electronics

MDPI

Article

Intelligent Meta-Heuristic-Based Optimization of Traffic Light Timing Using Artificial Intelligence Techniques

Mohammed A. Khasawneh * and Anjali Awasthi

Department of Information System and Engineering, Concordia University, Montreal, QC H3G 1M8, Canada; anjali.awasthi@concordia.ca

* Correspondence: m_khasa@live.concordia.ca

Abstract: This research examines worldwide concerns over traffic congestion, encompassing aspects such as security, parking, pollution, and congestion. It specifically emphasizes the importance of implementing appropriate traffic light timing as a means to mitigate these issues. The research utilized a dataset from Montreal and partitioned the simulated area into various zones in order to determine congestion levels for each individual zone. A range of prediction algorithms has been employed, such as Long Short-Term Memory (LSTM), Decision Tree (DT), Recurrent Neural Network (RNN), Auto-Regressive Integrated Moving Average (ARIMA), and Seasonal Auto-Regressive Integrated Moving Average (SARIMA), to predict congestion levels at each traffic light. This information was used in a mathematical formulation to minimize the average waiting time for vehicles inside the road network. Many meta-heuristics were analyzed and compared, with the introduction of an Enhanced Bat Algorithm (EBAT) suggested for addressing the traffic signal optimization problem. Three distinct scenarios are described: fixed (with a constant green timing of 40 s), dynamic (where the timing changes in real-time based on the current level of congestion), and adaptive (which involves predicting congestion ahead of time). The scenarios are studied with low and high congestion scenarios in the road network. The Enhanced Bat Algorithm (EBAT) is introduced as a solution to optimize traffic signal timing. It enhances the original Bat algorithm by incorporating adaptive parameter tuning and guided exploration techniques that are informed by predicted congestion levels. The EBAT algorithm provides a more effective treatment for congestion problems by decreasing travel time, enhancing vehicle throughput, and minimizing pollutant emissions.

Keywords: autoregressive integrated moving average; Harris–Hawks optimization; machine learning; particle swarm optimization; seasonal autoregressive integrated moving average

Citation: Khasawneh, M.A.; Awasthi, A. Intelligent Meta-Heuristic-Based Optimization of Traffic Light Timing Using Artificial Intelligence Techniques. *Electronics* **2023**, *12*, 4968. https://doi.org/10.3390/electronics12244968

Academic Editor: Felipe Jiménez

Received: 26 September 2023
Revised: 19 November 2023
Accepted: 29 November 2023
Published: 11 December 2023

1. Introduction

The rapid increase in the number of automobiles has contributed to congestion, pollution, and logistical delays. The financial toll of gridlock is rising. Specifically, Forbes reported in 2014 [1] that traffic congestion costs the United States economy USD 124 billion annually. Around one percent of the Gross Domestic Product (GDP) [2] of the European Union is lost due to traffic congestion costs. A major obstacle to creating smart cities is the exponential rise in vehicle ownership despite inadequate public transportation facilities. Increased traffic congestion and fuel costs are just two of the numerous negative consequences of dense vehicle numbers, which also include air and noise pollution, stress and disease, accidents, and high fuel costs. An increase in the number of individual vehicles is a result of the development of nations around the globe. This has led to heavier traffic in major urban centers. Yearly, traffic jams get worse in Indonesia. Indonesia, which in 2015 ranked No. 11 on the list of the world's most jammed countries behind only Brazil and Argentina, had an estimated jamming time of 40.58 min on average. With a time index of 49.44 min in 2017, the position moved up to second in 2017 [3]. We need a better traffic management system as a result. Time and money could be saved with efficient and trustworthy traffic management and control. With the advent of the IoT, we are one step closer

to completely automated, highly controlled processes and systems [4]. In order to detect, gather, and transmit data, IoT-based ITM sensors are integrated into autonomous vehicles and smart devices. Machine Learning (ML) has the potential to improve transportation in other ways as well. Traffic congestion, delays, and fatalities are all too common due to the various flaws in the current transportation management systems. Long waiting times, unnecessary fuel consumption, and rising carbon emissions are only some of the problems with the current vehicle traffic light signal control system. Drivers experience a great deal of stress as a result, and emergency and other high priority vehicles are delayed in their arrivals. Congestion can be alleviated through the smart management of traffic lights. Controlling traffic lights intelligently is essential for a smoothly running transportation network [5]. An intelligent traffic signal control system would automatically adapt to the current flow of traffic, as opposed to the static regulations used by the current system. Most traffic signals today are still programmed to operate on a predetermined schedule [6,7] rather than being based on observations of actual traffic patterns. Some recent research has proposed custom-made criteria based on actual traffic data [8,9]. Unfortunately, these rules remain statically specified, making them inflexible in the face of fluctuating traffic conditions. Existing research, however, has not yet validated the methodology using real-world traffic data and has instead focused solely on analyzing incentives without interpreting policy. An improved deep reinforcement learning model for managing traffic lights is proposed in this paper. To ensure the efficacy of our method, we use a massive collection of actual traffic data. Intriguing case studies of policies gleaned from actual data are also presented. Reinforcement learning [10–12] has been used to dynamically adapt traffic lights to current traffic conditions. Both representing the environment and modeling the relationship between the environment and the decision provide significant obstacles to the practical use of traditional reinforcement learning. In order to overcome these obstacles, researchers [11,12] have recently applied deep reinforcement learning methods, such as Deep Q-Learning (DQN), to the traffic light management problem. Many reasons have urged us to study the traffic congestion problem, since it might affect various sectors in our daily life. The first incentive is to reduce the congestion in the road network and hence the waiting time for roads users (drivers and pedestrians), which will directly affect each user's productivity in their daily life; it might be a small effect for each individual alone, but if we examine the change in a holistic manner, it will be a huge effect on a societal level. Another motivation is to decrease the CO and CO_2 gas emissions from vehicles, which will also have a massive impact on one of the main problems in the world at the current time, which is the destruction of the ozone layer that will directly affect people's health if not solved. In addition to that, a reduced fuel consumption is achieved, which gives scientists a longer time to look for other options such as renewable energy to replace the current energy sources that will run out in the near future. Congestion mitigation will also drastically increase the road users' satisfaction and quality of life, and decrease the number of car accidents, which could save lives around the world [13]. Many research papers have used meta-heuristic techniques to solve different problems in different fields. The researchers in [14] introduced a Two-Level Particle Swarm Optimization (TLPSO) method for managing credit portfolios, with the goal of reducing losses while staying below financial limits. Comparative investigations show that TLPSO, with its novel dual searching mechanism, outperforms conventional methods like the genetic algorithm and particle swarm optimization. Another paper [15] presented a strategy for improving disaster response logistics by balancing facility placement and transportation routes. Discrete particle swarm optimization and Harris–Hawks optimization were combined into a new hybrid algorithm that was introduced to solve this difficult issue. A COVID-19 case study conducted in Wuhan validated the method's efficacy, showing that it is both more accurate and more efficient than alternative approaches. Other studies have used meta-heuristic algorithms to find near-optimal solutions [16,17]. According to the No Free Lunch (NFL) theory, there is no single meta-heuristic that is suitable for any problem, which is why in this paper, we

investigated many meta-heuristics and compared them together to determine which one is better, as shown in the results section below.

After studying the literature of intelligent traffic lights, many research gaps have been discovered as explained in more detail in the related work section below. The problems that we will address are briefly summarized as follows:

- Increased road traffic congestion due to the increasing numbers of road users and the limited capacity of the current road infrastructure that cannot be extended easily, where this main problem could cause many subproblems such as increased waiting time, increased number of accidents and increased levels of CO and CO_2 gas emissions [18].
- A current problem exists in the proposed meta-heuristic algorithms in the literature concerning both the execution time and convergence speed, which are two important factors that should be minimized/improved when designing a meta-heuristic algorithm. This will help us in making faster decisions and better solutions that are nearer to the optimal solution.
- A current problem exists in the proposed machine learning models in the literature when it comes to the prediction accuracy of the solution. Better predictions will enable us to make more wise future decision and help solve the problem before occurring.

The selected algorithms in our proposed solution are renowned for their ability to achieve a balance between exploration (seeking out new regions) and exploitation (improving known good areas), which is essential in traffic light optimization, where both new solutions and improvement of old solutions are required. Additionally, they provide a favorable combination of randomization and deterministic principles, which aids in adjusting to the unpredictable characteristics of traffic patterns. The improved BAT algorithm, specifically, may provide a distinct benefit in terms of speed and convergence, rendering it more appropriate for real-time applications such as traffic signal optimization.

1.1. Motivations

Many reasons have urged us to study the traffic congestion problem, since it might affect various sectors in our daily life. The first incentive is to reduce the congestion in the road network and hence the waiting time for road users (drivers and pedestrians), which will directly affect each user's productivity in their daily life; it might be a small effect for each individual alone but if we examine the change in a holistic manner, it will be a huge effect on a societal level. Another motivation is to decrease the CO and CO_2 gas emissions from the vehicles, which also have a massive impact on one of the main problems in the world at the current time, which is the destruction of the ozone layer that will directly affect people's health if not solved. In addition to that, a reduced fuel consumption is achieved, which gives scientists a longer time to look for other options such as renewable energy to replace the current energy sources that will run out in the near future. The congestion mitigation will also drastically increase the road users' satisfaction and quality of life, and decrease the number of car accidents, which could save lives around the world.

1.2. Contributions

One of our contributions is in developing an enhanced approach for the BAT algorithm. In order to achieve this, we modified the exploration phase of the algorithm. In the BAT algorithm, there is an exploration phase, where the solution is generated randomly, and then its fitness is calculated and compared with the best solution obtained so far, and it is updated accordingly. In our modified version of the BAT algorithm, the exploration phase is not fully random but is semi-random, which means that the algorithm is directed to obtain a better solution by taking into consideration the congestion status. Another enhancement is to tune the parameters in the BAT algorithm to have a better result in terms of the convergence rate and the quality of the solution.

1.3. Organization

The rest of the paper is organized as follows. The study regarding existing solutions in the same field proposed in the literature is presented in Section 2. Section 3 introduces the Background of prediction algorithms which we used. Section 4 introduced the Prediction Techniques For Road Network Congestion. And Section 5 introduced our Proposed Methodology, Conclusions are shown in Section 6.

