A Novel Supervised Clustering Algorithm for Transportation System Applications

Mohammed H. Almannaa, Mohammed Elhenawy, and Hesham A. Rakha, *Senior Member, IEEE*

***Abstract*— This paper proposes a novel supervised clustering algorithm to analyze large datasets. The proposed clustering algorithm models the problem as a matching problem between two disjoint sets of agents, namely, centroids and data points. This novel view of the clustering problem allows the proposed algorithm to be multi-objective, where each agent may have its own objective function. The proposed algorithm is used to maximize the purity and similarity in each cluster simultaneously. Our algorithm shows promising performance when tested using two different transportation datasets. The first dataset includes speed measurements along a section of Interstate 64 in the state of Virginia, while the second dataset includes the bike station status of a bike sharing system (BSS) in the San Francisco Bay Area. We clustered each dataset separately to examine how traffic and bike patterns change within clusters and then determined when and where the system would be congested or imbalanced, respectively. Using a spatial analysis of these congestion states or imbalance points, we propose potential solutions for decision makers and agencies to improve the operations of I-64 and the BSS. We demonstrate that the proposed algorithm produces better results than classical *k*-means clustering algorithms when applied to our datasets with respect to a time event. The contributions of our paper are: 1) we developed a multi-objective clustering algorithm; 2) the algorithm is scalable (polynomial order), fast, and simple; and 3) the algorithm simultaneously identifies a stable number of clusters and clusters the data.**

***Index Terms*— Supervised clustering, high dimensional datasets and traffic operations, bike-sharing systems, urban computing, classification.**

I. Introduction

With the growth of new technologies, smart cities and urban areas are adapting advanced devices to control and monitor transportation networks and thus provide better service to the public and private sectors. These devices collect data through many sensors in the city's infrastructure. Agencies and researchers exploring the massive amounts of collected data often find it challenging to draw meaningful conclusions due the sheer size of the datasets. One way to deal with such data is to use clustering approaches.

In the transportation field, operating agencies (such as departments of transportation) have been collecting data to improve the efficiency of the transportation network and provide a better service for all transportation modes. Clustering the travel times or speeds of transportation modes could help operating agencies to better manage the transportation network. In particular, the collected data could be reduced to find the cluster centroids (i.e., the means of the clusters) that represent the entire data with respect to a time event such as time of day, day of month, and month of the year. This could help operating agencies answer several questions related to traffic operations such as, "Can we discriminate between recurrent congestion and outliers?" and "Can we identify how many time periods we need to plan for in terms of resource and congestion management?"

Clustering is an unsupervised learning technique that identifies the underlying structure of unlabeled data. The goal of clustering is to identify intrinsic groupings in an unlabeled dataset. Meaningful clustering depends on the clustering criterion used by the clustering algorithm. Accordingly, it is crucial to find the best criterion so that the clustering results will suit the needs of researchers a{Roberts, 1995 #46}nd agencies.

Clustering algorithms are used in many disciplines, such as computer vision to segment images [1], marketing to find similar customer behaviors [2], the insurance industry to identify fraud [3], [4], and in transportation to identify similar patterns in various modes of transport [5]–[7]. Clustering helps develop a deep understanding of similarity in data patterns. For example, traffic engineers can use clustering algorithms to identify similar traffic patterns on a highway during the day, week, or month, and then make use of the clustered patterns in the management of the system. Clustering has also been used to analyze bike sharing system (BSS) data [8], [9]. Some researchers have used a statistical model to predict bike availability at each station, while others have used clustering algorithms, such as traditional and non-traditional clustering [10]. Traditional clustering approaches, such as the k-median, DBSCAN, and fuzzy algorithms are good tools for clustering data, but give narrow results, as clusters are based on only one factor (i.e., distance or similarity). These

Manuscript received September 15, 2017; revised May 12, 2018 and November 22, 2018; accepted December 14, 2018. This work was supported in part by the Mid-Atlantic Transportation Sustainability University Transportation Center (MATS UTC) and in part by the Virginia Department of Transportation. The Associate Editor for this paper was V. Marzano. *(Corresponding author: Hesham A. Rakha.)*

M. H. Almannaa is with the Charles E. Via, Jr. Department of Civil and Environmental Engineering, Virginia Tech, Blacksburg, VA 24061 USA, with the Center for Sustainable Mobility, Virginia Tech Transportation Institute, Blacksburg, VA 24061 USA, and also with the Civil Engineering Department, King Saud University, Riyadh 2890588, Saudi Arabia (e-mail: almannaa@vt.edu).

M. Elhenawy is with the Centre for Accident Research and Road Safety, Queensland University of Technology, Brisbane, QLD 4059, Australia (e-mail: elhenawy@vt.edu).

H. A. Rakha is with the Charles E. Via, Jr. Department of Civil and Environmental Engineering, Virginia Tech, Blacksburg, VA 24061 USA, with the Bradley Department of Electrical and Computer Engineering, Virginia Tech, Blacksburg, VA 24061 USA, and also with the Center for Sustainable Mobility, Virginia Tech Transportation Institute, Blacksburg, VA 24061 USA (e-mail: hrakha@vt.edu).

Digital Object Identifier 10.1109/TITS.2018.2890588

clustering algorithms are unsupervised clustering that divides the observational points into clusters based on an objective function without considering natural labels in the dataset, such as the time of events (i.e., month of year, day of week, or time of day).

Recently, supervised clustering (non-traditional) approaches have been widely embraced as powerful tools that can take advantage of other attributes (labels) in the dataset [11]–[13]. Unlike traditional clustering techniques, the supervised technique clusters labeled data. Supervised algorithms use data labels to represent natural data groupings using the minimum possible number of clusters. Only the labels are used as an objective function, and distance and similarity are ignored [11], [14].

In this paper, we propose a new supervised clustering algorithm based on the college admission (CA) game theory algorithm [15] to maximize the reciprocal of the within-cluster sum of distances (similarity) and the cluster purity simultaneously. The proposed algorithm was used to answer several transportation-related research questions, such as which days or months exhibit similar patterns.

To evaluate the proposed algorithm, we tested it twice using two different transportation datasets. The first dataset is the station status of a BSS in the San Francisco Bay Area, which consists of bicycle count data. The second dataset is the speed data and instantaneous travel time collected along I-64 in Virginia. These data are real-valued data. The proposed algorithm successfully found meaningful underlying structures in each dataset when using natural data labeling (i.e., time of day, day of week, month of year [only for the I-64 dataset]). We then studied how bike or traffic patterns changed within each cluster, and addressed when and where the system would be imbalanced or congested.

II. Problem Statement

Operating agencies and transportation researchers have devoted significant attention to clustering approaches with the goal of clustering large datasets that contain traffic patterns (i.e., travel times or speeds) in transportation networks [5]–[7], [16]. They have adopted classical approaches such as k-means, Ward's hierarchical clustering algorithms, and density-based clustering. The purpose of using these clustering approaches is to (1) cluster traffic patterns with respect to a time event so that operators can have a temporal plan for operations planning purposes, and (2) discriminate between recurrent congestion and outliers. However, the aforementioned studies used classical clustering approaches that do not take advantage of natural time event labels (such as time of day, day of week, etc.). As for unsupervised clustering algorithms, they implicitly assume that clustering the data points based on similarity or distance leads to the ground truth of the clustering, which is not necessarily true. These algorithms cannot consider both similarity/distance and other domain knowledge information in the objective function. Consequently, clustering solutions do not help operators map the clustering solution to the network demand with regard to time events [17], [18].

In this research, we present a supervised clustering algorithm that attempts to find similar months, days, or hours within a day that have similar traffic patterns. We sacrifice the exact centroids of traffic patterns on account of having similar time events. The proposed algorithm is scalable (polynomial order), fast, and ready for practitioners. It makes no assumptions about the dataset and requires only one parameter, namely the number of clusters, which can be found using the consensus clustering (CC) technique (explained in section VII). It compromises between distance and purity in identifying clusters within the data.

