MovCloud: A Cloud-enabled Framework to Analyse Movement Behaviors

Shreya Ghosh*, Soumya K Ghosh†, Rajkumar Buyya‡
*†Department of Computer Science and Engineering, Indian Institute of Technology Kharagpur, India
‡Cloud Computing and Distributed Systems (CLOUDS) Lab, School of Computing and Information Systems
The University of Melbourne, Australia
Email: *shreya.cst@gmail.com, †skg@cse.iitkgp.ac.in, ‡ rbuyya@unimelb.edu.au

***Abstract*—Understanding human interests and intents from movement data are fundamental challenges for any location-based service. With the pervasiveness of sensor embedded smartphones and wireless networks and communication, the availability of spatio-temporal mobility trace (timestamped location information) is increasingly growing. Analysing these huge amount of mobility data is another major concern. This paper proposes a cloud-based framework named *MovCloud* to efficiently manage and analyse mobility data. Specifically, the framework presents a hierarchical indexing schema to store trajectory data in different spatio-temporal resolution, clusters the trajectories based on semantic movement behaviour instead of only raw latitude, longitude point and resolves mobility queries using MapReduce paradigm. MovCloud is implemented over Google Cloud Platform (GCP) and an extensive set of experiments on real-life data yield the effectiveness of the proposed framework. MovCloud has achieved $\sim 28\%$ better clustering accuracy and also executed three times faster than the baseline methods.**

***Index Terms*—Trajectory, Clustering, MapReduce, Cloud Computing, Deep Learning**

I. Introduction

With the increasing use of mobile-devices and advancement of location acquisition technologies and sensor networks, a huge amount of location traces are accumulated from varied moving agents such as people on the move, private vehicles and public transportation. These time-stamped sequences of latitude and longitude information are depicted as *trajectory trace*. The trajectories generated from the movement of agents provide unprecedented opportunity to discover implied knowledge and fosters several location-based services, namely, traffic resource management, ride-sharing services [1], next location prediction [2] and categorizing individuals from mobility traces [3]. However, clustering mobility patterns from the huge amount of historical traces is a challenging issue.

Interpreting human movement behaviour is pivotal for efficient urban planning and resource management. Mobility trace analysis not only provides information about the crowd-flow between varied places in different temporal-scales, it has a significant role in mapping *the intent of the move* [4]. For instance, the *intent* of a trip can be *commuting to workplace* or a *leisure travel*. This analysis helps in location-based advertising or even facilitates an effective transportation resource planning. Trajectory clustering is the primary method to group similar movement patterns in a cluster [5] and trajectory clustering has a wide range of applications. It has an ubiquitous applicability in traffic monitoring, next location prediction and activity analysis, since clustering helps to reduce the storage and computing time of any pattern mining task. Although, there are various methods [6],[7] to cluster similar movement patterns, but all of the existing works mainly focus on similarity measurements using raw spatial and/or temporal features (speed, distance, time etc.) of a trajectory. In this paper, our major goal is to cluster the trajectories that represent similar moving *behaviors* (semantic meaning) instead of only spatio-temporal features. For example, commuting from home to work is a moving behavior. Other examples include commuting for leisure travel, business travel, shopping or commuting for medical help. It may be noted that two trajectories having different spatial and temporal scales can represent the same moving behavior. For example, the commuting time from home to workplace may be fifteen minutes for one individual and for other person, it takes one hour. Clustering such moving behaviours do not work well with conventional supervised learning with hand-crafted feature set since human movement behaviours are diverse in nature. To achieve this, MovCloud deploys a novel deep neural network to cluster such movement trajectories.

Another concern in analysing mobility traces is the enlarging volume of time-series data, which makes the clustering a challenging task. The mobility information management task is non-trivial in a standalone database, due to this huge volume of moving data and the constant updates. A survey of big data analytics in cloud paradigm is presented in [8], however, the existing approaches fall short to store, manage and analyse spatio-temporal data. These necessitates a cloud-based framework to effectively manage and explore mobility traces in a timely manner.

Contributions: In this direction, the paper aims to build an end-to-end cloud-based framework which is capable to cluster the mobility pattern of a region and efficiently resolve the mobility-based queries. The major contributions of the paper can be summarized as:

- We propose a novel deep learning based trajectory clustering algorithm to group similar moving behaviours.
- A novel indexing and query processing is proposed which is executed in a distributive manner in the cloud.
- The system is also scalable to incorporate huge query-load in a given time-period.

- The framework is evaluated using real-life mobility traces and encouraging results have been found.

The rest of the paper is organized as follows. Section II presents a review of recent studies in this direction. A few preliminary concepts of the work and different modules of the framework are presented in section III. Section IV depicts the experimental evaluation and, finally, section V concludes the paper with future research directions.

II. Related Work

Trajectory clustering has widely attracted research attention in the data mining community [6], [7]. There are several distance or density based clustering algorithms which measure the similarity among the trajectory segments: *EDR (Edit Distance on Real sequence)*, *DTW (Dynamic Time Warping)* and *LCSS (Longest Common Subsequences)* [6]. The partition-and-group framework is proposed in [5] to discover common trajectories. The authors propose the framework namely, *TRACLUS*, a widely referred trajectory clustering approach, using the minimum description length (MDL) and a density-based line-segment clustering method. Hung *et al.* [9] present a trajectory aggregation algorithm by analysing spatially and temporally co-located data points. A novel temporal-constrained sub-trajectory cluster analysis is presented in [10], where the authors propose a novel indexing structure: *Representative Trajectory Tree*. There are also research advances to deploy deep learning architecture in trajectory data mining tasks. Yao *et al.* [11] propose a novel trajectory clustering approach by deep representation learning. The authors present a sequence to sequence auto-encoder to capture the movement behaviour sequences. A RNN based model (coupling the check-ins) called *TULER* is designed to capture the dependency of check-ins and to infer the latent patterns of users in [12]. However, no existing work focuses on clustering trajectory segments based on similar movement *behaviour* or *intent* of the moving agent.

