IEEE Access
Multidisciplinary | Rapid Review | Open Access Journal

Received June 7, 2019, accepted June 28, 2019, date of publication July 3, 2019, date of current version July 24, 2019.

Digital Object Identifier 10.1109/ACCESS.2019.2926675

A Stochastic Approach Towards Travel Route Optimization and Recommendation Based on Users Constraints Using Markov Chain

SHABIR AHMAD, ISRAR ULLAH, FAISAL MEHMOOD, MUHAMMAD FAYAZ, AND DOHYEUN KIM

Computer Engineering Department, Jeju National University, Jeju 63243, South Korea

Corresponding author: DoHyeun Kim (kimdh@jejunu.ac.kr)

This work was supported in part by the Energy Cloud Research and Development Program, through the National Research Foundation of Korea (NRF), funded by the Ministry of Science, ICT, under Grant 2019M3F2A1073387, and in part by the Ministry of Science and ICT (MSIT), South Korea, through the Information Technology Research Center (ITRC) Support Program, supervised by the Institute for Information and Communications Technology Planning and Evaluation (IITP), under Grant IITP-2019-2014-1-00743.

ABSTRACT Accurate analysis of tourist movement is essential for a country to devise sustainable policies for promoting and growing tourism. From the activities of tourists and the spots they visit, the amount of revenue generated for a particular region can be predicted. However, the tourist preferences evolve and vary from one user to another, and thus, a tourist spot favorite for one set of users is not preferred by another set of users. This paper aims to design and implement a novel application to recommend an optimal travel route based on user constraints. The user constraints can be the maximum time, distance, and popularity of a particular place. The real data are collected from the Wi-Fi routers installed at different tourist spots of Jeju Island, South Korea. We apply a Markov chain model to predict the popularity of different places on the short- and long-term bases. The popularity index alongside user constraints is provided to find optimal routes. A responsive web-based prototype is developed to collect user constraints, and, in response, recommends optimal routes using the Google Maps directory services. The results indicate the difference between the short- and long-term popularities to prove the effectiveness of the Markov chains in forecasting long-term behavior. The system is made responsive for all sizes of screens to make it uniformly serviceable on mobile phones. The accuracy of the system is computed based on the historical data and the recommendation system, and it is ascertained to fall between 95% and 100% all the time. Furthermore, the results are compared with popular state-of-the-art methods, and they are found to be significantly better than that in the long-term location prediction.

INDEX TERMS Markov chain, route optimization, route prediction, responsive systems, stochastic processes.

I. INTRODUCTION

Evolution of the global tourism industry has contributed a vital role in the economic advancement of a nation. In a survey in 2016, it was found that the World Travel and Tourism Council rated a 3.1% growth of the tourism, which was higher than the estimated global GDP growth (2.3%) [1]. The global tourism industry leads to significant growth regarding employment by affording 107.83 million jobs or 3.6% of total jobs in 2015, and it is forecasted that this will grow up to 135.88 million jobs by 2026. The capital investment in 2026 is expected to be USD 1254.2 billion, with global tourist visitors expected to be 1.93 billion, generating expenditure of USD 2056.0 billion. Thus, international travelers' investment shares much in the worldwide tourism industry.

Accurate prognostication of international visitor figures is becoming essential for authorities to be able to devise proper sustainable tourism and marketing policies. Governments should thoroughly examine the varying number of foreign sightseers. The variety of global tourism and their movement patterns have raised an uphill job of predicting popular trajectories and routes [2], [3]. The diversity of mobile phone

The associate editor coordinating the review of this manuscript and approving it for publication was Hao Shen.

has offered to ascend to a whole new range of location-based applications. Consequently, these applications are gaining popularity nowadays. For instance, with Google maps, one can comfortably experience the benefit of location-aware services such as requesting routes, famous places, getting information on eateries, and much more. In this paper, we design and implement a recommendation system to predict the popularity of a traveling route by considering other tourists historical trajectories (traces) that are generated by routers installed on various locations. Such data of paths predict the behavior of people's travel movements and pattern between different tourist spots. One primary goal of this work is to discover the most popular places using the real data collected for Jeju Island, South Korea. It is a significantly distinct work given the novelty of data from existing route planning methods that consider a rather old and in most cases, simulated data. The second goal of this work is the most popular route recommendation which covers the top most popular spots and at the same time considers user constraints and preferences. The optimal route is necessarily a statistical result derived from the actual traveling routes conducted by other people in the past. The significance of this work is useful mainly for visitors who are visiting Jeju Island for the first time and are entirely unfamiliar to the different traveling spots. For example, a tourist who is currently on Manjanggul Cave would want to travel nearest popular places within 5 hours. The system will find all the popular routes covering the top-most locations considering the route time within 5 hours. This approach is dynamic and varies according to user constraints. The route selection is entirely dependent on the number of places a user would want to visit and the total time and distance based on the user preferences. If a traveler's priority is time, the system will recommend the shortest route falling within the time constraint, but if the preference is the popularity of places, then the system will provide a route covering more popular sites instead of the shortest path. That being said, the shortest path is not necessarily the most favored route. Additionally, for different route planning constraints, different patterns should be considered, e.g., for tour planning, it is better to adopt the trajectories of previous tourists rather than local people's driving trajectories.

Given the current location and the travelers' preferences, one can check all existing routes covering a specified number of sites and the possible trajectories. Then the route with the top optimal index is deemed to be the most popular one. For instance, if a user wants to visit five popular locations with maximum 120 km distance as a constraint, the routes which fall in the range of 120 km, and at the same time more accessible will be recommended. All those routes which are not meeting any user constraints will be discarded. We find the popularity index of the route sets using the popularity of the locations. A route can be a combination of different trajectory segments, and the popularity of each segment can be determined by looking into historical data and finding the pattern of users. It is worth considering that the word ''popular'' is prejudiced. Different people might have varying ideas of explaining the popularity, and we intend to propose a reasonable one and address the problem of how to discover the optimal route combining trajectory segments. The result is based on a movement pattern in the dataset that we use in this work. We use the Markov Chain model to find the steady-state long-term behavior of the different spots and predict the popularity based on the state probabilities of each location.

In literature, many route recommendation systems endure. The more common of them are using driving patterns and behaviors [3]–[8] which are regarded as less related to the proposed system in a sense that it analyses users' traveling behaviors, but, in contrast, they largely concentrate on extracting the sequential patterns of objects' trajectories. These patterns can aid in recommending a drive turn at some junction in a general case, but they are not adequate and reasonable to find a popular route based on the number of visits to a certain location that falls in the route.

To address the above-mentioned limitations, the contribution of this paper primarily is to give a recommendation tool which would be equally effective to both government policymakers and sightseers. It should be noted that merely counting the number of mobile tourists is not enough to discover the popularity of a certain spot. The fundamental goal of this work is to utilize this information based on the dataset and find the pattern of movements in the long-term as well as short-term. The long-term and short-term popularity of these locations can be used by policymakers and law enforcement agencies to devise certain policies and adjust the budget accordingly. The mechanism of forecasting and popular route prediction is common and different machine learning techniques are used such as deep neural networks, decision tree and support vector machines but the problem with these techniques is that they predict the behavior irrespective of the amount of time in future the forecasting will hold accurate. Therefore, considering the goal of this work, they are not a suitable approach. We use a stochastic approach known as Markov Chain [9] which has inherent support of predicting long-term and short-term behaviors inside a dataset, making it an ideal candidate considering the scope of this work. In Markov Chain, we construct a transition matrix from raw data, which shows short-term transition probabilities from one location to another. Then with an initial state vector and transition matrix, the probabilities are converged to predict the long-term popularity index of different locations, and an intermediate result to capture the moving behaviors between sites and to facilitate the search of the popular route. Subsequently, the popularity of a route is defined as the summation of individual popularities of the locations covering the route. Thus we focus on recommending different trajectories based on cumulative popularity index and user preferences to create a general view of traffic which can be utilized in a broad spectrum of applications.

To achieve the goal, we propose to handle a few challenges which are stated below. First, we need to form a transition matrix from a dataset to demonstrate the popularity

of location in the short-term. We define those locations which have got highest probabilities as popular spots. It is a challenge because the data is in raw form and a lot of preprocessing is needed to render it in a form which lay a basis for transition matrix. For this, unique-location count algorithms are proposed to find unique locations from different routes in the data. Second, the values of popularity for locations and routes need to be converged to a steady-state. So, we can no longer use the unique count number location as a measurement. For this, we consider the Markov Chain model to find the steady-state convergence probabilities of each location. By doing so, we provide a reasonable way to measure the long-term popularity of a route and finally, combining these short-term and long-term popularities with user constraints and preferences to form the optimal route. The optimal route is maximizing the popularity index of location and minimizing the distance and time of the route. The algorithm proposed for this has the same end goal as the Dijkstra's algorithm [10], and the resulting route is a path consists of a series of locations that maximizes the product of transition probabilities. In a nutshell, the basis of the route planning approach in this work is to ''learn'' from history and recommend a route by digging the most popular path from a mobile travelers dataset and at the same time considering visitors constraints and preferences. Thus, the paper principally performs the following additions to the state-of-the-art solutions:

- It proposes and implements an algorithm to form the transition matrix using Markov Chain after some preprocessing operations of the dataset which has a collection of historical trajectories of mobile visitors inside Jeju Island, South Korea
- It proposes a popularity index formula based on steady-state convergence principle of the Markov Chain model and presents an algorithm to predict the long-term popularity of different locations of Jeju Island.
- It also proposes an optimization objective for a route, which is a function of the popularity index of sites in the route, distance, and time of the route. Thus, the formula aims at maximizing the popularity index and minimizing the distance and time.
- Finally, to prove the effectiveness of the algorithms, the paper develops an application which takes user constraints such as maximum time and distance and recommends an optimal route based on the objective functions.

The rest of this paper is organized as follows. In Section 2, the related work is described in detail, and different strategies and research relevant to route predictions and optimization are highlighted. The system model and proposed formulae are introduced in Section 3. Section 4 presents the conceptual architecture of the proposed work and exhibits a series of experiments performed in subsections. Moreover, the proposed algorithms are presented and explained, and the obtained results are elaborated. In Section 5, the optimization mechanism is discussed, and the user constraints and preferences are demonstrated with examples. Section 6 covers the implementation environment and outlines the tools and technologies used in incorporating this work. The recommendation system and the execution results are presented in Section 6. Section 7 provides the significance of the proposed system and evaluate its performance and accuracy with respect to other state-of-the-art methods, and finally, Section 8 concludes the paper.