III. Related Work

Clustering algorithms can be categorized into three main approaches: unsupervised (i.e., traditional), supervised, and semi-supervised. Unsupervised clustering algorithms assume the data are unlabeled (i.e., the relationship is unknown between the data points) and thus try to cluster them according to similarity or distance [19]. They implicitly assume that clustering the data points by distance or similarity leads to the ground truth of the clustering. The supervised clustering approach deals with labeled data (the relationship is known). There are a variety of supervised clustering algorithms. Some of these algorithms attempt to cluster data according to the labels (i.e., purity) and number of clusters [11]. Another algorithm uses the labels to learn the best similarity measure that produces the desirable clustering solution [20]. The semi-supervised clustering algorithms assume that part of the data is labeled and the rest is not. The known labels can be used to form constraints between pairs of data points in the form of must-link and cannot-link [21], [22] (which will not be covered in this paper as it is very different).

Two examples of unsupervised clustering algorithms are the well-known k-means and hierarchical clustering algorithms [23], [24]. The k-means simply partitions the data points into clusters, minimizing the distortion of each cluster [23]. The value of the model order (k) is set by the user based on personal knowledge or is chosen to maximize some criteria such as the clustering stability. At each iteration, the k-means algorithm assigns all the observation points to the clusters and updates the centroid of each cluster. Eventually, the k-means algorithm converges when the centroids stop moving.

The hierarchical clustering algorithm is a tree-based structure. It does not require the modeler to specify k apriori. Moreover, the dendrogram can be utilized to select the optimum number of clusters [24]. At every level of the tree-based structure, similar clusters are merged into one cluster. The key to this clustering algorithm is the criteria determining when and which two clusters can be merged. Different approaches are used, such as single linkage and complete linkage. The only difference between this algorithm and the k-means is the use of a similarity measure between clusters besides data points, but both use only similarity or the distance measure. More advanced unsupervised clustering algorithms have been proposed such as kernel k-means [25], kernel self-organizing maps [26], and kernel fuzzy c-means [27]. These algorithms attempt to cluster the data points by transforming them into

a higher dimensional feature space and then carry out the original clustering algorithm, which is based on the similarity or distance without considering other domain knowledge information.

Supervised clustering algorithms go a step further and endeavor to improve the unsupervised clustering algorithms by incorporating purity (i.e., labels) in the objective function [11], [14]. Purity means using labeled data to identify clusters that have a high probability density with respect to a single class. Eick *et al.* [11] proposed four different supervised clustering algorithms with the same objective function containing a linear combination of impurity and number of clusters. The aim is to minimize impurity and the number of clusters. However, these algorithms do not consider the similarity or distance measure. Spinelli [14] proposed a supervised clustering algorithm called Box Clustering that clusters data points into specific convex polygons with a fixed cluster impurity. Similar to Eick *et al.*'s [11] work, similarity was not incorporated in the objective function. Another approach to supervised clustering algorithms was given by Awasthi and Zadeh [28]. They assumed there is access for a teacher that can help improve the purity of the clusters [28]. Yet, this approach assumes that the teacher knows the ground truth of the data, which is not always the case in many datasets (i.e., assumes we have two datasets: training and test).

Recently, supervised clustering algorithms have been enhanced greatly by using a multi-objective approach [29]–[33]. This approach aims to optimize several clustering criteria such as similarity or compactness of the clusters and connectivity of the clusters. The goal is to compromise between these objective functions and produce a trade-off solution. This has led them to be widely introduced in data mining as a powerful way to effectively classify labeled datasets. Law *et al.* [32] proposed a multi-objective approach in a two-step process. In the first step, they used different clustering algorithms with different goals, and in the second step they integrated the output into a single partition. The labels of the datasets were only used for evaluating the clustering results but not in the objective function. Handl and Knowles [33] proposed a multi-objective evolutionary algorithm, maximizing the compactness and connectivity of the clusters simultaneously. This approach (i.e., the evolution optimization algorithm) gives many possible solutions (so called population approach) at each iteration, and thus the authors used a Pareto-based approach to select the non-dominate solutions that were created by the proposed algorithm.

None of the previous approaches used both purity as well as similarity in the objective function. Only a few supervised clustering algorithms had both purity (i.e., background information) and distance or similarity in the objective function, yet they suffer from complexity and having many assumptions and parameters, making them hard to interpret [24], [25]. For instance, Marcu [29] used the Dirichlet process prior to using a Bayesian approach to incorporate both similarity and purity. This approach is considered a generative model, meaning it estimates the joint probability distribution of the data between the observed data and the corresponding labels. This algorithm suffers from several drawbacks: (1) it is complex—one has to define the distribution of the data (which is usually unknown) and also has to use the Markov Chain Monte Carlo-based (MCMC) sampling to avoid intractability; (2) it cannot define a good distribution for the data due to its generative nature; and (3) it cannot deal with a large dataset, and thus the scalability is an issue. Forestier *et al.* [30] proposed a collaborative clustering algorithm that incorporates three components: cluster quality, class label, and link-based constraints. This approach selects a subset of the dataset as background knowledge randomly, causing it to be less stable. It also requires an expert who can tell which subset of the dataset to use as background knowledge.

In this paper, we propose a new supervised clustering algorithm with the ability to increase both cluster purity and member similarity simultaneously. The proposed algorithm is scalable, quick, and simple, considering only one parameter—the number of clusters. It compromises between distance and purity in identifying clusters within the data. It showed promising performance when applied to the I-64 and BSS datasets. It clustered the travel times and speeds of I-64 and bike availability with respect to a time event, giving operators more practical clustering results for operation planning purposes.

IV. The College Admission Algorithm

In 1962, Gale and Shapley [15] proposed the deferred acceptance algorithm as a solution to the stable marriage problem, in which an equal number of men and women are matched such that no player has an incentive to leave his/her matched partner. The stable marriage problem involves one-to-one matching. The college admission (CA) problem is another version of the stable marriage problem, though in this case the algorithm matches many to one. In the CA problem, there are a number of colleges and applicants that need to be matched. Each college has a ranked list of students they prefer, and each student has a ranked list of colleges they prefer. The size of the ranked list of students depends on the capacity of the college. The best-qualified candidates are offered admission first, followed by the lesser-qualified candidates.

This problem includes the uncertainty of the colleges not knowing which other colleges the students have applied to, and thus not knowing the ranked list of each student, or whether the student has been offered admission by other colleges. Consequently, the colleges are in a blind position with very little information, which prevents them from making the appropriate decision. This can result in an unbalanced situation in which some students are offered many admissions, while others are not offered any at all. Gale and Shapley presented a stable solution where each student would be accepted to the best possible college with regard to his or her list, and each college would have the best possible qualified student.

The CA algorithm finds a stable matching solution through a series of iterations. At each iteration, the colleges offer admission to the best-qualified students, and the students have to reply back by either accepting the offer or not. At the end of the iteration, some students have an admission and others do not. Colleges then update their list accordingly in the next iteration and offer admission to students who did not receive

an offer in the previous iterations, regardless of whether they have an admission or not. The students' lists do not change, but students can change their decision at each iteration if they are offered admission to a better college. The algorithm continues iterating until it reaches a stable matching solution.

