication in Multiagent Systems: Agent communication languages and conversation policies*, vol. 2650. Springer Science & Business Media, 2003.
[52] Y. Jia *et al.*, “Caffe: Convolutional architecture for fast feature embedding,” in *Proceedings of the 22nd ACM international conference on Multimedia*, 2014, pp. 675–678.
[53] “https://ffmpeg.org/ffmpeg.html.” .
[54] “http://scikit-image.org/.” .
[55] I. J. Timóteo, M. R. Araújo, R. J. Rossetti, and E. C. Oliveira, “TraSMAPI: An API oriented towards Multi-Agent Systems real-time interaction with multiple Traffic Simulators,” in *Intelligent Transportation Systems (ITSC), 2010 13th International IEEE Conference on*, 2010, pp. 1183–1188.

Appendix A

Road traffic scenes of different intersections

Figure A.1: Narahenpita Junction

Figure A.2: Ayurveda Junction

Figure A.3: Kanaththa Junction

Appendix B

Class diagram for traffic control optimization

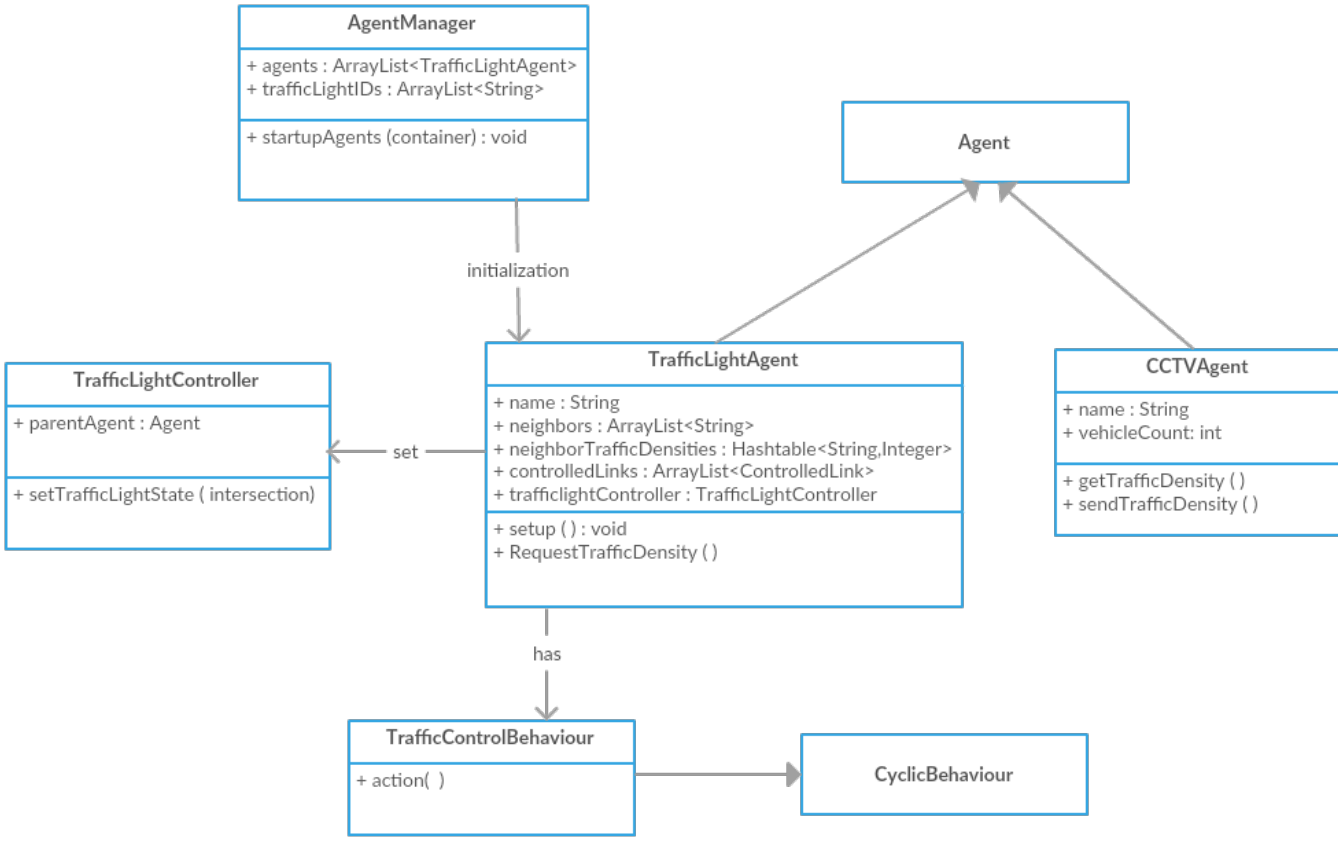

Figure B.1: Class diagram for traffic control optimization

Appendix C

Configuration details in Caffe Solver

Traffic_solver.prototxt

```
# The train/test net protocol buffer definition
net: "/Users/jwithanawasam/MachineLearning/Caffe/caffe/Traffic/traffic_train_test.prototxt"
# test_iter specifies how many forward passes the test should carry out.
# In the case of MNIST, we have test batch size 100 and 100 test iterations,
# covering the full 10,000 testing images.
test_iter: 100
# Carry out testing every 500 training iterations.
#test_interval: 500
test_interval: 500
# The base learning rate, momentum and the weight decay of the network.
#base_lr: 0.01
base_lr: 0.00009
momentum: 0.9
weight_decay: 0.0005
# The learning rate policy
lr_policy: "inv"
gamma: 0.0001
power: 0.75
# Display every 100 iterations
display: 100
# The maximum number of iterations
max_iter: 10000
# snapshot intermediate results
snapshot: 5000
snapshot_prefix: "/Users/jwithanawasam/MachineLearning/Caffe/caffe/Traffic/model"
# solver mode: CPU or GPU
solver_mode: CPU
```

Appendix D

Configuration details for Caffe deep neural network

Train_test.prototxt

```
name: "LeNet"
layer {
  name: "mnist"
  type: "Data"
  top: "data"
  top: "label"
  include {
    phase: TRAIN
  }
  transform_param {
