Incorporating LSTM Auto-encoders in Optimisations to Solve Parking Officer Patrolling Problem

WEI SHAO, School of Science, RMIT University, Australia
SIYU TAN, The Faculty of Computer Science and Engineering, Xi'an University of Technology, China
SICHEN ZHAO, School of Science, RMIT University, Australia
KYLE KAI QIN, School of Science, RMIT University, Australia
XINHONG HEI, The Faculty of Computer Science and Engineering, Xi'an University of Technology, China
JEFFREY CHAN, School of Science, RMIT University, Australia
FLORA D. SALIM, School of Science, RMIT University, Australia

The smart parking system is one of the most important problems in smart cities. Recently, an increasing number of sensors installed in parking spaces provide big spatio-temporal data which be used to analyse the parking situations in the city and help parking officers monitor the parking violations. Travelling Officer Problem was customised to formulate a path-finding problem that aims to maximise the probability of catching overstayed cars before they leave. One of the challenges is to extract effective features from the big spatio-temporal data and provide a data-driven solution to replace conventional solutions such as a simple rule-based system or single optimisation methods. In this paper, we propose a seamless end-to-end learning and optimisation framework that combines Long Short-Term Memory (LSTM) Auto-Encoder neural network, clustering and path-finding methods to solve the Travelling Officer Problem. Our extensive comparison experiments on a large scale real-world dataset have shown that our proposed solution outperforms any other single step methods or optimisation methods.

CCS Concepts: • **Theory of computation → Optimization with randomized search heuristics**; • **Computing methodologies → Cluster analysis**; **Neural networks**.

Additional Key Words and Phrases: Parking System, Travelling Officer Problem, Feature Extraction

ACM Reference Format:
Wei Shao, Siyu Tan, Sichen Zhao, Kyle Kai Qin, Xinhong Hei, Jeffrey Chan, and Flora D. Salim. 2020. Incorporating LSTM Auto-encoders in Optimisations to Solve Parking Officer Patrolling Problem. 1, 1 (March 2020), 22 pages. https://doi.org/10.1145/nnnnnnn.nnnnnnn

Authors' addresses: Wei Shao, wei.shao@rmit.edu.au, School of Science, RMIT University, Melbourne, Victoria, Australia; Siyu Tan, siyutan.cici@gmail.com, The Faculty of Computer Science and Engineering, Xi'an University of Technology, Xi'an, Shaanxi, China; Sichen Zhao, sichen.zhao@rmit.edu.au, School of Science, RMIT University, Melbourne, Victoria, Australia; Kyle Kai Qin, Kai.qin2@rmit.edu.au, School of Science, RMIT University, Melbourne, Victoria, Australia; Xinhong Hei, heixinhong@xaut.edu.cn, The Faculty of Computer Science and Engineering, Xi'an University of Technology, Xi'an, Shaanxi, China; Jeffrey Chan, jeffrey.chan@rmit.edu.au, School of Science, RMIT University, Melbourne, Victoria, Australia; Flora D. Salim, flora.salim@rmit.edu.au, School of Science, RMIT University, Melbourne, Victoria, Australia.

1 INTRODUCTION

With the evolution of Internet of Things (IoT) technology and rapid urbanisation, many significant challenges have been raised for city management and sensor data analytic. The combination of the IoT, AI solutions, and big data is a booming research area in urban planning that has brought new, interesting challenges towards the goal of future smart cities [5, 40].

Parking violation is one common problem in the daily life, which plays a critical role in traffic management of the city. Recently, the Melbourne City Council has installed thousands of in-ground sensors on the on-street car parking lots in the Melbourne CBD that are used to detect car parking and in violation. The sensors record the parking situation of each parking lot and upload records to the cloud server. The cloud system can notify nearby parking officers the location of the parking violation if the car has overstayed the maximum permitted period. Parking officers will place parking infringement notices on cars in violation if the officer can arrive to the specific location before the car in violation leaves. Since vehicles in breach are likely to leave within a short period, parking officers need to find a path to travel over those cars in violation before those cars escaping from parking lots.

The Travelling Officer Problem (TOP) is a new variant of Travelling Salesman Problem (TSP). Compared to the TSP, it also considers the time constraint and the state of the graph changes during the parking officer travelling process [26, 32]. The TOP was used to model the parking officer patrolling problem. During lunch and dinner time, many cars in the different locations in the CBD are likely to be in violation at the same time. Parking officers needs to arrive at each location and stick infringement notices to each car in violation before it leaves. Therefore, it is necessary to design a reasonable patrolling path for parking officers to catch cars in violation in time. As illustrated in Fig. 1, there are five parking violations occurring around the Argyle Square, the officer needs to visit each violation location and the travelling tour is formed differently according to the algorithm. The red tour is suggested by Ant Colony Optimisation (ACO) and the blue one is generated by First-come First-serve.

Traditional optimisation solutions to the TSP are not suitable for the TOP because those methods do not consider the time constraint and dynamic states of the graph [18] [29] [30] [39]. Therefore, it is necessary to propose some heuristic solutions which take both the historical data and time constraints into consideration. Shao *et al.* proposed two heuristic solutions, ACO and greedy with leaving probability estimation, to solve this problem [32]. Nevertheless, the problem is not completely solved because of many factors such as different leaving probability distributions of each area in the city, real-time performance, and the cost of human resources not being considered. Specifically, the leaving probability estimation model, which estimate the leaving probability of cars in violation when the parking officer arrives at the car in violation, is the essential component to solve TOP.

It is challenging to predict the parking violation event due to uncertainty. Furthermore, using a single probability model to estimate thousands of car parking spaces is inappropriate because the leaving probability model is heavily influenced by the spatial and temporal context of the certain parking lot. Previous research have been conducted to generate useful features to traditional solutions such as rule-based system or single optimisation methods [3, 4]. However, it is difficult to extract useful features from spatio-temporal information since the spatial and temporal information are from two different Euclidean spaces [33].

In this paper, we propose a framework with three steps: extracting features from historical data, clustering features and building the leaving probability model, and path planning using optimisation methods together with leaving probability model. We first propose Long Short-Term Memory (LSTM) auto-encoder model to extract features from spatio-temporal data and use the existing clustering methods to divide spatial space into sub-spaces. Finally, we

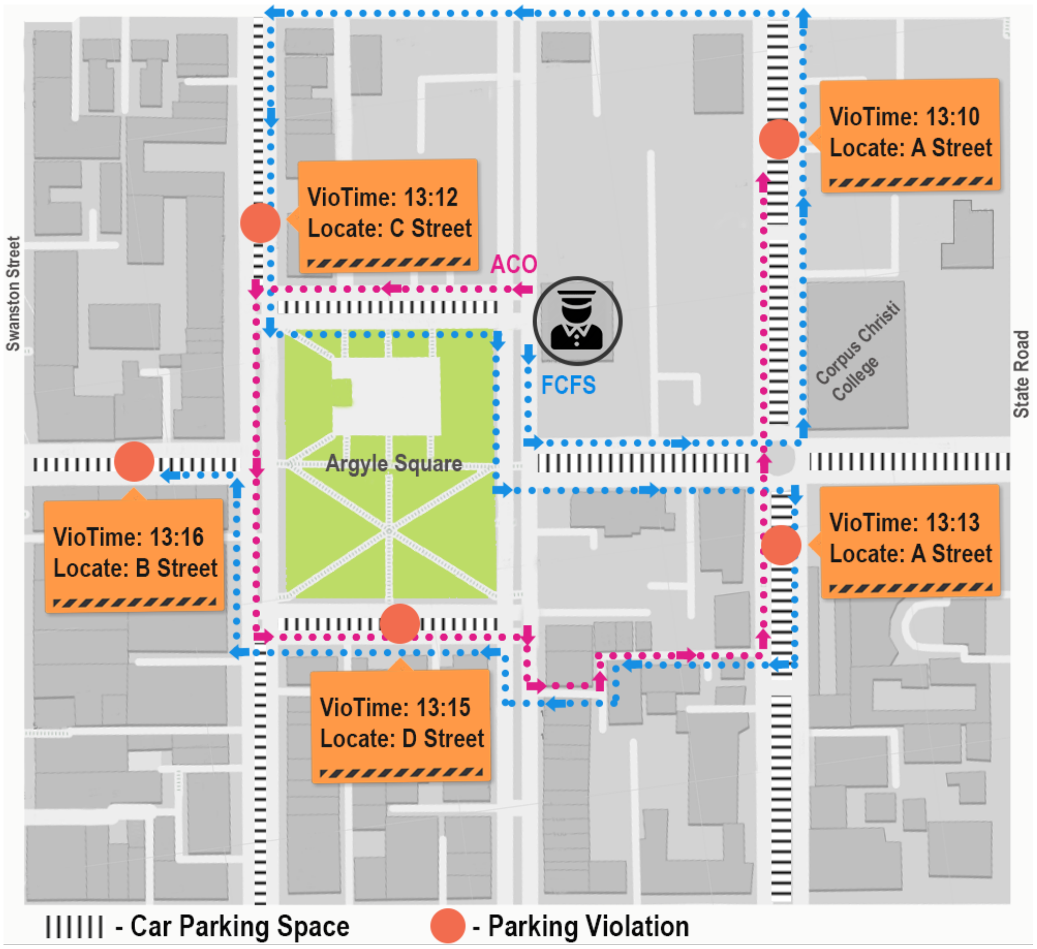


Fig. 1. Demonstration of finding path for officer via different algorithms in the Travelling Officer Problem.

incorporate four popular optimisation solutions – the Greedy algorithm, ant colony optimisation (ACO), the Genetic algorithm (GA) and simulated annealing (SA) and different leaving probability estimation model to solve the TOP. We found that different probability models have significantly different results on the performance of the optimisation solutions in TOP.