2. Related Work

The current traffic light control system uses a traffic light adjustment with a predetermined period that does not take into account real-time traffic or the traffic scenario. Without taking into account the current traffic conditions, a fixed-duration traffic light control system divides the total duration of the signal into equal portions for each phase. Sensors, such as inductive loop detectors, are used by various algorithms for traffic light management to gather traffic data [19]. The time between flashes of red and green is calculated based on the provided data. Such traffic light control systems, however, will become ineffective at times when the volume of traffic entering a city is particularly high [20]. Complex optimization problems can be solved with the help of a technique known as meta-heuristics. City traffic signals also make use of meta-heuristics. Several researchers split the meta-heuristics employed by urban traffic signals into two steps [21]. The first step is to coordinate amongst traffic lights to boost the efficiency of the meta-heuristic algorithms. The second is to switch to an adaptive time signal controller using meta-heuristics to determine the best possible green signal time. The first strategy involves optimizing the Artificial Bee Colony (ABC) algorithm's performance by locating its ARPD value's minimum and then performing a Harmony Search (HS). The artificial bee colony algorithm achieved the lowest average ARPD of 0.74. The second approach increased the vehicle's speed by 27.6%. Furthermore, we compare the effectiveness of different meta-heuristic algorithms for managing metropolitan traffic light schedules. The DBSCAN clustering technique, which is based on machine learning, is built into the system to detect accidental outliers. The proposed algorithm automatically modifies the timing of the lights at intersections to better accommodate traffic and anticipated pedestrian and vehicle movements. It not only allows for a more seamless movement, but also saves time on the road by easing traffic. When it comes to the frequency and severity of traffic accidents in metropolitan areas, intersections play a pivotal role. By alerting drivers to potential danger ahead, the Vehicular Ad hoc NETwork (VANET) can prevent accidents near crossroads. However, VANET performance should be improved to ensure that communications, especially safety messages, reach their intended recipients. By regulating the flow of data, we can reduce the likelihood of packet loss and delay and boost the dependability of VANETs. Using Road-Side Units (RSUs) at intersections, a centralized and targeted data congestion reduction technique was presented [22]. The suggested technique is divided into three parts: congestion detection, message grouping, and congestion reduction. To identify data congestion, this method monitors the channel utilization rate. Machine learning algorithms collect the messages, sort them, and then group them into categories. The K-means algorithm sorts the messages into groups according to their size, their reliability, and their nature. The transmission range, transmission rate, contention window size, and arbitration interframe spacing are all factors that the data congestion management unit takes into account while tailoring settings for individual clusters [23]. In order to reduce communication collisions, RSUs at intersections transmit the determined communication parameters to the stopped vehicles before red lights. In order to facilitate communications between vehicles and fixed infrastructure, ITS makes use of VANet. Dedicated Short-Range Communication (DSRC) and Wireless Access in a Vehicular Environment (WAVE) are two documents that specify the standards and protocols used in VANet communications. The IEEE 802.11p and IEEE1609 [24] standards both fall under the WAVE umbrella. Resources, network services, multi-channel operations, security services, etc., are all managed using these standards. Road-Side Units (RSUs) and On-Board Units (OBUs) are used in VANets to facilitate vehicle-

to-infrastructure and vehicle-to-vehicle communications, respectively. Road-Side Units (RSUs) are installed along the side of the road, while On-Board Units (OBUs) are mounted in moving vehicles [24,25]. An agent in this system emulates human learning by observing traffic patterns and adapting signal timing accordingly. The deep reinforcement learning strategy [26], one of the machine learning techniques widely recognized to handle these kinds of issues, is utilized to model brain behavior for the processing phase. To optimize rewards, such as reducing average waiting times in traffic control scenarios, reinforcement learning encourages agents to learn the best course of action by observing and interacting with their surroundings [27]. For dynamic traffic signal control systems, research on the reinforcement learning technique, which defines the states by the waiting line length [28,29], is sparse. However, the queue length does not always reflect actual traffic conditions. More accurate traffic light management, taking into account variations in vehicle types including ambulances, fire trucks, school buses, and police cars, is possible due to the proliferation of high-definition cameras and sensors made possible by technological progress [30]. A Dynamic and Intelligent Traffic Light Control System (DITLCS) was presented as a solution, which uses inputs such as real-time traffic data to address these problems and improve the system's overall efficiency. In addition, the proposed DITLCS can operate in three distinct modes: Fair Mode (FM), Priority Mode (PM), and Emergency Mode (EM), where the first prioritizes all vehicles equally, the second prioritizes vehicles based on their category, and the third prioritizes emergency vehicles above all others. Additionally, a deep reinforcement learning model was presented to change the traffic lights between phases (red, green, and yellow), and a fuzzy inference system chooses one of three modes (FM, PM, and EM) based on the data. They researchers used an open-source simulator (the Simulation of Urban Mobility (SUMO) systems) to test DITLCS by simulating it on a map of the city of Gwalior, India. Urban Traffic Light Scheduling Problems (UTSLPs) are studied by some authors, and they aim to minimize the sum of all vehicle and pedestrian delays within a specified time window. The UTLSP is first described using a centralized model that presents the cost functions and constraints of the two objectives [31]. To evaluate and rank approaches in terms of the two objectives, we use a metric that does not rely on a dominance strategy. Second, meta-heuristics like Harmony Search (HS) and Artificial Bee Colony (ABC) are used to find a solution to the UTLSP. There are four distinct approaches to managing traffic lights: fixed-time versus adaptive, and standalone versus collaborative. Some of the more prominent plans put forth in recent decades are MAXBAND [32,33], TRANSYT, SCOOT [34], OPAC [35], PRODYN [36], CRONOS [37], and RHODES [38]. The enormous computational cost of optimization becomes the key challenge for real-time scheduling because of the massive size of a typical traffic network, which can have thousands of road links and hundreds of intersections. Many researchers have presented different optimization strategies for the traffic light control problem. Case studies have been conducted to evaluate the proposed architecture in a real-world traffic network [39], and an architecture for optimizing traffic light cycles using a Genetic Algorithm (GA) and Cellular Automata Simulators (CAS) has been developed [40]. Case studies in real-world traffic networks have confirmed the substantial benefits of the suggested approach [41], and a PSO-based approach was proposed and deployed, coupled with a microscopic traffic simulator [42]. A model of a traffic network with continuous flow is discussed in [43], with the decision to change traffic lights being depicted as a discrete event. Meta-heuristic optimization techniques, such as the Genetic Algorithm (GA), Particle Swarm Optimization (PSO), Differential Evolution (DE), Harmony Search (HS), and the Artificial Bee Colony (ABC) algorithm, are among the most promising new methods for addressing traffic light management issues [44]. The objective of this work is to create advanced techniques using meta-heuristics to enhance the management of traffic at a single traffic light in Dhahran, Saudi Arabia. The work attempts to optimize signal timing plans in order to enhance the level of service (LOS) at intersections, using the Genetic Algorithm (GA) and Differential Evolution (DE). The findings indicate that both the Genetic Algorithm (GA) and Differential Evolution (DE) generate a methodical schedule for signal timings, resulting in a notable

decrease in travel time delay ranging from 15 to 35% as compared to the current conditions. While DE exhibits faster convergence to the target function, GA surpasses DE in terms of solution quality, specifically in minimizing vehicle delay. The validation results showcase the sufficiency and resilience of the proposed approaches, highlighting the significance of traffic signal control in intelligent transportation systems [45]. The use of an Adaptive Neuro-Fuzzy Inference System (ANFIS) is a viable approach to tackle the problem of traffic signal optimization. The ANFIS traffic signal controller employs meta-heuristic algorithms to determine the optimal duration for green lights at traffic signals, thereby minimizing both the queue length and the delay. The controller was simulated and implemented on intersections, showcasing exceptional efficacy in traffic prediction and control [46]. Another study presents the Meta-Heuristic Robust plan Approach (MHRA), which is a framework for fixed-time traffic lights that operates in an offline scenario-based manner. The MHRA evaluates the most effective signal schemes for different demand scenarios. The effectiveness of the framework was confirmed through numerical experiments. These experiments demonstrated that the framework surpasses nominal plans and consistently performs well even when faced with changing demand. Comparing MHRA with other scenario-based methodologies through benchmarking demonstrates its superior efficiency [47]. Another research paper discussed how to create an advanced traffic control protocol utilizing the Non-dominated Sorting Genetic Algorithm II (NSGA-II) at isolated signalized intersections in Dhahran, Saudi Arabia. The Measures of Effectiveness (MOEs) taken into account encompassed mean vehicle latency, overall vehicle stops, mean fuel consumption, and vehicular emissions. The simulations demonstrated that the proposed strategy effectively optimized performance metrics, resulting in a 16% to 23% improvement in MOEs compared to the current settings. A Synchro traffic light simulation and optimization program was used to conduct an optimization analysis. The results indicated that the proposed approach performed better than the Synchro optimization results in terms of the percentage reduction in MOE values [48]. In [49], the paper showcases a case study of a significant intersection in Timisoara, where the synchronization of traffic lights is accomplished by the utilization of firefly algorithms. Although these algorithms are effective, the study demonstrated that more enhancements to traffic flow are still possible. In [50], the authors provided a thorough analysis of the most recent developments in swarm intelligence and evolutionary approaches as they are applied to traffic control and optimization in metropolitan networks. The literature is classified according to meta-heuristics, optimization targets, and signal control parameters. This study examines the advantages and disadvantages of each approach, discusses the current difficulties, potential opportunities, and future direction of research, and conducts a thorough assessment of existing literature to fill in the gaps in traffic signal control systems.

These are the restrictions we found in the existing literature and included in our study. It is possible that the growing number of lights and complexity of the road network will overwhelm some algorithms, and our algorithm is scalable since it divides the city into different zones, where each zone has only 11 traffic lights in it. Other papers in the literature dealt with this problem as a local problem, meaning that it handles the congestion at each traffic light separately without communicating the congestion status between different traffic lights in the same zone. Most of the research work in the literature solves this problem by using one or two meta-heuristic algorithms; for our proposed solution, we compared seven different meta-heuristic algorithms to decide which one offers the best results, and we also enhanced the BAT algorithm to achieve better results by changing the exploration phase of the algorithm by making it semi-random and not fully random. Last but not least, we dealt with the problem as a two-objectives minimization problem, where the priority is given to emergency vehicles.

3. Background

3.1. Prediction Algorithms

3.1.1. Recurrent Neural Network (RNN)

Traditional neural networks are a class of Artificial Neural Networks (ANNs) and are a series of algorithms that are closely derived from the working of the human brain, where they interpret sensory data, whether historical or live data (offline or online data) by a type of machine perception, labeling or clustering raw input to recognize the patterns. An ANN usually contains many processors that work in parallel and are arranged in layers, where the first layer receives the input information, and each layer receives the output from the layer that precedes it [51]. Recurrent Neural Networks (RNNs) are a type of feedforward neural network that has an internal memory. An RNN is recurrent in nature, as it performs the same function for every input of data while the output of the current input depends on the past computation. After producing the output, it is copied and sent back into the recurrent network. For making a decision, it considers the current input and the output that it has learned from the previous input.

As shown in Figure 1, the RNN takes the X(0) from the sequence of the input and then outputs h(0), which together with X(1) is the input for the next step. Hence, h(0) and X(1) are the input for the next step. Similarly, h(1) from the second step is the input with X(2) for the next step, and so on. Due to this, it can be difficult to train RNNs to solve problems that need to learn long-term temporal dependencies for the reason that the gradient of the loss function vanishes exponentially with time (called the vanishing gradient problem), where RNN has short-term memory, and thus the LSTM is used to solve this problem.

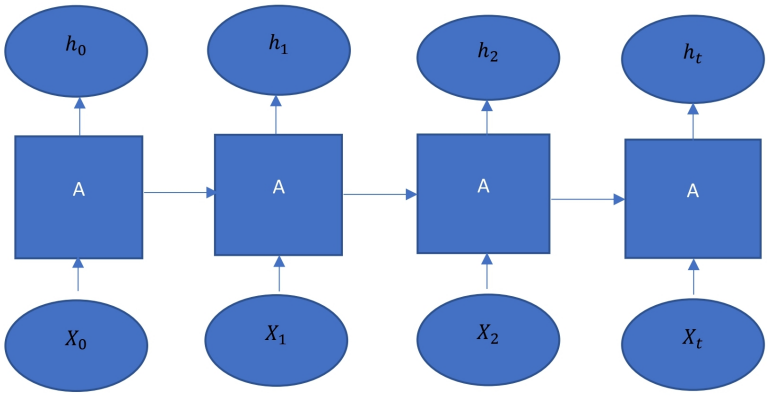

Figure 1. Recurrent neural network.