II. RELATED WORK

Optimal route recommendations based on historical movements data is a highly relevant topic. Route optimization and prediction are based on user preferences and available constraints. Sometimes, a user wishes for a route with less traffic [11], [12] while another time shortest path is preferred. Nevertheless, the main goal is still pattern extraction and mining [3], [6], [11], [13], [14], grouping similar routes and trajectories [15], [16], route prediction [7], [8], [17] and hot path mining [18], [19] including pattern mining [3], [6], [13], [14], trajectory clustering [15], [16], hot route discovery [18], [19], trajectory prediction [7], [8], [17] etc. Nevertheless, none of the afore-mentioned research approaches the challenge of discovering the most popular routes from one given location to another based on defined user constraints and preferences, and in many cases, uses simulated data. The work in this paper is based on real data and is mainly related to constraints-aware route planning, while the large bulk of previous work is dealing with a general prediction and mining challenges. The work in [16], [18] concerning the mechanism to discover hot routes are considered more closer to this work in recognizing paths which are often sighted by users. Sacharidis *et al.* [18] propose a density-based algorithm FlowScan for extracting popular routes as per the rule ''traffic density-reachable''. It is rather a route grouping algorithm which is based on the density of traffic. This idea is also portrayed in [15], [16], but they group trajectories by individual line segments. Sacharidis et al. also developed an on-line algorithm in [18] to search and maintain the list of paths which often senses motions by more than a specific number of mobile objects. These works are suited more for mining routes that are often toured from the whole world whereas our work is specific to Jeju Island to search the regularly visited routes or path segments for a query with a start location, maximum allowed time, maximum distance and number of places to visit.

The patterns mining techniques for trajectory prediction in [3], [6], [13], [14] could conceivably assist in predicting a popular path. In [6], Giannotti et al. investigated the enigma of distinct mining patterns known as T-pattern. T-pattern is a series of temporally annotated points, and the aim is to predict all T-patterns using Support Vector Machine. A T-pattern could be perceived as a central pattern which symbolizes movement through a series of points. Thus, if the starting location and destination are just right on the series, it is suggested a recommended route and vice versa. Nevertheless, this work is based on start and end locations and doesn't

consider the waypoint in between, so this idea does not go well with the paper context, where the number of waypoints and constraints need to be considered. Apart from the above literature, a more recent idea is of the region of interest (ROI) which is employed for estimating a route as a series of tokens, but this is not adequately correct for pointing precise navigation for path devising goal.

Likewise, in [3] and [14], the existing idea of sequential pattern mining algorithms is utilized for investigating regular trajectory segments that are visited more often. In [13], the movement in periodic fashion is mined, and the areas are examined too. In the pre-processing phase of our solution, an algorithm is produced for retrieving frequently visited location by using the movement path and path segments, and subsequently, the transition matrix. The grounds of this algorithm is comparable to the density-based clustering algorithm DBSCAN in [20], which identifies a density-based SCAN. In [21], Cao et al. also propose an approach to retrieve a road network from trajectories. However, their method is designed primarily for identifying edges while road intersections are not elegantly clarified. The work in [22] are intended mainly for discovering road intersections, but they require an underlying roadmap available in advance for training a classifier, while in our algorithm, the trajectories may be un-constraint and the roadmap availability is not assumed. Other similar work includes planning routes by analyzing traffic contingency [23], seeking alike routes [24]–[28], shortest path [10], [29], shortest path on time-dependent networks [30], predicting the fastest route by mining speed patterns [31], etc. Nevertheless, all the work above is not able to address the problem of obtaining and acquiring the popularity of a route. Similarly, the use of a Markov Chain is commonly used in predicting the stochastic behavior of the data. In [32]–[34], Markov Chain is used for arterial route travel time distribution and road congestion prediction. A variation of Markov Model has also been in use for route popularity [35], [36]. Grey prediction models [37] have also attracted considerable regard since these models can characterize an unexplored system from limited data [38]–[40], without expecting conformance to statistical hypotheses, such as normal distributions. The extensively applied grey model with a first order differential equation and a single variable, GM(1,1), for example, can be set up using only four recent sample data points [40], [41]. The above instances in the literature indicate the significance of Markov chains in case of stochastic behavior in tourism data; however, the use of Markov Chain is not limited to this context but also is widely used in network control system [42], [43] in which the stochastic behavior is non-linear. Markov jump [44], [45] is used to reliably profile event-triggered control issue in such systems. Similarly, Deep Neural Networks [46] and Recurrent Neural Networks (RNN) [47] are also widely adopted in location prediction considering the data is time series. However, for data having a long sequence of routes, the performance and accuracy of these algorithms suffer significantly. In such cases, the Markov Chain and Hidden Markov Chain are the preferred choices. In this work, we use a simplified Markov Chain model, and much of the preprocessing is handled with powerful data processing tools to efficiently predict and recommend an optimal route based on user preferences and constraints, which is, to the best of authors knowledge, the first of its kind for Jeju Island, South Korea region.

III. SYSTEM MODEL

The main goal of this work is to analyze the stochastic behavior of the real data collected from routers and to predict the famous spots and routes from the data. The block representation of the proposed system is depicted in figure 1. The popularity indexes of all the locations in short-term and in long-term are provided as inputs alongside user-defined constraints. The system processes these functions according to optimization objective function, which is described in subsequent sections. The system returns the optimal route based on user constraints and the popularity of the locations.

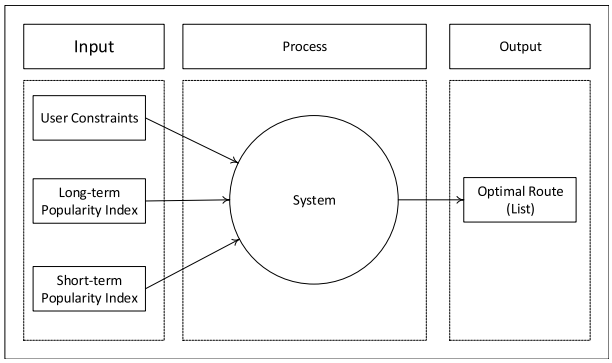


FIGURE 1. System block representation.

The flow diagram of the system is exhibited in figure 2. The first step is to analyze the input flow data to find stochastic characteristics of different locations using Markov Chain. Using the data, we see the transition probabilities from one place to another. The transition matrix is then provided to the Markov Chain model to find steady state probabilities. Similarly, from the transition matrix, short-term probabilities are computed which are taken as the probabilities of specific locations from one location to another. In addition to the probability transition matrix, a distance transition matrix is also computed based on the latitude and longitude of all the tourist sites. These three parameters are provided alongside user constraints, which are time, current location, and maximum distance. Once the optimization objective function receives all the input parameters, it computes the optimal route, which meets user constraints and at the same time popular both in the short and long-term.

In the following subsections, we describe a systematic approach for finding the input parameters for the optimization function. The parameters are the popularity index of locations

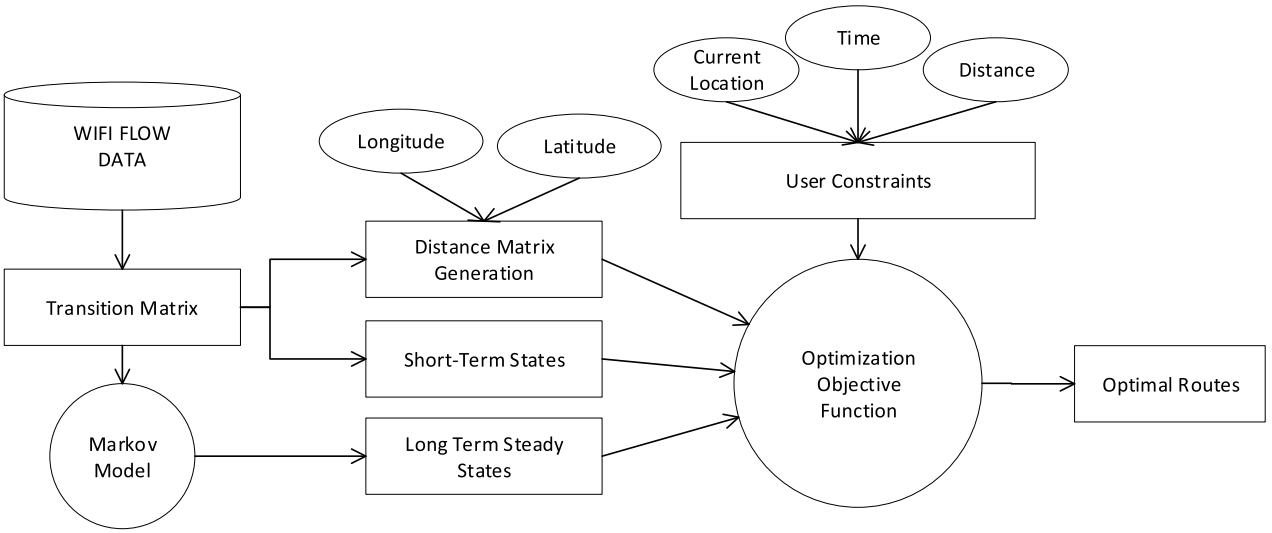


FIGURE 2. Detailed system flow diagram.

in short-term, the popularity index of locations in long-term, distance matrix, and user preference weights.

A. MARKOV CHAIN BASED POPULARITY FORECASTING

The proposed approach is based on the statistical concept of the Markov Chain and aims to forecast the popularity measure of all places of Jeju Island in long-term while characterizing the stochastic behavior of the real data we collected.

Markov process is widely used in modeling the dynamics of stochastic systems and the state transitions of complex stochastic systems. A Markov process $X(t), t \in T$ is a stochastic process with the property that, given the system state at time t, $X(t)$, for a time $s > t$, the system state $X(s)$ is not influenced by the system states, $X(u)$ for $u < t$ that is, prior to the time t. P is a transfer matrix; the matrix elements are not negative, and the sum of all the various elements is equal to 1, expressed in the probability of each element. The element in the matrix is the probability that the transition occurs from the previous spot to current spot in our example use case. In simple words, a stochastic system is said to be meeting the Markov property if the future states are not dependent on the steps that led up to the current state. In formal term the for locations $l = (l_o, l_1, l_2 - - - l_n)$ the markov property is:

$$\begin{aligned} PX(ln) = xn \mid X(l_1) = x_1, X(l_2) = x_2, \ldots.., X(l_{n-1}) = x_{n-1} \\ = PX(ln) = x_n \mid X(ln-1) = x_{n-1} \end{aligned} \tag{1}$$

The probability of X_{n+1} being in state j, given that Xn is in state i is called the one-step transition probability. Multi-step transition probability can be calculated according to one-step transition probability and the Markov property as shown below:

$$P(Xn+1 = xj \mid Xn) \tag{2}$$

$$\begin{aligned} &P(Xn+2 \mid xn) \\ &= \int_{+1} P(Xn+2, Xn+1 \mid Xn) dXn+1 \\ &= \int P(Xn+2 \mid Xn+1) P(Xn+1 \mid Xn) dXn+1 \end{aligned} \tag{3}$$

A transition matrix P for Markov Chain X at time instant t is an $n \times n$ matrix which contains the probabilities of transition from one state to another. In particular, given an ordering of a matrix's rows and columns by the state space S, the $(i, j)^{th}$ element if the matrix P is given by

$$P_{i,j} = P(X_{t+1} = j \mid X_t = i) \tag{4}$$

For instance, if at time t a tourist is on location l_1 the probability that the user moves to location l_2 will be on P_{1_2} location of the transition matrix P.