Table I: Comparisons of existing works and MovCloud framework

Feature	Related Works			MovCloud
	[5], [9], [10]	[13]	[14], [15]	
Trajectory Clustering	✓	✘	✘	✓
Trajectory Clustering (semantics)	✘	✘	✘	✓
Indexing	✓	✘	✘	✓
Query-Processing	✘	✓	✘	✓
Scalability	✘	✘	✓	✓
Geo-Spatial Cloud	✘	✓	✘	✓

There are few works on *geo-spatial cloud* and approaches to speed up the computations. Shashi Shekhar *et al.* [16] present the emerging challenges to analyse spatial big data and how cloud paradigm is used to solve them. There are several spatial data processing systems, namely, Hadoop-GIS [14] and SpatialHadoop [17] which utilize *MapReduce* [15] paradigm for power and cost effectiveness. Sijie Ruan *et al.* [18] present a cloud-based trajectory data pre-processing framework based on Spark. *Traj-Cloud* is presented in [13], where the framework provides map-matching and query-processing services. Li *et al.* [19] present an efficient framework to resolve path-query from massive trajectory database on cloud. There are several interesting applications of analysing human movement history and activity-patterns [20]. For instance, optimal route prediction [21], finding correlations of mobility patterns and other contexts are studied in [22], [23], where authors observe the correlations between movement history and academic performances of students.

Table I summarizes the features of other existing works and MovCloud. To the best of our knowledge, there is no existing work which clusters trajectory segments by analysing the semantic behaviour of the moving agents. Furthermore, MovCloud presents novel MapReduce based query processing and trajectory clustering approaches.

III. MovCloud Framework

Fig. 1 illustrates the proposed framework, MovCloud, consisting three major modules. In the pre-processing step, the raw GPS log is associated with the semantic (geo-tagged) information. In the next step, the trajectory is segmented and stored followed by building a trajectory-index. Finally, the clustering is carried out and the query mining module is presented. Few preliminary terms are defined as follows:

- Road Network (R): Road network of a region is represented by a directed graph $R = (V_R, E_R)$ where V_R represents all intersecting points of the road-segments, i.e., either starting or ending points of each such road-fragment and E_R is the set of edges or roads in the map.

$$E_R = \{v_{R_i} \xrightarrow{v_{R_j}} |(v_{R_i}, v_{R_j}) \in V_R\}$$

- GPS Trajectory (G): The location information as $p :< latitude, longitude >$ pairs in chronologically ordered time-series is termed as GPS trajectory or Trajectory Trace.

$$G(p_1, p_2, \ldots, p_n) = \{(lat_1, lon_1, t_1) \longrightarrow (lat_2, lon_2, t_2) \longrightarrow \cdots \longrightarrow (lat_n, lon_n, t_n)\} \quad (1)$$

  where $(lat_i, lon_i) \subseteq R^2$, $t_i \subseteq R$ and $t_1 < t_2 \ldots t_n$. The trajectory is formed by connecting the location information on increasing time-ordering.
- POI-taxonomy: POI or *Point-of-interest* refers to a specific-type of landuse associated with a location, which typically depicts the social functional region of a place such as *residential building*, *academic premise*, *entertainment area*. Several apis namely *Google Place API*[1] and *Foursquare*[2] provide such place information to associate raw GPS trace with their POI-tags. In this work, the POIs of the region-of-interest are classified in a tree-structured taxonomy where the leaf nodes represent the

[1]Google Place API: https://developers.google.com/maps/documentation/javascript/examples/geocoding-reverse

[2]Foursquare: https://developer.foursquare.com/places-api

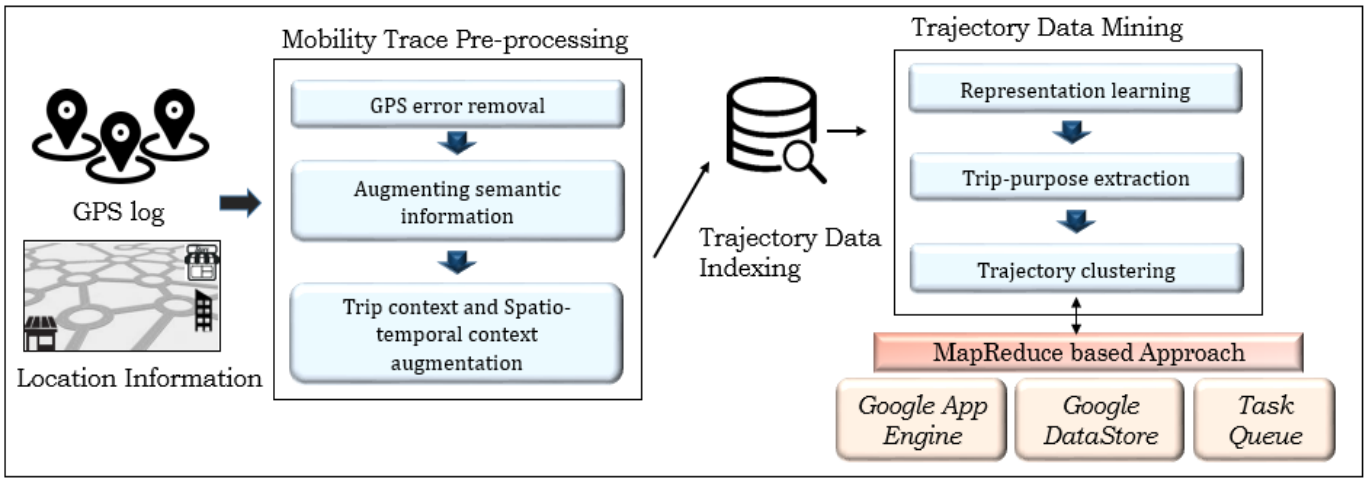

Figure 1: Workflow of MovCloud framework

specific POI-information and nodes at the top represent more generic POI information.