3.1.2. Long Short-Term Memory (LSTM)

LSTM is similar to the control flow in RNN and has internal mechanisms called gates that can regulate the flow of information; these gates can learn which data in a sequence are important or not and thus keep them or throw them away. As shown in Figure 2, the LSTM has three gates: the forget gate, input gate, and output gate. The forget gate is used to decide which information to keep or forget. Information from a previous hidden state and information from the current input is passed through the sigmoid function that decided which values will be updated by transforming values to be between [0,1]; values between [0,1] closer to 0 will be removed and values closer to 1 will be kept [52].

The cell state acts as a transport highway that transfer relevant information all the way down to the sequence chain, and it can be considered as the memory of the network.

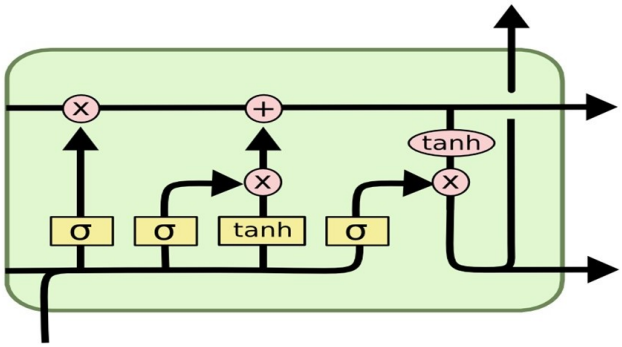

Figure 2. Long short-term memory.

3.1.3. Decision Tree

A decision tree algorithm is a popular machine-learning technique used for both classification and regression tasks. It is a type of supervised learning algorithm that takes a set of input features and produces a decision tree as its output. This decision tree is essentially a flowchart-like structure where each internal node represents a decision based on a particular feature, each branch represents the outcome of that decision, and each leaf node represents a class label or a predicted value [53].

The primary goal of a decision tree algorithm is to create a model that can make accurate predictions or classifications by learning patterns and relationships within the input data. Here is a simplified overview of how a decision tree algorithm works:

Data Splitting:

The algorithm starts with the entire dataset at the root node of the tree.

Feature Selection:

It selects the best feature from the dataset to split the data into subsets. The "best" feature is typically chosen based on criteria that aim to maximize the separation between different classes or minimize the variance within each subset. Common criteria include the Gini impurity, entropy, or mean squared error.

Recursive Process:

The algorithm recursively applies the same process to each subset of data, creating child nodes for each split. This process continues until a stopping condition is met. This condition could be a certain depth of the tree, a minimum number of samples in a node, or other similar criteria.

Leaf Node Labeling:

Once the recursive process is complete, the leaf nodes are assigned class labels (for classification tasks) or predicted values (for regression tasks), usually based on the majority class in a classification problem or the average value in a regression problem.

Prediction:

To make predictions for new data, one starts at the root node and traverses the decision tree by following the path that corresponds to the feature values of the new data. One ends up at a leaf node, and the class label or predicted value associated with that leaf node become the model's prediction.

3.1.4. Auto-Regressive Integrated Moving Average (ARIMA)

The Auto-Regressive Integrated Moving Average (ARIMA) is a popular time-series forecasting approach. It is well-suited to time series data such as stock prices, sales figures, and weather patterns, since it predicts links between observations and lagged values.

3.1.5. Seasonal ARIMA (SARIMA)

SARIMA (Seasonal ARIMA), an extension of ARIMA, can be used to model seasonal patterns in time series data. It adds additional terms to model seasonal fluctuations, increasing prediction precision for repeating data.

3.2. Meta-Heuristics Algorithms

3.2.1. Phases in Meta-Heuristics Algorithms

The phrases "exploration" and "exploitation" refer to two basic features of the behavior of meta-heuristic algorithms like evolutionary algorithms, swarm intelligence, and simulated annealing as they search for optimal solutions in a complex search space.

1. Exploration: First, we have exploration, which is the process of examining a large portion of the solution space for novel and varied solutions. The algorithm places a premium on trying out fresh solutions, even if they do not look promising at first. By searching for other regions of the solution space, exploration aims to prevent getting stuck in local optima (suboptimal solutions). In other words, the goal of exploration is to keep the population of solutions diverse and to stop it from settling too quickly into a suboptimal zone.
2. Exploitation: The second strategy, "exploitation", is a thorough investigation of proven methods in order to hone and enhance them. The algorithm's focus throughout the exploitation phase is on using the insights acquired from previously found optimal solutions to further enhance and perfect them. This is usually done through local search techniques that center on making minor adjustments close to probable solutions. The goal of exploitation is to improve upon already promising solutions so that they converge on the optimal solution.

3.2.2. BAT Meta-Heuristic Algorithm

The Bat Algorithm (BA) is an optimization technique that mimics the way bats use echolocation. In 2010, Xin-She Yang introduced it as a meta-heuristic optimization technique for dealing with difficult optimization issues. Applications of BA can be found in many different fields, including engineering, economics, medicine, and more due to BA's versatility and effectiveness with continuous and discrete optimization issues.

The bat algorithm, at its core, is a simulation of bat foraging behavior; each "bat" stands for a possible answer to an optimization issue. The program is modeled after the way bats fly by sending out ultrasonic pulses, listening for echoes, and altering their flight path accordingly. A series of mathematical equations and operators were developed to include this behavior into the optimization procedure.

Each bat in the bat algorithm's population represents a possible solution, and the algorithm works by moving these bats throughout the solution space. In order to identify the best possible or nearly best possible solution, the algorithm iteratively improves upon these candidates. As shown in Algorithm 1, the main parts and procedures of the bat algorithm are as follows:

- The algorithm begins by creating a population of bats with uniformly distributed initial positions in the solution space (referred to as "initialization"). Each bat has a unique pulse emission rate and loudness value that influences how it flies.
- Emission and Movement: From their current locations, bats emit ultrasonic pulses, with the intensity of the pulse decreasing with each repetition. Bats make course corrections using information from the radiated pulses as well as their own past positions and velocities. By flying about, bats can test out many strategies and eventually settle on the most effective one.
- Pulse Frequency and Velocity: A bat's search radius surrounding its current position is determined by the frequency of its pulses. More global exploration is encouraged by bats with higher pulse frequencies, whereas local exploitation is prioritized by those

with lower frequencies. Each bat's speed is revised every time its pulse frequency or volume is detected.

- Local Search and Global Search: When looking for interesting regions in the solution space, bats undertake a local search around their current places. They also alter their speeds in the direction of the greatest solution identified so far in the population, which aids in worldwide research.
- The loudness and pulse frequency of each bat are adjusted to reflect their current performance. Bats with better solutions have their volume and frequency settings preserved, while bats with worse solutions have their settings lowered to promote more research.
- At each iteration, the objective function of the optimization problem is used to assess the fitness of each bat's solution. Bats who come up with superior solutions contribute to improving the overall best one.
- Termination: The algorithm repeats this process until a convergence condition is fulfilled or a predetermined number of iterations have passed. In the end, one has the optimal solution discovered by any bat.

Algorithm 1 Bat Algorithm

Data: Objective function $f(x)$, Population size N, Number of generations G, Frequency scaling factor α, Pulse emission rate γ, Lower bound L, Upper bound U
Result: Optimal solution x^*
Initialize population X with random solutions;
Initialize pulse rates r_i and loudness A_i for each bat;
Initialize best solution x^*;
for $t \leftarrow 1$ **to** G **do**
  for $i \leftarrow 1$ **to** N **do**
    Generate a new solution x_i by adding random step to X_i;
    if *rand()* $< r_i$ **then**
      Generate a new solution x_i by adding a random step and a fraction of the current best solution;
    end
    Evaluate the fitness of x_i;
    if *rand()* $< \gamma$ *and* $f(x_i) < f(x^*)$ **then**
      Accept x_i as the new best solution x^*;
    end
    Update r_i and A_i using equations;
  end
end

Because it strikes a good balance between exploration and exploitation, the bat algorithm can be applied to a wide variety of optimization problems. Its flexibility and durability come from the fact that it can dynamically modify bats' search behavior via pulse frequency and loudness adjustments. The efficiency of the bat algorithm, like that of any optimization algorithm, might shift based on the details of the problem and the values chosen for its parameters. Researchers are still looking at new ways to tweak and refine the algorithm so that it can more quickly and accurately address difficult optimization challenges.

The Bat Algorithm (BA) is a nature-inspired optimization algorithm that mimics the echolocation behavior of bats. It is often used for solving optimization problems. The key equations of the bat algorithm are as follows:

Position Update

The position of each bat is updated based on its current position, velocity, and the pulse rate of the bat. The updated position x_i of the i-th bat is given by:

$$x_i^{t+1} = x_i^t + \epsilon \cdot (x_{\text{best}} - x_i^t) + \alpha \cdot A \cdot \text{rand}() \quad (1)$$

where:

- x_i^{t+1} is the updated position of the i-th bat at time $t+1$.
- x_i^t is the current position of the i-th bat at time t.
- x_{best} is the current best solution found by any bat.
- ϵ is a random scaling factor.
- α is the loudness of the bat.
- A is the pulse rate of the bat.
- rand() generates a random number between 0 and 1.

Velocity Update

The velocity v_i of each bat is updated to move towards the updated position. The velocity update equation is given by:

$$v_i^{t+1} = v_i^t + (x_i^{t+1} - x_i^t) \quad (2)$$

where:

- v_i^{t+1} is the updated velocity of the i-th bat at time $t+1$.
- v_i^t is the current velocity of the i-th bat at time t.

4. Prediction Techniques for Road Network Congestion

The road congestion prediction procedure begins with Phase 1. The initial step is data gathering, which includes recording information about things like traffic volumes, vehicle speeds, road occupancy, and other characteristics over time. Different prediction methods are implemented on top of this dataset. Data collection and processing algorithms like recurrent neural networks, long short-term memory networks, decision trees, autoregressive integrated moving average, and seasonal ARIMA are used to forecast traffic congestion. The accuracy of these prediction algorithms is then assessed by comparing the projected values to real-world congestion data and using metrics such as Mean Squared Error (MSE) and Mean Absolute Error (MAE). After the models have been verified, they are used to make predictions about traffic congestion, which can shed light on how the situation will unfold in the future. Adaptability to shifting traffic circumstances is ensured by repeating this forecasting method throughout several instances or time periods (N instances) [54]. The results of these forecasts are saved in a special congestion database, which is subsequently used for analysis and decision making in the future. In the second phase, meta-heuristic algorithms are used to improve traffic flow. The green time for traffic lights at various junctions can be strategically adjusted with the help of algorithms like the BAT algorithm, particle swarm optimization, the Cuckoo search algorithm, and the Jaya algorithm. The ultimate goal is to reduce the amount of time cars spend waiting at these junctions. Congestion will be reduced, traffic flow will be improved, and a more effective transportation system will be created thanks to this optimization process. When Phase 1 predictions are combined with Phase 2 optimization efforts in Figure 3, a unified system is produced that can not only anticipate congestion but also take proactive steps toward its reduction by optimizing traffic signal timings. Transportation systems, in general, and traffic management in particular, stand to benefit greatly from this all-encompassing strategy. The success of both stages is dependent on the accuracy of the data acquired, the efficacy of the prediction algorithms, and the efficacy of the optimization process.