Transition matrix shows the transition probabilities from one state to another in the short-term. These probabilities are converged into a steady state matrix by multiplying with an initial state vector repeatedly. Let I is the initial state vector having any arbitrary distribution.

$$I = [P(l_1)\, P(l_2)\, P(l_3) \ldots P(l_n)$$

The I after nth multiplication is given by:

$$u^{(n)} = uP^{(n)} \tag{5}$$

If the transition matrix is irreducible and periodic, the n-step transition matrix converges to a stationary distribution with each column different than the short-term transition

matrix; that is,

$$\lim_{k\to\infty} p^k = \pi \tag{6}$$

B. DISTANCE CALCULATION

In order to compute the distance transition matrix, for all the locations in dataset, the distance is calculated using the Haversine formula. The Haversine formula calculates the distance between two locations on earth, considering it as a sphere if their latitudes and longitudes are known as used in recent studies [48], [49]. It is instrumental in navigation and is sometimes regarded as a special case of spherical trigonometry. For any two locations $[l_1, l_2]$ on earth, the Haversine of the central angle between them is given by

$$hav(\frac{2*d}{D}) = hav(\phi_2 - \phi_1) + cos(\phi_1)cos(\phi_2)hav(\gamma_2 - \gamma_1) \tag{7}$$

where ϕ represents the latitude of the location, γ is the longitude of the location, d is the distance and D is the diameter of the earth. hav is the haversine function which is given by:

$$hav(\theta) = sin^2(\frac{\theta}{2}) = \frac{1 - cos(\theta)}{2} \tag{8}$$

combining equation (7) and (8), the distance d is found as:

$$d = \frac{D}{2} hav^{-1}(hav(\phi_2 - \phi_1) + cos(\phi_1)cos(\phi_2)hav(\gamma_2 - \gamma_1)) \tag{9}$$

or

$$d = Dsin^{-1}(\sqrt{hav(\phi_2 - \phi_1) + cos(\phi_1)cos(\phi_2)hav(\gamma_2 - \gamma_1)}) \tag{10}$$

putting the values of Haversine function from equation (8) gives rise to

$$d = Dsin^{-1}(\sqrt{sin^2(\frac{\phi_2 - \phi_1}{2}) + cos(\phi_1)cos(\phi_2)sin^2(\frac{\gamma_2 - \gamma_1}{2})}) \tag{11}$$

C. OPTIMIZATION OBJECTIVE FUNCTION

The objective of the optimization function is to find a route which has less distance and time and covers more popular sites at the same time. Therefore, the optimization function tends to minimize distance and time and maximize popularity index. Based on these assumption the optimal index is formulated. The first parameter to consider is the popularity index (P_i) and the objective is to maximize it as shown in equation: (12)

$$P_i = max(p_s, p_l) \tag{12}$$

where p_s is the short-term probability and p_l is the long-term probability. The time and distance constraints need to be minimized by the objective function. i.e,

$$C_i = min(d, t) \tag{13}$$

Based on equation (12) and (13) the optimization index formula is given by

$$\psi = \frac{\alpha_1 p_s + \alpha_2 p_l}{\beta_1 d + \beta_2 t} \tag{14}$$

where ψ is the optimization function and the route having a maximum of ψ is the most recommended route and α, and β are user preferences.

IV. CONCEPTUAL ARCHITECTURE

The goal of this paper is to perform a series of experiments to refine and preprocess the dataset and extract some useful patterns. Once the data has been preprocessed, the next step is to apply the Markov Chain model for learning the stochastic behavior of the data regarding the long-term popularity index of input locations. The proposed methodology is summarized in figure 3.

There are four layers; input, process, learning, and optimization. The dataset is taken as raw input and sent to the process layer. The process layer performs a series of experiments to refine the raw data. Experiments include tagging of data, cleaning of data to remove redundant records, feature extraction for extracting columns of interest, and finally distance matrix for all the locations. The Markov Chain model is applied in the learning phase to predict the stochastic behavior of the data. In this paper, we are interested in the popularity index of all the locations of Jeju Island. Once we get the popularity index and distance data, we apply optimization based on user preferences to find an optimal route which covers the most popular locations within specified user constraints. In the following subsections, a description of all the experiments performed and algorithms developed are presented. Table 1 summarizes the notations used in all algorithms.

A. EXPERIMENTAL DATA

As mentioned earlier, real data based on the mobile tourist is gathered for the year 2017-2018 from different locations of Jeju Island, South Korea. The data has attributes like date of the connection. The date is split across month, half and quarter for granularity purposes. Apart from this, the moving path is recorded, showing the tourist movements across different locations. Additionally, the number of tourists are counted for the route, and the duplicate count is also noted down, which represents the number of tourists who travel part of the route. The difference between the total path count and duplicate path count is that in case of total path count, the mobile tourist covered the whole trajectory while in case of duplicate the tourist covered part of the trajectory. A chunk of the dataset is shown in figure 4.

B. DATA PREPROCESSING

In this part, several experiments have been performed to present the data in a form which can comfortably be consumed by the Markov Chain model. In this stage, locations

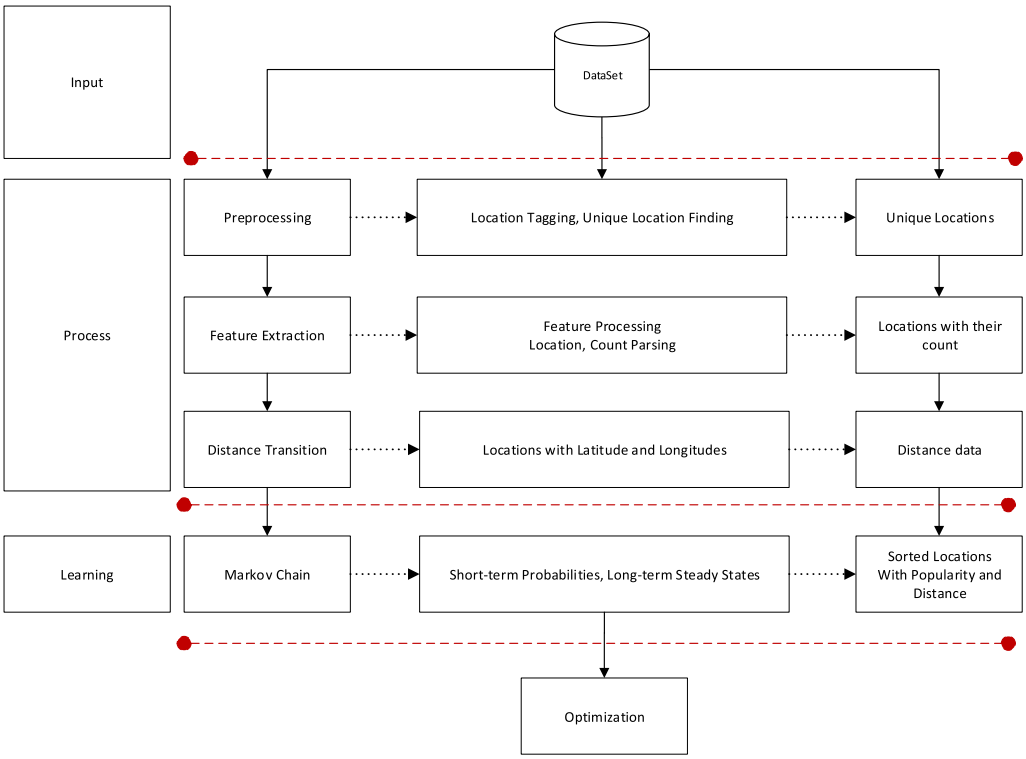


FIGURE 3. Proposed methodology.

TABLE 1. Summary of symbols and notation used in different algorithms.

Symbol	Description	Algorithm	Symbol	Description	Algorithm
Δ	dataset	1,2,4	Γ	Time transition Matrix	4
ν	location	1,2	α	Prefences vector	4
mp	move_path	1	π	external output from webform	4
r	row in loop	1,2,3,4	β	Constraint vector	4
iter	loop counter	1	loc_i	current location	4
loc	location	1,4	$P \Leftrightarrow loc_i$	Probability transition of from current location to destination	4
tot	total count	1,4	$loc[c]$	candidate locations	4
tpc	total path count	1	ζ	optimization index	4
i	index	2,3,4	I_{min}	Minimum Index	4
c	column	2,3,4	I_{max}	Maximum Index	4
Ξ	Transition Matrix	2,3,4	ω	distance transition matrix	3,4
Ω	Pandas Dataframe	2,3,4	R	Radius of earth	3
Σ	Sum function	2	ϕ_1	Source Latitude	3
ϕ	Latitude	3,4	δ	difference function	3
γ	Longitude	3,4	d	distance	3

have been tagged for efficient processing. A location is tagged using the index of the location and the first three letters of the location. For instance, Seopjikoji is tagged as SEO-002. Once the locations are labeled, the tagged places are replaced in the preprocessed dataset. Secondly, the trajectories are in an arrow separated string, so algorithm 1 is used to parse the list and make a panda data frame to show the location and the total tourists passed through the location. The symbols and acronyms used in the algorithm have already defined in table 1.

The result of the algorithm 1 returns the total unique locations and their corresponding count. Table 2 shows chunk of the result of algorithm 1. The computational complexity of the algorithm is computed, taking into account two distinct operations. Firstly, the iterative operations shown in lines 3 to 6 are the most expensive in terms of time and space. It has two loops, the outer of which is traversing the whole dataset whereas the inner one is traversing the unique locations in the route. If the number of rows in the dataset is taken as n and the number of location in nth row is taken as k,

dt_year	dt_month	dt_half	season_code	move_path	total_path_count	total_duplicate_path_count
2016	12	2	4	Manjanggul->Bijarim	1	1
2016	12	2	4	Manjanggul->Seopjikoji	5	5
2016	12	2	4	Manjanggul->Seongsanilchul Provincial Marine Park	12	12
2016	12	2	4	Manjanggul->Jeongbang Waterfall, Xu Fu exhibition, Chilshimni food specialized street	1	1
2016	12	2	4	Manjanggul->Jusangjeolli	1	1
2016	12	2	4	Bijarim->Seopjikoji	1	1
2016	12	2	4	Bijarim->Seongsanilchul Provincial Marine Park	1	1
2016	12	2	4	Bijarim->Cheonjiyeon Waterfall, Saeyeongyo office	1	1
2016	12	2	4	Sanbangsan Mountain->Seongsanilchul Provincial Marine Park	1	1
2016	12	2	4	Sanbangsan Mountain->Jeongbang Waterfall, Xu Fu exhibition, Chilshimni food specialized street	2	2
2016	12	2	4	Sanbangsan Mountain->Jusangjeolli	5	5
2016	12	2	4	Sanbangsan Mountain->Jusangjeolli->Cheonjiyeon Waterfall, Saeyeongyo office->Jeongbang Waterfall, Xu Fu exhibiti	1	1
2016	12	2	4	Seopjikoji ->Manjanggul	1	1
2016	12	2	4	Seopjikoji ->Seongsanilchul Provincial Marine Park	17	17
2016	12	2	4	Seopjikoji ->Seongsanilchul Provincial Marine Park->Manjanggul	1	2
2016	12	2	4	Seopjikoji ->The mystery of the road	1	1
2016	12	2	4	Seopjikoji ->Cheonjiyeon Waterfall, Saeyeongyo office	1	1
2016	12	2	4	Seongsanilchul Provincial Marine Park->Manjanggul	3	3
2016	12	2	4	Seongsanilchul Provincial Marine Park->Seopjikoji	30	30
2016	12	2	4	Seongsanilchul Provincial Marine Park->Seopjikoji ->The mystery of the road	1	7
2016	12	2	4	Seongsanilchul Provincial Marine Park->Seopjikoji ->Jejumok government office	1	1
2016	12	2	4	Seongsanilchul Provincial Marine Park->The mystery of the road	1	1
2016	12	2	4	Seongsanilchul Provincial Marine Park->Jeju Natural World Heritage Center	1	1
2016	12	2	4	Seongsanilchul Provincial Marine Park->Jusangjeolli	6	6
2016	12	2	4	Seongsanilchul Provincial Marine Park->Halla Arboretum	2	2
2016	12	2	4	The mystery of the road->Seongsanilchul Provincial Marine Park	2	2
2016	12	2	4	The mystery of the road->Historic Site of Anti-Yuan Movement->Sanbangsan Mountain->Jusangjeolli->Seongsanilchul	1	1

FIGURE 4. Experimental dataset.