- Stay-point (S) and Trajectory Segment ($Traj\_Seg$): The stay-point depicts set of GPS-points which are spatially close and the time-differences of the visits are within a temporal threshold. Typically, stay-points of a trace represent that user has stopped in these locations and spent a specific amount of time-duration (T). Stay-point ($S(B, Pl, T)$) is represented by the *bounding box* (B), POI-information and T time-duration. The bounding-box (*polygon*, spatial data-type [24]) encloses the area covered by the user in her T stay-duration.
  The trajectory segment is defined as:

$$Traj\_Seg = \{S_i(B_i, Pl_i, T_i) \longrightarrow (p_1, t_1) \longrightarrow \dots \longrightarrow (p_n, t_n) \longrightarrow S_j(B_j, Pl_j, T_j)\} \quad (2)$$

  where B_i and B_j are two stay-points. It may be noted that each $Traj\_Seg$ consists of two stay-points and the GPS log between start and end stay-points.

- Movement Behaviour: This term holds the semantic meaning or *intent* of a trip. In this work, we consider 48 such trip-purposes (few trip-purposes are mentioned in section IV-A).

The problem definition of this work can be summarized as:

(i) Given the GPS trajectories $\varkappa = \{G_1, G_2, \dots, G_n\}$, generate the clusters of trajectories with similar movement behaviour.

(ii) Resolve mobility-based queries efficiently in timely manner.

A. Mobility Trace Pre-processing

In the pre-processing step, the error removal is carried out. It is checked such that all the GPS points are strictly ordered on increasing time-stamp. The duplicate points with same time-stamp are removed.

In the GPS data cleaning step, two types of GPS errors are observed: (i) large scale error or outlier, i.e., completely different position from the actual location due to low number of satellites in view or device error, and (ii) small or random errors. The large GPS errors are removed by deploying *Kalman Filtering technique* [25], where first the error is modeled by adding random Gaussian noise to the actual GPS points. The noise vector is drawn from Gaussian probability distribution of mean difference between actual GPS point and deviated point. Furthermore, speed-based GPS error detection is also implemented, where the deviation between the speed-limit and speed at previous GPS location is estimated and the points outside the threshold are filtered. It may be noted, that among 759328 trajectory-segments used in the study, there are 52356 large scale errors, and the filtering technique is able to modify 45034 such instances. The percentage of random errors are quite low (approx $\sim 2\%$) in the dataset and largely removed in the Mapmatching process.

The trajectory is represented by episodes of *stop* (stay-point) and *move*. Using density-based clustering and distance, time thresholds, different possible *stay-points* are extracted. In the next step, geo-tagged information for each *stop* points are extracted and appended to enrich the semantic information of the raw traces. For tagging landuse information, *iterative reverse geocoding*(IRG) is used. Additionally, we build the POI-taxonomy where *<latitude, longitude, POI>* are stored for any subsequent computation. Finally, for semantic enrichment, the road network structure of the region is extracted from OSM[3]. The features related to underlying road-structure (such as, length, width etc. of the road-segments), the connectivity and continuity (like, intersection of roads) are appended in this step. Here, we have used the map-matching algorithm named *AntMapper* [26] which considers both topological information and global similarity measurement.

The most significant contexts are *time* and *geotagged location* of the trajectory trips to infer the movement behaviour. Although, the collected raw GPS traces do not have these semantic information, however, we append semantic informa-

[3]OpenstreetMap: https://www.openstreetmap.org/

tion such as nearest POIs of the start and destination of any trip. The intuition is that individual visit specific places for some activity or intent. Furthermore, the movement patterns of individuals exhibit high level of spatio-temporal regularity. The *day of the week*, *stay-duration*, *POI-type*, *timestamp* of the visit directly influence the semantic-label or the trip-purpose. Moreover, the other contexts also indirectly influence this semantics. For example, the *class-time schedules* of students and faculty members or in general *appointment schedule* of individuals helps to accurately map their intent for the movements. All these information are embedded with the raw trajectory trips in this pre-processing step.

The pre-processing step of MovCloud is carried out in a VM [Ubuntu 16.04, 15GB memory] of *Google App Engine*. The road-network and POI information are stored in *Google Cloud Big Query Storage*. The map-matching and semantic enrichment algorithms are implemented in the VM, which calls *Google Place API* service and stores the data in a database [Oracle Spatial and Graph] with spatial extension.

B. Trajectory Data Indexing and Storage

To facilitate a fast access of trajectory or location search process, we propose a hierarchical schema to store the trajectory traces.

First, the map-matched trajectory-segments are taken as the input, where each trajectory has an associated road-edge segment. The region-of-interest is divided into rectangular grids and the geo-hash codes of each grids are calculated. Geohash code of the grids represent the spatial location on the earth surface using unique alphanumeric strings. We aim to store the trajectory information by representing the road-map as *Quadtree* [1]. Typically, a quadtree is a tree data structure having exactly four children of each internal node. Quadtrees are utilized to partition a two-dimensional space by recursively subdividing it into four quadrants or regions. Here, using fixed-size grids has a major disadvantage. The cost of accessing trajectory data depends on the spatio-temporal resolution of the query. If we select big grids, then it will be inadequate for smaller queries, since they will contain many other trajectory segments which do not have any intersection with the query. On the other side, too small grids will require more number of disk seeks to access a given solution.