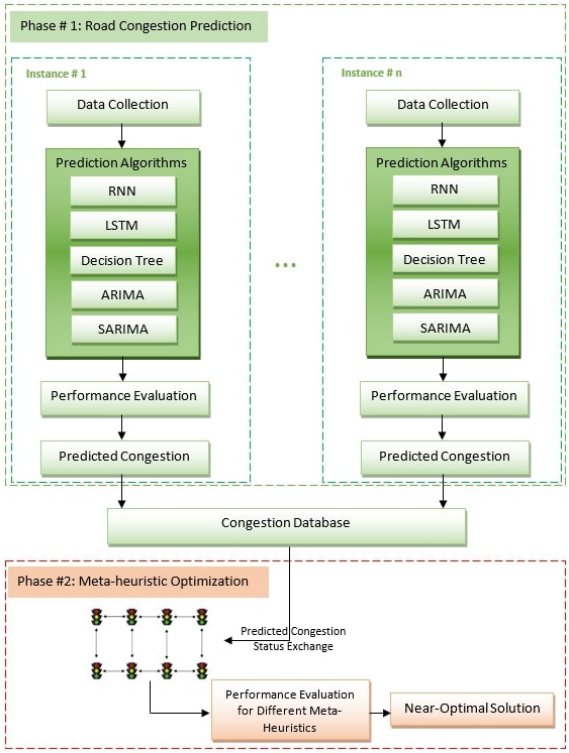

Figure 3. Proposed Methodology.

4.1. Dataset Description

The dataset as shown in Table 1 contains data about intersections and their associated properties. The "INT ID" (Intersection Number) uniquely identifies each intersection. Different intersections in the dataset can be distinguished by this identifier. The names or labels for each intersection are stored in the "Intersection Name" field. The names of the two streets that meet at a certain point are often listed there. Each intersection's location is denoted by its region's name in the "ARRONDISSEMENT" (Region Name) field. The position of a junction is a factor that can be used to classify it. The dataset also includes geographical coordinates. The "Longitude" field stores the longitude coordinate, which identifies the geographical east–west location of a certain intersection. The "Latitude" field stores the north–south coordinate of an intersection on Earth's surface. For mapping and other spatial work, these coordinates are indispensable.

Table 1. Dataset description.

Name of Feature	Description
INT_ID (Intersection Number)	This field would likely contain a unique identifier for each intersection in the dataset.
Intersection Name	This field would store the name or label associated with the intersection, and the names of the two streets that form the intersection.
ARRONDISSEMENT (Region Name)	This field could include the name of the region or area where the intersection is located.
Longitude	This feature would hold the geographical longitude coordinate of the intersection's position on the Earth's surface.

Table 1. *Cont.*

Name of Feature	Description
Latitude	This feature would contain the geographical latitude coordinate of the intersection's position on the Earth's surface.
Congestion Level (Number of Cars)	Low congestion (0 to 10 cars at traffic light), medium congestion (11 to 30 cars at traffic light), or high congestion (31 to 50 cars at traffic light)
Number of Intersections	2345
Number of Zones	213

4.2. Zone-Based Analysis

As discussed earlier, there are many options for studying the road congestion problem. The first option is to study this problem in a local manner, which means that each traffic light only considers the congestion level at its intersection without any consideration of the congestion level at other traffic lights, and adjust the green timing for each traffic light accordingly. The other option is to have a global view of all the congestion levels at each traffic light and then adjust the green timing for each traffic light according to the local congestion level at the specific traffic light and according to the congestion level at other traffic lights in the city of Montreal as shown in Figure 4 [54]. In our study, we divided the simulation area into different zones and studied our problem for each zone separately, where each traffic light should adjust their green timing dynamically according to the congestion level at this specific traffic light and the congestion levels at all the neighboring traffic lights in the same zone.

No matter how big the network is, even if it is a large-sized network, it will be divided into zones and the prediction algorithm will be separately applied to each zone; in our simulation each zone had 11 traffic lights in it, which makes our proposed algorithm scalable.

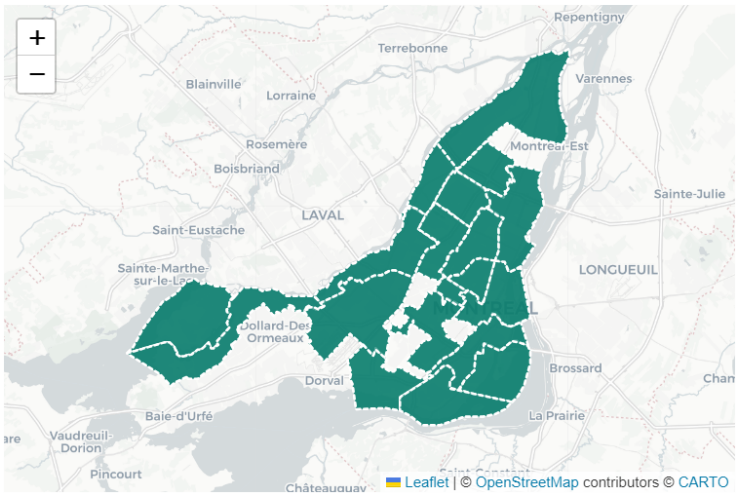

Figure 4. Simulation area in Montreal [55].

4.3. Performance Measures

When evaluating the performance of a predictive model, several measures can help assess its accuracy and effectiveness. In this section, we will discuss four commonly used performance measures: Mean Absolute Error (MAE), Root Mean Squared Error (RMSE), R-squared (R^2), and the visualization of prediction vs. actual values.

4.3.1. Mean Absolute Error (MAE)

The mean absolute error measures the average absolute difference between predicted and actual values. It is defined as:

$$MAE = \frac{1}{n}\sum_{i=1}^{n}|y_i - \hat{y}_i| \tag{3}$$

where:

- n is the number of data points.
- y_i represents the actual value of the i-th data point.
- $\hat{y}_i$ represents the predicted value of the i-th data point.

4.3.2. Root Mean Squared Error (RMSE)

The root mean squared error is another measure of the average prediction error. It takes the square root of the average of the squared differences between the predicted and actual values:

$$RMSE = \sqrt{\frac{1}{n}\sum_{i=1}^{n}(y_i - \hat{y}_i)^2} \tag{4}$$

4.3.3. R-Squared (R^2)

R-squared, also known as the coefficient of determination, quantifies the proportion of the variance in the dependent variable that is predictable from the independent variables. It is given by:

$$R^2 = 1 - \frac{\sum_{i=1}^{n}(y_i - \hat{y}_i)^2}{\sum_{i=1}^{n}(y_i - \bar{y})^2} \tag{5}$$

where:

- $\bar{y}$ is the mean of the actual values of y_i.

4.4. Problem Formulation

$$\begin{aligned}
\text{Minimize} \quad & \alpha_1 \cdot \sum_{i=1}^{I}\sum_{j=1}^{J}\sum_{k=1}^{K} w_{ijk}^{p} t_{ijk}^{q} + \alpha_2 \cdot \frac{1}{E}\sum_{e\in E} T_e \\
\text{subject to} \quad & t_{ijk}^{p} \geq d_{ij}^{p} + s_{ij}^{p} \\
& t_{ijk}^{p} \geq t_{i(k-1)j}^{p} + \tau_{ij}^{p} - (1 - \gamma_{ij}^{p})M \\
& t_{ijk}^{p} - t_{i(k-1)j}^{p} \leq \delta_{ij}^{p} \\
& \sum_{p=1}^{P}\sum_{k=1}^{K} \gamma_{ij}^{p} w_{ijk}^{p} \leq C_{ij}
\end{aligned}$$

where:

- α_1 and α_2 are the weights for the two objective functions.
- w_{ijk}^{p} is the number of vehicles in phase p at intersection i and lane group j during time interval k.
- t_{ijk}^{p} is the total waiting time for all vehicles in phase p at intersection i and lane group j during time interval k.
- T_e is the total time taken by emergency vehicle e to travel from its origin to its destination.
- E is the total number of emergency vehicles.
- d_{ij}^{p} is the duration of phase p at intersection i and lane group j.
- s_{ij}^{p} is the yellow and all of the red time at intersection i and lane group j during phase p.
- τ_{ij}^{p} is the duration of the green time at intersection i and lane group j during phase p.

- γ_{ij}^{p} is a binary variable that equals 1 if phase p is active at intersection i and lane group j, and 0 otherwise.
- M is a large constant.
- δ_{ij}^{p} is the maximum allowed difference between the green time at intersection i and lane group j in successive phases.
- C_{ij} is the maximum number of vehicles that can be served per hour at intersection i and lane group j.

The first objective function aims to minimize the overall waiting time for all vehicles, while the second objective function aims to minimize the average waiting time for all emergency vehicles. These two objectives are conflicting, as reducing the waiting time for all vehicles may increase the waiting time for emergency vehicles.

The four constraints included in the optimization problem are:

Safety constraint (C1): The total green time allocated to all phases at all intersections should not exceed the time budget, T.

Feasibility constraint (C2): The green time allocated to each phase at each intersection should not exceed the total cycle length, g_i.

Smoothness constraint (C3): The change in green time between any two consecutive phases at each intersection should not exceed the maximum change in green time, K_i.

Saturation constraint (C4): The green time allocated to each phase at each intersection should not exceed the maximum green time, c_i.

Including these constraints is important to ensure that the optimization problem produces solutions that are both safe and feasible, while also minimizing waiting times for all vehicles, including emergency vehicles.

4.5. Results and Analysis

We divided our extensive case study into several zones, each of which contained 19 traffic signals. Our focus was on using advanced machine learning algorithms to predict and address congestion before it occurred. This proactive strategy enabled us to make educated judgments based on the algorithms' forecasts.

We thoroughly analyzed the efficacy of our machine learning algorithms in the ensuing graphical representations. We used key measures including the Mean Square Error (MSE), Mean Absolute Error (MAE), and Coefficient of Determination (R^2) to quantify their performance. We separated the dataset into two portions to thoroughly validate our models: an 80% training subset and a 20% testing subset. Using this method, we were able to effectively analyze the model's predictive skills throughout 300 iterations.

Figures 5–8 depicts the congestion level forecasts provided by each separate machine learning method, along with the related MSE, MAE, and R^2 values. The results clearly show that the decision tree algorithm performed better than the other machine learning approaches. This superiority was demonstrated by its faster convergence rate, lower loss values, and significantly greater coefficient of determination.

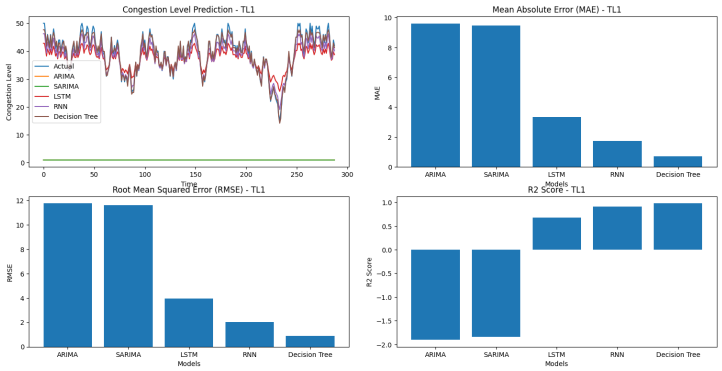

Figure 5. TL1.

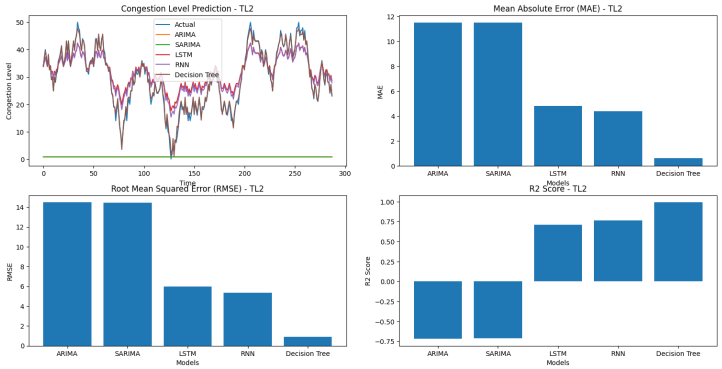

Figure 6. TL2.