We conduct extensive experiments on a large-scale real-world parking dataset. The experimental results show that the combination of our proposed models outperforms other solutions and our proposed heuristic methods are capable of solving the TOP. Additionally, we also compare the cost of human resources with the solutions generated from different combinations of optimisation solutions and probability estimation models.

In summary, this paper makes the following contributions:

- We establish a data-driven framework combined with feature extraction, clustering and optimisation methods to solve an orienteering problem – the Travelling Officer Problem (TOP).

- We propose a LSTM Auto-encoder based solution to extract essential features for parking violation leaving probability estimation.
- We conducted extensive comparison experiments on a real-world large scale dataset to show the different combination of clustering methods, optimisation methods and feature extraction methods on Travelling Officer Problem.

The organisation of this paper is as follows: Section 2 discusses the related works in both optimisation area and smart parking system area. In Section 3, we will introduce the background knowledge and our previous studies. The framework of solutions and the corresponding algorithms are described in Section 4. In Section 5, we present and discuss the experimental results. Section 7 concludes the paper.

2 RELATED WORK

Parking services are becoming increasingly important to providing citizens with high-quality living experiences in major cities worldwide. Many efforts have been devoted to establishing an intelligent parking system to enhance parking services [7]. With the technological progression in wireless networks, sensor communication and smart devices, an Intelligent Parking Assistant (IPA) architecture was established by Barone *et al.* to facilitate the management of public parking and improve the urban mobility. The system can detect the state of any on-street parking slots in real time, which enables the customers to reserve parking bays effectively and also helps deal with overstaying cars timely. An architecture of integrated smart parking system that brings multiple parking service providers under a unified platform to provide one-stop parking information services is proposed in [2]. Ji *et al.* developed a new car parking system that could be integrated into the IoT architecture of smart cities [23]. In this system, the sensor layer utilises sensing technologies for car parking access control. In addition, the communication layer is responsible for transmitting car access information via wireless network to the application layer, where a number of parking services are provided to the public. To further enhance the experiences of searching free car parking space, Caicedo *et al.* proposed a methodology for predicting real-time availability of parking spaces in the intelligent parking reservation systems [9]. Rahaman *et al.* studied real-time parking situations considering various contexts (e.g. weather, time, congestion) to better manage airport ground transport movements [27, 28, 31].

Previous studies show that an inefficient parking service could lead to an increase in urban traffic congestion and air pollution, and illegal parking is a main cause to intensify this impact [7] [12]. Furthermore, it has been pointed out by Cullinane and Polak that illegal parking on the street could also cause a loss of revenue from parking bays, non-compliance with laws and even accidents [12]. Thus, Cullinane and Polak studied the causes and patterns of illegal parking, and the possible relationships between illegal parking and enforcement. The results shown that inadequate resources that deployed for parking enforcement could undermine the effectiveness of the management. In 2012, a similar study was conducted by Spiliopoulou and Antoniou to analyse illegal parking behaviours in several cities in Greece [37]. The investigation recommended the authorities to apply a certain level of enforcement at different regions to ensure compliance to parking regulations. The solutions for a smart parking management system have brought a great deal of benefits to the citizens. However, there is a very few literature that aim at assisting the parking officers to deal with on-street parking violations of vehicles. Recently, Dinh and Kim developed a new IoT-cloud-base system for enhancing the administration of cat parking violations in the City of Melbourne [14]. When the violations are detected by on-ground sensors at parking bays, the system records and computes the travelling distance between an officer and the violation locations, then recommends the shortest path for the officer to handle the parking violation. The results show that the system can improve the work of officers in finding parking violations and fines collection.

The TOP can be regarded as a variant of the TSP which is a well-known NP-hard problem. TSP has been solved by many meta-heuristic algorithms that could be the potential solutions of finding optimal path for an officer in TOP. A parallel genetic algorithm was used by Miihlenbe and Kindermann to solve TSP in 1989, which can simulate each individual of population on a separate processor [25]. Two important differences in this parallel genetic algorithm are that the selection is conducted locally on a neighbourhood and each individual of solutions is improved by hill climbing. To further improve the efficiency of finding optimal solution on TSP, Braun enhanced the underlying basic operators, such as making the offspring consists of two sub-strings from the parents via the order crossover and applied 2-opt and or-opt heuristic as the local optimisation [8]. In 2003, Simulated Annealing (SA) was developed by Song *et al.* based on Grand Canonical Ensemble for the TSP. The experimental results show that SA is a simple algorithm and has an excellent performance on finding high-quality solutions [36]. Moreover, an artificial Ant Colony algorithm was used by Dorigo and Gambardella on solving both symmetric and asymmetric TSP in 1997. The results indicate that the algorithm can successively create shorter feasible tours by using a pheromone trail as information that accumulated on the edges of the TSP graph [15].

Most of the route planning problems can be seen as a generalisation of TSP. One well known query called the Trip Planning Query (TPQ), which aims at finding the best trip from two different locations that passes at least one point from each given category such as a gas station or a post office. To evaluate the efficiency of TP queries, Soma *et al.* proposed a new algorithm to refine the search space as an elliptical region using geometric properties, with the protection of location privacy of the users [35]. Haryanto *et al.* proposed an efficient indexing technique called IG-Tree to accelerate the efficiency on best path planning. The new indexing technique incorporates both keywords about locations and spatial information of road networks when searching the best tour[20].

3 BACKGROUND

3.1 On-Street Parking Infringements

The in-ground sensor system has been set up to monitor the states of parking bays around Melbourne CBD and the parking dataset is available online for the public to use and study. In 2016, there are approximately 399,000 parking violations occurred in the city centre. Each record contains the information of a parking event, such as area name, bay id, arrive time, departure time and parking rule. Furthermore, the coordinates of each parking bay is published on the platform along with the sensor records. The entire CBD is divided into different regions by the city council, and each region is monitored by a parking officer.

Fig. 2 is the heat-map of the yearly parking violations distribution around Melbourne CBD in 2016. Totally, the parking events of 1,120 parking bays are covered in this data. And the position in deep red colour is the area where the degree of parking violation is specially intensive. As we can see that, many places with heavy parking infringements are located in the west and east of the map. The relatively dispersed distribution of parking violations renders us an opportunity to apply the technique of clustering in building probabilistic models and finding optimal path for officer.

3.2 Travelling Officer Problem

Shao *et al.* proposed Travelling Officer Problem (TOP) to modelling the procedure that parking officers seeking for cars in violation within limited time. [32]. Formally speaking, the parking nodes are modelled as a graph $\mathcal{G}$ =