To resolve this issue, we maintain a hierarchical structure where at the top level bigger-size grids store large location information and at the bottom level small location information are stored. Similarly, the temporal information is also sub-divided into different levels based on the resolution (3 hours, 1 hour, 15minutes etc.). Each cell is defined by the geo-hash code and an list of trajectory information is stored in it. The major objective is to reduce the memory footprint of the index. To achieve this, we only consider the non-empty grid-cells. Cloud Spanner of GCP is used to store these information which supports horizontal scalability.

C. Trajectory Data Mining

In this section, we describe the proposed approach to cluster trajectory such that similar movement behaviour are grouped into a cluster, even if they are space and time invariant. The process is deployed in MapReduce paradigm for time and cost effectiveness.

We aim to cluster the trajectories based on the *trip-purposes* of the individuals in a given ROI (region-of-interst). The input of the task is few labelled trajectories (10% of the complete mobility traces) of m users for n days, and the output is labels for all users' trips. The labelled trajectories are used as training data for the mapping task. To carry out the task, we need to capture the similar movement behaviours from the training data to learn the mobility semantics. To this end, deep hierarchical models can generate useful representations of mobility data and distinguish between different trip-purposes albeit they can have similar spatio-temporal features. Here, we propose a deep architecture based mobility clustering framework which involves (i) classify the trajectory-segments into any of the semantic labels (or trip-purposes). and (ii) grouping similar movement traces considering all features and contexts.

Deep Neural Network (DNN) is a feed-forward neural network which maps inputs (feature set) to required outputs. To solve more complex problems, several variations of DNNs are proposed. Most simpler model is with one input layer, one hidden layer and one output layer. The interconnected neurons map the input layers to output layer:

$$y = f(Wx + b) \tag{3}$$

where the output and input vectors are y and x respectively. In our set-up, y represents the semantic label of the trajectory and x contains the mobility features such as timestamp, duration, POI-tag or trip-distance etc. $f(.)$ is the activation function. The bias (b) and weight vectors (W) are learned by deploying gradient based algorithm. While DNN can only operate on a fixed-size sliding window and thus unable to capture the context-shifts. The *LSTM (Long Short Term Memory)* - a type of RNN (Recurrent Neural Network) is suitable to learn the long term dependency of the time-series data and determine the output vector more efficiently. Figure 2 illustrates the basic building blocks of the network. The movement behavioural features (M_f) of the trajectories are extracted in the first step. In order to extract features, the trajectory-segments are fragmented into fixed-length *traj_slider*s, such that each traj_slider contains one stay-point. Few attributes are *stay-duration* (t_{sd}), *average speed*(Δv) and *timestamp*. Sample traj_seg and traj_slider and few attributes are shown:

$$\begin{aligned} Traj\_Seg &= \{(S_i, t_i), (p_1, t_1), \ldots, (p_a, t_a), \ldots (S_j, T_j)\} \\ Traj\_Slider &= \{(S_i, t_i), (p_1, t_1), \ldots, (p_a, t_a)\} \\ t_{sd} &= t_1 - t_i \\ \Delta v &= \frac{\sum_{m=1}^{a}(v_{m+1} - v_m)}{(a-1)} \end{aligned} \tag{4}$$

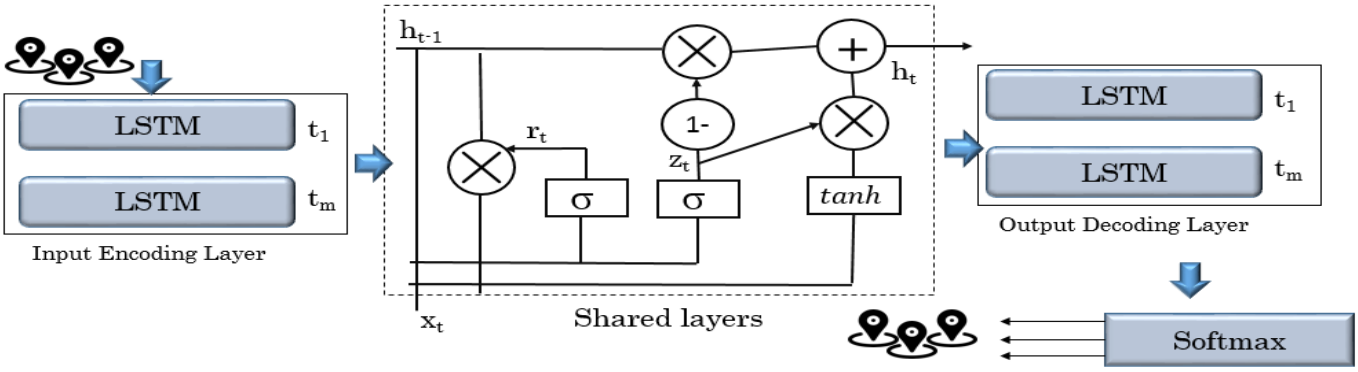

Figure 2: Deep learning architecture to group similar movement behaviour

Similarly we extract other features (M_f) by traversing each record of the *traj_slider*. Next, for each trajectory data, the differences of features are extracted. The fixed length deep representation of the trajectory is carried out here. Next, two steps are followed to extract the trip-purposes:

- A *Gated recurrent unit (GRU)* is used which is similar to $LSTM$. The reset and update gates of GRU is formally defined as:

$$z_t = \sigma(W_z \cdot [h_{t-1}, x_t])$$
$$r_t = \sigma(W_r \cdot [h_{t-1}, x_t])$$
$$h_t = (1 - z_t) * h_{t-1} + z_t * \tanh(W \cdot [r_t * h_{t-1}, x_t] \quad (5)$$

  W_z, W_r and W are the weight matrices. The *update gate* (z_t) helps to extract the required information and the *reset gate* (r_t) determines how much past information needs to be eliminated.
- Finally, the trajectory-segments are mapped to the semantic labels. The representation of the trajectory-segments (l_G) produced by the previous block is fed into the next layer. Here, a *softmax* function is used.

$$l_{G_i} = softmax(W_{G_i} h_{G_i} + b_{G_i}) \quad (6)$$

In this stage, we find the semantic labels of all trajectory segments, and finally we group them in different clusters using classical clustering algorithm K-means and the clusters are obtained. Since, it is observed that back-propagation procedure to derive the LSTM gradient is the most time expensive step. Therefore, we execute this step parallely on each subset of the data in the map phase. Then, the weight increment is computed and updated by the reducer function. In the next step, MovCloud resolves mobility queries using MapReduce paradigm. In this work, we have considered *R-query* (Range-Query) and *T-query* (Trajectory-query) [1] to resolve efficiently. The R-query ($RQ(Q_s, Q_t)$) returns all trajectory line segments intersecting the given spatial (Q_s) and temporal (Q_t) extent.

$$RQ(Q_s, Q_t) \rightarrow G_k \quad (7)$$

Algorithm 1 : Query Processing - A MapReduce based approach

Input: Trajectory Traces G, Spatial Extent location E_s, Time Interval E_T, Query context Q_c
Output: $< G_k >$ ▷ Query Result: Trajectory Segments

1: $function\ MAPPER(E_s, E_T, G[])$
2: $geo - hash(j) \leftarrow generate\ geo - hash\ code(E_s)$
3: $a \leftarrow Find\ QuadTree\ index(G)$ ▷ Search the QuadTree
4: $B \leftarrow a \cap j$
5: **for** $all\ b_i \in B$ **do**
6: $L \leftarrow ExtractTraj(b_i)$ ▷ Extract trajectory information within the cell
7: $EMIT(t, L)$ ▷ t: Temporal information
8: **end for**
9: $function\ REDUCER(L, t)$
10: Sort the segments based on t
11: $L_1 \rightarrow Eliminate(L, E_T)$
12: $G \rightarrow Eliminate(L_1, Q_c)$
13: $G_k \rightarrow Combine(G)$ ▷ Aggregate all trajectory segments from the reducer phase
14: $Print < G_k >$

The T-Query or trajectory-based query returns all trajectory line segments of a moving agent (m) in the temporal interval (Q_t).

$$TQ(m, Q_t) \rightarrow G_k \quad (8)$$

Algorithm 1 presents the proposed approach. In the map phase, the index is searched against the spatial extent of the query and the intersecting trajectory segments are extracted. In the reducer phase, the segments are sorted based on temporal information, and refinement of the trajectory segments send by the mapper function is carried out. Typically, the segments not satisfying temporal constraint or other query context (other attributes such as POI-name, POI-type) are eliminated and rest of the trajectories are reported.

The deep learning architecture is implemented using Google Tensorflow[4]. The implementation is done on the top of the

[4]Tensorflow: https://www.tensorflow.org/

Google App Engine, including DataStore and Task Queues. Two Cloud SQL instances are created, where one is executed from Google App engine. The other instance has the database access permission. To add storage capacity, automatic storage increase is enabled.

IV. Performance Evaluation

This section evaluates different modules of MovCloud and compares our approach with others (K-means, DTW and other query processing approaches [19], [27], [28]).

A. Dataset

The mobility datasets are collected voluntarily from the students, staffs and faculty members of two Indian Institutes, *Indian Institute of Technology Kharagpur (IIT KGP)* and *National Institute of Technology, Warangal, Telangana (NIT W)*. A total of 145 subjects from IIT KGP and 72 from NIT W participated in the survey. Mobility log from the participants are collected from their GPS-enabled smartphones and Google Map Timeline for a span of 28 months. The subjects are requested to upload their movement history weekly basis through a web-form[5]. A sub-set of the data is available in the link[6].

Table II: Top 10 semantic labels of mobility (trip purpose + activity) traces and their counts in the dataset

ID	Semantic Label	Count
s_1	Commuting to Office/WorkPlace	736.3×10^2
s_2	Shopping	287.2×10^2
s_3	Business Travel	158.6×10^2
s_4	Commuting for medical help	148.5×10
s_5	Leisure Travel	371.2×10^2
s_6	Commuting to sports-complex	258.9×10
s_7	Commuting to hangout spot	479.6×10^2
s_8	Commuting to University/Lecture Hall	668.1×10^3
s_9	Commuting to restaurant/cafeteria	378.5×10^2
s_{10}	Commuting to auditorium/ movie-complex	178.2×10^2

Moreover, the volunteers logged the semantics (trip-purposes) of their movement history. We have provided a list (total 48) of *trip-purpose* or *activity* and the individuals can identify the intent of the movements for each trajectory-segment.

Few of the semantic labels of trajectories are reported in Table II. We have also provided the total number (*count* in Table II) of such trip-purposes in the datasets. The activity-survey is designed in a way such that users can log the activities performed or trip-purposes in different time-slots. In the experimental set-up, 70% of the movement traces are used for training the module, 20% and 10% for testing and validating respectively.