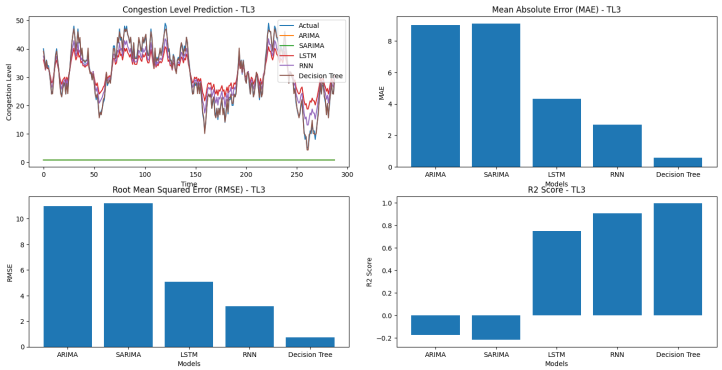

Figure 7. TL3.

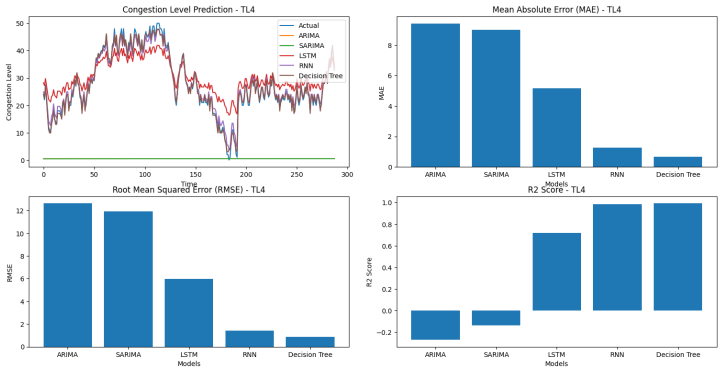

Figure 8. TL4.

5. Proposed Methodology

5.1. Enhanced BAT Algorithm

One of our contributions is to develop an enhanced approach for the BAT algorithm. In order to achieve this, we modified the exploration phase of the algorithm. In the BAT algorithm, there is an exploration phase, where the solution is generated randomly and then its fitness is calculated and compared with the best solution obtained so far and updated accordingly. In our modified version of the BAT algorithm, the exploration phase is not fully random but is a semi-random exploration phase, which means that the algorithm is directed to obtain a better solution by taking into consideration the congestion status, as shown in the equation below:

$$x_i \leftarrow x_i + C \times (x^* - x_i)$$

Another enhancement is to tune the parameters in the BAT algorithm to have better results in terms of the convergence rate and the quality of the solution. Below is the Algorithm 2 of our proposed enhanced BAT algorithm.

Algorithm 2 Enhanced Bat Algorithm

Data: Objective function $f(x)$, Population size N, Number of generations G, Frequency scaling factor α, Pulse emission rate γ, Lower bound L, Upper bound U
Result: Optimal solution x^*
Initialize population X with random solutions;
Initialize pulse rates r_i and loudness A_i for each bat;
Initialize best solution x^*;
for $t \leftarrow 1$ **to** G **do**
  for $i \leftarrow 1$ **to** N **do**
    Generate a new solution x_i by adding random step to X_i;
    if *rand()* $< r_i$ **then**
      Generate a new solution x_i by adding a random step and a fraction of the current best solution;
      Enhanced exploration term: $x_i \leftarrow x_i + C \times (x^* - x_i)$;
    end
    Evaluate the fitness of x_i;
    if *rand()* $< \gamma$ *and* $f(x_i) < f(x^*)$ **then**
      Accept x_i as the new best solution x^*;
    end
    Update r_i and A_i using equations;
  end
end

5.2. Results and Analysis

As shown in Figures 9 and 10 below, we compared different meta-heuristic algorithms in terms of the convergence rate. As shown in the results in Figure 9, under the high congestion category, the enhanced BAT algorithm outperformed the other algorithms in terms of solution quality; it minimized the objective function and converged to the near-optimal solution after 25 iterations, which is defined as the average waiting time for each vehicle in the road network. The second-best algorithm was the BAT algorithm, which converged to its local minimum solution after almost 30 iterations. In this specific scenario, where the exploration phase is semi-random, which means that the exploration phase benefits from the predicted congestion status at each traffic light before adjusting the green timing for each traffic light in the zone, the initial solution was around 9 for both BAT and EBAT. As seen in Figure 10, under the low congestion category, the enhanced BAT algorithm outperformed the other algorithms in terms of solution quality; it minimized the objective function and converged to the near-optimal solution after 25 iterations, which is defined as the average waiting time for each vehicle in the road network. The second-best algorithm was the BAT algorithm, which converged to its local minimum solution after almost 28 iterations. In this specific scenario, where the exploration phase is semi-random, which means that the exploration phase benefits from the predicted congestion status at each traffic light before adjusting the green timing for each traffic light in the zone, the initial solution was around 6.5 for both BAT and EBAT.

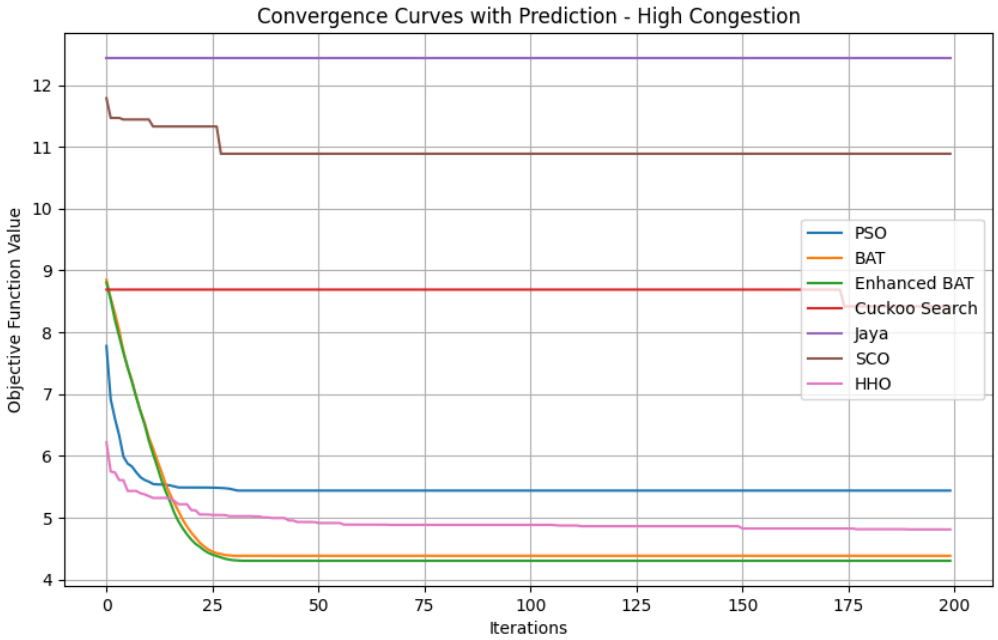

Figure 9. Convergence curves with prediction—high congestion.

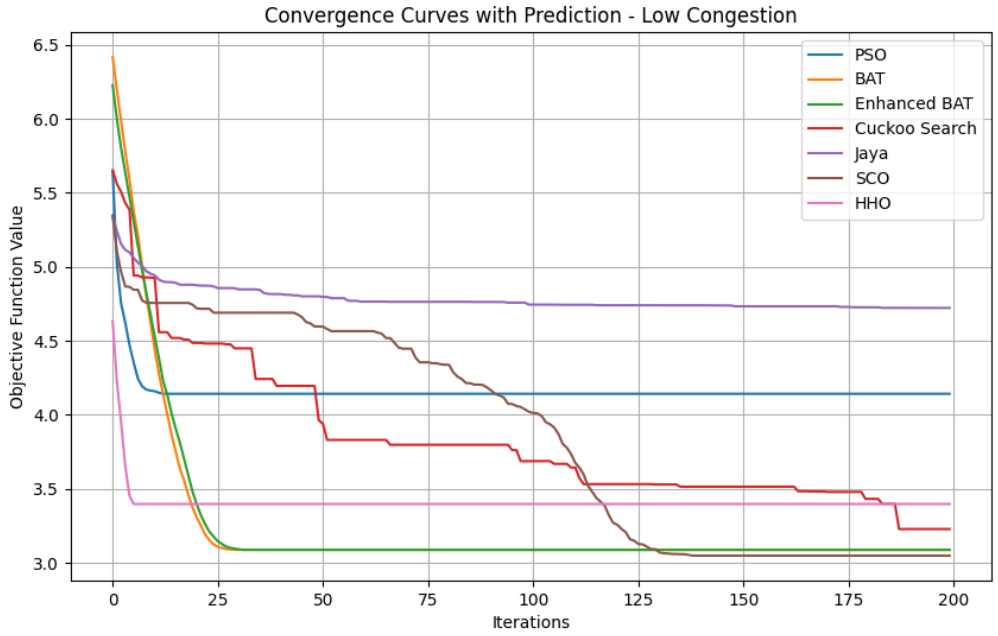

Figure 10. Convergence curves with prediction—low congestion.

Figures 11 and 12 below show the results of the other scenario, which is also a dynamic scenario, where the timing for the traffic lights changes dynamically based on the current congestion status and not the predicted congestion status as in the previous scenario. We compared different meta-heuristic algorithms in terms of the convergence rate. As shown in the results in Figure 11, under the high congestion category, the enhanced BAT algorithm outperformed the other algorithms in terms of solution quality; it minimized the objective function and converged to the near-optimal solution after 30 iterations. The second-best algorithm was the BAT algorithm, which converged to its local minimum solution after almost 37 iterations. In this specific scenario, nothing has changed in the exploration phase, and the initial solution was around 15.5 for both BAT and EBAT. We noticed that the solution quality for the EBAT and BAT algorithms was almost the same, which is because the enhanced BAT algorithm in this case has only one improvement, which is the parameter

tuning, but there is no improvement in the exploration phase as in the previous scenario. As seen in Figure 12, under the low congestion category, the enhanced BAT algorithm outperformed the other algorithms in terms of solution quality; it minimized the objective function and converged to the near-optimal solution after 30 iterations. The second-best algorithm was the BAT algorithm, which converged to its local minimum solution after almost 30 iterations. In this specific scenario, nothing has changed in the exploration phase, and the initial solution was around 13.8 for both BAT and EBAT. We noticed that the solution quality for the EBAT and BAT algorithms was almost the same, which is because the enhanced BAT algorithm in this case has only one improvement, which is the parameter tuning, but there is no improvement in the exploration phase as in the previous scenario.

Figures 13 and 14 show the results of the third scenario, which is the fixed-approach scenario, where the timings for the traffic lights are fixed at all times. We compared different meta-heuristic algorithms in terms of convergence rate. As shown in the results, in Figure 13, under the high congestion category, the enhanced BAT algorithm outperformed the other algorithms in terms of solution quality; it minimized the objective function and converged to the near-optimal solution after 30 iterations. The second-best algorithm was the BAT algorithm, which converged to its local minimum solution after almost 37 iterations. In this specific scenario, nothing has changed in the exploration phase, and the initial solution was between 19 and 19.5 for both BAT and EBAT. We noticed that the solution quality for the EBAT and BAT algorithms was almost the same, which is because the enhanced BAT algorithm in this case as well has only one improvement, which is the parameter tuning, but there is no improvement in the exploration phase as in the first scenario. As seen in Figure 14, under the low congestion category, the enhanced BAT algorithm outperformed the other algorithms in terms of solution quality; it minimized the objective function and converged to the near-optimal solution after 30 iterations. The second-best algorithm was the BAT algorithm, which converged to its local minimum solution after almost 35 iterations. In this specific scenario, nothing has changed in the exploration phase, and the initial solution was between 6 and 6.5 for both BAT and EBAT. We noticed that the solution quality for the EBAT and BAT algorithms was almost the same, which is because the enhanced BAT algorithm in this case as well has only one improvement, which is the parameter tuning, but there is no improvement in the exploration phase as in the first scenario.