    scale: 0.00390625
  }
  data_param {
    source: "/Users/jwithanawasam/MachineLearning/Caffe/caffe/Traffic/training/train_lmdb"
    batch_size: 16
    backend: LMDB
  }
}
layer {
  name: "mnist"
  type: "Data"
  top: "data"
  top: "label"
  include {
    phase: TEST
  }
  transform_param {
    scale: 0.00390625
```

```
  }
  data_param {
    source: "/Users/jwithanawasam/MachineLearning/Caffe/caffe/Traffic/testing/test_lmdb"
    batch_size: 16
    backend: LMDB
  }
}
layer {
  name: "conv1"
  type: "Convolution"
  bottom: "data"
  top: "conv1"
  param {
    lr_mult: 1
  }
  param {
    lr_mult: 2
  }
  convolution_param {
    num_output: 20
    kernel_size: 5
    stride: 1
    weight_filler {
      type: "xavier"
    }
    bias_filler {
      type: "constant"
    }
  }
}
layer {
  name: "pool1"
  type: "Pooling"
  bottom: "conv1"
```

```
  top: "pool1"
  pooling_param {
    pool: MAX
    kernel_size: 2
    stride: 2
  }
}
layer {
  name: "conv2"
  type: "Convolution"
  bottom: "pool1"
  top: "conv2"
  param {
    lr_mult: 1
  }
  param {
    lr_mult: 2
  }
  convolution_param {
    num_output: 50
    kernel_size: 5
    stride: 1
    weight_filler {
      type: "xavier"
    }
    bias_filler {
      type: "constant"
    }
  }
}
layer {
  name: "pool2"
  type: "Pooling"
  bottom: "conv2"
```

```
  top: "pool2"
  pooling_param {
    pool: MAX
    kernel_size: 2
    stride: 2
  }
}
layer {
  name: "ip1"
  type: "InnerProduct"
  bottom: "pool2"
  top: "ip1"
  param {
    lr_mult: 1
  }
  param {
    lr_mult: 2
  }
  inner_product_param {
    num_output: 500
    weight_filler {
      type: "xavier"
    }
    bias_filler {
      type: "constant"
    }
  }
}
layer {
  name: "relu1"
  type: "ReLU"
  bottom: "ip1"
  top: "ip1"
}
```

```
layer {
  name: "ip2"
  type: "InnerProduct"
  bottom: "ip1"
  top: "ip2"
  param {
    lr_mult: 1
  }
  param {
    lr_mult: 2
  }
  inner_product_param {
    num_output: 2
    weight_filler {
      type: "xavier"
    }
    bias_filler {
      type: "constant"
    }
  }
}
layer {
  name: "accuracy"
  type: "Accuracy"
  bottom: "ip2"
  bottom: "label"
  top: "accuracy"
  include {
    phase: TEST
  }
}
layer {
  name: "loss"
  type: "SoftmaxWithLoss"
```