[5]GPS:https://form.jotform.me/61262371630448

[6]DataSet:https://drive.google.com/drive/folders/1BpM-K3clH6XYpSHkFe12aGsG8n1AclI4?usp=sharing

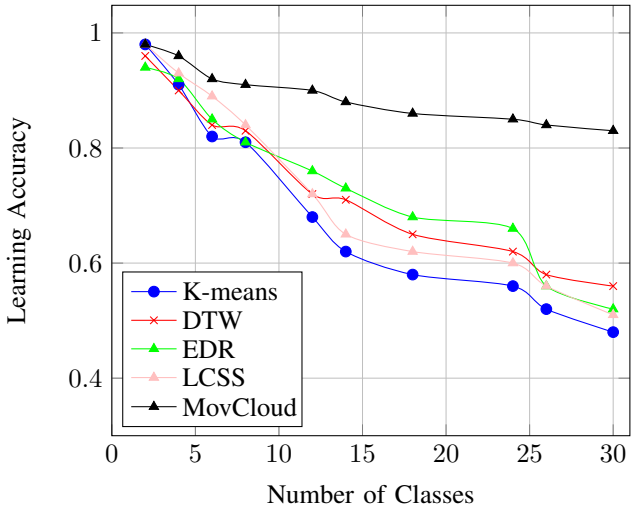

Figure 3: Comparison of learning accuracy of clustering method with baseline methods and MovCloud

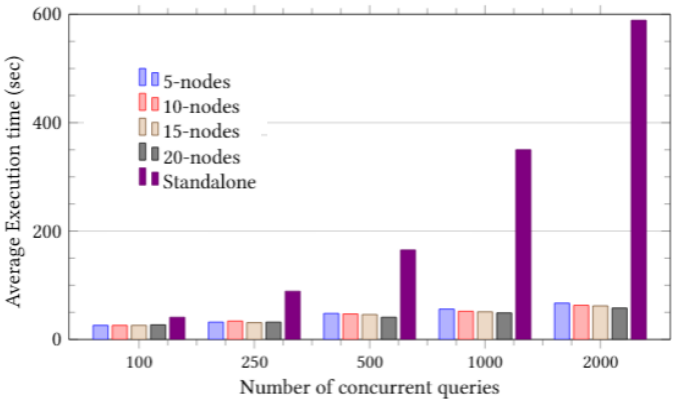

Figure 4: Query (Range Query) Concurrency

B. Experimental Observations

The major goal of this paper is to cluster users' movement trajectories based on their movement behaviour. We evaluate the clustering approach by five validation metrics, namely, *overall similarity* (O), *precision* (Pre), *recall* (Re), *cohesion* (Co) and *separation* (Se) [6]. Table III reports the normalized values of 10 clusters ($C_1, \ldots, C_{10}$). Here, the clusters are numbered based on the same ordering of semantic labels presented in Table II. The overall similarity depicts the agglomeration degree and compact clusters have less O value. The closeness of the trajectories within the cluster and separateness with other clusters are defined by Co and Se values. MovCloud has achieved 0.84 cohesion value and 0.79 separation values - which support the correctness of our proposed approach.

Figure 3 shows the learning accuracy of the proposed clustering algorithm compared to other baseline methods. The experiment has carried out with 30 classes and accuracy is compared with four other classical clustering technique.

Table III: MovCloud Clustering Validation Result

Cluster ID	Overall Similarity (O)	Precision (Pre)	Recall (Re)	Cohesion (Co)	Separation (Se)
C_1	0.18	0.721	0.28	0.792	0.756
C_2	0.32	0.56	0.47	0.581	0.602
C_3	0.21	0.68	0.30	0.702	0.684
C_4	0.16	0.82	0.26	0.84	0.791
C_5	0.36	0.61	0.52	0.671	0.708
C_6	0.22	0.65	0.33	0.684	0.702
C_7	0.304	0.58	0.42	0.616	0.64
C_8	0.258	0.64	0.37	0.672	0.703
C_9	0.33	0.53	0.42	0.582	0.546
C_{10}	0.27	0.66	0.36	0.672	0.658