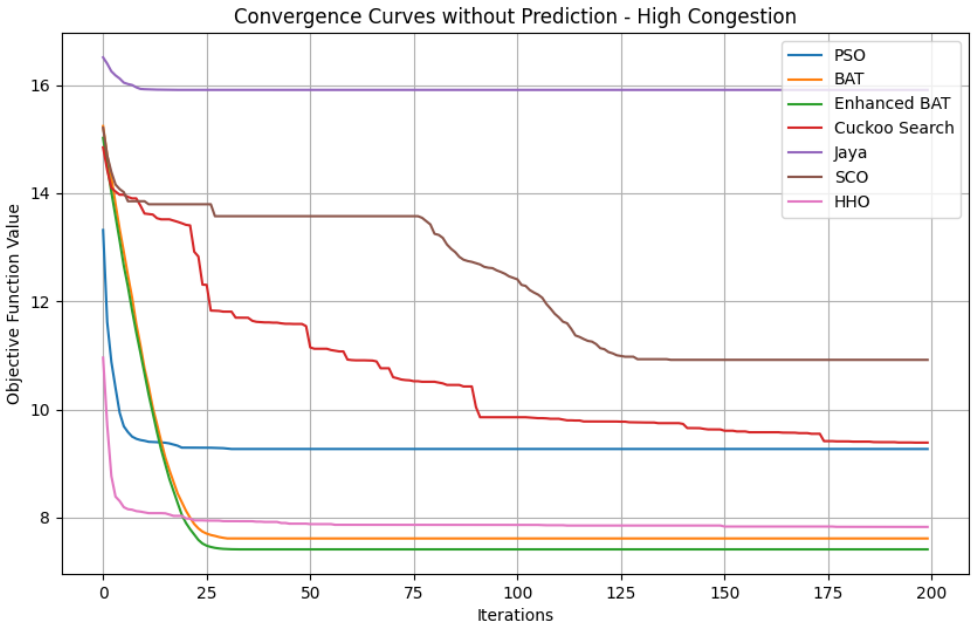

Figure 11. Convergence curves without prediction—high congestion.

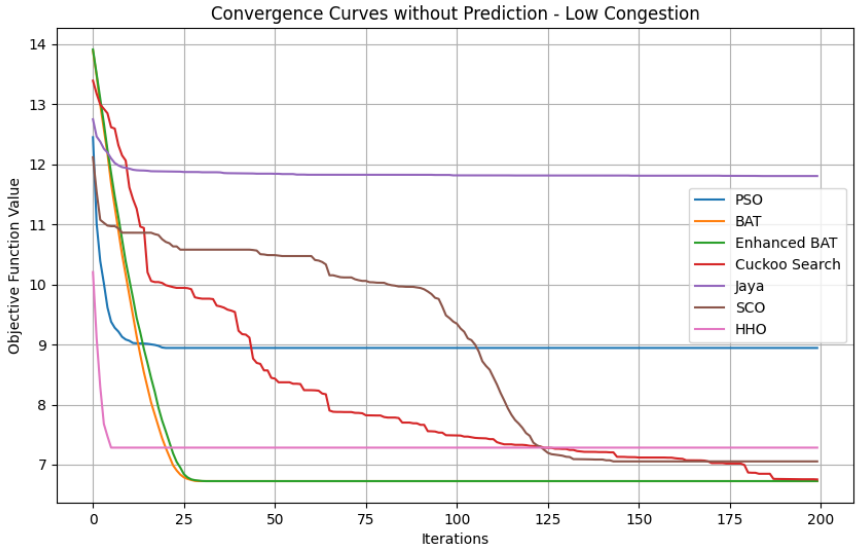

Figure 12. Curves without prediction—low congestion.

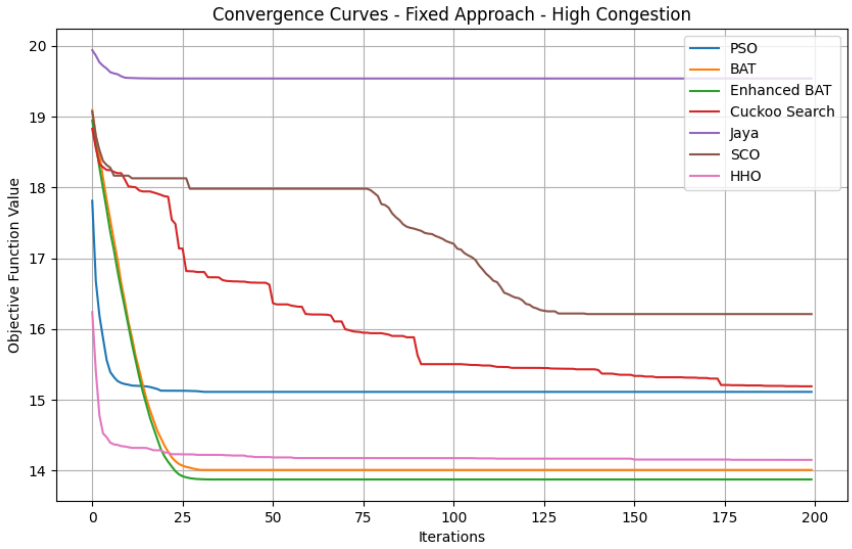

Figure 13. Convergence curves—fixed approach—high congestion.

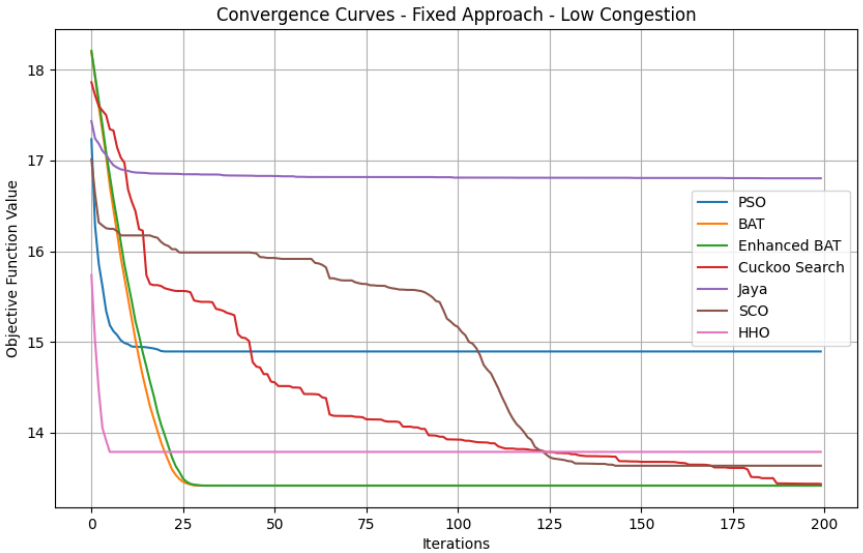

Figure 14. Convergence curves—fixed approach—low congestion.

As can be seen from the convergence rate in all the investigated scenarios, the initial solution is the key for a better solution and fast convergence.

For example, in the fixed approach scenario because of the bad initial solution due to the fixed green timing regardless of the congestion at each traffic light, the EBAT algorithm converged after 44 iterations and the BAT algorithm converged after 29 iterations, and the initial solutions were 19 for both of them.

On the other hand, the dynamic with no prediction approach converged after 42 iterations for the EBAT and 27 iterations for the BAT algorithm, with the initial solutions being 15 for both the EBAT and BAT algorithms.

In the third scenario, which is the dynamic with prediction approach, the EBAT algorithm converged after 37 iterations and the BAT algorithm converged after 24 iterations, with the initial solutions being 9 for both EBAT and BAT algorithms.

In terms of execution time, there was slight difference between the different algorithms with a reasonable execution time except for the JAYA, SCO, and HHO algorithms, as shown in Figures 15–20, where the execution time for the third approach, which is the dynamic with prediction approach, was the highest, and the execution time for the fixed approach was the lowest.

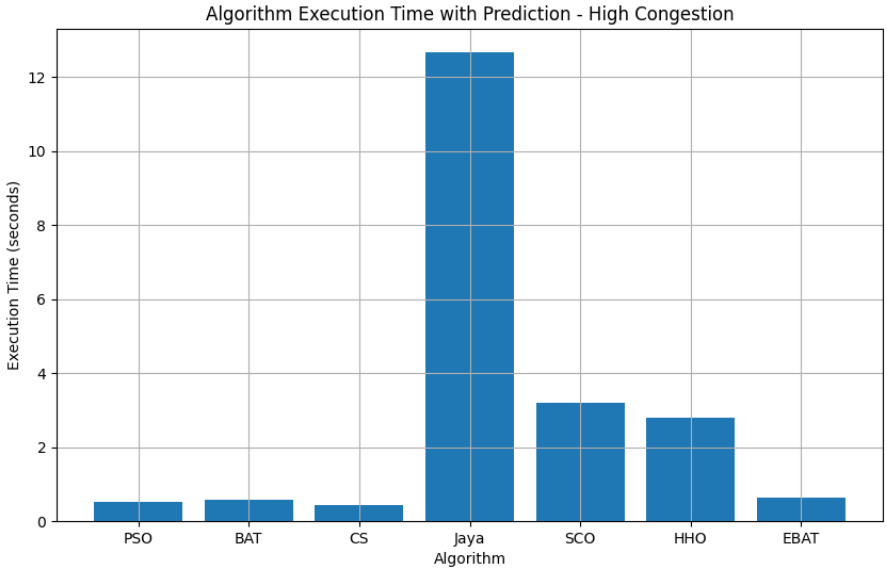

Figure 15. Execution time with prediction—high congestion.

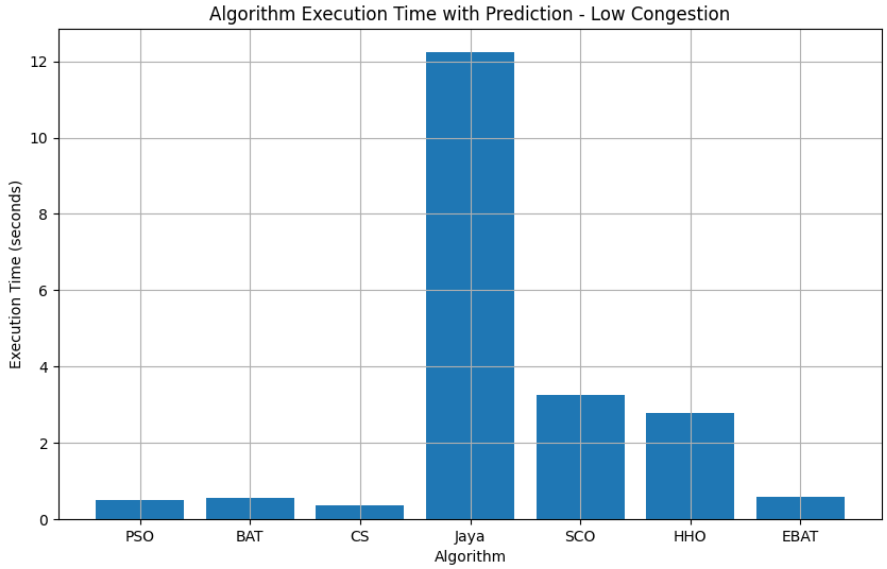

Figure 16. Execution time with prediction—low congestion.