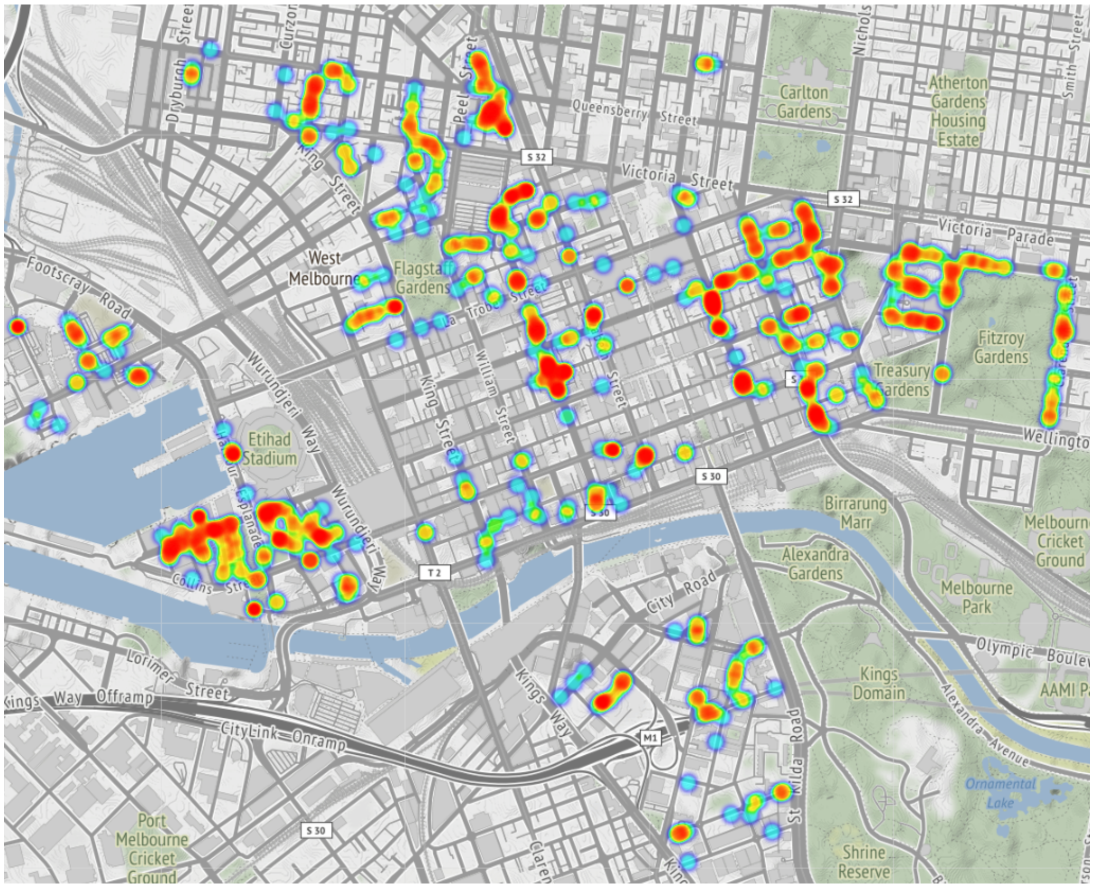


Fig. 2. Distribution of yearly parking violations around Melbourne CBD areas in 2016.

$(\mathbf{V},\ C(u,\ w)), u,\ w \in \mathbf{V}$, where $\mathbf{V}$ denotes the parking nodes, and $C(u,\ w)$ denotes the time consumption for a parking officer walk from node u to w. For each node $x_i \in \mathbf{V}$, there are two possible state: 0 denotes there is no car in violation at the node x_i, otherwise 1. Let $f_{j,t} \in \{0,\ 1\}$ denotes the state of node j at time t.

Let T be the total time budget. Solution $\mathbf{S} = \{(x_1, t_1), (x_2, t_2), ..., (x_{|\mathbf{S}|}, t_{|\mathbf{S}|})\}$ is a path consists of nodes and timestamps, where $x_i \in \boldsymbol{v}$ and denote the i_{th} node in the path, and $t_i \in Time$ denote arriving time at the node x_i. We simplify our path $\mathbf{s}$ to $x_1, x_2, \ldots, x_{|S|}$. Let $\mathbb{R}$ denote the fines parking officer collected. In this paper, we assume the $\mathbb{R}$ is a constant value.

A formal definition of this problem is as follows:

$$\arg\max_{S} \sum_{x_i \in S} f_{i,t} \cdot \mathbb{R}$$

s.t.

$$\sum_{i=1}^{|S|-1} C(x_i,\ x_{i+1}) \ \leq T \tag{1}$$

$$t_k = \sum_{i=2}^{k} C(x_i,\ x_{i+1}), \text{for } 1 < k < |S| \tag{2}$$

$$t_1 = 1 \tag{3}$$

4 METHODOLOGY

In this section, we will describe the framework used in this paper. As illustrated in Fig. 3, there are mainly three components of the proposed framework. The first component is the feature extraction. We use a deep learning neural network called LSTM auto-encoder to extract and predict the spatio-temporal features from historical data. In the second component, we apply two clustering methods to features of each parking lot and group the leaving probability model of each parking space. At last, we combine four different optimisation methods and probability model, and test it on the real-world dataset. We will introduce the details of each component in the following sections.

4.1 LSTM Auto-encoder

In this paper, we are using an LSTM auto-encoder to extract latent features from the violation time distribution data. The main reasons that we choose the LSTM auto-encoder are listed below:

(1) The dimensionality of the original violation time distribution data is too high, a dimension reducing method is needed to prevent the potential occurrence of the 'curse of dimensionality'.
(2) Since the next section in our architecture, is an unsupervised clustering section, a method that can self-evaluate is needed to help improve the performance of this feature extraction during the training process.
(3) The violation time distribution data in this section is a sequence data, the chosen method should have the ability to preserve the sequence characteristics of the input data.

An auto-encoder is a feed-forward neural network model which try to learn a compressed representation of its input [21]. Since the input is also used as the label to evaluate the performance of the model, it is technically an unsupervised learning method which using the supervised learning manner. This model provides a way to evaluate the performance of the feature extraction model without any label [6].

The typical design of an auto-encoder is in an Encoder-Decoder structure. Intuitively, the purpose of the first part in this 2-phase structure is to extract features from the input which contains all that is needed to reproduce the input in the second phase. After each round of the training process, the error between this reconstructed output and the original input is calculated to evaluate the performance of this model.

Once the model is trained, the reconstruction aspect of the model can be discarded. The output of the remaining part is a fixed length vector representation that provides a compressed extraction of the input and can be used as input of other models.

The reason we choose the LSTM auto-encoder instead of the ordinary one is that the violation time distribution data in this paper is a sequences data, the compressed representation should also contain the potential sequence information. The normal auto-encoder structure might not satisfy this criteria the LSTM, which is a type of recurrent neural network,

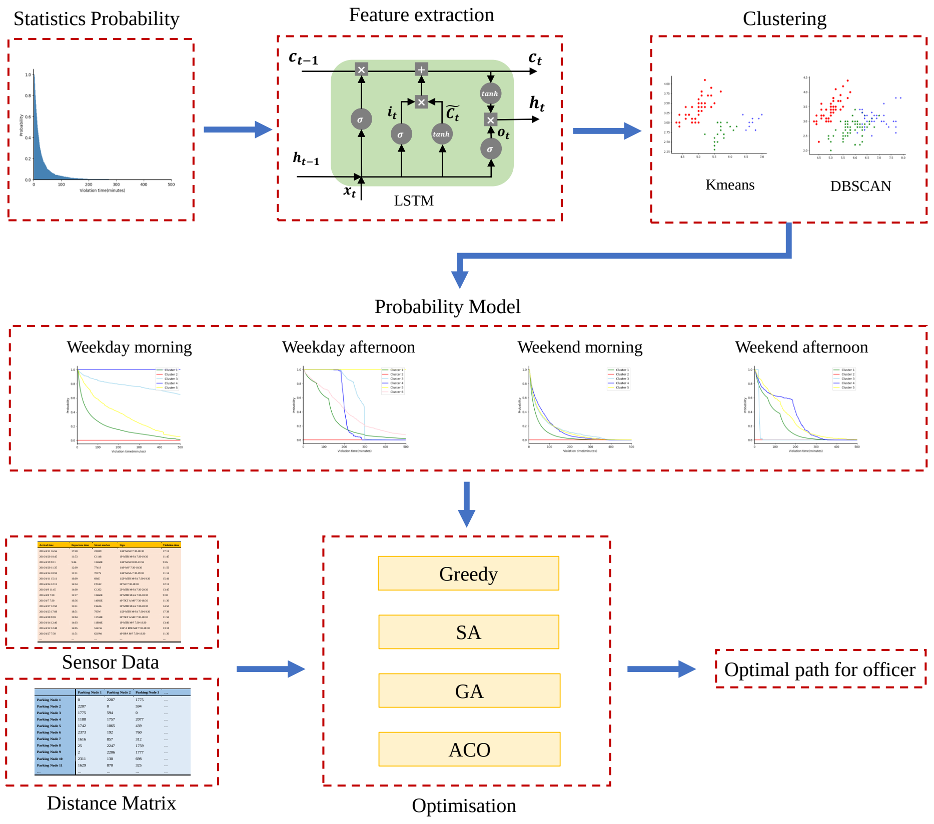


Fig. 3. Framework of proposed model

is specifically designed to fit the sequence input data [22]. It is capable of learning the complex dynamics within the temporal ordering of input sequences, which in our case, is the potential length of the violation.