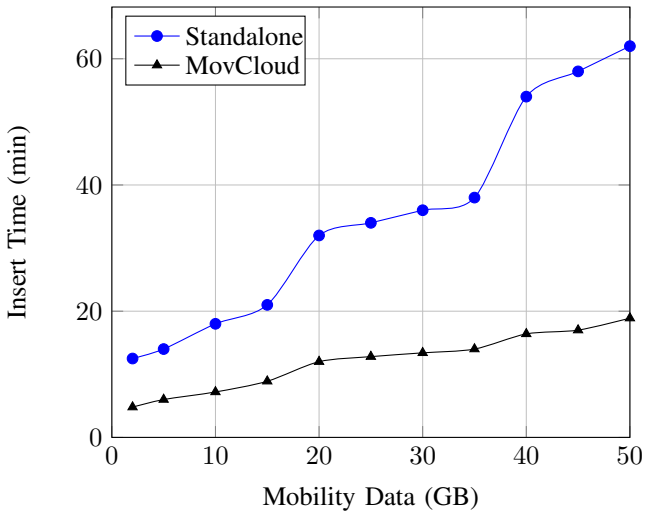

Figure 5: Indexing Efficiency of MovCloud

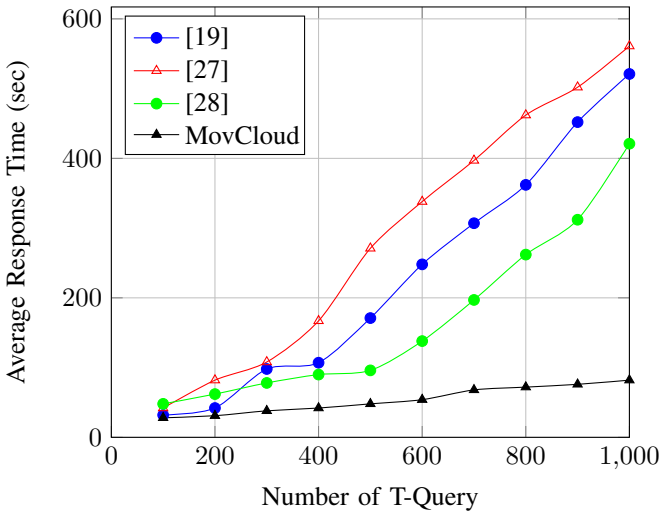

Figure 6: Comparison of execution time of T-Query Processing

With the increasing number of classes, the accuracy of other clustering techniques reduce drastically, while MovCloud has manitained an accuracy value within 0.83 to 0.98. Other baselines show accuracy value in the range of 0.56 to 0.48, when the number of classes is 30. The average improvement of clustering accuracy over other baselines is $\sim$ 28%. It is observed that MovCloud has outperformed other baselines in a large margin. The reason behind obtaining more accurate results than other methods, is that other approaches rely only on spatial and temporal features of the raw trajectory, and fall short on capturing the semantic meaning or intent of the movement trajectories.

Figure 4 illustrates the different number of concurrent queries and the average execution time of MovCloud with different number of nodes in the framework. We have evaluated the framework with 100, 25, 500, 1000 and 2000 concurrent queries and report the average execution time. Along with the standalone system, varied number of nodes (5-20) are used to show the variations of execution time. The framework is scalable to handle query-load at any given instance. Our approach using MapReduce paradigm allows MovCloud to handle concurrent queries compared to a standalone server set-up.

The indexing efficiency of our framework is shown in Figure 5. It is observed that 50GB data insertion takes less than 20 minutes, whereas standalone needs more than 60 minutes. Cloud Spanner of GCP is utilized in MovCloud framework appropriately to support horizontal scalability. We have compared MovCloud with [19], [27] and [28] by varying number of T-Query (Figure 6). In the experimental set-up, the numbers of T-Query are varied from 100 to 1000 range. The other baselines take in the range of $32s - 48s$ and $421s - 561s$ for 100 and 1000 concurrent queries respectively. The response time of MovCloud is $28s$ and $82s$ in the same set-up. It is observed that the average response time of MovCloud is significantly better than these existing works. The distributed way of query handling and novel indexing schema of MovCloud are the key reasons of this performance.

In summary, the MovCloud is capable to cluster trajectory segments efficiently and shows better learning accuracy than baseline clustering methods. Since, the approaches are implemented using MapReduce paradigm, the average execution time of query processing is much lesser than standalone. MovCloud reduces the data insertion time by almost 1/3rd

than standalone set-up using the novel indexing scheme and distributed platform. It is observed that, for 2000 concurrent queries, MovCloud takes around 45secs (20 nodes), while standalone needs almost 600secs - which is useless in any real-life application.

V. Conclusions and Future work

This paper presents an end-to-end cloud-based framework *MovCloud* conducive to index, store and analyse mobility traces of individuals. The proposed clustering approach is capable to cluster users' movement trajectories based on similar movement behaviour. Furthermore, the query processing over MapReduce paradigm provides an edge over other existing query solving approaches in terms of computational time. The hierarchical indexing of trajectory information using QuadTree is a novel proposition. The framework is implemented over an interface of Google cloud platform and an extensive set of experiments illustrates the effectiveness of the framework.

In future, we would like to extend the present cloud-based framework to provide varied mobility services such as personalized route recommendation, trip-planner. Using the same framework, we will deploy a ride-sharing service, where similar mobility trips will be clustered. Also, we will accommodate more semantic trip-lables and validate with different mobility datasets instead of only traces from academic campuses to make the framework more generic. We strongly believe that proposed framework will act as a foundation of several cloud-enabled spatio-temporal mobility analytics.

Acknowledgement

The authors would like to thank the volunteers of IIT Kharagpur and NIT W for sharing their movement traces. The work is partially supported by TCS Research Scholarship grant.

References

[1] Y. Zheng, "Trajectory data mining: an overview," *ACM Transactions on Intelligent Systems and Technology (TIST)*, vol. 6, no. 3, p. 29, 2015.

[2] S. Ghosh and S. K. Ghosh, "Modeling individual's movement patterns to infer next location from sparse trajectory traces," in *2018 IEEE International Conference on Systems, Man, and Cybernetics (SMC)*, pp. 693–698, IEEE, 2018.

[3] S. Ghosh and S. K. Ghosh, "Modeling of human movement behavioral knowledge from gps traces for categorizing mobile users," in *Proceedings of the 26th International Conference on World Wide Web Companion*, pp. 51–58, ACM, 2017.

[4] S. Ghosh and S. K. Ghosh, "Thump: Semantic analysis on trajectory traces to explore human movement pattern," in *Proceedings of the 25th International Conference Companion on World Wide Web*, pp. 35–36, 2016.

[5] J.-G. Lee, J. Han, and K.-Y. Whang, "Trajectory clustering: a partition-and-group framework," in *Proceedings of the ACM SIGMOD International conference on Management of data*, pp. 593–604, ACM, 2007.

[6] G. Yuan, P. Sun, J. Zhao, D. Li, and C. Wang, "A review of moving object trajectory clustering algorithms," *Artificial Intelligence Review*, vol. 47, no. 1, pp. 123–144, 2017.