bottom: "ip2"
bottom: "label"
top: "loss" }

Appendix E

Image Classification using PyCaffe

```
import numpy as np
import matplotlib.pyplot as plt
import caffe

DEPLOY_LOCATION =
'/Users/jwithanawasam/MachineLearning/Caffe/caffe/Traffic/traffic_deploy.prototxt'
MODEL_LOCATION =
'/Users/jwithanawasam/MachineLearning/Caffe/caffe/Traffic/model_iter_10000.caffemodel'
IMAGE_LOCATION = '/Users/jwithanawasam/MachineLearning/Caffe/caffe/Traffic/
/images/traffic_img_00001.jpeg'

caffe.set_mode_cpu()
net = caffe.Classifier(DEPLOY_LOCATION, MODEL_LOCATION)
input_image = caffe.io.load_image(IMAGE_LOCATION)
plt.imshow(input_image)
prediction = net.predict([input_image])
plt.plot(prediction[0])
print 'Traffic density state:', prediction[0].argmax()
plt.show()
```

Appendix F

Road traffic simulation using SUMO

Node definition

```
<nodes version="0.13" xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xsi:noNamespaceSchemaLocation="http://sumo-sim.org/xsd/nodes_file.xsd">
    <location netOffset="0.00,0.00" convBoundary="0.00,0.00,300.00,300.00"
origBoundary="0.00,0.00,300.00,300.00" projParameter="!"/>

    <node id="0/0" x="0.00" y="0.00" type="priority"/>
    <node id="0/1" x="0.00" y="150.00" type="traffic_light" tl="0/1"/>
    <node id="0/2" x="0.00" y="300.00" type="priority"/>
    <node id="1/0" x="150.00" y="0.00" type="traffic_light" tl="1/0"/>
    <node id="1/1" x="150.00" y="150.00" type="traffic_light" tl="1/1"/>
    <node id="1/2" x="150.00" y="300.00" type="traffic_light" tl="1/2"/>
    <node id="2/0" x="300.00" y="0.00" type="priority"/>
    <node id="2/1" x="300.00" y="150.00" type="traffic_light" tl="2/1"/>
    <node id="2/2" x="300.00" y="300.00" type="priority"/>
</nodes>
```