V. The Proposed Algorithm

Knowing some similarities in the dataset is a great advantage to clustering algorithms. It can efficiently and effectively advance the outcome of the algorithm and create meaningful clusters. Accordingly, we developed a novel supervised clustering algorithm that is based on the CA algorithm [15]. The proposed algorithm takes advantage of the natural labeling of the data (i.e., day of week, time of day) and models the clustering problem as a cooperative game. In this game, two disjointed sets of players join the game to identify a stable match. The first player's set consists of the centroids (clusters), and the second player's set consists of the data examples (data points). Each centroid orders the data points in its preference list based on the distance from the centroid to the data point. Alternatively, each data point orders the centroids in its preference list based on the purity. For example, a data point that has label h will give preference to the centroid that has the proportion of members with label h. In other words, a data point gives higher preference to centroids when the majority of its members have the same label as its own label. Through a series of iterations, the proposed algorithm tries to match between the clusters, which want to minimize distances, and data points, which want to maximize purity, until it converges. It should be noted that cluster purity is the number of objects of the largest class in this cluster divided by the cardinality of the cluster, as presented in Eq. (1). The similarity measure is computed using Eq. (2). The algorithm terminates when the stopping criteria of Eq. (3) are met.

$$purity\,(c_i)^t = \max_m \left(\frac{n_i^m}{n_i}\right) \tag{1}$$

$$similarity\,(c_i)^t = \sum\nolimits_{x_j \in c_i} 1/d\left(\boldsymbol{x}_j, \boldsymbol{c}_i\right) \tag{2}$$

$$\alpha \left| \frac{\sum_{i=1}^{K} purity\,(c_i)^t - \sum_{i=1}^{K} purity\,(c_i)^{t-1}}{\sum_{i=1}^{K} purity\,(c_i)^{t-1}} \right| + (1-\alpha) \times \left| \frac{\sum_{i=1}^{K} similarity\,(c_i)^t - \sum_{i=1}^{K} similarity\,(c_i)^{t-1}}{\sum_{i=1}^{K} similarity\,(c_i)^{t-1}} \right| < \varepsilon \tag{3}$$

where t is the iteration number, n_i is the number of objects in cluster i (cardinality of cluster i), $i \in \{1, \ldots, K\}$, n_i^m is the number of the class (m) in cluster i, $m \in \{1, \ldots, M\}$, d is the distance between $\boldsymbol{x}_j$ and $\boldsymbol{c}_i$, $\boldsymbol{c}_i$ is the centroid of cluster i, $i \in \{1, \ldots, K\}$, $j \in \{1, \ldots, N\}$, N is the number of data points, $\boldsymbol{x}_j$ is the data vector j, α is a weighting factor (0.5 in our case), and ε is the stopping criteria threshold (0.0005 in our case).

We observe that one advantage of the proposed algorithm is that we do not need to write the entire objective function of the algorithm. Thus, we remove the normalization problem. However to stop the algorithm we normalize the purity difference by simply dividing by the previous purity and do the same with the similarity.

The following is a description of the proposed algorithm assuming the model order K is known:

1) Randomly choose K points as the initial centroids $\boldsymbol{c}_i$, $i \in \{1, \ldots, K\}$.
2) Form K clusters by assigning all points to the closest centroid using $L1$ norm distance where $\boldsymbol{x}_j$ is assigned to the centroid that satisfies $\min_{\boldsymbol{c}_i} \left\|\boldsymbol{x}_j - \boldsymbol{c}_i\right\|_1$.
3) Recompute the centroid of each cluster by computing the median. The median is computed in each single dimension.
4) Find the cardinality of each cluster.
5) Compute the within-clusters class distribution matrix P.
6) $P = \begin{bmatrix} \frac{n_1^1}{n_1} & \cdots & \frac{n_1^M}{n_1} \\ \vdots & \ddots & \vdots \\ \frac{n_K^1}{n_K} & \cdots & \frac{n_K^M}{n_K} \end{bmatrix}$
7) Each centroid $\boldsymbol{c}_i$ creates its preference list of points $\boldsymbol{x}_j \forall j \in \{1, \ldots, N\}$ based on $\left\|\boldsymbol{x}_j - \boldsymbol{c}_i\right\|_1 = \sum_{d=1}^{D} \left\|x_{dj} - c_{di}\right\|$, where D is the dimension of the data vector $\boldsymbol{x}_j$.
8) Each point creates its preference list based on the P matrix. For example a point from class m will create its preference list based on column m of the P matrix.
9) Find the best match using the CA algorithm.
10) Recompute the centroids and the P matrix based on the outcome of CA.
11) Evaluate the stopping criteria using Eq. 3.
12) While the stopping criteria are not satisfied, repeat steps 7–12.

To illustrate this algorithm, let us assume we have N data points and want to group them into three clusters as shown in Fig.1. The data points' labels are known. These labels could be any observed labels, such as the day of the week ($M = 7$). Moreover, we assume that the true number of clusters is three. The question we want to answer is how to partition the N data points such that similar data points in terms of distance and true labels are grouped together. By effectively partitioning the N data points, we can answer questions such as which days of the week have similar bike availability across the network.

In the first step, the proposed algorithm first randomly chooses three points as centroids for the three clusters. Then, it will partition the data points based on distance to get an estimate of the cardinality of each cluster and the P matrix. After that, each data point builds its preference list and each centroid builds its preference list, as shown in Fig. 1.

In the second step, the proposed algorithm, through a series of iterations, will try to find matches between clusters and data points and provide a stable match using the CA algorithm. At the end of this step, all points should be matched with one of the three clusters.

After successfully matching the point with clusters, the centroid and P matrix of the three clusters will be recalculated. The algorithm will repeat the entire process of building new preference lists, matching, and calculating new centroids and

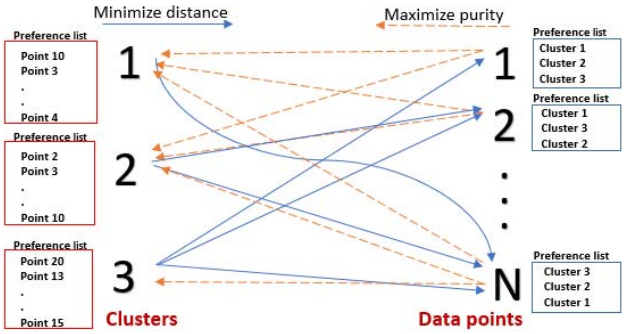

Fig. 1. CA based clustering.

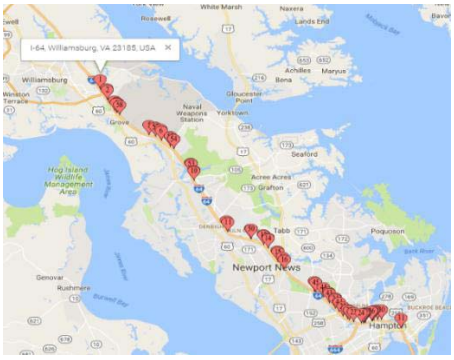

Fig. 2. Virginia study site (I-64) (source: Google Maps).

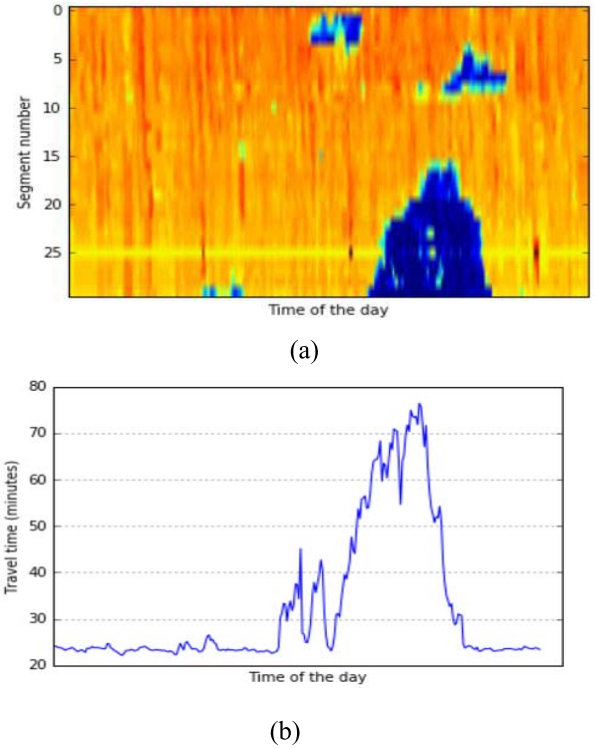

Fig. 3. Different views of traffic data (a) Visualization of the spatiotemporal traffic speed matrix where blue areas are congested. (b) Travel time vector corresponding to the spatiotemporal traffic speed matrix.

the P matrix. The algorithm will stop when there is no significant improvement in the purity and similarity.