Algorithm 1 Unique Location With Count of Tourists Passed Through It

1: $\Delta \leftarrow Read\,\Delta$
2: $\nu \leftarrow EmptyList$
3: **for** r, $iter$ *in* Δ **do**
4: $\quad tpl \leftarrow \Delta[mp].split(\rightarrow)$
5: $\quad$ **for** r *in* tpl **do**
6: $\quad\quad \nu \leftarrow Append\ \nu(r, \Delta[tpc])$
7: **for** r, $iter$ *in* ν **do**
8: $\quad \nu[loc] \leftarrow \nu[loc] + \nu[tot]$

TABLE 2. A small chunk of preprocessed dataset showing tagging of locations and their respective count.

Index	Tag	Location	Total Count
1	Mang-001	Mangangul	2120
2	Seop-002	Seopjikoji	1110
3	Song-003	Songaksan	1222
4	Samb-004	Sambangsan	3770
5	Bija-005	Bijarim	2901
6	Hall-006	Halla Arboretum	2160

then the computational complexity is $O(nk)$. Secondly, lines 7,8 traverse the unique location extracted, and thus, it also contributes marginally. If the total location is represented as l, then the overall time complexity of algorithm 1 is $O(nk+l)$. The space complexity of algorithm 1 is $O(l)$ where l is the number of unique location extracted as a result of algorithm 1.

The total visits per location which are parsed from the real data, serve a useful purpose for finding the pattern for the popularity of the location. Figure 5 shows a circular line graph which shows location tags on the circumference of the circle and their respective count inside the ring. The inner sub-circles display different ranges of counts. For instance, the innermost circle represents those locations whose count is more than 25000 and the outer most circle denote that the count is in the range 0–5000.

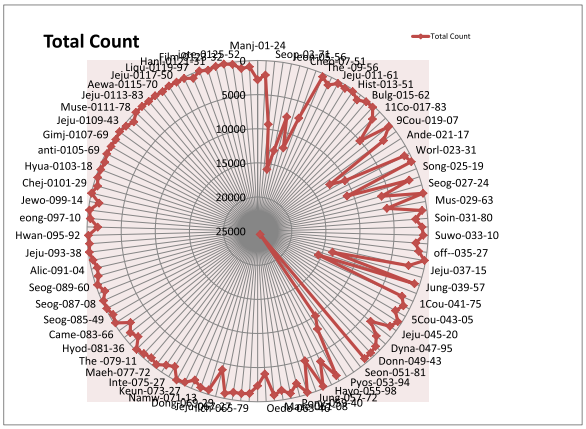


FIGURE 5. Total visit per location.

C. TRANSITION MATRIX BASED SHORT-TERM LOCATION POPULARITIES

Transition Matrix is one of the preliminary steps in applying the Markov Chain model on a dataset. In this paper, states are the existence of tourists on a certain location, and transitions are the movement of tourists from one location to another to form a trajectory segment. Thus, the rows and the columns show the locations and the values of the matrix are the transition probabilities. For this use case, the input data is the unique tagged locations with their respective counts as computed in algorithm 1. The data is a two-dimensional data frame. The indices show locations while columns represent the counts. The total count for each transition is computed to form the transition matrix. A transition is represented as $LocA \Rightarrow LocB$, so a string matching algorithm is applied to search for every row in the dataset dynamically, and if found, the respective count against that index is noted. In the end, the duplicate transition is summed up to see the unique transitions. Algorithm 2 describes the method to find the transition matrix. The symbols and acronyms used in the algorithm have already defined in table 1.

Algorithm 2 Transition Matrix Calculation From the Dataset and the Count of Every Locations

1: $< Read >\nu, \Delta$
2: ▷ Create a DataFrame with index and columns to unique locations
3: $\Xi \leftarrow \Omega[i = \nu, c = \nu]$
4: **for** i *in* Ξ **do**
5: **for** c *in* Ξ **do** ▷ Initialize all diagonal to 0 because there is no self-transitions
6: **if** $\Delta[loc][i] == \Delta[loc][c]$ **then**
7: $\Xi[i][c] \leftarrow 0$
8: **else**
9: $q \leftarrow selectiif\ \Xi[i] \rightarrow \Xi[c]inmp$
10: $\Xi \leftarrow \Delta[q]$
11: $tc \leftarrow \Delta[tc][i]$
12: $\Xi[i][c] \leftarrow tc$
13: $\Sigma[r] \leftarrow \Sigma[r = 1]$
14: $\Xi \leftarrow \frac{\Xi[r=1]}{\Sigma[r]}$

The time complexity of algorithm 2 is illustrated by considering lines 4 to 12. As there are two *for* loops each dealing with traversing the unique locations, therefore the time complexity is in the order of $O(n^2)$ where n is the number of unique locations. It appears from the complexity that the algorithm is quadratic in scale but since the n is very small considering the tiny geographical area of Jeju island, so the overall impact of is minimal. Furthermore, it also generates and populates a panda matrix of nXn order, so the space complexity is also $O(n^2)$.

The transition matrix generated as a result of algorithm 2 is summarized in table 3. The first row and the first column show the locations and rest of the rows show the transition probabilities. The matrix is a hollow matrix because the probability of going from one location to itself is always 0.

D. STEADY STATE CONVERGENCE BASED ON LONG-TERM LOCATION PROBABILITIES

As discussed in Section 3, in the Markov Chain, if an initial state vector representing any arbitrary distribution is repeatedly multiplied with the transition matrix, it converges to a steady state. The steady state is a state which can't be changed after further multiplication with the transition matrix. In this state, the system is characterizing the long-term behavior of the data. In this subsection, we outline the result of steady state convergence and depicts how it converges to a steady state. Figure 6 shows the effect of long-term probabilities for $n = 100$.

The convergence of the probabilities is validated by comparing the n = 100 and n = 200, and the difference was entirely negligible. n is made to increase from 5 to 200, and initially, the difference is visible, but after 50 repetitions, the states are converged, and no matter how many times more we multiply the initial state's vector, the result is always same. It is depicted in figure 7

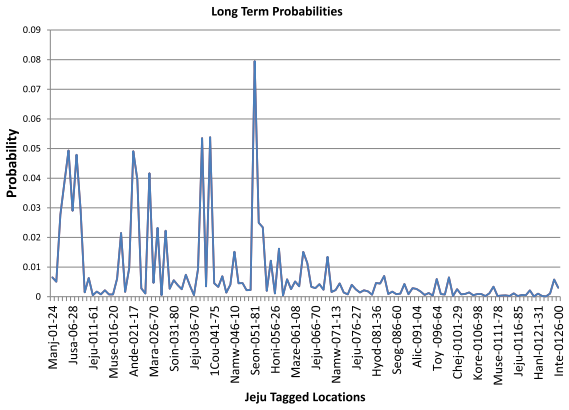


FIGURE 6. Steady state long-term probabilities.

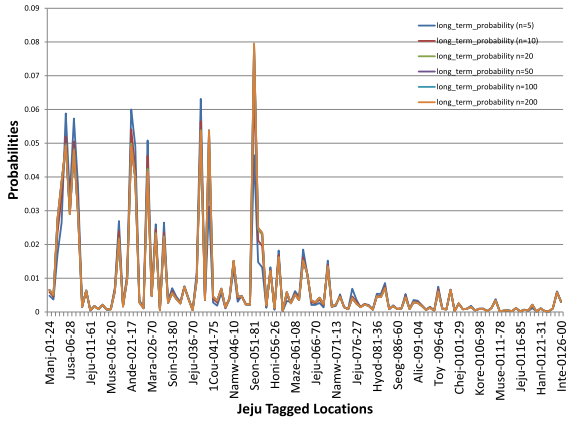


FIGURE 7. The effect of state convergence for varying number of n.

The difference between $n = 5$, $n = 10$ and $n = 200$ is considerably noticeable but the difference between 100 and 200 is not visible which means the probabilities values are almost same for $n = 100$ and $n = 200$. Thus, it can be confirmed that the states at $n = 100$ are converged and shows the long-term popularities of locations.

V. PROPOSED OPTIMAL ROUTE SELECTION

In this paper, optimal routes are recommended based on user constraints and the popularity of data in a long-term and short-term basis. The long-term and short-term popularity of the data is computed based on the Markov Chain model. In this section of the paper, the aim is to define user constraints and to describe how it affects a typical route recommendation system if one of these constraints varies. One of the primary issues in any recommendation system is the time and distance constraints. The optimal route must satisfy user constraints for effective recommendations. In the following subsection, we describe the mechanism of approaching each user constraint.

A. DISTANCE TRANSITION MATRIX

In describing the model, we derive the formula for distance calculation using equations. In the Markov Chain model,

TABLE 3. A small chunk of transition matrix generated as a result of algorithm 2.

	Manj-01-88	Bija-02-26	Seop-03-97	Seon-04-72	Jeon-011-82	Jusa-07-17
Manj-01-88	0	0.077300613	0.047852761	0.469325153	0.016564417	0.007361963
Bija-02-26	0.098804781	0	0.071713147	0.192031873	0.013545817	0.010358566
Seop-03-97	0.013979031	0.018472292	0	0.19404227	0.011649193	0.011316359
Seon-04-72	0.063945956	0.02276513	0.222654081	0	0.015454377	0.041458449
Jeon-011-82	0.003365385	0.000600962	0.007932692	0.015745192	0	0.049639423
Jusa-07-17	0.002650897	0.000611746	0.009584013	0.015497553	0.064641109	0

stochastic modeling probabilities of state transition are represented in a matrix called transition matrix. For distance, we also represent a distance transition matrix which shows the distance of each location to another. This transition matrix helps in correlating the popularity of locations with its distance. Algorithm 3 describes the mechanism of how to compute the distance transition matrix. The symbols and acronyms used in the algorithm have already defined in table 1.