Edges definition

```
<edges version="0.13" xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" xsi:noNamespaceSchemaLocation="http://
sumo-sim.org/xsd/edges_file.xsd">
    <edge id="0/0to0/1" from="0/0" to="0/1" priority="-1" numLanes="1" speed="13.90"/>
    <edge id="0/0to1/0" from="0/0" to="1/0" priority="-1" numLanes="1" speed="13.90"/>
    <edge id="0/1to0/0" from="0/1" to="0/0" priority="-1" numLanes="1" speed="13.90"/>
    <edge id="0/1to0/2" from="0/1" to="0/2" priority="-1" numLanes="1" speed="13.90"/>
    <edge id="0/1to1/1" from="0/1" to="1/1" priority="-1" numLanes="1" speed="13.90"/>
    <edge id="0/2to0/1" from="0/2" to="0/1" priority="-1" numLanes="1" speed="13.90"/>
    <edge id="0/2to1/2" from="0/2" to="1/2" priority="-1" numLanes="1" speed="13.90"/>
    <edge id="1/0to0/0" from="1/0" to="0/0" priority="-1" numLanes="1" speed="13.90"/>
    <edge id="1/0to1/1" from="1/0" to="1/1" priority="-1" numLanes="1" speed="13.90"/>
    <edge id="1/0to2/0" from="1/0" to="2/0" priority="-1" numLanes="1" speed="13.90"/>
    <edge id="1/1to0/1" from="1/1" to="0/1" priority="-1" numLanes="1" speed="13.90"/>
    <edge id="1/1to1/0" from="1/1" to="1/0" priority="-1" numLanes="1" speed="13.90"/>
    <edge id="1/1to1/2" from="1/1" to="1/2" priority="-1" numLanes="1" speed="13.90"/>
    <edge id="1/1to2/1" from="1/1" to="2/1" priority="-1" numLanes="1" speed="13.90"/>
    <edge id="1/2to0/2" from="1/2" to="0/2" priority="-1" numLanes="1" speed="13.90"/>
    <edge id="1/2to1/1" from="1/2" to="1/1" priority="-1" numLanes="1" speed="13.90"/>
    <edge id="1/2to2/2" from="1/2" to="2/2" priority="-1" numLanes="1" speed="13.90"/>
    <edge id="2/0to1/0" from="2/0" to="1/0" priority="-1" numLanes="1" speed="13.90"/>
    <edge id="2/0to2/1" from="2/0" to="2/1" priority="-1" numLanes="1" speed="13.90"/>
    <edge id="2/1to1/1" from="2/1" to="1/1" priority="-1" numLanes="1" speed="13.90"/>
    <edge id="2/1to2/0" from="2/1" to="2/0" priority="-1" numLanes="1" speed="13.90"/>
    <edge id="2/1to2/2" from="2/1" to="2/2" priority="-1" numLanes="1" speed="13.90"/>
    <edge id="2/2to1/2" from="2/2" to="1/2" priority="-1" numLanes="1" speed="13.90"/>
    <edge id="2/2to2/1" from="2/2" to="2/1" priority="-1" numLanes="1" speed="13.90"/>
</edges>
```

Connections definition

```
<connections version="0.13" xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xsi:noNamespaceSchemaLocation="http://sumo-sim.org/xsd/connections_file.xsd">

    <connection from="0/0to0/1" to="0/1to1/1" fromLane="0" toLane="0"/>
    <connection from="0/0to0/1" to="0/1to0/2" fromLane="0" toLane="0"/>

    <connection from="0/0to1/0" to="1/0to2/0" fromLane="0" toLane="0"/>
    <connection from="0/0to1/0" to="1/0to1/1" fromLane="0" toLane="0"/>

    <connection from="0/1to0/0" to="0/0to1/0" fromLane="0" toLane="0"/>

    <connection from="0/1to0/2" to="0/2to1/2" fromLane="0" toLane="0"/>

    <connection from="0/1to1/1" to="1/1to1/0" fromLane="0" toLane="0"/>
    <connection from="0/1to1/1" to="1/1to2/1" fromLane="0" toLane="0"/>
    <connection from="0/1to1/1" to="1/1to1/2" fromLane="0" toLane="0"/>

    <connection from="0/2to0/1" to="0/1to0/0" fromLane="0" toLane="0"/>
    <connection from="0/2to0/1" to="0/1to1/1" fromLane="0" toLane="0"/>

    <connection from="0/2to1/2" to="1/2to1/1" fromLane="0" toLane="0"/>
    <connection from="0/2to1/2" to="1/2to2/2" fromLane="0" toLane="0"/>

    <connection from="1/0to0/0" to="0/0to0/1" fromLane="0" toLane="0"/>

    <connection from="1/0to1/1" to="1/1to2/1" fromLane="0" toLane="0"/>
    <connection from="1/0to1/1" to="1/1to1/2" fromLane="0" toLane="0"/>
    <connection from="1/0to1/1" to="1/1to0/1" fromLane="0" toLane="0"/>

    <connection from="1/0to2/0" to="2/0to2/1" fromLane="0" toLane="0"/>
</connections>
```