VI. Datasets

To evaluate the proposed algorithm, we used two different datasets for two different transportation modes: a dataset containing the spatiotemporal traffic speed matrices collected on I-64 and a dataset containing station status from a BSS in the San Francisco Bay area. The following two subsection will briefly explain each dataset.

A. I-64 Speed and Instantaneous Travel Time Data

The selected I-64 eastbound study site was located between Hampton and Williamsburg in Virginia as shown in Fig. 2. There are 30 segments along the selected 26-mile stretch of freeway. The number of lanes in the selected stretch ranges from two to six, and the speed limit varies from 55 to 65 mph. This is the major commuting corridor between Richmond and Virginia Beach. INRIX probe data from 2013–2016 were used in this research effort, and data reduction was conducted to extract the daily traffic speed matrices. One-minute average speeds (or travel times) are available in the raw data for each roadway segment. Initially, the daily speed data were sorted by time and location from the raw data into a two-dimensional (spatiotemporal) matrix. In order to reduce the stochastic noise and measurement error, the speed matrices were aggregated by 5-minute intervals. The missing data in the aggregated speed matrices were estimated using the moving average for a 3×3 window.

Preprocessing the traffic data resulted in the creation of a spatiotemporal traffic speed matrix for each day, as shown in Fig. 3(a). In order to cluster speeds, we considered multivariate speed vectors as the inputs to the proposed clustering algorithm. The speed vectors had 30 dimensions, and each element was the speed at one segment at a certain time (i.e., one column of the spatiotemporal traffic speed matrix).

We also reduced the 2-D spatiotemporal traffic speed matrix into a 1-D vector by computing the instantaneous travel time. The instantaneous travel time method is simple, and assumes the segment speed does not change for the entire trip. The travel time calculation using the instantaneous approach is shown in equation (4).

$$\text{instantaneous travel time at } t_0 = \sum_{i=1}^{30} \frac{L_i}{v_i^{t0}} \tag{4}$$

where L_i is the length of segment i, v_i^{to} is the speed at segment i at the departure time t_0, and 30 is the total number of segments.

The travel time vector corresponding to the spatiotemporal traffic speed matrix is shown in Fig. 3(b). After converting the spatiotemporal traffic speed matrices to travel time vectors, the vectors were used as data input to the proposed algorithm. Each travel time vector had 288 dimensions, and each element was the instantaneous travel time at a certain time of the day.

B. BSS Station Status in the San Francisco Bay Area

We used docking station data collected from August 2013 to August 2015 in the San Francisco Bay area. The docking

station data include station ID, number of bikes available, number of docks available, and time of recording. The time data included year, month, day of month, day of week, hour, and minute at which the docking station data were recorded. As the station data were documented every minute for 70 stations in San Francisco over 2 years, it was necessary to reduce the size of the dataset by sampling station data once at every quarter-hour instead of once at every 1 minute and obtaining the exact values without any smoothing process. This was done to reduce the complexity of the data and take a global view of bike availability in the entire network every 15 minutes, with the goal of finding the similarity between these views and clustering them based on this similarity and recorded time. Similarity refers to bike availability in all stations, while recorded time refers to day of week and hour of day. We discarded other time attributes such as year, day of month, and minute in the analysis as they might not have a significant impact on bike availability.

During the data processing phase, we found that numerous stations had recently been added to the network and others had been terminated, making it necessary to clean the dataset by eliminating any entries missing docking station data. This reduced the number of entries from approximately 70,000 to 48,000. Each entry included the availability of bikes at the 70 stations with the associated time (day of week and hour of day). The availability of bikes represents the coordination measure for each entry, which is used in the k-median method to determine the entry closeness measure. This resulted in each entry constituting 70 dimensions (70 stations).

VII. Clustering Results and Discussion

In this section, we present the results of the aforementioned proposed algorithm using the I-64 traffic and BSS station status datasets. We first demonstrate the technique used to select the model order, and then we show the results for each dataset with respect to month of year, day of week, and time of day.

A. Model Order Selection—Consensus Clustering (CC)

Finding clustering for similar days of the week or similar hours of the day is not straightforward, as we do not know the natural grouping for day of week or hour of day (i.e., number of clusters). In cluster analysis, determining the number of clusters is called model order selection. In this research effort, we used a well-known model order selection technique called consensus clustering to determine the number of clusters [34]. This method looks for the model order that yields the most stable clustering solution. By stable clustering we mean that, given the model order, nearly the same paired data points are grouped together each time the clustering algorithm is run using different initial centroids (i.e., the centroids we begin the algorithm with) [35]. The CC method begins by assuming that the number of clusters is K, and then the dataset is clustered B times (using different initial centroids). A consensus matrix (CM) which is an $N \times N$ matrix (N is the number of the data points), is built for this model order K. This matrix identifies the number of times each two data points are grouped in the same cluster divided by B. Then the algorithm increases K by one and redoes the clustering and the consensus matrix for the new model order. The algorithm continues doing this until it has scanned the whole range of model orders required. At this point, the best model order is chosen visually by drawing the cumulative distribution function (CDF) of the CM at each model order against the consensus index $c_index \in [0, 1]$ (Eq. 5). The CDF for a particular CM is defined over the range [0, 1] as follows:

$$CDF\,(c\_index) = \frac{\sum_{i<j} 1\{CM(i,j) \le c\_index\}}{N(N-1)/2} \qquad (5)$$

where $1\{\ldots\}$ denotes the indicator function, $CM(i, j)$ denotes entry (i, j) of the consensus matrix CM, and N is the number of rows (and columns) of CM.

The outcome of the CDF is that for the correct model order the elements of the CM will only have zeros and ones. So we estimate the CDF for different model orders and choose the cleanest CM which has the flatter CDF. In other words, every CDF curve represents a different model order (number of clusters), and the flatter the curve, the more stable the model order. We applied this algorithm to the two aforementioned datasets to find the best number of clusters using the proposed

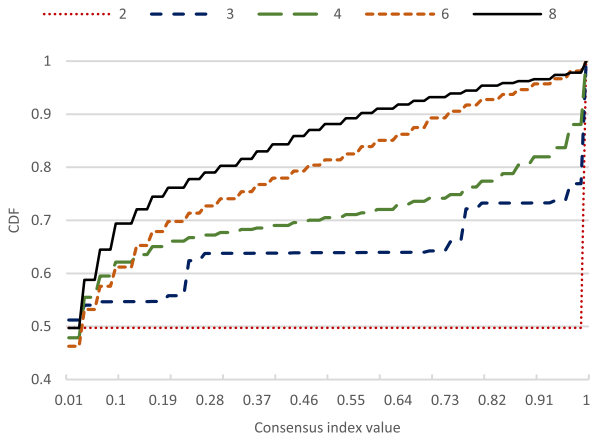

Fig. 4. CDF against consensus index value for each cluster – time of day using speed vectors from I-64.