As shown in the Fig. 4, by following the architecture used by Srivastava *et al.* [38], there are two separate LSTM layers, the first one is the auto-encoder part that maps our input into a vector representation and the second one takes the output of the encoder to reconstruct the input. Intuitively, the more close the reconstructed target sequence is to the original input data, the better extraction this vector representation is. This model will be pre-trained for 200 epochs and once this model achieves a desired level, the decoder part of this model is removed. The left part can produce a fixed length vector from the original sequence data and it will be used as the input of the following clustering section.

4.2 Clustering

Clustering methods used in the spatio-temporal prediction problem has been proved effective [18, 34]. It is because that clustering methods can summarise the sparse data and distinguish different groups of temporal data. In parking system, parking events for some parking lots are sparse. It is difficult to predict the parking event time using only a couple of parking records in a complete year.

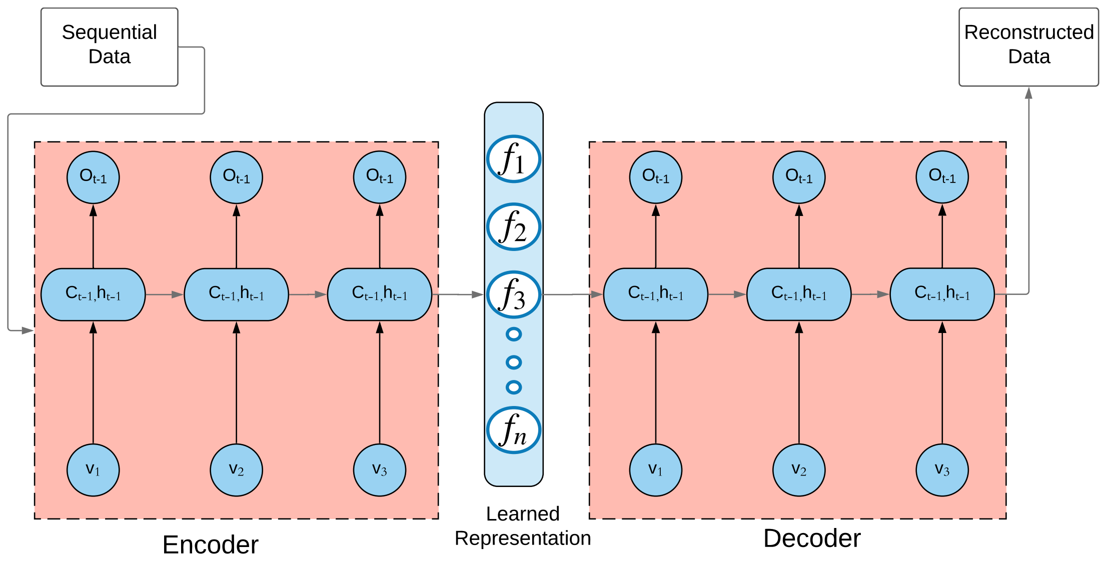


Fig. 4. The LSTM Auto-encoder model structure

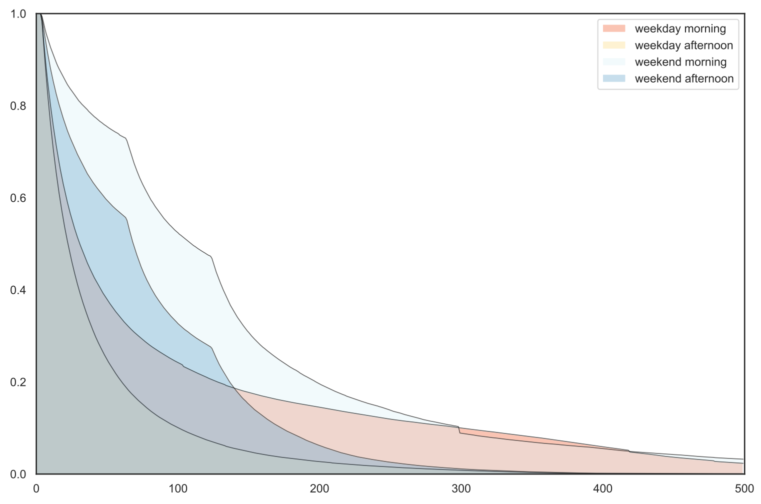


Fig. 5. The difference among the distribution of violation time during different time period

As shown in Fig. 5, four sections in this graph represent the average parking violation time distribution on weekday morning, weekday afternoon, weekend morning, weekend afternoon respectively and the difference among those lines are be easily spotted. Therefore, a rigid universal probability model might not the fit the real-world need at some certain

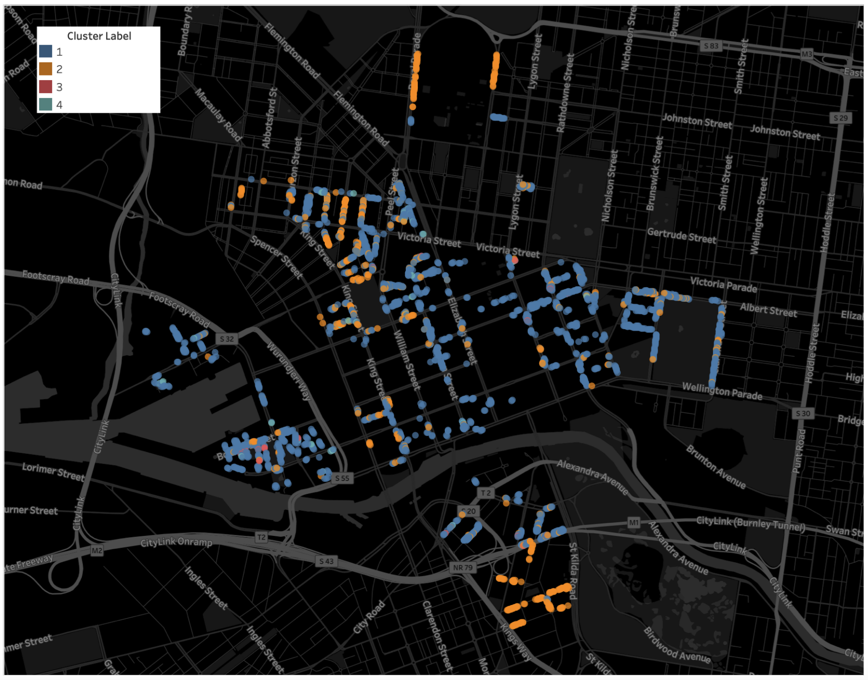
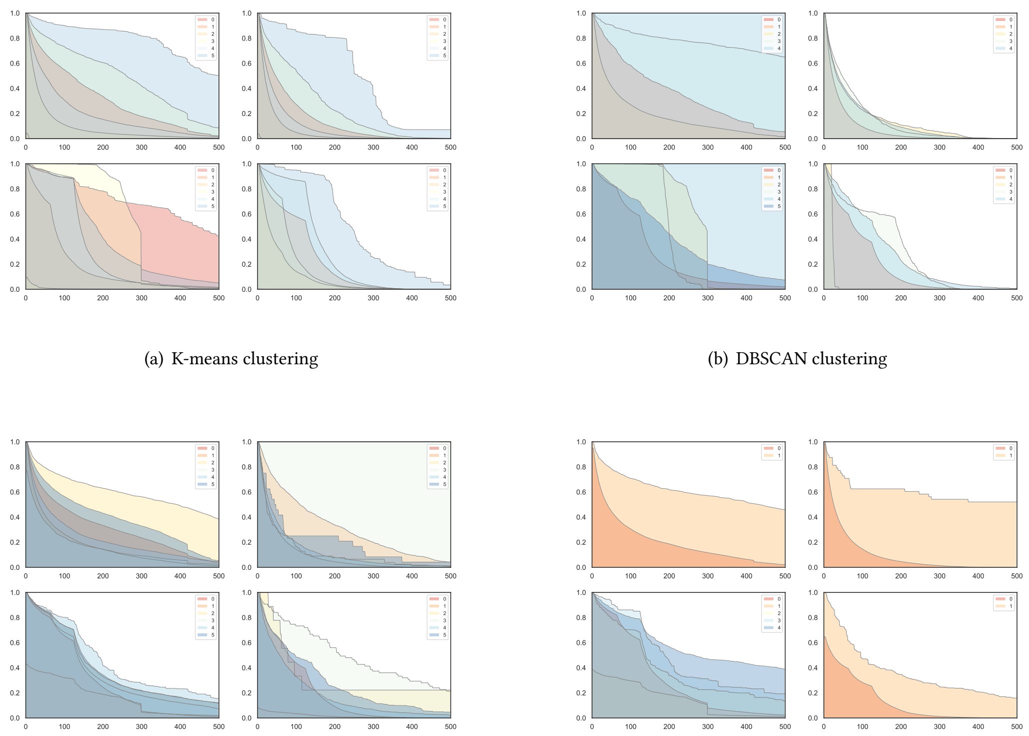


Fig. 6. The DBSCAN clustering result for all parking bays in the morning of the weekend after LSTM Auto-encoder.

periods of time. Besides the significant difference among the distribution of the violation in different time periods, the difference between different parking bays is also noticeable, due to some specific factors such as location; traffic density around it; special parking rules and etc. It might be helpful for the officer to determine its next target if they are partitioned based on their hidden characteristics, rather than use a universal equation for all parking bays [32].