Routes definition for vehicles

```
<?xml version="1.0" encoding="UTF-8" standalone="no"?><routes xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xsi:noNamespaceSchemaLocation="http://sumo-sim.org/xsd/routes_file.xsd"><vType accel="1.8" color="red" decel="4.5"
id="eme" length="5" maxSpeed="140" sigma="0.5" vClass="emergency"/><vType accel="0.4" color="blue" decel="8.5"
id="bus" length="10" maxSpeed="40" sigma="0.5" vClass="bus"/><vType accel="2.0" decel="4.5" id="nor" length="5"
maxSpeed="120" sigma="0.5"/><vehicle depart="1.00" id="0" type="nor">
        <route edges="2/2to2/1 2/1to1/1 1/1to1/2 1/2to2/2"/>
    </vehicle><vehicle depart="5.00" id="1" type="eme">
        <route edges="2/2to1/2 1/2to1/1 1/1to0/1 0/1to0/2 0/2to1/2 1/2to2/2 2/2to2/1"/>
    </vehicle><vehicle depart="9.00" id="2" type="eme">
        <route edges="2/1to2/2 2/2to1/2 1/2to1/1 1/1to0/1 0/1to0/0 0/0to1/0 1/0to1/1"/>
    </vehicle><vehicle depart="13.00" id="3" type="eme">
        <route edges="1/1to2/1 2/1to2/0 2/0to1/0 1/0to1/1 1/1to1/2 1/2to2/2 2/2to2/1"/>
    </vehicle><vehicle depart="17.00" id="4" type="eme">
        <route edges="0/2to0/1 0/1to1/1 1/1to1/2"/>
    </vehicle><vehicle depart="21.00" id="5" type="eme">
        <route edges="0/2to1/2 1/2to1/1 1/1to1/0 1/0to0/0 0/0to0/1"/>
    </vehicle><vehicle depart="25.00" id="6" type="eme">
        <route edges="2/0to2/1 2/1to2/2 2/2to1/2"/>
</vehicle></routes>
```

Traffic signal phase and cycle definition

```
<tlLogics version="0.13" xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xsi:noNamespaceSchemaLocation="http://sumo-sim.org/xsd/tllogic_file.xsd">
    <tlLogic id="0/1" type="static" programID="0" offset="0">
        <phase duration="31" state="GgrrGG"/>
        <phase duration="4" state="ygrryy"/>
        <phase duration="6" state="rGrrrr"/>
        <phase duration="4" state="ryrrrr"/>
        <phase duration="31" state="rrGGGr"/>
        <phase duration="4" state="rryyyr"/>
    </tlLogic>
    <tlLogic id="1/0" type="static" programID="0" offset="0">
        <phase duration="31" state="rrGGGg"/>
        <phase duration="4" state="rryyyg"/>
        <phase duration="6" state="rrrrrG"/>
        <phase duration="4" state="rrrrry"/>
        <phase duration="31" state="GGGrrr"/>
        <phase duration="4" state="yyyrrr"/>
    </tlLogic>
    <tlLogic id="1/1" type="static" programID="0" offset="0">
        <phase duration="31" state="rrrGGgrrrGGg"/>
        <phase duration="4" state="rrryygrrryyg"/>
        <phase duration="6" state="rrrrrGrrrrrG"/>
        <phase duration="4" state="rrrrryrrrrry"/>
        <phase duration="31" state="GGgrrrGGgrrr"/>
        <phase duration="4" state="yygrrryygrrr"/>
        <phase duration="6" state="rrGrrrrrGrrr"/>
        <phase duration="4" state="rryrrrrryrrr"/>
    </tlLogic>
    <tlLogic id="1/2" type="static" programID="0" offset="0">
        <phase duration="31" state="GgrrGG"/>
        <phase duration="4" state="ygrryy"/>
        <phase duration="6" state="rGrrrr"/>
        <phase duration="4" state="ryrrrr"/>
        <phase duration="31" state="rrGGGr"/>
        <phase duration="4" state="rryyyr"/>
    </tlLogic>
    <tlLogic id="2/1" type="static" programID="0" offset="0">
        <phase duration="31" state="GGGgrr"/>
        <phase duration="4" state="yyygrr"/>
        <phase duration="6" state="rrrGrr"/>
        <phase duration="4" state="rrryrr"/>
        <phase duration="31" state="GrrrGG"/>
        <phase duration="4" state="yrrryy"/>
    </tlLogic>
```

Appendix G

Retrieve controlled links for a given traffic light

In order to control traffic signal sequences, all the information related to controlled link such as incoming lane and outgoing lane is required. The relevant code segments and explanation on how to retrieve controlled link information for a given traffic light from SUMO via TRACI and TRASMAPI is given here.