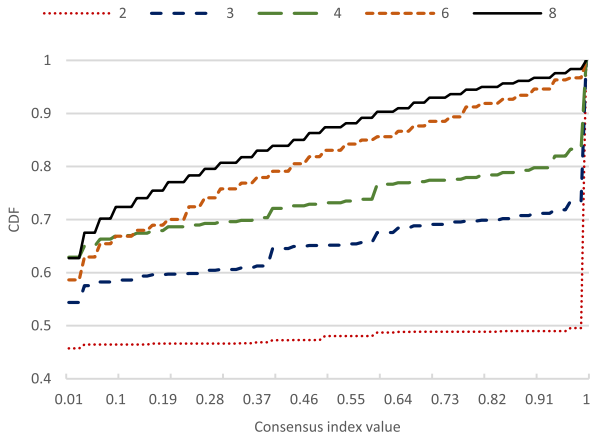

Fig. 5. CDF against consensus index value for each cluster – time of day using BSS station status data.

algorithm. To illustrate, in Fig. 4 and Fig. 5 we give two examples with regard to the time-of-day label for the two datasets. As the figure shows, the most stable model order for time of day for both datasets was determined to be $K = 2$. Consequently, we analyzed the data in more detail for $K = 2$. Similarly, the optimal number of clusters for day of week was $K = 2$ and $K = 3$ for the I-64 and BSS datasets respectively. The optimal number of clusters for the month of year was $K = 2$ for the I-64 dataset.

B. Instantaneous Travel Time Clustering Results

In order to demonstrate that traditional clustering analysis cannot provide a satisfactory answer to our aforementioned research questions, we clustered the I-64 travel time vectors using the k-means and hierarchical clustering techniques. Accordingly, we clustered travel time vectors blindly (without considering the natural label day of week), and then found the distribution of days of the week inside each cluster in order to determine which clusters dominated on certain days. However, the results of the k-means and hierarchical clustering did not show any obvious grouping of similar days. For example, in Fig. 6(a), the left panel shows that the first centroid (cluster) for k-means has members of all days with almost the same percentage except Fridays. The right panel of Fig. 6(a) shows that the second centroid (cluster) has members of all days but has a high percentage of Fridays and Thursdays. The same thing applies to the hierarchical technique in Fig. 6(b). This means that these clusters are not homogeneous, and thus we cannot identify a grouping for a particular day. Therefore, both the k-means and hierarchical clustering algorithms did not help us determine which days were similar.

That is due to the fact that k-means and hierarchical clustering algorithms try to cluster the observations using only distance as the parameter, and thus neglect the existence of the natural label. These results make it difficult to come up with any meaningful conclusion that would be helpful for operating agencies.

On the other hand, when we used the proposed algorithm to cluster the same dataset and use the month-of-year label, the optimal number of clusters is $K = 2$, and the results are given in Fig. 7(a). The months are grouped into two clusters: summer (May, June, July, and August), and the rest of the months of the year in the other cluster. The associated pattern of travel time for these two clusters is given in Fig. 7(b). The results can help the operators distinguish between the travel times.

The results look more homogeneous clusters with respect to the chosen time event, thus providing a basis for a better temporal plan for operations planning purposes with regard to months of the year. Specifically, the results reveal how vehicle travel times vary over the months and what the expected travel time is every hour for each month. The variance in travel time patterns for the two clusters offers insight into the worst and best travel time scenarios for every hour and every month. Generally, the longer travel time is, the more congestion happens at this particular time. One potential mitigation would be to use traffic control strategies to prevent the onset of traffic congestion.

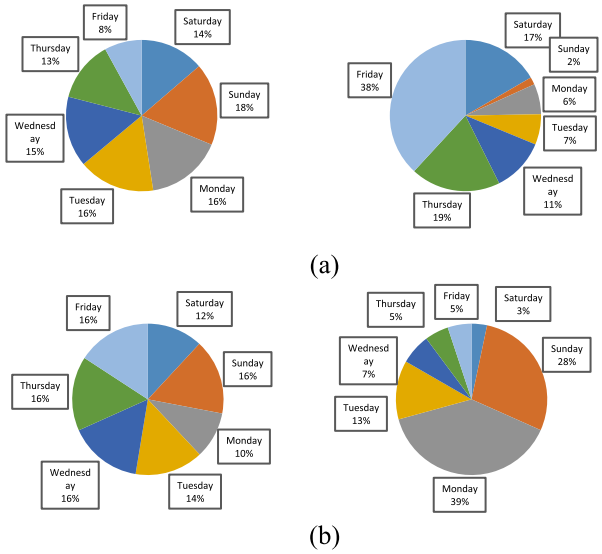

Fig. 6. Distribution of days of the week within each of the two clusters using the travel time vectors (a) k-means and (b) hierarchical algorithms.

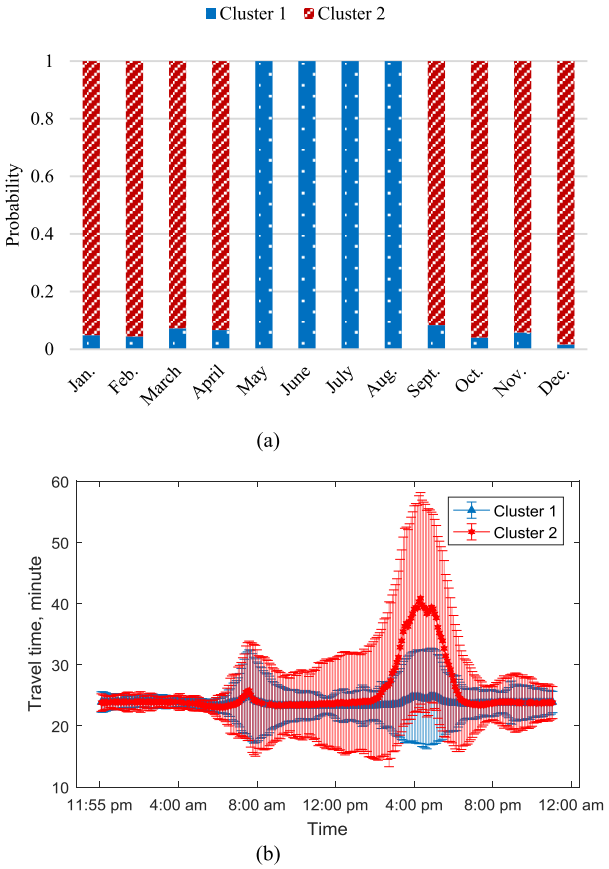

Fig. 7. Month of year clustering results (a) Probability of month being in one of the two clusters ($K = 2$). (b) Variance and average travel time of vehicles in the two clusters at different times.

In order to find the natural grouping for day of week, we applied the proposed algorithm to the same data and used day of week as the label. We set $K = 2$, which is the optimal number of clusters found using the CC method. Analysis of the travel time data reveals that the two clusters are (1)

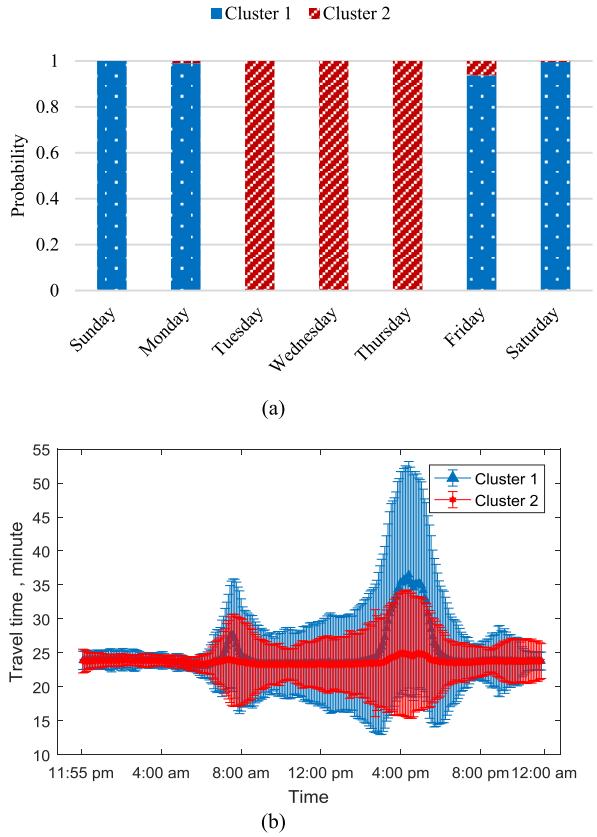

Fig. 8. Day of week clustering results (a) Probability of day of week being in one of the two clusters ($K = 2$). (b) Variance and average travel time of vehicles in the two clusters at different times.