Algorithm 3 Distance Transition Matrix

1: $< Read > \nu, loc(\phi, \gamma)$ ▷ Create a DataFrame with index and columns to unique locations
2: $\omega \leftarrow \Omega[i = \nu, c = \nu]$
3: **for** *i in* ω **do**
4: **for** *c in* ω **do** ▷ Initialize all diagonal to 0 because there is no self-distance
5: **if** $\Delta[loc][i] == \Delta[loc][c]$ **then**
6: $\omega[i][c] \leftarrow 0$
7: **else**
8: $R \leftarrow 6373.0$
9: $\phi_1 \leftarrow Radian(loc[\phi][\omega[i]])$
10: $\gamma_1 \leftarrow Radian(loc[\gamma][\omega[i]])$
11: $\phi_2 \leftarrow Radian(loc[\phi][\omega[c]])$
12: $\gamma_2 \leftarrow Radian(loc[\gamma][\omega[c]])$
13: $\delta(\phi_1, \phi_2) \leftarrow \phi_2 - \phi_1$
14: $\delta(\gamma_1, \gamma_2) \leftarrow (\gamma_2 - \gamma_1)$
15: $d \leftarrow Equation\ (11)$
16: $\omega[i][c] \leftarrow d$

The time complexity of algorithm 3 is very much identical with that of algorithm 2 in a sense that bulk of the computation is performed in two *for* loops, as shown in lines 3 until 16. As there are two *for* loops, each dealing with traversing the unique locations; therefore, the time complexity is in the order of $O(n^2)$. In this case, it generates and populates a panda matrix of nXn order from the distance values, so its space complexity is also in the order $O(n^2)$. The result of applying algorithm 3 is summarized in table 4. It shows the input location and its distance in kilometers to all other location.

B. TIME TRANSITION MATRIX

In addition to distance, time is also one constraint that can be tackled and is crucial for any recommendation system. Ideally, time is computed based on distance and speed. However, the speed of the vehicle varies according to road conditions, traffic flow, the slope of the road, and many more. Also, the type of vehicle plays a role in such formulations. In this paper, we consider slope and distance for average time calculation of a specific route. The slope is a function of steepness difference of source and destination location. As it is known from elementary math that $speed = \frac{distance}{time}$ and this implies to $time = speed\ x\ distance$. The slope can be found by a popular slope-intercept formula $slope = mx + b$. If $b = 0$ then both formulae are same and the slope is responsible for the speed of a vehicle. That being said, the average slope of a place can be found by elevation of source and destination. The average slope can be given by equation (15)

$$slope = \frac{(E_d - E_s)}{Distance} \tag{15}$$

where E_d is the elevation of the destination location and E_s is the elevation of the source location. The elevation can be found by utilizing Open Elevation API [50] to which the latitude and longitude of a certain location are passed, and the elevation of the location is returned as a response. Figure 8 shows a typical API request and the response. We give the latitude and longitude of location Manjangul, which is 33.52 and 126.77, respectively. These parameters are passed as a query string to open-elevation API, and in response, we get a JSON string which shows the elevation as 85.

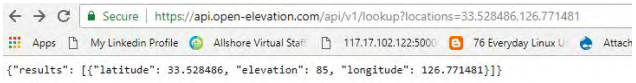


FIGURE 8. Sample API request for Manjanggul Location.

If the slope is a positive value, it means the route from source to destination is a climbing route, and otherwise, if it is a negative value.

C. USER PREFERENCES

In designing a recommendation system that is aware of user choices, it must consider them by recording them. Sometimes the user gives preference to one constraint than to others. For instance, if for a user, time is more critical, then it would mean the weight of the time is more than other parameters and so one. The user preferences are represented in weights, which is static in the start but is dynamically updated for the same user based on the history and pattern of travelers. In this paper, we design a preference vector which shows the weight of short-term popularity, long-term popularity, distance, and time.

TABLE 4. A small chunk of distance transition matrix generated as a result of algorithm 3.

	Manj-01-24	Bija-02-95	Seop-03-71	Seon-04-50	Jeon-05-56	XuF-12-13	Chil-12-22
Manj-01-24	0	5.544662694	18.72765287	17.69262135	36.59542198	36.63709799	36.792135
Bija-02-95	5.544662694	0	13.2931756	12.71709066	35.31411978	35.36586606	35.53310156
Seop-03-71	18.72765287	13.2931756	0	4.02898732	38.7159609	38.78627746	38.96708496
Seon-04-50	17.69262135	12.71709066	4.02898732	0	41.81675382	41.88424861	42.06417376
Jeon-05-56	36.59542198	35.31411978	38.7159609	41.81675382	0	0.081862698	0.256068926
XuF-12-13	36.63709799	35.36586606	38.78627746	41.88424861	0.081862698	0	0.180987595
Chil-12-22	36.792135	35.53310156	38.96708496	42.06417376	0.256068926	0.180987595	0

TABLE 5. Specification of various tools and technologies used in implementing the proposed work.

System Parameter	Value
Operating System	Windows
CPU	Intel (R) Core(TM) i5-4570 CPU @ 3.20 GHz
Primary Memory	8 GB
Integrated Development Environment (IDE)	PyCharm, Sublime Text 3, Jupyter Notebook
Framework	Flask Micro Framework
Core Programming Language	Python
Libraries	Jinja 2, CSV Parser, Bootstrap 3, HTML 5/CSS3
Data Processing	Pandas, Numpy
Persistance	CSV

VI. IMPLEMENTATION ENVIRONMENT

This section provides an overview of the tools and technologies used in implementing and evaluating the proposed recommendation system. The tools and technologies being used in this work are summarized in table 5.

The primary programming language which has been used as a core is Python. The latest version at the time of implementing this project is 3.6. Python is a common yet very powerful programming language. It is extensively exercised for producing stellar applications be it web-based, desktop-based applications or even a data-extensive application performing data analysis and simulation on massive sets of data. Python is an open-source programming language, and a vast community of developers is contributing patches and modules for any applications. The choice of Python as a core programming language has been made for numerous reasons. First, it is equally successful among researchers doing data-extensive processing and for web programmers developing cutting-edge applications. Second, it is effortless to learn and adapt. Finally, it has strong API support, and security patches are developed actively. For dataset processing, we use a very popular library of Python known as Pandas. Pandas library makes use of the Numpy library to enable efficient data processing. The primary unit of data is presented in the form of a data frame. Data frame stores data and exposes various helper methods to retrieve, search, update, and to do any operation with the data much like database queries. The recommendation system is designed and implemented using a python framework named Flask. Flask is a Model-View-Controller based mini-framework. It is mini-framework in a sense that it does not utilize the full-fledged features of an MVC framework. It uses only those modules which are required on demand and consequently, it brings efficiency in its core. Modules of the framework are loosely coupled and can be utilized as per demand. For Front-End technologies, we use HTML5, CSS3, and JavaScript. We also use Bootstrap 3 and a powerful Front-End framework for re-using the code and making the tool responsive to run on mobile screens too [51]. The significant part of the presentation has been developed with the help of a Jinja Templating Engine. As part of the implementation of this work, a Web-based Graphical User Interface (GUI) based on latest web tools like HTML5, CSS3, Javascript and Bootstrap is developed to enable users to interact with the user interface with best user experience screens. Bootstrap 3 is used for providing the application with the ability to adapt to screens of any width. Thus, the proposed application works both on mobile screens as well as on desktop screens.

VII. EXECUTION RESULTS

In this section, we describe different user preferences and constraints and apply the proposed optimization formula, as described in section 2. The optimization formula is based on the optimization index and is needed to be maximized. The optimization index is based on long-term location probabilities, short-term location probabilities, distance, and time. For maximizing the optimization index, distance and time need to be minimized, and location popularities need to be maximized. Thus, a more popular and closer location has higher optimization index than a location with less popularity and high distance and time. Algorithm 4 shows the mechanism of finding candidate routes based on user constraints. The symbols and acronyms used in the algorithm have already defined in table 1.

In the optimization approach, as presented in algorithm 4, different transition matrices are read alongside the dataset. The transition matrices are split based on the initial locations. Afterward, based on the distance and time constraints, those locations are included in the candidate subset which can safely satisfy the user constraints. The optimization index is then computed based on the formulae described in section 3 and then sorted to get the most popular top places. The computational complexity of algorithm 4 is computed based on the traversing time of Pandas data frame according to user preferences. Pandas exploit efficient searching techniques using a variety of functions, and the time is even further decreased if the data frame is sorted beforehand. Consequently, the worst-case execution time is still linear.

Algorithm 4 Optimization Index Computation Based on Markov Chain

$< Read > \leftarrow \Xi$
$< Read > \leftarrow \omega$
$< Read > \leftarrow \Gamma$
$\alpha \leftarrow < Read > \pi$
$\beta \leftarrow < Read > \pi$
$loc_i \leftarrow < Read > loc$
$p \Leftrightarrow loc_i \leftarrow \Xi.find[loc_i]$
$d \Leftrightarrow loc_i \leftarrow \omega.find[loc_i]$
$t \Leftrightarrow loc_i \leftarrow \Gamma.find[loc_i]$ ▷ Return Candidate Locations which is within Maximum Distance
$loc[c] \leftarrow P \Leftrightarrow loc_i[P \Leftrightarrow loc_i < \beta[d_max]]$
$i \leftarrow loc[c].index$
▷ Return Only those location which are candidate
$P[loc[c]] \leftarrow P \Leftrightarrow loc_i.find[P \Leftrightarrow loc_i.isin(i)]$
$\zeta \leftarrow \Omega(i = loc[c])$
for *i in* $loc[c]$ **do**
    $key \leftarrow loc[c][i]$
    $p \leftarrow loc[c][i]$
    $d \leftarrow loc[c][key]$
    ▷ Initialize all diagonal to 0
    if $d == 0$ **then**
        $\zeta[key] \leftarrow 0$
        $ContinueLoop$
    else
        $I_{max} \leftarrow \alpha[p_s] * p + \alpha[p_l] * p$
        $I_{min} \leftarrow \alpha[d] * d + \alpha[t] * t$
        $temp \leftarrow \frac{I_{max}}{I_{min}}$
        $\zeta[key] \leftarrow temp$
    ▷ Sort the OptimizationIndexes value of the Series
$loc[top] \leftarrow \zeta.Sort(asc = 0)[\beta[loc[tot]]]$

Another major computationally expensive operation is the iterative approach to go through the candidate location and find the route which gives rise to the shortest path. If the worst-case execution time is considered the overall complexity of the algorithm is $O(3n + clogc + c)$ where n is the number of rows in the dataset and c is the number of candidate locations which meet user preferences. $clogc$ is the sorting time for the shortest path, and $logc$ is the searching time.

A. RESULTS

In this part of the paper, different user constraints and preferences are varied to observe the different recommended routes satisfying constraints and giving more weights to the users' preferences. The cases for which the proposed recommendation system is tested are summarized in table 6.