In this paper, we choose two clustering method, K-means and Density-based spatial clustering of applications with noise (DBSCAN), to cluster the parking bays. K-means is an algorithm that partitions all node in the space based on their distances to all the cluster cores and updates the core each iteration [19]. It is fairly simple but requires some insights about the dataset itself to determine how many clusters work best. Silhouette index was used to select the number of cluster. DBSCAN, on the other hand, uses the concept of reachability [16] and can self-determine the number of clusters in the result. But it might fail if there are clusters with different density exists in the dataset.

For comparison, we will perform these two methods on the raw input data, i.e. the violation time distribution data without LSTM auto-encoder and do the same clustering based on the vector representation from auto-encoder as well. Fig. 6 shows the clustering result, and the parking bays located in the same street are usually put into the same cluster, since they tend to share the same circumstances. However, the parking bays that in the same cluster is not necessarily located closely in case of geographical locations. That is possibly due to the hidden sequence features they shares, which are preserved by the LSTM auto-encoder.



(a) K-means clustering (b) DBSCAN clustering

(c) K-means clustering with LSTM auto-encoder (d) DBCSAN clustering with LSTM auto-encoder

Fig. 7. Leaving probability model of each cluster.

4.3 Dynamic Temporal Probability Model

As mentioned previously in section 3, the distribution of violation time changes even for the same parking bay during the different time periods, a universal probability model mentioned in [32] might lose its effectiveness at certain circumstances. In this paper, we used a method to construct the dynamic temporal probability model based on historical violation data.

After the previous clustering section, we group the nodes that in the are in the same cluster and use their grouped up historical statistic violation time data to construct a dynamic temporal probability model for this cluster during this specific time period. So the time-based probabilistic model can be described as Eq. 4:

$$P_{v_j}^m = P_{c_i}^m(t) = P_{c_i}^m(t_{dep} - t_{vio}) \tag{4}$$

where v_i is a target node and c_i is the label of the cluster this node belongs to. And m denotes the different time period when the violation occurs.

Besides using this method on the clustering output, we also calculate the dynamic temporal probability model on the whole dataset without any clustering method which can show insights about the effectiveness of the clustering.

The exponential distribution mentioned in [32] is also used in this experiment as the baseline. As illustrated in Fig. 7, we segment the temporal space into four categories: weekday morning, weekday afternoon, weekend morning and weekend afternoon. From observation, we found that the distribution of each temporal segment are significantly different from each other. Therefore, it also proves that it is necessary to clustering the parking space in terms of both spatial space and temporal space.

4.4 Path-Finding Algorithm

In this section, we revise the two solutions mentioned by Shao *et al.*, which is greedy and ant colony based optimisation framework, and proposed two new solutions - the simulated annealing algorithm and a genetic algorithm solution. All four of them are based on the probability estimation model we get from the previous section.

4.4.1 Greedy Algorithm. The greedy heuristic is a fairly simple heuristic which will always choose the local optimum at every decision-making point [11]. Although there is no guarantee on finding the global optimum, greedy algorithm is still the most popular and sometime even the best solution for solving a NP-hard problem [10]. As shown in Algorithm 1, in our experiment, the patrolling officer will count the expected reward for all nodes that are currently having a violation event, and then choose the node with the highest expected reward as its next destination, which suggests the highest possible fines per minute he expected to get. And then it will perform the action of moving to that location and select its next target using the same method.

Algorithm 1 The Greedy Algorithm with Probability Estimation Model

```
Input: a given graph $G = (V, E)$
   a start node $v_c$
Output: a solution $S$
1: init solution $S = \phi$
2: while $Cost(S) < T_{max}$ do
3:    calculate $C_i = Cost(V_i, v_c), i \in [1, n]$, $n$ is the number of nodes and $V_c$ is the current node
4:    updates $p_i(t_i)$ for all nodes, $t_i = t_{vio} + C_i$ is the time that has been violated
5:    update the possible target set $\Omega$
6:    if $\Omega = \phi$ then
7:        stay at the current location $v_c$ for a certain period of time
8:    else
9:        calculate $E_i = \frac{p_i(t_i)}{t_i}$ for all possible targets in $\Omega$
10:       select the node with the highest expectation $E_t$
11:   end if
12:   add the next location to $S$
13:   perform the moving action and update the current state
14: end while
```

4.4.2 Simulated Annealing Algorithm. The simulated annealing algorithm (SA) is a meta-heuristic algorithm derived from a statistical mechanics [24] and the main concept of this algorithm is that it will accept a worse solution which might help to avoid falling into a local optimum. And this acceptance rate will decrease over time in simulating the cooling process, which will provide unnecessary 'teleport' while the solution space is revealed more and more. Since in TOP, the search space is discrete, SA can be used and the precise algorithm is shown in Algorithm 2:

Algorithm 2 The Simulated Annealing Algorithm with Probability Estimation Model

```
Input: a given graph $G = (V, E)$
    a start node $v_c$
    a cooling rate $r$
    a start temperature $m$
Output: a solution $S$
1: init a solution set $S = \{v_1, v_2, v_3, ...v_m\}$ based on Greedy Algorithm, $m$ is the length of solution $S$
2: calculate the benefits for this solution $B(S)$
3: while $m > m_{min}$ and not reach the max iteration $M$ do
4:     randomly select two nodes $v_i, v_j$ from $S, 1 \leq i \leq j \leq m$
5:     exchange their order to construct the new solution $S^{'} = \{v_i, ..., V_j, ..., v_i, ..., v_n\}$
6:     trim or extend solution $S^{'}$ to make $Cost(S^{'})$ close to but still under $T_{max}$
7:     calculate the benefits $B(S^{'})$ for new solution
8:     if $B(S) < B(S^{'})$ then
9:         $S = S^{'}$
10:    else
11:        accept new solution $S^{'}$ with Metropolis Algorithm
12:    end if
13:    $m = m * r$
14: end while
```

The first thing is to obtain a basic solution based on the Greedy algorithm. At each iteration, randomly choose two nodes v_i, v_j in the solution S, and then swap their place. Trim or extend the solution to make it as close to the maximum time limit as possible while still under it. And using the total benefits of the new solution, if it is higher than the benefits this officer can get from the old one, it will accept it. It the total benefit of the new solution is lower, then the officer will use the Metropolis Algorithm to determine whether to accept it or not [17].

4.4.3 Genetic Algorithm. Genetic algorithm (GA) is an algorithm inspired by biological operators, and inherit the term fitness, mutation and crossover [13]. The algorithm will exploit the current knowledge by choosing the parents who have a high fitness value, while keep exploring new solutions which might help to avoid local optimum by applying random mutations and crossovers when forming its next generation.

As shown in Algorithm 3, it will be firstly initialized with a basic set which has a population size of k. For each generation, the fitness of every individual will be calculated and the ones with higher fitness will more likely to be chosen as parents. After a bootstrap sampling procedure, where data are sampled randomly with replacement, some mutation and crossover actions will be performed based on their respective rate and results in the next generation. The algorithm will return the best solution with the highest benefit in the last generation.

4.4.4 Ant Colony Optimization Algorithm. The name of the ant colony algorithm (ACO) is very self-explanatory, which is derived from the path-finding method using by ant colonies [15]. A real ant is capable of finding the shortest path to the food base on the pheromone trace previously left by other ants. The ACO imitate the strategies used by ant colonies, maintaining exploring other solutions by roulette selecting, and exploit the current knowledge by choosing the path with the highest pheromone value, which means it is chosen by most other ants.