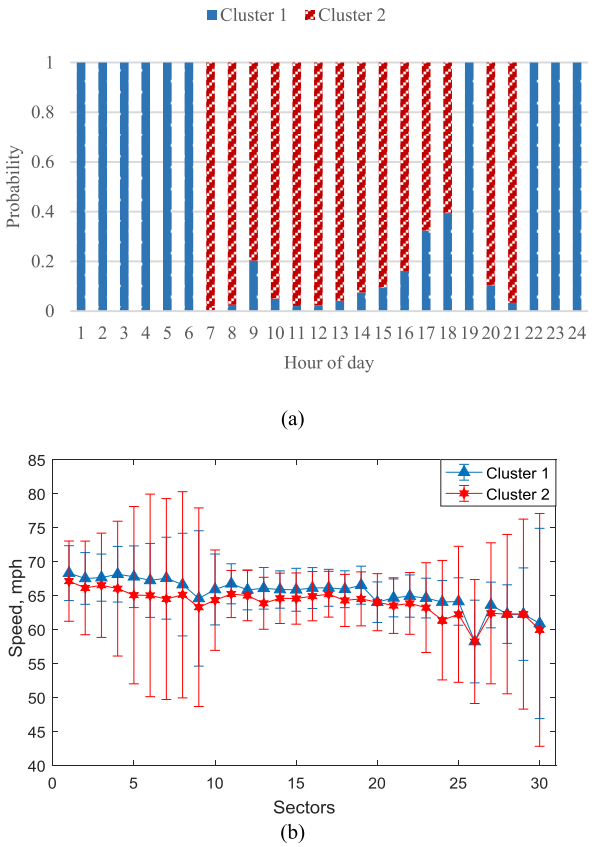

Fig. 9. Time of day clustering result. (a) Probability of hour being in one of the two clusters ($K = 2$). (b) Cluster's centroid and variance of the two clusters at different sectors at I-64.

Wednesday, Thursday and Friday (cluster 1), and (2) the rest of the weekdays (cluster 2). The results of the clustering are given in Fig. 8(a) and (b), which shows the probability of a day of the week being in one of the two clusters and the pattern of each cluster.

As shown previously, the clustering results look more homogeneous with respect to the chosen time events, month of year and day of week, providing the basis for a better temporal plan for operations planning purposes with regard to either months or days. Specifically, the results reveal how the travel times of vehicles vary over months or the days of the week and what the expected travel time is every hour for each month and day of the week. The variance in travel time patterns for the clusters demonstrates when the worst and best travel time scenarios might happen for every hour, month, and day of the week. Generally, the longer the travel time is, the greater the congestion that happens at this particular time. One potential technique that could be utilized is to use traffic control strategies to regulate the flow of traffic prior to the onset of congestion to prevent or delay the anticipated traffic congestion.

C. Speed Vectors Clustering Results

Unlike the data clustering described in the previous section, we clustered the speed vectors with respect to the time-of-day label. After determining the optimal number of clusters ($K = 2$) using CC, we explored how the data points were partitioned between the two clusters. The results of the two clusters are presented in Fig. 9(a), which shows the probability of each hour being in one of the two clusters. Each cluster is associated with a pattern for the expected speed of vehicles for each sector as provided in Fig. 9(b).

The two clusters are dominated by peak and non-peak hours: (1) 7:00 a.m. to 6:00 p.m. and 8:00 p.m. to 10:00 p.m. are in the first cluster, and (2) 10:00 p.m. to 6:00 a.m. and 7:00 p.m. are in the second cluster. This pattern differs from previous research showing that peak hours are from 8:00 a.m. to 6:00 p.m. and non-peak hours are the rest of the day [36]. Unlike most of the previous research, our research shows that the hours of 8:00 p.m. to 10:00 p.m. belong to the peak hours. This would make sense, as people often go home after work, rest for a few hours or so, then go back out for shopping, dinner, etc.

It should be noted that the pattern represents the centroid of the cluster (the median of the cluster) so that the exact values of the pattern are not shown here. The pattern can serve as an indication of when the speed will drop at a specific location (i.e., sector). Two observations can be made from Fig. 9(b). First, the trends of the two clusters generally follow each other, which might be linked to the geometric design of the highway (i.e., number of lanes, grade, etc.). Second, the variance of the pattern of the second cluster is greater

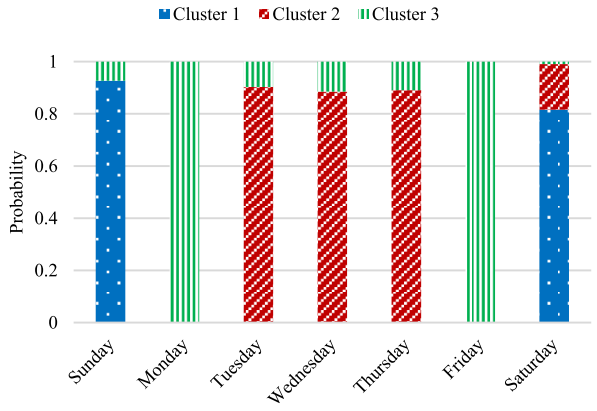

Fig. 10. The probability of the day of week being in one of the three clusters ($K = 3$).

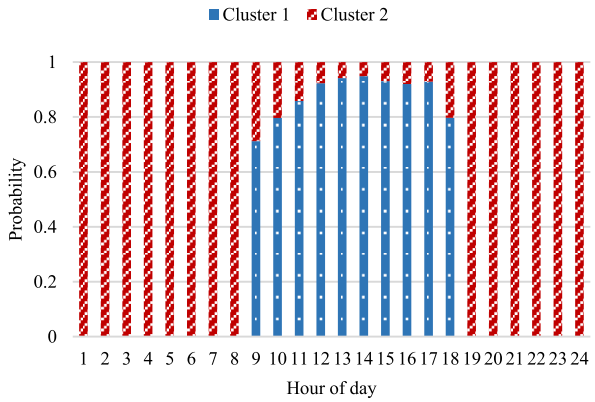

Fig. 12. Probability of hour being in one of the two clusters ($K = 2$).

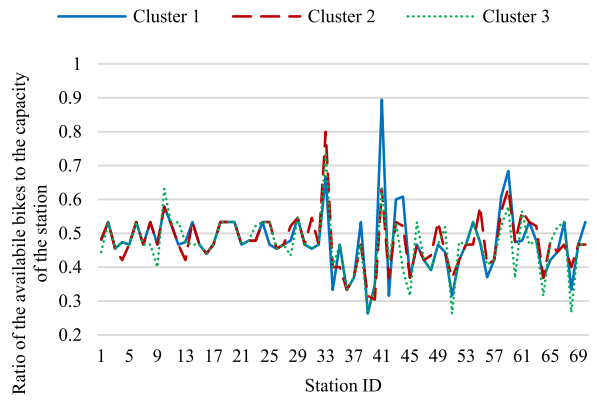

Fig. 11. The ratio of the available bikes to station capacity for the three clusters at station in the network.