TRASMAPI acts as a Java API to communicate with TRACI API in SUMO to get the traffic information of the traffic control simulation dynamically.

Once we connect to the SUMO simulation environment via a TCP connection using given port as client, we can communicate with SUMO using TCP messages. TRACI API provides a protocol, either to control the environment or to retrieve information using commands that are issued as TCP messages. TRACI command to retrieve traffic light information is 0xa2. For a given command, there will be set of variables that represents different details related to that particular command. Variable to get links controlled by a traffic light is 0x27. When, SUMO receives 0x27 command, it will return the incoming lane, outgoing lane and via lane for a given traffic light that are embedded as a CompoundObject as TCP message called 0xb2 response variable. The structure of the CompoundObject is given below.

integer	controlled links	...	controlled links
Length (number of signals)	**links controlled by signal 0**	**...**	**links controlled by signal n-1**

Controlled links:

int	stringlist	...	stringlist
number of controlled links	**link 0**	**...**	**link n-1**

Source:
http://www.sumo.dlr.de/wiki/TraCI/Traffic_Lights_Value_Retrieval#Command_0xa2:_Get_Traffic_Lights_Variable

Relevant code segments for the above custom implementation in TRASMAPI is given below.

```
// Receive controlled links information from SUMO
public ArrayList<ControlledLink> getControlledLinks(){

        Command command = new
        Command(Constants.CMD_GET_TL_VARIABLE);
        Content content = new Content(Constants.TL_CONTROLLED_LINKS,id);

        command.setContent(content);

        RequestMessage request = new RequestMessage();
        request.addCommand(command);

        ResponseMessage response = SumoCom.query(request);
        Content rspContent = response.validate( (byte)
        Constants.CMD_GET_TL_VARIABLE, (byte)
        Constants.RESPONSE_GET_TL_VARIABLE,
        (byte)  Constants.TL_CONTROLLED_LINKS, (byte)
        Constants.TYPE_COMPOUND);

        controlledLinks = rspContent.getControlledLinksFromCompoundObject();

        return controlledLinks;
}

// Retrieve controlled links information from CompoundObject in TRACI
public ArrayList<ControlledLink> getControlledLinksFromCompoundObject(){

        ArrayList<ControlledLink> controlledLinks = new
        ArrayList<ControlledLink>();

        int currentPointer = 5;
        int skipCount = 6;
        int integerCount = 4;

        int numberOfSignals = readInt(currentPointer);
        currentPointer += integerCount;

        for (int signal=0; signal < numberOfSignals; signal++)
        {
                ControlledLink link = new ControlledLink();

                currentPointer = currentPointer + skipCount + integerCount;
                String incoming = new String();
                int linkCharLength_incoming = readInt(currentPointer);
                currentPointer += integerCount;
```

```
                for (int character=0; character < linkCharLength_incoming;
                character++){
                        byte charact = varValue.get(currentPointer);
                        incoming += (char) charact;
                        currentPointer++;
                }
                link.incoming = incoming;
                String outgoing = new String();
                int linkCharLength_outgoing = readInt(currentPointer);
                currentPointer += integerCount;

                for (int character=0; character < linkCharLength_outgoing;
                character++){

                        byte charact = varValue.get(currentPointer);
                        outgoing += (char) charact;
                        currentPointer++;
                }
                link.outgoing = outgoing;

                String tls_index = new String();
                int linkCharLength_tls = readInt(currentPointer);
                currentPointer += integerCount;

                for (int character=0; character < linkCharLength_tls;
                character++){
                        byte charact = varValue.get(currentPointer);
                        tls_index += (char) charact;
                        currentPointer++;
                }
                link.tls_index = tls_index;
                controlledLinks.add(link);
        }
        return controlledLinks;
}
```