The complete algorithm is shown is Algorithm 4. At each round there will be a total number of l ants, each ant will get a solution, with is a set of nodes based on the previous knowledge, i.e. the pheromone matrix. And at the end of this

Algorithm 3 The Genetic Algorithm with Probability Estimation Model

```
Input: a given graph $G = (V, E)$
    a start node $v_c$
Output: a best solution $S_b$
1: init a solution $\Phi(S) = \{S_1, S_2, S_3, ...S_k\}$ based on Greedy Algorithm, $k$ is the population size
2: calculate the benefits for this solution $B(S_i), i \in [1, k]$
3: $S_b = argmax(B(S_i))$
4: while $iter < iter_{max}$ do
5:     for $i = 1$ to $k$ do
6:         calculate the fitness $f_i = exp(B(S_i))$
7:         bootstrap select $k$ solution to form a new solution set $\Phi(S^{'})$ with probability $p_i = \frac{f_i}{\Sigma_{i=1}^{k} f_i}$ for each solution $S_i$ in $\Phi(S)$
8:     end for
9:     for $i = 1$ to $k$ do
10:        $r_c$ is a random number between 0 and 1, $p_c$ is the crossover probability.
11:        if $r_c < p_c$ then
12:            Randomly select another solution $S_j^{'} \in \phi(S^{'})$
13:            Randomly choose partial solutions from both $S_i^{'}$ and $S_j^{'}$ and concatenate them to obtain a new solution $S_i^{''}$
14:            $S_i^{'} = S_i^{''}$
15:        else
16:            $S_i^{'} = S_i^{'}$
17:        end if
18:        for $index = 1$ to the length of $S_i^{'}$ do
19:            $r_m$ is a random number between 0 and 1, $p_m$ is the mutation probability.
20:            if $r_m < p_m$ then
21:                Randomly select two point in $S_i^{'}$ and swap those two point and obtain a new solution $S_i^{''}$
22:                $S_i^{'} = S_i^{''}$
23:            else
24:                $S_i^{'} = S_i^{'}$
25:            end if
26:        end for
27:    end for
28:    record new best solution $S_b$
29:    $\Phi(S) = \Phi(S^{'})$
30:    $iter+ = 1$
31: end while
```

round, this matrix will be updated with the solutions constructed by the ants in this round. And after all ants finished their actions, we will get the best solution so far, using the final pheromone matrix. More details are explained below:

$$Q(path_{ij}) = \frac{[\tau(path_{ij})]^{\alpha}[\eta(path_{ij})]^{\beta}}{\Sigma_{k \in V}[\tau(path_{ik})]^{\alpha}[\eta(path_{ik})]^{\beta}} \tag{5}$$

$$\eta(path_{ij}) = \left[\frac{1}{Cost(v_i, v_j)}\right]^{\alpha} [E_j]^{\beta} \tag{6}$$

Algorithm 4 The Ant Colony Optimization Algorithm with Probability Estimation Model

```
Input: a given graph $G = (V, E)$
    a start node $v_c$
    the ant count $l$ for each iteration
Output: a best solution $S_b$
1: init the pheromone matrix $\tau_{ij} = \frac{1}{Cost(v_i, v_j)}$, $Cost(v_i, v_j)$ is the travel time between two nodes $v_i, v_j$
2: while $iter < iter_{max}$ do
3:    for $i = 1$ to $l$ do
4:       while $Cost(S) < T_{max}$ do
5:          selected next node by probability roulette $Q()$
6:       end while
7:       calculate the benefit $B(S_i)$ for this ant's solution $S_i$
8:       update the best solution $S_b$
9:    end for
10:   update the pheromone matrix $\tau$
11:   $iter+ = 1$
12: end while
```

where S_{ij} means the potential path from node i to node j, V is the set of all vertices. E_j is the reward per unit time from node v_i to v_j. α and β are two constants represent the weight of pheromones and weight of visibility, respectively. This equation is used to calculate the probability an ant will explore other routes based on the two preset constants α and β.

$$\tau_{ij}(iter) = (1-\rho)\tau_{ij}(t-1) + \Sigma_{k=1}^{l}\tau_{ij}^{k}(iter) \tag{7}$$

$$= (1-\rho)\tau_{ij} + \Sigma_{k=1}^{l}\tau_{ij}^{k}(iter-1) + \omega * P_j \tag{8}$$

where ρ is the pheromone evaporation coefficient and ω is the pheromone enhancement coefficient. And this equation shows how to update the pheromone matrix after each iteration.

5 EXPERIMENT RESULTS

In this section, we discuss the improvement achieved by addressing TOP with the methods proposed. We will first describe the experimental settings include running environment, evaluation metric and simulation settings. Then we will show and discuss two sets of experiments. The first set of experiment shows the weekly results yield from different combination of probability model and optimisation methods. The second experiments shows the average performance by different combination of probability models and optimisation methods using three different criteria.

5.1 Evaluation Settings

In this section, we will show our experimental environment, simulation conditions and parameters settings.[1]

5.1.1 Evaluation Environment. Optimisation methods are complied and tested on a Linux server (CPU: Intel(R) Xeon E5-2690 2.60GHz). Features extraction and clustering are executed on Google Colab. We do grid search to select

[1]https://github.com/zschaoihen/LSTM-Autoencoder-Parking

the hyper-parameters of all models. Grid search is a brutal force hyper-parameters searching method which test all combinations of different hyper-parameters and select the best one.

For LSTM auto-encoder model, we test the $Batch\_size = \{32, 64, 128, 256\}$, $Learning\ rate = \{0.01, 0.03, 0.001, 0.003\}$. For K-means clustering, and we test the $K = \{2, 3, 4, 5, 6, 7, 8, 9, 10\}$. And for DBSCAN clustering, we test the $min\_samples = \{5, 6, 7\}$, $eps = \{0.3, 0.4, 0.5, 0.6, 0.7, 0.8, 0.9, 1.0, 1.1, 1.2, 1.3\}$.

Table 1 lists all hyper-parameter of models and methods we used in the experiments. For fair comparison, we select parameters for LSTM auto-encoder based model and without LSTM auto-encoder based model, respectively. Table 1 only shows the parameter setting of the best performance.

5.1.2 Evaluation Metric. In TOP, the most important criterion is the number of rewards $\mathbb{R}$. Except the rewards obtained everyday, some other factors such as human resource cost and catch accuracy should also be taken into account because City Council also need to reduce the workload of parking officers. Therefore, we propose two other criteria: Rewards/Visited nodes and Rewards/Distance. The former one suggests the accuracy of chosen path, and the latter one denotes the cost-effective of chosen path. Additionally, we choose the rewards/The number of total car parking violations (Rewards/Total Vio Num) instead of rewards because the number of violations varies everyday. Although we propose three evaluation metrics to measure the performance of different approaches, only Rewards/Total Vio Num is the critical criteria in our problem. Therefore, our problem is a single-objective problem.

For all experiments, we use the real-world data extracted from Melbourne in-ground parking space sensors which is available through City of Melbourne's Open Data Platform [1]. We have discussed with Melbourne City Council and simulate the parking officer patrolling process. We select a complete year data from 2016. We select 10 month data for training and 1 month data for validation. Regarding the working conditions of the simulations, they are the followings:

- Parking officer speed: 70 meter/minute
- Working hours from 7am to 7 pm per day
- The start position is the central railway station
- In order to simplified the problem, we assume parking officers will only change the direction when they are at the parking space.
- All distance are calculated by drive distance from Google Map.

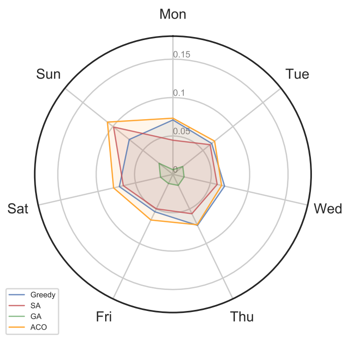
5.2 Experimental Results

In the first set of experiment, we random select one week data from November which is only used for validation. We leverage parking data from other 11 month in this year as the training dataset. Fig. 8 shows the overall daily performance for different combinations of optimisation solutions and leaving probability model in a week. For each subfigure, we show the ratio of rewards obtained to total potential rewards using one probability model combined with four optimisation methods. From the observation of Fig. 8, we can draw a couple of conclusion as follows: 1) All experiments show that all combination of methods perform best on Sunday. We have investigated from dataset and found that parking violations are more dense because the parking rule on Sunday is different from other days. Therefore, the parking officers can collect more fines from cars in violation on Sunday. 2) Ant colony optimisation (ACO) outperforms other optimisation methods with any probability model, and GA perform worst among all optimisation solutions. 3) Clustering method boost the performance of all results compared with optimisation methods with exponential probability model and average model. 4) Overall, the combination of LSTM auto-encoder method, clustering method and optimisation perform better than other combinations.