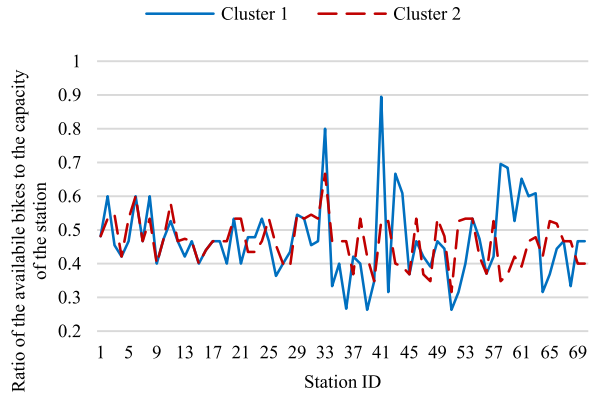

Fig. 13. Available bikes of the two clusters for each station in the network.

than the first cluster. That makes sense, as non-peak hours tend to be less deterministic and have more randomness than peak hours.

Similar to the month-of-year and day-of-week clustering results, we can see that the clusters are time homogeneous and thus can be used by operators to implement traffic operations plans to manage I-64 temporally and spatially. For instance, the sections with high variance (e.g., section 9) would be promising candidates for improvement. One potential improvement is controlling merging and diverging traffic.

D. Bike Station Status Clustering Results

First, we clustered the bike station data using the day-of-week label, and the optimal number of clusters found using the CC method was $K = 3$. The results of the three clusters are presented in Fig. 10, which shows the probability of each day being in one of the three clusters. The three clusters are dominated by specific days: (1) Saturdays and Sundays, (2) Mondays and Fridays, and (3) finally Tuesday, Wednesday, and Thursday. This pattern differs from previous research [37] that showed bike patterns grouped into two clusters (weekend and weekdays). Our research shows that the weekdays can be split into groups: (a) Mondays and Fridays, and (b) Tuesdays, Wednesdays, and Thursdays. This appears to be logical, as the beginning and the end of the week are different from the rest of the weekdays.

Each cluster is associated with a pattern for the availability of bikes at each station. The patterns of the ratio of the available bikes to the station capacity for the three clusters are provided in Fig. 11.

Three observations can be made from Fig. 11. First, the three patterns of the three clusters generally follow the pattern of the stations' capacity, which could be the result of system operators' rebalancing efforts. Second, the patterns of the three clusters show fluctuations in the bike activities; none of the days of the week has the highest activity for the entire network, which depends on both spatial and temporal factors. Third, several stations appear more likely to be empty or full on either weekdays or weekends. The difference in demand between the three clusters appears clearly for some stations, but not others. For example, the bike activities for cluster 1 (Tuesday, Wednesday, and Thursday) and cluster 3 (Saturday and Sunday) are similar for some stations in the network. That can be seen in stations 58 and 59 (San Francisco Caltrain 2–330 Townsend and San Francisco Caltrain–Townside at 4th). When taking a closer look at the location of these two stations, we found that they are located close to the Caltrain station. Accordingly, the similarity between these two clusters can be linked to the train timetable.

Second, we clustered the bike sharing data using the hour-of-the-day label to find the hours of the day that have similar patterns. Only the station data at the beginning of each hour were considered. The optimal number of clusters was found to be two ($K = 2$). The analysis of the data reveals that the two clusters are peak (cluster 2) and non-peak (cluster 1) hours, confirming previous research. The results of the clustering are shown in Fig. 12 and Fig. 13, which give the probability of an hour being in one of the two clusters and the pattern of each cluster. It can be concluded from Fig. 13 that when the patterns of the two clusters are lined up, the bike activity in the peak and non-peak hours is the same.

Generally, both clustering results for day of week and time of day are time homogeneous, making it possible for operators of the BSS to manage the bike stations and propose temporal and spatial plans. The clustering results give them a general view of the status of stations and clarify where the imbalances would happen with respect to time of day and day of week, leading to better monitoring of the system as a whole.

VIII. Conclusion

The paper describes the development of a useful tool for agencies and researchers to cluster similar transportation patterns with respect to time-based events. A new supervised clustering algorithm was proposed to benefit from the background knowledge of the dataset along with similarity. Unlike other similar supervised clustering algorithms, the proposed algorithm is scalable given that it involves low computational times. It takes advantage of the natural labeling of the data (i.e., day of week, time of day) and models the clustering problem as a cooperative game and simultaneously clusters and identifies the stable number of clusters.

The algorithm was tested on two different datasets, namely a travel time dataset along a section of I-64 in Virginia and BSS station status data from the San Francisco Bay area. Three types of background knowledge were used: month of year, day of week, and hour of day. The proposed algorithm was run on the two datasets and produced more meaningful clusters considering the background knowledge. The resultant clusters appear to be more time homogenous, giving the potential for operators to better manage the transportation modes per time event. Specifically, the algorithm provides insight for the clusters that operators can use to anticipate and plan for congestion on I-64 and imbalances in the BSS.

We have shown that the proposed algorithm outperforms the classical k-means clustering algorithm, which did not reveal any obvious grouping of similar days. We proved that the proposed algorithm produced better results than these classic clustering algorithms when applied to any labeled dataset with respect to a time event.

References

[1] P. Arbeláez, M. Maire, C. Fowlkes, and J. Malik, "Contour detection and hierarchical image segmentation," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 33, no. 5, pp. 898–916, May 2011.

[2] J. A. Roberts, "Profiling levels of socially responsible consumer behavior: A cluster analytic approach and its implications for marketing," *J. Marketing Theory Pract.*, vol. 3, no. 4, pp. 97–117, 1995.

[3] E. W. T. Ngai, Y. Hu, Y. H. Wong, Y. Chen, and X. Sun, "The application of data mining techniques in financial fraud detection: A classification framework and an academic review of literature," *Decis. Support Syst.*, vol. 50, no. 3, pp. 559–569, 2011.

[4] S. Thiprungsri and M. A. Vasarhelyi, "Cluster analysis for anomaly detection in accounting data: An audit approach," *Int. J. Digit. Accounting Res.*, vol. 11, pp. 69–84, Jul. 2011.

[5] W. Weijermars and E. van Berkum, "Analyzing highway flow patterns using cluster analysis," in *Proc. IEEE Intell. Transp. Syst.*, Sep. 2005, pp. 308–313.

[6] M. Elhenawy, H. Chen, and H. A. Rakha, "Dynamic travel time prediction using data clustering and genetic programming," *Transp. Res. C, Emerg. Technol.*, vol. 42, pp. 82–98, May 2014.

[7] C. T. Calafate, D. Soler, J.-C. Cano, and P. Manzoni, "Traffic management as a service: The traffic flow pattern classification problem," *Math. Problems Eng.*, vol. 2015, p. 14, 2015.

[8] J. E. Froehlich, J. Neumann, and N. Oliver, "Sensing and predicting the pulse of the city through shared bicycling," in *Proc. IJCAI*, 2009, pp. 1420–1426.

[9] P. Vogel, T. Greiser, and D. C. Mattfeld, "Understanding bike-sharing systems using data mining: Exploring activity patterns," *Procedia-Social Behav. Sci.*, vol. 20, pp. 514–523, Jan. 2011.

[10] C. Etienne and O. Latifa, "Model-based count series clustering for bike sharing system usage mining: A case study with the Vélib'system of Paris," *ACM Trans. Intell. Syst. Technol.*, vol. 5, no. 3, p. 39, 2014.

[11] C. F. Eick, N. Zeidat, and Z. Zhao, "Supervised clustering—Algorithms and benefits," in *Proc. 16th IEEE Int. Conf. Tools Artif. Intell.*, Nov. 2004, pp. 774–776.