[7] P. C. Besse, B. Guillouet, J.-M. Loubes, and F. Royer, "Review and perspective for distance-based clustering of vehicle trajectories," *IEEE Transactions on Intelligent Transportation Systems*, vol. 17, no. 11, pp. 3306–3317, 2016.

[8] M. D. Assunção, R. N. Calheiros, S. Bianchi, M. A. Netto, and R. Buyya, "Big data computing and clouds: Trends and future directions," *Journal of Parallel and Distributed Computing*, vol. 79, pp. 3–15, 2015.

[9] C.-C. Hung, W.-C. Peng, and W.-C. Lee, "Clustering and aggregating clues of trajectories for mining trajectory patterns and routes," *The International Journal on Very Large Data Bases (VLDB Journal)*, vol. 24, no. 2, pp. 169–192, 2015.

[10] N. Pelekis, P. Tampakis, M. Vodas, C. Doulkeridis, and Y. Theodoridis, "On temporal-constrained sub-trajectory cluster analysis," *Data Mining and Knowledge Discovery*, vol. 31, no. 5, pp. 1294–1330, 2017.

[11] D. Yao, C. Zhang, Z. Zhu, J. Huang, and J. Bi, "Trajectory clustering via deep representation learning," in *Proceedings of the International Joint Conference on Neural Networks (IJCNN)*, pp. 3880–3887, IEEE, 2017.

[12] Q. Gao, F. Zhou, K. Zhang, G. Trajcevski, X. Luo, and F. Zhang, "Identifying human mobility via trajectory embeddings.," in *Proceedings of the International Joint Conference on Artificial Intelligence (IJCAI)*, vol. 17, pp. 1689–1695, 2017.

[13] S. Ghosh and S. K. Ghosh, "Traj-cloud: a trajectory cloud for enabling efficient mobility services," in *Proceedings of the 11th International Conference on Communication Systems & Networks (COMSNETS)*, pp. 765–770, IEEE, 2019.

[14] A. Aji, F. Wang, H. Vo, R. Lee, Q. Liu, X. Zhang, and J. Saltz, "Hadoop gis: a high performance spatial data warehousing system over mapreduce," *Proceedings of the VLDB Endowment*, vol. 6, no. 11, pp. 1009–1020, 2013.

[15] J. Dean and S. Ghemawat, "Mapreduce: simplified data processing on large clusters," *Communications of the ACM*, vol. 51, no. 1, pp. 107–113, 2008.

[16] S. Shekhar, V. Gunturi, M. R. Evans, and K. Yang, "Spatial big-data challenges intersecting mobility and cloud computing," in *Proceedings of the 11th ACM International Workshop on Data Engineering for Wireless and Mobile Access*, pp. 1–6, ACM, 2012.

[17] A. Eldawy and M. F. Mokbel, "Spatialhadoop: A mapreduce framework for spatial data," in *Proceedings of the 31st international conference on Data Engineering*, pp. 1352–1363, IEEE, 2015.

[18] S. Ruan, R. Li, J. Bao, T. He, and Y. Zheng, "Cloudtp: A cloud-based flexible trajectory preprocessing framework," in *Proceedings of the 34th International Conference on Data Engineering (ICDE)*, pp. 1601–1604, IEEE, 2018.

[19] R. Li, S. Ruan, J. Bao, Y. Li, Y. Wu, and Y. Zheng, "Querying massive trajectories by path on the cloud," in *Proceedings of the 25th ACM SIGSPATIAL International Conference on Advances in Geographic Information Systems*, p. 77, ACM, 2017.

[20] S. Ghosh, S. K. Ghosh, R. D. Das, and S. Winter, "Activity-based mobility profiling: A purely temporal modeling approach.," in *Proceedings of the 27th International Conference Companion on World Wide Web*, pp. 409–416, 2018.

[21] S. Ghosh, A. Chowdhury, and S. K. Ghosh, "A machine learning approach to find the optimal routes through analysis of gps traces of mobile city traffic," in *Recent Findings in Intelligent Computing Techniques*, pp. 59–67, Springer, 2018.

[22] S. Ghosh and S. K. Ghosh, "Exploring human movement behaviour based on mobility association rule mining of trajectory traces," in *International Conference on Intelligent Systems Design and Applications*, pp. 451–463, Springer, 2017.

[23] S. Ghosh and S. K. Ghosh, "Exploring the association between mobility behaviours and academic performances of students: a context-aware trajgraph (ctg) analysis," *Progress in Artificial Intelligence*, vol. 7, no. 4, pp. 307–326, 2018.

[24] S. Shekhar and S. Chawla, *Spatial databases: a tour*. Pearson, 2003.

[25] E. J. Krakiwsky, C. B. Harris, and R. V. Wong, "A kalman filter for integrating dead reckoning, map matching and gps positioning," in *IEEE PLANS'88., Position Location and Navigation Symposium, Record.'Navigation into the 21st Century'.*, pp. 39–46, IEEE, 1988.

[26] Y.-J. Gong, E. Chen, X. Zhang, L. M. Ni, and J. Zhang, "Antmapper: An ant colony-based map matching approach for trajectory-based applications," *IEEE Transactions on Intelligent Transportation Systems*, vol. 19, no. 2, pp. 390–401, 2017.

[27] H. Liu, J. Xu, K. Zheng, C. Liu, L. Du, and X. Wu, "Semantic-aware query processing for activity trajectories," in *Proceedings of the Tenth ACM International Conference on Web Search and Data Mining*, pp. 283–292, ACM, 2017.

[28] S. Shang, L. Chen, C. S. Jensen, J.-R. Wen, and P. Kalnis, "Searching trajectories by regions of interest," *IEEE Transactions on Knowledge and Data Engineering*, vol. 29, no. 7, pp. 1549–1562, 2017.