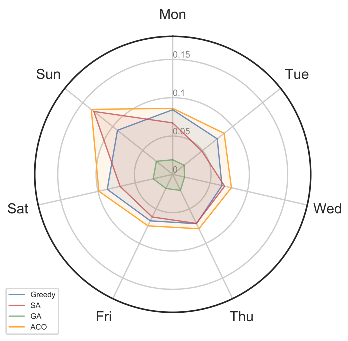
Table 1. Parameter configuration

Encoder	LSTM auto-encoder			
Pre-train epochs	200			
Batch size	256			
Optimiser type	Adam			
Learning rate	0.001			
Time Period	Weekday morning	Weekday afternoon	Weekend morning	Weekend afternoon
Clustering Method	K-means			
n_clusters	6	6	6	6
Clustering Method	LSTM+K-means			
n_clusters	6	6	6	6
Clustering Method	DBSCAN			
eps	1.3	0.4	1.3	1
min_samples	7	5	7	6
Clustering Method	LSTM+DBSCAN			
eps	0.3	0.3	0.3	0.3
min_samples	7	7	7	7
Optimisation Method	SA			
r	0.995			
m	33.5857			
m_{min}	0.00001			
M	5000			
Optimisation Method	GA			
p_c	0.7			
p_m	0.3			
k	500			
$iter_{max}$	50			
Optimisation Method	ACO			
l	10			
$iter_{max}$	20			
α	1.0			
β	10.0			
ρ	0.5			
ω	10			

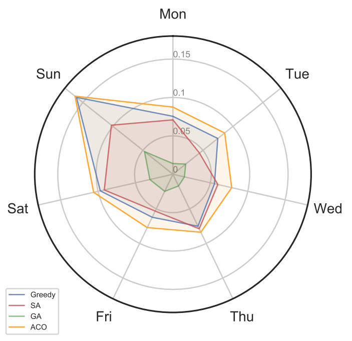
The second experiment shows the comparable experiments with different combination of four optimisation solutions and six leaving probability model. We evaluate the experimental results with three different criteria. The results are shown in Table 2. We conducted the experiments weekly and repeated it for one month. The average performance of the combination of DBSCAN, LSTM auto-encoder and ACO outperform other combinations in the term of rewards/visited nodes. The average performance of the combination of K-means, LSTM auto-encoder and ACO outperform other combination in terms of the second and third criterion. In brief, the LSTM auto-encoder approach yields better results



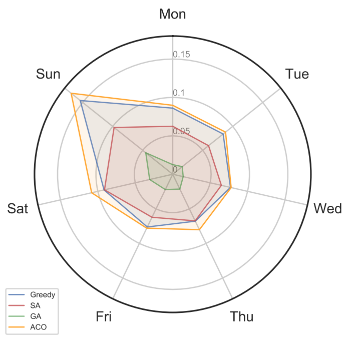
(a) The result using the exponential distribution



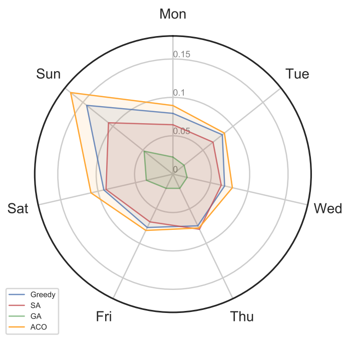
(b) The result using the average distribution of the historical data



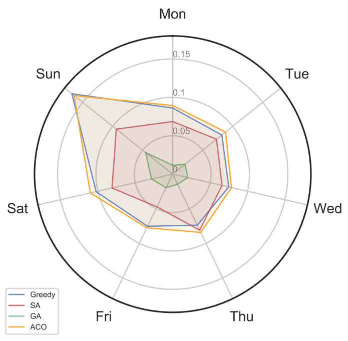
(c) The result using K-means clustering



(d) The result using DBSCAN clustering



(e) The result using K-means clustering after LSTM auto-encoder



(f) The result using DBSCAN clustering after LSTM auto-encoder

Fig. 8. The comparison result of different combination of optimisation methods and probability model in a week.

for all experiments relative to other approach without feature extraction. We prove therefore that integrating spatio-temporal features extraction step using LSTM auto-encoder and clustering methods to summarise leaving probability model can yield better results than their simple optimisation solution version. It is clear that establishing a proper leaving probability model for each car parking lot is significantly important to improve the performance of optimisation solutions.

Table 2. Pairwise comparison

Algorithm	Greedy	SA	GA	ACO
Leaving Probability Model	Rewards/Visited Nodes			
K-means+LSTM auto-encoder	0.903643	0.875328	0.662555	0.902561
DBSCAN+LSTM auto-encoder	0.908119	0.875273	0.586463	**0.914410**
K-means	0.909446	0.886700	0.636853	0.909301
DBSCAN	0.893755	0.875273	0.551561	0.906447
Exponential	0.888761	0.869388	0.432583	0.890102
Average	0.903504	0.857803	0.649487	0.899305
Leaving Probability Model	Rewards/Distance			
K-means+LSTM auto-encoder	0.004789	0.004167	0.001466	**0.005721**
DBSCAN+LSTM auto-encoder	0.003933	0.003933	0.001226	0.005709
K-means	0.004526	0.003800	0.001325	0.005643
DBSCAN	0.004975	0.003834	0.001348	0.005683
Exponential	0.003572	0.003141	0.001031	0.004101
Average	0.004087	0.003982	0.001101	0.005161
Leaving Probability Model	Rewards/Total Vio Num			
K-means+LSTM auto-encoder	0.088325	0.077291	0.026300	**0.099095**
DBSCAN+LSTM auto-encoder	0.094621	0.072967	0.022810	0.098863
K-means	0.085825	0.071614	0.024298	0.097455
DBSCAN	0.091021	0.072218	0.023086	0.098780
Exponential	0.068097	0.062479	0.015147	0.076413
Average	0.077569	0.074297	0.021541	0.091200

To show the robustness of the proposed technique, we apply ANOVA test ($n = 10$) to different approaches, and show the result in Table 3. We can conclude that there is a statistically significant difference in the evaluation metric between the different probability models since the F ratio is greater than critical value ($p < 0.05$) for all methods. Meanwhile, it also concludes that there is no statistically significant difference in the performance of the same approaches with the same probability model for every time ($p < 0.05$).

6 DISCUSSION AND FUTURE WORK

There are still have some weakness of the work. First, our method does not rely on any specific dataset, and our framework can be generalised to other spatio-temporal dataset. In the future, we plan to apply our framework to more spatio-temporal datasets. Second, a deeper data analytic is needed. Although we have shown that our proposed method performs well, it also needs to explain the reason and the correlation between the data distribution and the performance of different combinations of methods. Third, our model consists many different inconsistent steps. In the future, we plan to propose an end-to-end deep learning model with optimisation approaches to search for uncertain spatio-temporal events. Fourth, in this paper, we assume that paring slots are independent. However, we investigated that the parking states are influenced by neighbour parking slots. Therefore, the performance is likely to be improved if we take it into account.

Table 3. ANOVA results

Algorithm	Greedy	SA	GA	ACO
Evaluation Metrics	F			
Rewards/Visited Nodes	4.32E+28	2.79E+5	3.46E+1	5.86E+4
Rewards/Distance	7.79E+30	7.12E+6	1.87E+1	1.19E+3
Rewards/Total Vio Num	1.34E+31	3.05E+6	7.39E+1	1.17E+3
Evaluation Metrics	P-value			
Rewards/Visited Nodes	0	7.69E-118	1.13E-15	1.63E-99
Rewards/Distance	0	8.54E-156	1.01E-10	6.62E-54
Rewards/Total Vio Num	0	7.31E-146	6.64E-23	9.85E-54

7 CONCLUSION

This paper propose a new framework to address Travelling Officer Problem. Unlike previous studies, where the focus was on optimisation methods and a simple exponential probability leaving probability estimation, we first extract spatio-temporal features from historical data and establish a more accurate probability model using clustering methods. We also applied more optimisation methods to larger dataset and combined with different leaving probability model. Based on extensive multiple comparison experiments, the results shows that LSTM auto-encoder and clustering algorithm could significantly improve the performance of each optimisation solution. Our proposed framework outperforms previous work.

ACKNOWLEDGMENTS

At the time of writing, Wei Shao was supported by Northrop Grumman Corporations USA. This paper was partially supported by Australian Government's Smarter Cities and Suburbs Project grant with Mornington Peninsula Shire. We also acknowledge the support of Australian Research Council *DP190101485.* Siyu Tan and Xinhong Hei were supported by Natural Science Foundation of China (No. 61773313), and Key Research and Development Program of Shaanxi Province (2017ZDXM-GY-098, 2019TD-014).

REFERENCES

[1] 2016. On-street Car Parking Sensor Data - 2016. https://data.melbourne.vic.gov.au/Transport-Movement/On-street-Car-Parking-Sensor-Data-2016/dj7e-rdx9

[2] Sabbir Ahmed, Mohammad Saidur Rahman, and Mohammad Saiedur Rahaman. 2019. A Blockchain-Based Architecture for Integrated Smart Parking Systems. In *2019 IEEE International Conference on Pervasive Computing and Communications Workshops (PerCom Workshops)*. IEEE, 177–182.