[12] A. Bar-Hillel, T. Hertz, N. Shental, and D. Weinshall, "Learning distance functions using equivalence relations," in *Proc. 20th Int. Conf. Mach. Learn. (ICML)*, 2003, pp. 1–8.

[13] J. Sinkkonen, S. Kaski, and J. Nikkilä, "Discriminative clustering: Optimal contingency tables by learning metrics," in *Proc. Eur. Conf. Mach. Learn.* Helsinki, Finland, Springer, 2002, pp. 418–430.

[14] V. Spinelli, "Supervised box clustering," *Adv. Data Anal. Classification*, vol. 11, no. 1, pp. 179–204, 2017.

[15] D. Gale and L. S. Shapley, "College admissions and the stability of marriage," *Amer. Math. Monthly*, vol. 69, no. 1, pp. 9–15, 1962.

[16] M. Elhenawy and H. A. Rakha, "Automatic congestion identification with two-component mixture models," *Transp. Res. Rec., J. Transp. Res. Board*, vol. 2489, pp. 11–19, May 2015.

[17] M. Elhenawy and H. Rakha, "Applying cluster analysis techniques to traffic operations," Virginia Dept. Transp., Richmond, VA, USA, Tech. Rep. 00124, 2017, p. 59.

[18] U. Demiryurek, B. Pan, F. Banaei-Kashani, and C. Shahabi, "Towards modeling the traffic data on road networks," in *Proc. 2nd Int. Workshop Comput. Transp. Sci.* 2009, pp. 13–18.

[19] D. Xu and Y. Tian, "A comprehensive survey of clustering algorithms," *Ann. Data Sci.*, vol. 2, no. 2, pp. 165–193, 2015.

[20] T. Finley and T. Joachims, "Supervised clustering with support vector machines," in *Proc. 22nd Int. Conf. Mach. Learn.*, 2005, pp. 217–224.

[21] S. Basu, A. Banerjee, and R. Mooney, "Semi-supervised clustering by seeding," in *Proc. 19th Int. Conf. Mach. Learn.*, Sydney, NSW, Australia, 2002, pp. 19–26.

[22] S. Basu, M. Bilenko, and R. J. Mooney, "Comparing and unifying search-based and similarity-based approaches to semi-supervised clustering," in *Proc. ICML-Workshop Continuum Labeled Unlabeled Data Mach. Learn. Data Mining*, 2003, pp. 42–49.

[23] J. A. Hartigan and M. A. Wong, "Algorithm AS 136: A k-means clustering algorithm," *J. Roy. Stat. Soc. Ser. C (Appl. Statist.)*, vol. 28, no. 1, pp. 100–108, 1979.

[24] S. C. Johnson, "Hierarchical clustering schemes," *Psychometrika*, vol. 32, no. 3, pp. 241–254, 1967.

[25] B. Schölkopf, A. Smola, and K.-R. Müller, "Nonlinear component analysis as a kernel eigenvalue problem," *Neural Comput.*, vol. 10, no. 5, pp. 1299–1319, Jul. 1998.

[26] D. MacDonald and C. Fyfe, "The kernel self-organising map," in *Proc. 4th Int. Conf. Knowl.-Based Intell. Eng. Syst. Allied Technol.*, Aug./Sep. 2000, pp. 317–320.

[27] Z.-D. Wu, W.-X. Xie, and J.-P. Yu, "Fuzzy c-means clustering algorithm based on kernel method," in *Proc. 5th Int. Conf. Comput. Intell. Multimedia Appl. (ICCIMA)*, 2003, pp. 49–54.

[28] P. Awasthi and R. B. Zadeh, "Supervised clustering," in *Proc. Adv. Neural Inf. Process. Syst.*, 2010, pp. 91–99.

[29] H. Daumé, III, and D. Marcu, "A Bayesian model for supervised clustering with the Dirichlet process prior," *J. Mach. Learn. Res.*, vol. 6, pp. 1551–1577, Sep. 2005.
[30] G. Forestier, P. Gançarski, and C. Wemmert, "Collaborative clustering with background knowledge," *Data Knowl. Eng.*, vol. 69, no. 2, pp. 211–228, 2010.
[31] N. Chen *et al.*, "An evolutionary algorithm with double-level archives for multiobjective optimization," *IEEE Trans. Cybern.*, vol. 45, no. 9, pp. 1851–1863, Sep. 2015.
[32] M. H. C. Law, A. P. Topchy, and A. K. Jain, "Multiobjective data clustering," in *Proc. IEEE Comput. Soc. Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun./Jul. 2004, p. 2.
[33] J. Handl and J. Knowles, "An evolutionary approach to multiobjective clustering," *IEEE Trans. Evol. Comput.*, vol. 11, no. 1, pp. 56–76, Feb. 2007.
[34] S. Monti, P. Tamayo, J. Mesirov, and T. Golub, "Consensus clustering: A resampling-based method for class discovery and visualization of gene expression microarray data," *Mach. Learn.*, vol. 52, nos. 1–2, pp. 91–118, 2003.
[35] Y. Şenbabaoğlu, G. Michailidis, and J. Z. Li, "Critical limitations of consensus clustering in class discovery," *Sci. Rep.*, vol. 4, p. 6207, Aug. 2014.
[36] S. I.-J. Chien and C. M. Kuchipudi, "Dynamic travel time prediction with real-time and historic data," *J. Transp. Eng.*, vol. 129, no. 6, pp. 608–616, 2003.
[37] A. Kaltenbrunner, R. Meza, J. Grivolla, J. Codina, and R. Banchs, "Urban cycles and mobility patterns: Exploring and predicting trends in a bicycle-based public transport system," *Pervasive Mobile Comput.*, vol. 6, no. 4, pp. 455–466, 2010.

Mohammed H. Almannaa received the B.Sc. degree *(magna cum laude)* in civil engineering from King Saud University, Riyadh, Saudi Arabia, in 2012, and the M.Sc. degree in civil engineering from Virginia Tech, Blacksburg, VA, USA, in 2016, where he is currently pursuing the Ph.D. degree in civil engineering. He is currently a Teaching Assistant with the Civil Engineering Department, King Saud University. His research interests include, but are not limited to, eco-driving, highway transportation safety, intelligent transportation systems, and bike sharing systems.

Mohammed Elhenawy received the Ph.D. degree in computer engineering from Virginia Tech (VT), Blacksburg, VA. He was a Post-Doctoral Researcher with the VT Transportation Institute, for three years. He is currently a Research Fellow with the Center for Accident Research and Road Safety, Queensland University of Technology. He has authored or co-authored over 37 intelligent transportation system (ITS) related papers. His research interests include machine learning, statistical learning, and game theory and their application in ITSs and cooperative ITSs.

Hesham A. Rakha (M'04–SM'18) received the B.Sc. degree (Hons.) in civil engineering from Cairo University, Cairo, Egypt, in 1987, and the M.Sc. and Ph.D. degrees in civil and environmental engineering from Queen's University, Kingston, ON, Canada, in 1990 and 1993, respectively. He is currently the Samuel Reynolds Pritchard Professor of engineering with the Charles E. Via, Jr. Department of Civil and Environmental Engineering, Virginia Tech, Blacksburg, where he is also a Courtesy Professor with the Bradley Department of Electrical and Computer Engineering. He is the Director of the Center for Sustainable Mobility, Virginia Tech Transportation Institute. His areas of research are traffic flow theory, traffic modeling and simulation, traveler and driver behavior modeling, artificial intelligence, dynamic traffic assignment, traffic control, energy and environmental modeling, and safety modeling. He is a member of the IEEE, the ITE, the ASCE, the SAE, and the TRB. He is a Professional Engineer in ON, Canada.