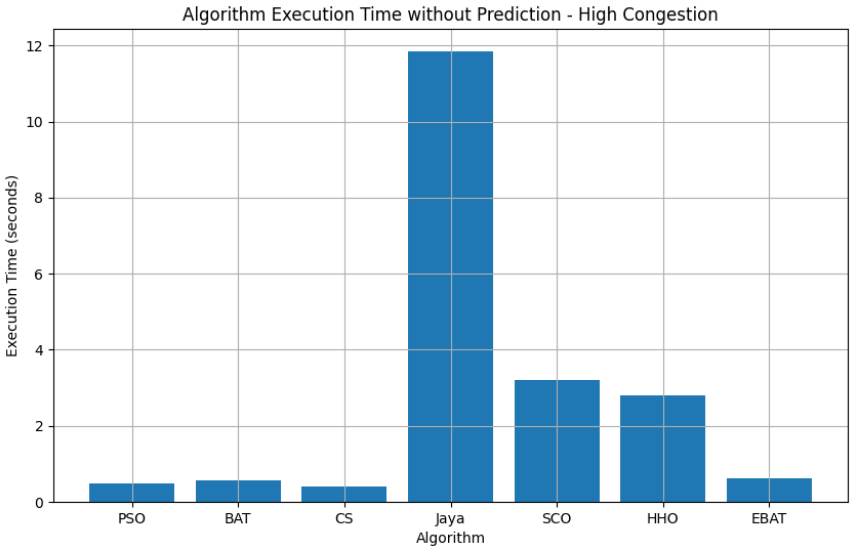

Figure 17. Execution time without prediction—high congestion.

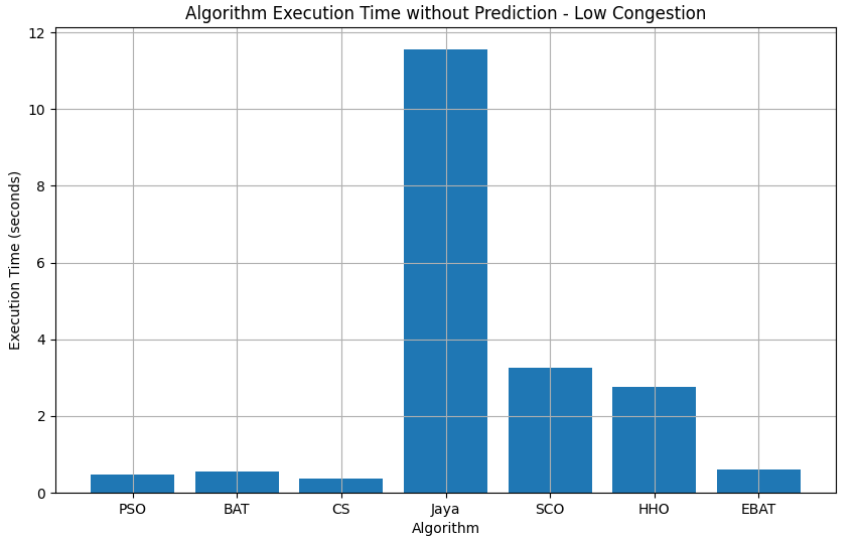

Figure 18. Execution time without prediction—low congestion.

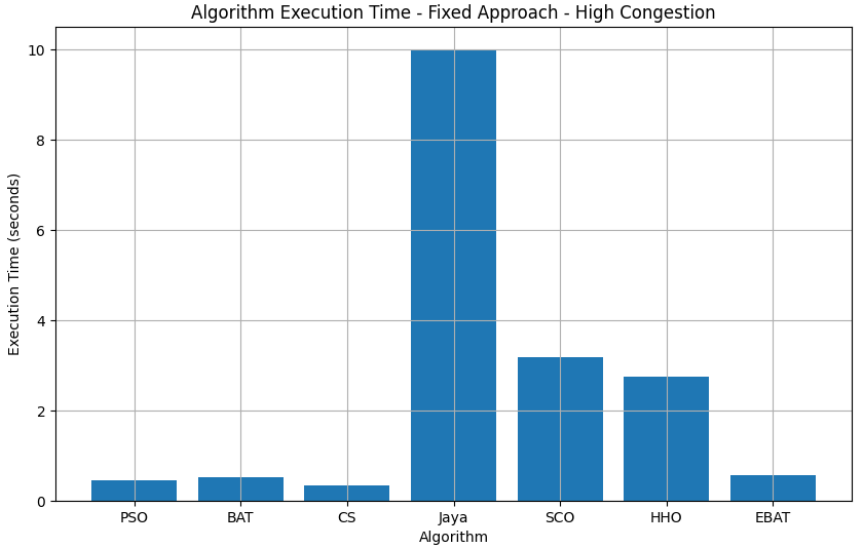

Figure 19. Execution time—fixed—high congestion.

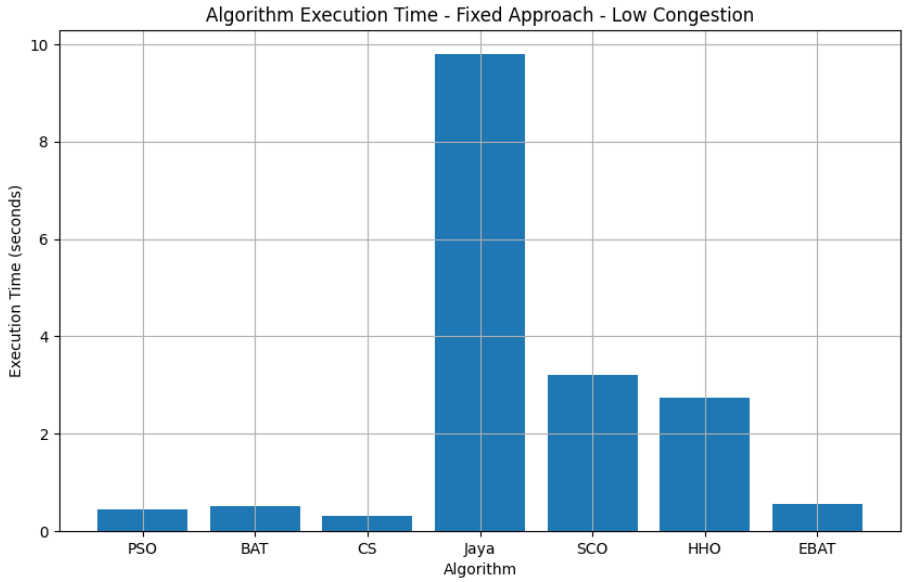

Figure 20. Execution time—fixed—low congestion.

6. Conclusions

In our paper, road congestion prediction is a process that involves two phases: Phase 1 and Phase 2. Phase 1 involves gathering comprehensive data on traffic-related variables, such as traffic volumes, vehicle speeds, and road occupancy, which have been taken from the dataset, and then we used sophisticated prediction methodologies like recurrent neural networks, long short-term memory networks, decision trees, auto-regressive integrated moving average, and seasonal ARIMA. The effectiveness of these algorithms was assessed using metrics like Mean Squared Error (MSE) and Mean Absolute Error (MAE), ensuring precision in prediction.

Phase 2 presents a proactive approach to traffic optimization using meta-heuristic algorithms, such as the EBAT, the BAT algorithm, particle swarm optimization, Cuckoo search, and the Jaya algorithm. The vehicle waiting times have been reduced, mitigating congestion and promoting more efficient traffic movement. The integration of these insights with the optimization efforts in Phase 2 results in an integrated framework that not only predicts congestion but also implements proactive strategies to mitigate it through traffic signal timing optimization.

The successful completion of both phases depends on the precision of data, the efficacy of prediction algorithms, and the efficiency of the optimization process. The proposed technique has significant implications for contemporary contexts, and we aimed to address congestion issues using advanced machine learning techniques and by conducting a comprehensive case study. The optimization targets involve minimizing the total and average waiting times for all vehicles, including emergency vehicles. The optimization model incorporates time budgets, phase cycle lengths, smoothness of phase transitions, and maximum green times to ensure feasibility and safety.

There are other areas we could pursue for future work. These directions not only enhance our current research but also tackle emerging difficulties and opportunities in traffic management and urban development. As an illustration, we can focus on investigating the incorporation of real-time traffic data from IoT devices, social media, and other sources to improve the accuracy and responsiveness of our algorithm. A further avenue of research is expanding our investigation to encompass optimization for multi-modal traffic networks, encompassing pedestrians, motorcycles, and public transportation, thus guaranteeing a more comprehensive approach to urban mobility. An additional improvement could involve prioritizing the enhancement of the resilience and security of the traffic management system, protecting it against cyber threats and guaranteeing its reliability.

In conclusion, this study provides a thorough investigation that forms a foundation for a complete approach to managing congestion and optimizing traffic, which has the potential to significantly transform transportation systems and rethink traffic control principles.

Author Contributions: Conceptualization, M.A.K.; Formal analysis, M.A.K.; Writing—original draft, M.A.K.; Writing—review & editing, A.A.; Visualization, M.A.K.; Supervision, A.A. All authors have read and agreed to the published version of the manuscript.

Funding: This research received no external funding.

Data Availability Statement: Data are contained within the article.

Conflicts of Interest: The authors declare no conflict of interest.