[3] Md Ridwan Al Iqbal, Mohammad Saiedur Rahaman, and Syed Irfan Nabil. 2012. Construction of decision trees by using feature importance value for improved learning performance. In *International Conference on Neural Information Processing*. Springer, 242–249.

[4] MD Ridwan Al Iqbal, Saiedur Rahman, Syed Irfan Nabil, and Ijaz Ul Amin Chowdhury. 2012. Knowledge based decision tree construction with feature importance domain knowledge. In *2012 7th international conference on electrical and computer engineering*. IEEE, 659–662.

[5] Vito Albino, Umberto Berardi, and Rosa Maria Dangelico. 2015. Smart cities: Definitions, dimensions, performance, and initiatives. *Journal of urban technology* 22, 1 (2015), 3–21.

[6] Pierre Baldi. 2012. Autoencoders, unsupervised learning, and deep architectures. In *Proceedings of ICML workshop on unsupervised and transfer learning*. 37–49.

[7] Rosamaria Elisa Barone, Tullio Giuffrè, Sabato Marco Siniscalchi, Maria Antonietta Morgano, and Giovanni Tesoriere. 2013. Architecture for parking management in smart cities. *IET Intelligent Transport Systems* 8, 5 (2013), 445–452.

[8] Heinrich Braun. 1990. On solving travelling salesman problems by genetic algorithms. In *International Conference on Parallel Problem Solving from Nature*. Springer, 129–133.

[9] Felix Caicedo, Carola Blazquez, and Pablo Miranda. 2012. Prediction of parking space availability in real time. *Expert Systems with Applications* 39, 8 (2012), 7281–7290.

[10] Vasek Chvatal. 1979. A greedy heuristic for the set-covering problem. *Mathematics of operations research* 4, 3 (1979), 233–235.

[11] Thomas H Cormen, Charles E Leiserson, Ronald L Rivest, and Clifford Stein. 2009. *Introduction to algorithms*. MIT press.

[12] Kevin Cullinane and John Polak. 1992. Illegal parking and the enforcement of parking regulations: causes, effects and interactions. *Transport Reviews* 12, 1 (1992), 49–75.

[13] Lawrence Davis. 1991. Handbook of genetic algorithms. (1991).

[14] Thanh Dinh and Younghan Kim. 2016. A novel location-centric IoT-cloud based on-street car parking violation management system in smart cities. *Sensors* 16, 6 (2016), 810.

[15] Marco Dorigo and Luca Maria Gambardella. 1997. Ant colony system: a cooperative learning approach to the traveling salesman problem. *IEEE Transactions on evolutionary computation* 1, 1 (1997), 53–66.

[16] Martin Ester, Hans-Peter Kriegel, Jörg Sander, Xiaowei Xu, et al. 1996. A density-based algorithm for discovering clusters in large spatial databases with noise.. In *Kdd*, Vol. 96. 226–231.

[17] Walter R Gilks, Sylvia Richardson, and David Spiegelhalter. 1995. *Markov chain Monte Carlo in practice*. Chapman and Hall/CRC.

[18] Carlos Groba, Antonio Sartal, and Xosé H. Vázquez. 2018. Integrating forecasting in metaheuristic methods to solve dynamic routing problems: Evidence from the logistic processes of tuna vessels. *Engineering Applications of Artificial Intelligence* 76, July (nov 2018), 55–66. https://doi.org/10.1016/j.engappai.2018.08.015

[19] John A Hartigan and Manchek A Wong. 1979. Algorithm AS 136: A k-means clustering algorithm. *Journal of the Royal Statistical Society. Series C (Applied Statistics)* 28, 1 (1979), 100–108.

[20] Anasthasia Agnes Haryanto, Md Saiful Islam, David Taniar, and Muhammad Aamir Cheema. 2018. IG-Tree: an efficient spatial keyword index for planning best path queries on road networks. *World Wide Web* (2018), 1–41.

[21] Geoffrey E Hinton and Ruslan R Salakhutdinov. 2006. Reducing the dimensionality of data with neural networks. *science* 313, 5786 (2006), 504–507.

[22] Sepp Hochreiter and Jürgen Schmidhuber. 1997. Long short-term memory. *Neural computation* 9, 8 (1997), 1735–1780.

[23] Zhanlin Ji, Ivan Ganchev, Máirtín O'Droma, Li Zhao, and Xueji Zhang. 2014. A cloud-based car parking middleware for IoT-based smart cities: Design and implementation. *Sensors* 14, 12 (2014), 22372–22393.

[24] Scott Kirkpatrick, C Daniel Gelatt, and Mario P Vecchi. 1983. Optimization by simulated annealing. *science* 220, 4598 (1983), 671–680.

[25] H Miihlenbein and J Kindermann. 1989. The dynamics of evolution and learning - Toward genetic neural networks. *Connectionism in perspective* (1989), 173–198.

[26] Kyle K Qin, Wei Shao, Yongli Ren, Jeffrey Chan, and Flora D Salim. [n.d.]. Solving multiple travelling officers problem with population-based optimization algorithms. *Neural Computing and Applications* ([n. d.]), 1–27.

[27] Mohammad Saiedur Rahaman, Margaret Hamilton, and Flora D Salim. 2017. Predicting Imbalanced Taxi and Passenger Queue Contexts in Airport.. In *PACIS*. 172.

[28] Mohammad Saiedur Rahaman, Margaret Hamilton, and Flora D Salim. 2017. Queue context prediction using taxi driver knowledge. In *Proceedings of the Knowledge Capture Conference*. ACM, 35.

[29] Mohammad Saiedur Rahaman, Margaret Hamilton, and Flora D Salim. 2018. Coact: A framework for context-aware trip planning using active transport. In *2018 IEEE International Conference on Pervasive Computing and Communications Workshops (PerCom Workshops)*. IEEE, 645–650.

[30] Mohammad Saiedur Rahaman, Yi Mei, Margaret Hamilton, and Flora D Salim. 2017. CAPRA: A contour-based accessible path routing algorithm. *Information Sciences* 385 (2017), 157–173.

[31] Mohammad Saiedur Rahaman, Yongli Ren, Margaret Hamilton, and Flora D Salim. 2018. Wait Time Prediction for Airport Taxis Using Weighted Nearest Neighbor Regression. *IEEE Access* 6 (2018), 74660–74672.

[32] Wei Shao, Flora D. Salim, Tao Gu, Ngoc Thanh Dinh, and Jeffrey Chan. 2018. Traveling Officer Problem: Managing Car Parking Violations Efficiently Using Sensor Data. *IEEE Internet of Things Journal* 5, 2 (2018), 802–810. https://doi.org/10.1109/JIOT.2017.2759218

[33] Wei Shao, Flora D. Salim, Andy Song, and Athman Bouguettaya. 2016. Clustering Big Spatiotemporal-Interval Data. *IEEE Transactions on Big Data* 2, 3 (2016), 190–203. https://doi.org/10.1109/TBDATA.2016.2599923

[34] Wei Shao, Yu Zhang, Bin Guo, Kai Qin, Jeffrey Chan, and Flora D. Salim. 2019. Parking Availability Prediction with Long Short Term Memory Model. In *Green, Pervasive, and Cloud Computing*, Shijian Li (Ed.). Springer International Publishing, 124–137. https://doi.org/10.1007/978-3-030-15093-8_9

[35] Subarna Chowdhury Soma, Tanzima Hashem, Muhammad Aamir Cheema, and Samiha Samrose. 2017. Trip planning queries with location privacy in spatial databases. *World Wide Web* 20, 2 (2017), 205–236.

[36] Chi-Hwa Song, Kyunghee Lee, and Won Don Lee. 2003. Extended simulated annealing for augmented TSP and multi-salesmen TSP. In *Proceedings of the International Joint Conference on Neural Networks, 2003.*, Vol. 3. IEEE, 2340–2343.

[37] Christina Spiliopoulou and Constantinos Antoniou. 2012. Analysis of illegal parking behavior in Greece. *Procedia-Social and Behavioral Sciences* 48 (2012), 1622–1631.

[38] Nitish Srivastava, Elman Mansimov, and Ruslan Salakhudinov. 2015. Unsupervised learning of video representations using lstms. In *International conference on machine learning*. 843–852.

[39] Pieter Vansteenwegen, Wouter Souffriau, and Dirk Van Oudheusden. 2011. The orienteering problem: A survey. *European Journal of Operational Research* 209, 1 (2011), 1–10. https://doi.org/10.1016/j.ejor.2010.03.045

[40] Andrea Zanella, Nicola Bui, Angelo Castellani, Lorenzo Vangelista, and Michele Zorzi. 2014. Internet of things for smart cities. *IEEE Internet of Things journal* 1, 1 (2014), 22–32.