In each case, we vary the initial starting location and user preferences. The user preferences are their priorities to certain weights. For instance, a user might be more interested in a popular location in short-term than in the long-term. In this case, the value of *sw* will be the highest. Similarly, if a user would like to emphasize more on maximum time, then the *tw* weight will be the highest of all values and the route recommended will be within the time the user has specified. On the other hand, in finding the optimal route, a user can specify some constraints. For instance, the initial location, the total number of locations the user wants to visit, the total distance at maximum, and the maximum time he wants to spend on the route. Table 7 to 12 summarize the top optimal locations which satisfy user constraints as well as user preferences. These places recommend the highly popular tourist spots based on the experimental data and at the same time, meet user constraints.

From the cases mentioned above, it is clear that the route selection is based on the popularity of the locations in the short-term and long-term as well as user preferences. For instance, in case 2 and case 3, the only difference is that the

TABLE 6. Summary of different user constraints and preference considered as inputs to the recommendation system.

Case	User Preferences				User Constraints			
	sw	lt	tw	dw	Initial Location	No. of Locs	Max. Dist	Max. Time
1	0.2	0.4	0.2	0.4	Manjanggul	5	120	180
2	0.1	0.7	0.1	0.1	Bijarim	4	120	120
3	0.8	0.05	0.1	0.1	Bijarim	4	120	120
4	0.2	0.6	0.1	0.3	Seongsanilchul	8	150	180
5	0.6	0.2	0.1	0.3	Seongsanilchul	8	150	180
6	0.5	0.4	0.05	0.05	Ilchulland	6	120	120

TABLE 7. Case 1 Result: Route = Gujwa gym->Bijarim->Seongsanilchul->Pyoseon->Andeok.

S.No	Location	Distance	Optimal Index
1	Gujwa gym	3.227364	0.012052
2	Bijarim	5.5	0.015251
3	Seongsanilchul Provincial Marine Park	17.6	0.029898
4	Pyoseon Haevichi beach	23.14	0.002189
5	Andeok-myeon	50.8	0.00195

TABLE 8. Case 2 Result: Route = Manjanggul->Gujwa gym->Seongsanilchul->Pyoseon.

S.No	Location	Distance	Optimal Index
1	Manjanggul	5.544663	0.025628
2	Gujwa gym	8.589099	0.004669
3	Seongsanilchul Provincial Marine Park	12.717091	0.035776
4	Pyoseon Haevichi beach	18.346219	0.009385

TABLE 9. Case 3 Result: Route = Manjanggul->Gujwa gym->Ilchulland->Seongsanilchul.

S.No	Location	Distance	Optimal Index
1	Manjanggul	5.544663	0.1408
2	Gujwa gym	8.589099	0.02854
3	Ilchulland	12.31	0.009808
4	Seongsanilchul Provincial Marine Park	12.71	0.120317

TABLE 10. Case 4 Result: Route = 2C->1C->Bij->Py->Manj->Nam->Jus->And.

S.No	Location	Distance	Optimal Index
1	2Course	1.807906	0.006338
2	1Course	3.98194	0.006304
3	Bijarim	12.717091	0.001975
4	Pyoseon Haevichi beach	17.455676	0.002907
5	Manjanggul	17.692621	0.00313
6	Namwon-eup	28.58515	0.001099
7	Jusangjeolli	53.961962	0.001579
8	Andeok-myeon	60.831231	0.001636

TABLE 11. Case 5 Result: Route = 2C->1C->Ilch->Bij->Py->Manj->Jus->And.

S.No	Location	Distance	Optimal Index
1	2Course	1.807906	0.009352
2	1Course	3.98194	0.012866
3	Ilchulland	12.52544	0.001661
3	Bijarim	12.717091	0.003823
4	Pyoseon Haevichi beach	17.455676	0.001606
5	Manjanggul	17.692621	0.007433
7	Jusangjeolli	53.961962	0.001884
8	Andeok-myeon	60.831231	0.000626

TABLE 12. Case 6 Result: Route = BMC->Pony->Pyo->JHD->Bij->Seon.

S.No	Location	Distance	Optimal Index
1	Blue Mountain Coffee theme park	0.539186	3.548627
2	Pony Valley	6.199235	0.118831
3	Pyoseon Haevichi beach	6.200371	0.075852
4	Jeju Herb Dongsan	6.461041	0.071938
5	Bijarim	12.313274	0.037965
6	Seongsanilchul Provincial Marine Park	12.52544	0.021446

user gives more preference to the location popularity in the long-term and the resultant route is significantly different for both cases.

B. CLIENT APPLICATION

In this subsection of the paper, we describe the client front application, which has developed using Flask and Google MAPS API. The application lets users enter their preferences and constraints using a web form. Once the form is submitted, the route summary and the Google Map directions are returned as a response. Figure 9 shows the main interface for the application.

The main components of the interface are user preference form, Google map, and a table shows the routing summary. Figure 10 and Figure 11 show the result for 7 locations. The routing summary displays the candidate locations which meet user constraints and according to user preferences.

The application is responsive to adapt on every screen and thus can be used pretty easily on mobile screens and even tablet devices. Figure 12 shows the screenshots of various interfaces on mobile size screens.

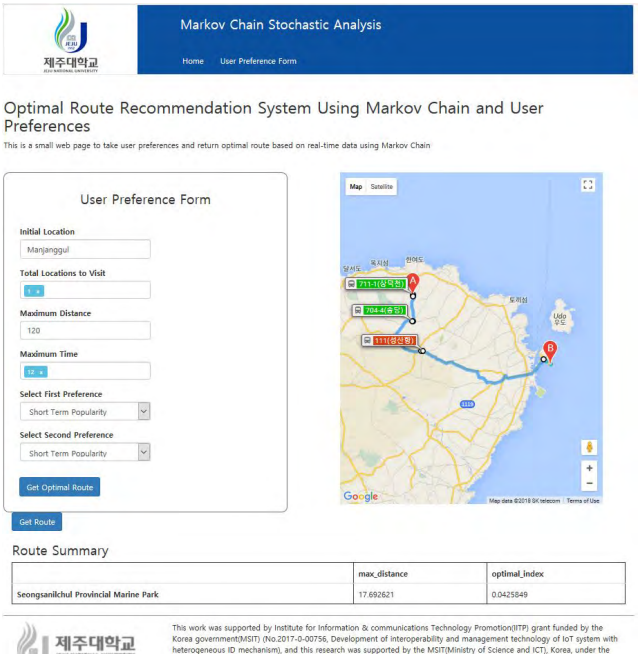


FIGURE 9. Main interface.

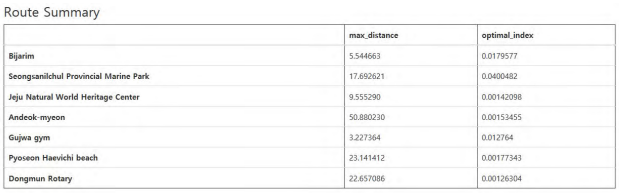


FIGURE 10. Routing summary.

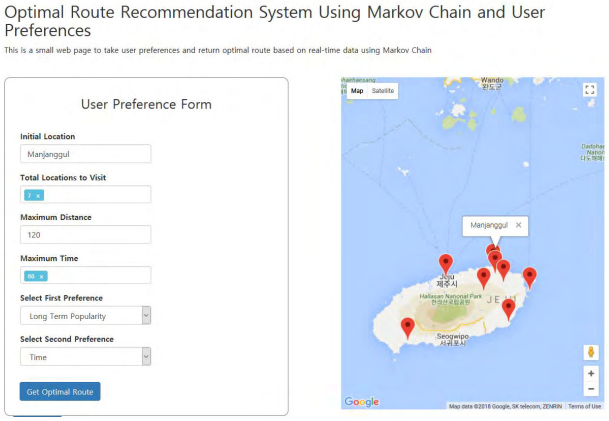
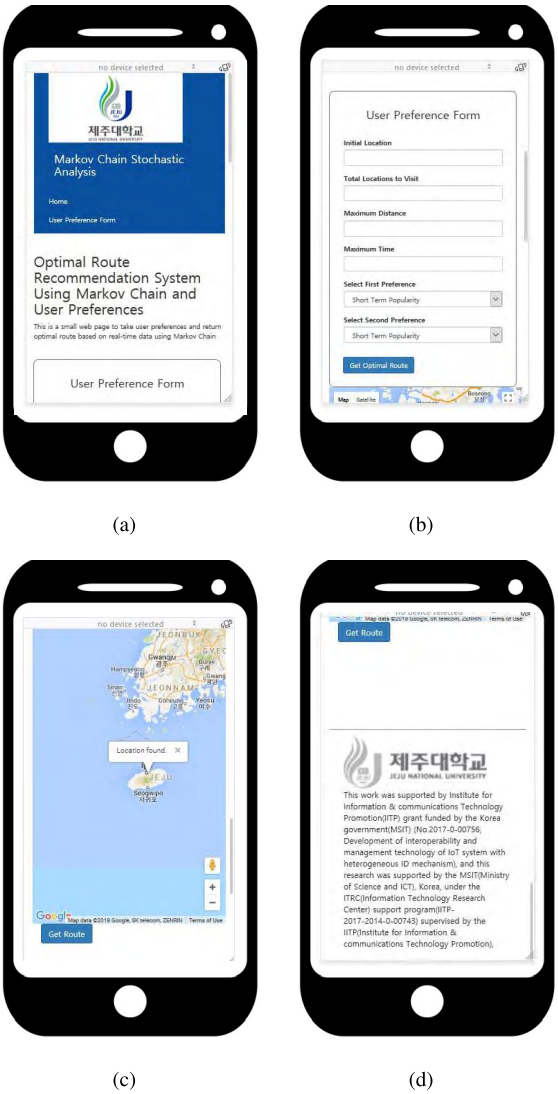


FIGURE 11. Sample form and corresponding location points on google map.

VIII. DISCUSSION AND SIGNIFICANCE

In this part of the paper, we aim to signify the contribution of this work and discuss the results achieved. The significance of this work is evaluated with the accuracy analysis, evaluation of the efficiency of the proposed solution with other state-of-the-art methods, and finally with its



(a) (b)

(c) (d)

FIGURE 12. Mobile interfaces of client system.

computational complexity. As discussed in the earlier sections, the Markov Chain model characterizes the behavior of the data stochastically by analyzing the historical pattern within the data. For examining the accuracy of the system, we run the application 100 times. The run of the system is not manually done but rather performed using Selenium testing environment. In the parameter, we have specified the sample size, and the resultant route is recorded for each cycle of the running life. The route tags are then decoded to actual names, and the generated route string is compared in the dataset for confirmation of the route. In the data some routes were found exactly similar, some routes were exactly dissimilar while some of them were route sharing some similar and dissimilar routes. We found the accuracy of the system using the following equation:

$$Error(\%) = \left(\frac{Di + pDi\prod(1-\lambda) + S_p\prod\lambda}{S + S_p\prod\lambda}\right)100 \quad (16)$$

where S is the number of similar routes, S_p is the number of partially similar routes, λ is the number of similar hops in a route, Di is the number of dissimilar routes, and pDi is the number of partially dissimilar routes. The percentage accuracy of the system is computed using $100-Error$. The result of the first 10 routes, 15, routes, and 20 routes is shown in figure 13.