References

1. Guerrini, F. Traffic Congestion Costs Americans $124 Billion A Year, Report Says. *Forbes*, 14 October 2014, p.1.
2. The Economist. The Cost of Traffic Jams. Available online: https://www.economist.com/blogs/economist-explains/2014/11/economist-explains-1 (accessed on 3 November 2014).
3. Tan, M.K.; Chuo, H.S.E.; Chin, R.K.Y.; Yeo, K.B.; Teo, K.T.K. Optimization of Urban Traffic Network Signalization using Genetic Algorithm. In Proceedings of the 2016 IEEE Conference on Open Systems (ICOS), Langkawi, Malaysia, 10–12 October 2016; pp. 87–92.
4. Khasawneh, M.A.; Awasthi, A. Literature Review on Smart City SoS Perspective. In *Fleet Management and Planning for Sustainable Connected Mobility Systems*; IGI Global: Hershey, PA, USA, 2021.
5. Gao, K.; Zhang, Y.; Zhang, Y.; Su, R.; Suganthan, P.N. Meta-Heuristics for Bi-Objective Urban Traffic Light Scheduling Problems. *IEEE Trans. Intell. Transp. Syst.* **2019**, *20*, 2618–2629. [CrossRef]
6. Miller, A.J. Settings for fixed-cycle traffic signals. *J. Oper. Res. Soc.* **1963**, *14*, 373–386. [CrossRef]
7. Webster, F.V. *Traffic Signal Settings*; Road Research Technical Paper 39; Road Research Laboratory: London, UK, 1958.
8. Cools, S.-B.; Gershenson, C.; D'Hooghe, B. Self-organizing traffic lights: A realistic simulation. In *Advances in Applied Self-Organizing Systems*; Springer: London, UK, 2013; pp. 45–55.
9. Porche, I.; Lafortune, S. Adaptive look-ahead optimization of traffic signals. *J. Intell. Transp. Syst.* **1999**, *4*, 209–254.
10. Kuyer, L.; Whiteson, S.; Bakker, B.; Vlassis, N. Multiagent reinforcement learning for urban traffic control using coordination graphs. In *Machine Learning and Knowledge Discovery in Databases*; Springer: Berlin/Heidelberg, Germany, 2008; pp. 656–671.
11. van der Pol, E.; Oliehoek, F.A. Coordinated Deep Reinforcement Learners for Traffic Light Control. In Proceedings of the 30th Conference on Neural Information Processing Systems (NIPS 2016), Barcelona, Spain, 5–10 December 2016.
12. Wiering, M.A. Multi-agent reinforcement learning for traffic light control. In Proceedings of the Seventeenth International Conference on Machine Learning (ICML'2000), Stanford, CA, USA, 29 June–2 July 2000; pp. 1151–1158.
13. Kumar, N.; Rahman, S.S.; Dhakad, N. Fuzzy Inference Enabled Deep Reinforcement Learning-Based Traffic Light Control for Intelligent Transportation System. *IEEE Trans. Intell. Transp. Syst.* **2021**, *22*, 4919–4928. [CrossRef]
14. Lu, F.-Q.; Huang, M.; Ching, W.-K.; Siu, T.K. Credit portfolio management using two-level particle swarm optimization. *Inf. Sci.* **2013**, *237*, 162–175. [CrossRef]
15. Yan, T.; Lu, F.; Wang, S.; Wang, L.; Bi, H. A hybrid metaheuristic algorithm for the multi-objective location-routing problem in the early post-disaster stage. *J. Ind. Manag. Optim.* **2023**, *19*, 4663–4691. [CrossRef]
16. Wen, H.; Wang, S.X.; Lu, F.Q.; Feng, M.; Wang, L.Z.; Xiong, J.K.; Ma, C. Colony search optimization algorithm using global optimization. *J. Supercomput.* **2021**, *78*, 6567–6611. [CrossRef]
17. Lu, F.; Chen, W.; Feng, W.; Bi, H. 4PL routing problem using hybrid beetle swarm optimization. *Soft Comput.* **2023**, *27*, 17011–17024. [CrossRef]
18. Tsai, C.-W.; Teng, T.-C.; Liao, J.-T.; Chiang, M.-C. An effective hybrid-heuristic algorithm for urban traffic light scheduling. *Neural Comput. Appl.* **2021**, *33*, 17535–17549. [CrossRef]
19. Wei, H.; Zheng, G.; Yao, H.; Li, Z.J. IntelliLight: A Reinforcement Learning Approach for Intelligent Traffic Light Control. In Proceedings of the 24th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining, London, UK, 19–23 August 2018.
20. Sartikha; Ardiyanto, I.; Sulistyo, S. A Study on Metaheuristics for Urban Traffic Light Scheduling Problems. In Proceedings of the 2018 10th International Conference on Information Technology and Electrical Engineering (ICITEE), Bali, Indonesia, 24–26 July 2018; pp. 545–550.
21. Roy, A.; Das, S.K. QM2RP: A QoS-based mobile multicast routing protocol using multi-objective genetic algorithm. *Wirel. Netw.* **2004**, *10*, 271–286. [CrossRef]
22. Radhika, S.C.S.; Ranjith, M.K.S.; Sasirekha, N. An intelligent IoT Enabled Traffic queue handling System Using Machine Learning Algorithm. In Proceedings of the 2022 International Conference on Innovative Computing, Intelligent Communication and Smart Electrical Systems (ICSES), Chennai, India, 15–16 July 2022; pp. 1–9. [CrossRef]

23. Jaleel, A.; Hassan, M.A.; Mahmood, T.; Ghani, M.U.; Rehman, A.U. Reducing Congestion in an Intelligent Traffic System with Collaborative and Adaptive Signaling on the Edge. *IEEE Access* **2020**, *8*, 205396–205410. [CrossRef]
24. Zeadally, S.; Hunt, R.; Chen, Y.-S.; Irwin, A.; Hassan, A. Vehicular ad hoc networks (VANETS): Status, results, and challenges. *Telecommun. Syst.* **2012**, *50*, 217–241. [CrossRef]
25. Golestan, K.; Jundi, A.; Nassar, L.; Sattar, F.; Karray, F.; Kamel, M.; Boumaiza, S. Vehicular ad-hoc networks (VANETs): Capabilities, challenges in information gathering and data fusion. In *Autonomous and Intelligent Systems*; Springer: Berlin/Heidelberg, Germany, 2012; pp. 34–41.
26. Shen, M.; Zhan, Z.H.; Chen, W.N.; Gong, Y.J.; Zhang, J.; Li, Y. Bi-velocity discrete particle swarm optimization and its application to multicast routing problem in communication networks. *IEEE Trans. Ind. Electron.* **2014**, *61*, 7141–7151. [CrossRef]
27. Taherkhani, N.; Pierre, S. Centralized and Localized Data Congestion Control Strategy for Vehicular Ad Hoc Networks Using a Machine Learning Clustering Algorithm. *IEEE Trans. Intell. Transp. Syst.* **2016**, *17*, 3275–3285. [CrossRef]
28. Kumar, V.; Kumar, S. Energy balanced position-based routing for lifetime maximization of wireless sensor networks. *Ad Hoc Netw.* **2016**, *52*, 117–129. [CrossRef]
29. Wang, H.; Meng, X.; Li, S.; Xu, H. A tree-based particle swarm optimization for multicast routing. *Comput. Netw.* **2010**, *54*, 2775–2786. [CrossRef]
30. Haghighat, A.T.; Faez, K.; Dehghan, M.; Mowlaei, A.; Ghahremani, Y. GA-based heuristic algorithms for QoS based multicast routing. *Knowl.-Based Syst.* **2003**, *16*, 305–312. [CrossRef]
31. Kumar, N.; Rahman, S.S. Deep Reinforcement Learning with Vehicle Heterogeneity Based Traffic Light Control for Intelligent Transportation System. In Proceedings of the 2019 IEEE International Conference on Industrial Internet (ICII), Orlando, FL, USA, 11–12 November 2019; pp. 28–33. [CrossRef]
32. Robertson, D.I. TRANSYT method for area traffic control. *Traffic Eng. Control* **1969**, *10*, 276–281.
33. Li, M.-T.; Gan, A. Signal timing optimization for oversaturated networks using TRANSYT-7F. *Transp. Res. Rec. J. Transp. Res. Board* **1999**, *1683*, 118–126. [CrossRef]
34. Hunt, P.B.; Robertson, D.L.; Bretherton, R.D.; Royle, M.C. The SCOOT on-line traffic signal optimisation technique. *Traffic Eng. Control* **1982**, *23*, 190–192.
35. Gartner, N.H. *OPAC: A Demand-Responsive Strategy for Traffic Signal Control*; U.S. Department of Transportation: Washington, DC, USA, 1983.
36. Farges, J.-L.; Henry, J.J.; Tufal, J. The PRODYN real time traffic algorithm. *IFAC Proc. Vol.* **1983**, *16*, 305–310.
37. Boillot, F.; Blosseville, J.M.; Lesort, J.B.; Motyka, V.; Papageorgiou, M.; Sellam, S. Optimal signal control of urban traffic networks. In *Proceedings of the 6th IEEE International Conference Road Traffic Monitoring and Control, 28–30 April 1992;* The Institution: London, UK, 1992; pp. 75–79.
38. Sen, S.; Head, K.L. Controlled optimization of phases at an intersection. *Transp. Sci.* **1997**, *31*, 5–17. [CrossRef]
39. Sanchez, J.; Galan, M.; Rubio, E. Applying a traffic lights evolutionary optimization technique to a real case: 'Las Ramblas' area in Santa Cruz de Tenerife. *IEEE Trans. Evol. Comput.* **2008**, *12*, 25–40. [CrossRef]
40. Sanchez, J.J.; Galán, M.; Rubio, E. Genetic algorithms and cellular automata: A new architecture for traffic light cycles optimization. In Proceedings of the 2004 Congress on Evolutionary Computation (IEEE Cat. No.04TH8753), Portland, OR, USA, 19–23 June 2004; pp. 1668–1674.
41. García-Nieto, J.; Alba, E.; Olivera, A.C. Swarm intelligence for traffic light scheduling: Application to real urban areas. *Eng. Appl. Artif. Intel.* **2012**, *25*, 274–283. [CrossRef]
42. Garcia-Nieto, J.; Olivera, A.C.; Alba, E. Optimal cycle program of traffic lights with particle swarm optimization. *IEEE Trans. Evol. Comput.* **2013**, *17*, 823–839. [CrossRef]
43. Göttlich, S.; Herty, M.; Ziegler, U. Modeling and optimizing traffic light settings in road networks. *Comput. Oper. Res.* **2015**, *55*, 36–51. [CrossRef]
44. Cheng, J.; Wu, W.; Cao, J.; Li, K. Fuzzy Group-Based Intersection Control via Vehicular Networks for Smart Transportations. *IEEE Trans. Ind. Inform.* **2017**, *13*, 751–758. [CrossRef]
45. Jamal, A.; Rahman, M.T.; Al-Ahmadi, H.M.; Ullah, I.; Zahid, M. Intelligent Intersection Control for Delay Optimization: Using Meta-Heuristic Search Algorithms. *Sustainability* **2020**, *12*, 1896. [CrossRef]
46. Shahkar, A.; Oruç, E.; Yelghi, A. Traffic Signal Prediction Based on ANFIS and Metaheuristic Algorithms Applied to a Vissim-Based Simulated Intersection. *Res. Sq.* 2023, *in press*. [CrossRef]
47. Shirke, C.; Sabar, N.; Chung, E.; Bhaskar, A. Metaheuristic approach for designing robust traffic signal timings to effectively serve varying traffic demand. *J. Intell. Transp. Syst.* **2022**, *26*, 343–355. [CrossRef]
48. Al-Turki, M.; Jamal, A.; Al-Ahmadi, H.M.; Al-Sughaiyer, M.A.; Zahid, M. On the Potential Impacts of Smart Traffic Control for Delay, Fuel Energy Consumption, and Emissions: An NSGA-II-Based Optimization Case Study from Dhahran, Saudi Arabia. *Sustainability* **2020**, *12*, 7394. [CrossRef]
49. Szatmari, L.; Prodan, L.; Iovanovici, A.; Avramoni, D. Road Intersection Optimization with Resource-Constrained Metaheuristic: A Case Study. In Proceedings of the 2022 IEEE 20th Jubilee International Symposium on Intelligent Systems and Informatics (SISY), Subotica, Serbia, 15–17 September 2022. [CrossRef]

50. Jamal, A.; MAl-Ahmadi, H.; Muhammad Butt, F.; Iqbal, M.; Almoshaogeh, M.; Ali, S. Metaheuristics for Traffic Control and Optimization: Current Challenges and Prospects. In *Search Algorithm—Essence of Optimization*; IntechOpen: London, UK, 2023. [CrossRef]
51. Ilhan, T.; Mehmet, S. Fuzzy logic and deep Q learning based control for traffic lights. *Alex. Eng. J.* **2022**, *67*, 343–359. [CrossRef]
52. Younes, M.B.; Boukerche, A.; Rango, F.D. SmartLight: A smart efficient traffic light scheduling algorithm for green road intersections. *Ad Hoc Netw.* **2023**, *140*, 103061. [CrossRef]
53. Dheeraj, J.; Neetesh, K.; Anuj, S.; Yash, D.; Naveen, D. Adaptive neuro-fuzzy enabled multi-mode traffic light control system for urban transport network. *Appl. Intell.* **2022**, *53*, 1–22. [CrossRef]
54. Khasawneh, M.A.; Awasthi, A. Intelligent Traffic Light Control. In Proceedings of the 2nd African International Conference on Industrial Engineering and Operations Management (IEOM), Harare, Zimbabwe, 7–10 December 2020.
55. Towards Data Science. Available online: https://towardsdatascience.com/highlighting-click-data-on-plotly-choropleth-map-377e721c5893 (accessed on 28 November 2023).