The experiments have been automatically performed using Selenium IDE, and the resultant route is searched in the dataset. In every case, the number of similar routes and partially similar routes was more than partially dissimilar and completely dissimilar nodes. The maximum error found in the experiments were 5% for $iteration = 10$. The reason for this is the insufficiency of the data. For this said case, only 3 routes using the locations were found out of which, one was completely similar, one was having a single hop different, and one was partially dissimilar. Those locations which are too often visited has higher accuracy than those who are comparably less popular.

The second factor is the evaluation of the proposed system in terms of accuracy and time with other state-of-the-art methods. The dataset is a time series data and is not labeled, so recurrent neural network, grey model, and deep neural network are chosen to compare their performance gain with Markov Chain-based recommendation system. Additionally, the goal of the paper is not only the forecasting of popular spots in the short-term but also in the long-term. For this, the data is split across 4 quarters of the year 2017. The first quarter is validated with the 4th quarter as long-term forecasting, whereas the 2nd quarter is validated with the 3rd quarter as a short-term evaluation. For accuracy calculation, equation (16) is used. The results of the experiments are summarized in a bar graph, as shown in figure 14. The short-term accuracy of the recurrent neural network is slightly higher than the Markov Chain, but in the long-term, its accuracy is significantly lower. The deep neural network has the highest difference of accuracy in terms of short-term and high-term evaluation whereas grey model's accuracy is slightly lower than others, but the accuracy is stable in long-term as well which makes it the second best solution to Markov Chain. It is the reason why most of the modern studies are focused towards the combination of grey and Markov Chain to make better forecasting, particularly in tourism industries. Similarly, Long Short-Term Memory (LSTM) RNN could potentially improve the long-term accuracy, but it still has the limitation of long sequence which makes it not ideal in this case considering the route length in the dataset is long on average.

Another major aspect of the proposed recommendation system is the data processing time of the system. In addition to selecting an efficient solution based on the nature of the data, we use powerful Pandas API which is made for big data processing and even with a massive set of data it takes considerably less time than other technologies. There are efficient methods of searching, sorting, and other operations within the data, which makes it super helpful in massive

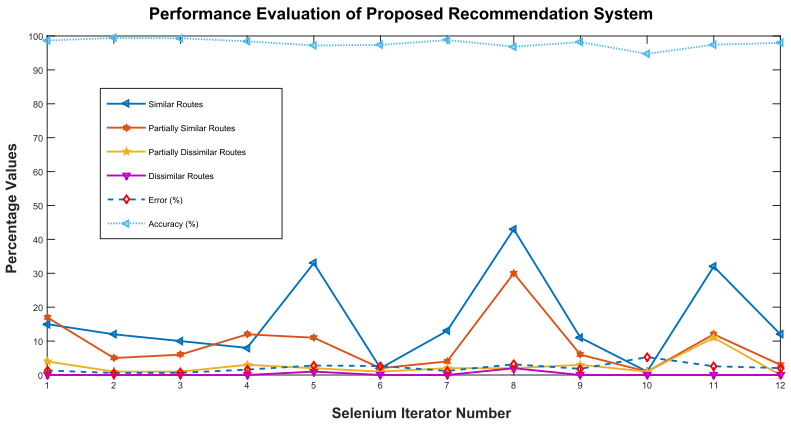


FIGURE 13. Performance evaluation of the proposed recommendation system with the historical data.

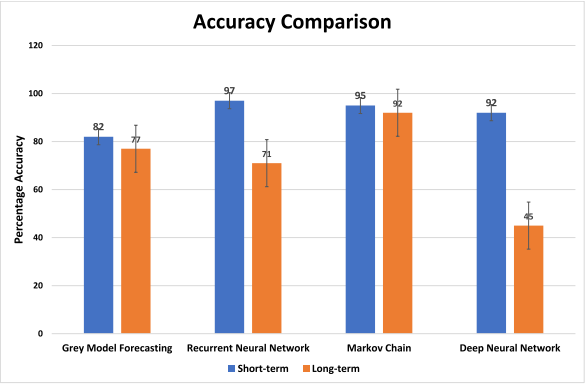


FIGURE 14. Accuracy evaluation of the proposed recommendation system with state-of-the-art methods.

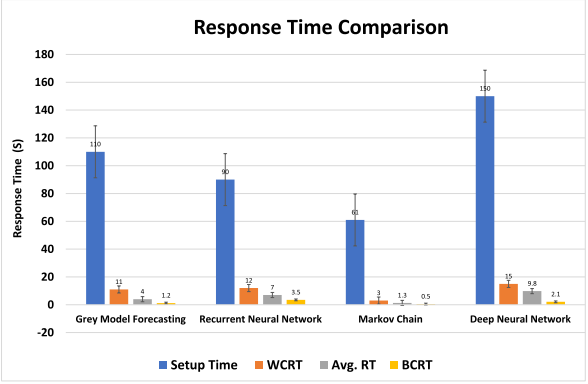


FIGURE 15. Response time evaluation of the proposed recommendation system with state-of-the-art methods.

datasets processing. There are more than 36000 records in the dataset and processing them using any other techniques, would have taken a lot more time. The main slow operation was the processing the whole data and forming transition matrix, which barely took 2 minutes, which is very much small given the massive amount of data. For a complete epoch of the data, it took 5 minutes to extract the stochastic behavior of the system and converge long-term probabilities. The algorithm complexity also suggests that the slowest operation is in the order of $O(n^2)$ in case of forming transition matrices, but since the number of unique locations in Jeju is very limited [up to 36], so it would not affect the overall performance.

To systematically evaluate the response time of the proposed recommendation system from applying user preference query to displaying of results, it is automatically tested 20 times using selenium. The setup time, the worst-case response time (WCRT), average response time, and best-case response time (BCRT) are recorded. The setup time is the slowest time, but it is done only once to find steady states. In the case of deep neural networks, the setup time is equivalent to training time. The response time comparison is shown in figure 15. The setup time of all the methods is significantly different. While the deep neural network is on the slowest slide, but it can be further enhanced by tweaking the number of epochs and hidden layer at the cost of accuracy. The Markov Chain model uses steady state matrix formation as a setup phase, which is the fastest among all. The access time is the time taken by the system to take user preference query and return the optimal route. For all the methods, this time is in the order of a few seconds. In the case of Markov Chain, the worst-case time is only 3 seconds, whereas the average response time is just over a second, which is very fast considering the massive set of data. Grey model is the next faster solution with the average response time of only 4 seconds. Other methods also perform reasonably well but considering all the measure such as long-term accuracy, setup time and WCRT; the Markov Chain is by far the smarter choice among the state-of-art methods.

IX. CONCLUSION

In this paper, we have worked around a real dataset based on the locations of Jeju Island, South Korea. We have extracted the visitor's movement routes and patterns and analyzed the stochastic behavior of the data using Markov Chain model.

As part of the Markov Chain model, we formed different transition matrices indicating probabilities transition, distance, and time transitions among different locations. The long-term steady states predictions alongside user preferences and constraints helped in finding optimal routes within user constraints. A recommendation system is developed to take user constraints such as current location, number of locations to visit, maximum time and maximum distance from users and return optimal routes on Google Maps having maximum index within input constraints. The application is responsive, so it can easily be accessed on mobile phones as well. The performance is evaluated with respect to different state-of-the-art methods, and it has been learned that the system accuracy is always within 95 to 100% whereas the long-term accuracy which is one of the central goals of this paper is the highest among all solutions. Based on the performance and the novelty of data, the proposed recommendation system is regarded as a significant contribution to the state-of-the-art.

CONFLICT OF INTERESTS

The authors declare that there is no conflict of interests regarding the publication of this paper.

REFERENCES

[1] Y.-C. Hu, "Predicting foreign tourists for the tourism industry using soft computing-based Grey-Markov models," *Sustainability*, vol. 9, no. 7, p. 1228, 2017.

[2] S. Mao, E. Tu, G. Zhang, L. Rachmawati, E. Rajabally, and G.-B. Huang, "An automatic identification system (AIS) database for maritime trajectory prediction and data mining," in *Proc. ELM*. Cham, Switzerland: Springer, 2016, pp. 241–257.

[3] H. Gonzalez, J. Han, X. Li, M. Myslinska, and J. P. Sondag, "Adaptive fastest path computation on a road network: A traffic mining approach," in *Proc. 33rd Int. Conf. Very Large Data Bases*, 2007, pp. 794–805.

[4] I. Haldankar, M. Tiwari, G. Usha, and S. Aruna, "An intelligent framework for road safety and driver behavioral change detection system using machine intelligence," in *Computational Vision and Bio Inspired Computing*. Cham, Switzerland: Springer, 2018, pp. 913–924.

[5] L. Li, J. Miao, S. Fang, I.-H. Yang, and J. Wang, "Method and system to predict vehicle traffic behavior for autonomous vehicles to make driving decisions," U.S. Patent 0 143 644 A1, May 24, 2018.

[6] F. Giannotti, M. Nanni, F. Pinelli, and D. Pedreschi, "Trajectory pattern mining," in *Proc. 13th ACM SIGKDD Int. Conf. Knowl. Discovery Data Mining*, 2007, pp. 330–339.

[7] A. Monreale, F. Pinelli, R. Trasarti, and F. Giannotti, "Wherenext: A location predictor on trajectory pattern mining," in *Proc. 15th ACM SIGKDD Int. Conf. Knowl. Discovery Data Mining*, 2009, pp. 637–646.

[8] H. Jeung, Q. Liu, H. T. Shen, and X. Zhou, "A hybrid prediction model for moving objects," in *Proc. IEEE 24th Int. Conf. Data Eng.*, Apr. 2008, pp. 70–79.

[9] C. M. Grinstead and J. L. Snell, *Introduction to Probability*. Providence, RI, USA: American Mathematical Socity, 2012. [Online]. Available: https://math.dartmouth.edu/~prob/prob/prob.pdf

[10] H. W. Corley and I. D. Moon, "Shortest paths in networks with vector weights," *J. Optim. Theory Appl.*, vol. 46, no. 1, pp. 79–86, 1985.

[11] Y. Zheng, Y. Li, C.-M. Own, Z. Meng, and M. Gao, "Real-time predication and navigation on traffic congestion model with equilibrium Markov chain," *Int. J. Distrib. Sensor Netw.*, vol. 14, no. 4, 2018.

[12] S. Salman and S. Alaswad, "Alleviating road network congestion: Traffic pattern optimization using Markov chain traffic assignment," *Comput. Oper. Res.*, vol. 99, pp. 191–205, Nov. 2018.

[13] N. Mamoulis, H. Cao, G. Kollios, M. Hadjieleftheriou, Y. Tao, and D. W. Cheung, "Mining, indexing, and querying historical spatiotemporal data," in *Proc. 10th ACM SIGKDD Int. Conf. Knowl. Discovery Data Mining*, 2004, pp. 236–245.

[14] Y. Zheng, L. Zhang, X. Xie, and W.-Y. Ma, "Mining interesting locations and travel sequences from GPS trajectories," in *Proc.18th Int. Conf. World wide Web*, 2009, pp. 791–800.

[15] J.-G. Lee, J. Han, and K.-Y. Whang, "Trajectory clustering: A partition-and-group framework," in *Proc. ACM SIGMOD Int. Conf. Manage. Data*, 2007, pp. 593–604.

[16] J.-G. Lee, J. Han, X. Li, and H. Gonzalez, "TraClass: Trajectory classification using hierarchical region-based and trajectory-based clustering," *Proc. VLDB Endowment*, vol. 1, no. 1, pp. 1081–1094, 2008.

[17] Y. Tao, C. Faloutsos, D. Papadias, and B. Liu, "Prediction and indexing of moving objects with unknown motion patterns," in *Proc. ACM SIGMOD Int. Conf. Manage. Data*, 2004, pp. 611–622.

[18] D. Sacharidis, K. Patroumpas, M. Terrovitis, V. Kantere, M. Potamias, K. Mouratidis, and T. Sellis, "On-line discovery of hot motion paths," in *Proc. 11th Int. Conf. Extending Database Technol. Adv. Database Technol.*, 2008, pp. 392–403.

[19] X. Li, J. Han, J.-G. Lee, and H. Gonzalez, "Traffic density-based discovery of hot routes in road networks," in *Proc. Int. Symp. Spatial Temporal Databases*. Cham, Switzerland: Springer, 2007, pp. 441–459.

[20] M. Ester, H.-P. Kriegel, J. Sander, and X. Xu, "A density-based algorithm for discovering clusters in large spatial databases with noise," in *Proc. KDD*, vol. 96. 1996, pp. 226–231.

[21] L. Cao and J. Krumm, "From GPS traces to a routable road map," in *Proc. 17th ACM SIGSPATIAL Int. Conf. Adv. Geograph. Inf. Syst.*, 2009, pp. 3–12.

[22] A. Fathi and J. Krumm, "Detecting road intersections from GPS traces," in *Proc. Int. Conf. Geographic Inf. Sci.* Cham, Switzerland: Springer, 2010, pp. 56–69.

[23] M. Hua and J. Pei, "Probabilistic path queries in road networks: Traffic uncertainty aware path selection," in *Proc. 13th Int. Conf. Extending Database Technol.*, 2010, pp. 347–358.

[24] L. Chen, M. T. Özsu, and V. Oria, "Robust and fast similarity search for moving object trajectories," in *Proc. ACM SIGMOD Int. Conf. Manage. Data*, 2005, pp. 491–502.

[25] M. Vlachos, G. Kollios, and D. Gunopulos, "Discovering similar multidimensional trajectories," in *Proc. 18th Int. Conf. Data Eng.*, 2002, pp. 673–684.

[26] D. Pfoser, C. S. Jensen, and Y. Theodoridis, "Novel approaches to the indexing of moving object trajectories," in *Proc. VLDB*, 2000, pp. 395–406.

[27] R. Sherkat and D. Rafiei, "On efficiently searching trajectories and archival data for historical similarities," *Proc. VLDB Endowment*, vol. 1, no. 1, pp. 896–908, 2008.

[28] Z. Chen, H. T. Shen, X. Zhou, Y. Zheng, and X. Xie, "Searching trajectories by locations: An efficiency study," in *Proc. ACM SIGMOD Int. Conf. Manage. Data*, 2010, pp. 255–266.

[29] A. V. Goldberg and C. Harrelson, "Computing the shortest path: A search meets graph theory," in *Proc. 16th Annu. ACM-SIAM Symp. Discrete Algorithms*, 2005, pp. 156–165.

[30] B. Ding, J. X. Yu, and L. Qin, "Finding time-dependent shortest paths over large graphs," in *Proc. 11th Int. Conf. Extending Database Technol. Adv. Database Technol.*, 2008, pp. 205–216.

[31] E. Kanoulas, Y. Du, T. Xia, and D. Zhang, "Finding fastest paths on a road network with speed patterns," in *Proc. 22nd Int. Conf. Data Eng. (ICDE)*, Apr. 2006, p. 10.

[32] M. Ramezani and N. Geroliminis, "On the estimation of arterial route travel time distribution with Markov chains," *Transp. Res. B, Methodol.*, vol. 46, no. 10, pp. 1576–1590, 2012.

[33] H. Q. Vu, G. Li, R. Law, and B. H. Ye, "Exploring the travel behaviors of inbound tourists to Hong Kong using geotagged photos," *Tourism Manage.*, vol. 46, pp. 222–232, Feb. 2015.

[34] J. Yeon, L. Elefteriadou, and S. Lawphongpanich, "Travel time estimation on a freeway using discrete time Markov chains," *Transp. Res. B, Methodol.*, vol. 42, no. 4, pp. 325–338, 2008.

[35] Z. Chen, H. T. Shen, and X. Zhou, "Discovering popular routes from trajectories," in *Proc. IEEE 27th Int. Conf. Data Eng.*, Apr. 2011, pp. 900–911.

[36] M. L. Hazelton, S. Lee, and J. W. Polak, "Stationary states in stochastic process models of traffic assignment: A Markov Chain Monte Carlo approach," in *Proc. 13th Int. Symp. Transp. Traffic Theory*, Pergamon, Turkey, 1996, pp. 341–357.

[37] D. Ju-Long, "Control problems of grey systems," *Syst. Control Lett.*, vol. 1, no. 5, pp. 288–294, 1982.

[38] S. Liu and J. Y. L. Forrest, *Grey Systems: Theory and Applications*. Cham, Switzerland: Springer, 2010.
[39] S. Liu, Y. Yang, and J. Forrest, *Grey Data Analysis: Methods, Models and Applications*. Cham, Switzerland: Springer, 2017.
[40] L. Suganthi and A. A. Samuel, "Energy models for demand forecasting—A review," *Renew. Sustain. Energy Rev.*, vol. 16, no. 2, pp. 1223–1240, 2012.
[41] J. Cui, S.-F. Liu, B. Zeng, and N.-M. Xie, "A novel grey forecasting model and its optimization," *Appl. Math. Model.*, vol. 37, no. 6, pp. 4399–4406, 2013.
[42] P. Antsaklis and J. Baillieul, "Guest editorial special issue on networked control systems," *IEEE Trans. Autom. Control*, vol. 49, no. 9, pp. 1421–1423, Sep. 2004.
[43] W. Zhang, K. Xing, J. Li, and M. Chen, "Adaptive synchronization of delayed reaction-diffusion FCNNs via learning control approach," *J. Intell. Fuzzy Syst.*, vol. 28, no. 1, pp. 141–150, 2015.
[44] J. Wang, M. Chen, H. Shen, J. H. Park, and Z.-G. Wu, "A Markov jump model approach to reliable event-triggered retarded dynamic output feedback H_∞ control for networked systems," *Nonlinear Anal. Hybrid Syst.*, vol. 26, pp. 137–150, Nov. 2017.
[45] H. Shen, Y. Zhu, L. Zhang, and J. H. Park, "Extended dissipative state estimation for Markov jump neural networks with unreliable links," *IEEE Trans. Neural Netw. Learn. Syst.*, vol. 28, no. 2, pp. 346–358, Feb. 2017.
[46] D. Kong and F. Wu, "HST-LSTM: A hierarchical spatial-temporal long-short term memory network for location Prediction," in *Proc. IJCAI*, 2018, pp. 2341–2347.
[47] N. C. Chen, W. Xie, R. E. Welsch, K. Larson, and J. Xie, "Comprehensive predictions of tourists' next visit location based on call detail records using machine learning and deep learning methods," in *Proc. IEEE Int. Congr. Big Data (BigData Congress)*, Jun. 2017, pp. 1–6.
[48] A. Onueam, N. Choi, T. Kim, D. Lee, and H. Jung, "Adaptive road segmentation of openstreetmap data using haversine formula," in *Proc. Int. Conf. Future Inf. Commun. Eng.*, 2018, vol. 10, no. 1, pp. 297–299.
[49] T. Nadu, S. Aryasomayajula, S. Singh, R. Gupta, and P. Sawant, "Anti-theft vehicle tracking with automatic police notifying using haversine formula," Int. J. Advance Res., Ideas Innov. Technol., Tamil Nadu, India, Tech. Rep., 2018. vol. 4, no. 2, pp. 584–588. [Online]. Available: https://www.ijariit.com/manuscripts/v4i2/V4I2-1375.pdf
[50] *Open Elevation Api*. Accessed: Oct. 16, 2018. [Online]. Available: https://api.open-elevation.com/api/v1/lookup?locations=lat,long
[51] *Bootstrap*. Accessed: Oct. 15, 2018. [Online]. Available: https://getbootstrap.com/

SHABIR AHMAD received the B.S. degree in computer system engineering from the University of Engineering and Technology, Peshawar, Pakistan, and the M.S. degree in computer software engineering from the National University of Science and Technology, Islamabad, Pakistan, in 2013. He is currently pursuing the Ph.D. degree with the Department of Computer Engineering, Jeju National University, South Korea. He is currently a Faculty Member with the Software Engineering Department, University of Engineering and Technology, Peshawar. His research interests include the Internet-of-Things application, cyber–physical systems, and intelligent systems.

ISRAR ULLAH received the M.C.S. degree from the Institute of Computing and Information Technology (ICIT), Gomal University, Pakistan, in 2004, and the M.S. degree in computer science from the National University of Computer and Emerging Sciences (NUCES), Islamabad, Pakistan, in 2009. He is currently pursuing the Ph.D. degree with the Computer Engineering Department, Jeju National University, South Korea. His research work is focused on the application of prediction and optimization algorithms to the build IoT-based solutions. His research interests include analytical modeling, network simulation, and analysis of optimization algorithms.

FAISAL MEHMOOD received the M.C.S. degree from the University Institute of Information Technology, PMAS-University of Arid Agriculture, Rawalpindi, Pakistan, in 2014. He is currently pursuing the master's degree in computer engineering with Jeju National University, South Korea. He moved to Jeju, South Korea, in 2018. He has professional experience in the software development industry and in academic as well. His research interests include the IoT, big data, and machine learning.

MUHAMMAD FAYAZ received the M.S. degree in computer science from SZABIST, Islamabad, Pakistan, in 2014, and the M.Sc. degree in computer science from the University of Malakand, Chakdara, Pakistan. He is currently pursuing the Ph.D. degree with the Department of Computer Engineering, Jeju National University, South Korea.

DOHYEUN KIM received the B.S. degree in electronics engineering from Kyungpook National University, South Korea, in 1988, and the M.S. and Ph.D. degrees in information telecommunication from Kyungpook National University, South Korea, in 1990 and 2000, respectively. He was with the Agency of Defense Development (ADD), from 1990 to 1995. Since 2004, he has been with Jeju National University, South Korea, where he is currently a Professor with the Department of Computer Engineering. From 2008 to 2009, he was a Visiting Researcher with the Queensland University of Technology, Australia. His research interests include sensor networks, M2M/IOT, energy optimization and prediction, intelligent service, and mobile computing.