MoSum: Spatio-temporal Mobility Summary of Individuals

Abstract—**Mobility data of people is being increasingly recorded by location sensing applications such as GPS traces and Cellular Network Records. Such large-scale location data of people is capable of providing rich mobility context information about how, where and when an individual moves. These insights are useful in several domains such as hyper-targeted advertising, city transportation planning and cellular network planning. While mobility data is capable of providing such interesting information, techniques to summarize a spatio-temporal mobility of a individual is non-existent. In this work, we propose an "'Mobility Summary (MoSum)", a system for summarizing and quantifying how an individual moves in space and time. MoSum provides a novel way to store mobility signatures of a person, which inturn provides a powerful mechanism for solving various use-cases that can rely on regular movement pattern of an individual (such as next-location prediction and anomalous movement detection).**

I. Introduction

Advances in GPS-based applications and ubiquitous connectivity has enabled collection of vast amounts of location data describing the movement of humans and animals. Currently, location traces of millions of people are being collected by various applications [1]. In addition, location traces of a vast majority of the population are inherently collected by cellular network operators in the form of Call Detail Records, which continuously log the base-station to which the user is connected [2]. This large-scale location traces of people enables understanding the movement pattern of objects, and opens a plethora of location-enabled applications such as location prediction, mobility-intent identification and anomaly detection.

Large scale human mobility data enables solving interesting problems and generating new revenue streams for the enterprises. For example, the transportation departments now spend millions of dollars once in a few years in *travel pattern surveys*, which only samples a sub-5% of population, to plan the bus and train networks [9]. Such expensive and exhaustive methods can be easily replaced by analysis of location traces, which can provide real-time and fine-granular data from a significantly larger sample of people. Similarly, location data can be analyze user interactions in physical space. Insights from location data enable determining user interests, demographics and who they hangout with based on where, how and when people go to different locations such as stadiums and malls. Hence, location data – similar to social networking data – enables enterprises to create new revenue streams.

A primary challenge in enabling new applications is inferring insightful movement patterns from the raw location traces. Existing studies have focused on identifying hangouts of an individual, such as home/work and other frequently visited places [5]. Hangouts provide a spatio-temporal signature of the person in terms of where the person hangs out. However, such algorithms does not indicate the mobility pattern of the user.

Mobility Summary of an individual succinctly describes frequent paths taken by the user. Mobility Summary is the natural abstraction for many higher level applications that utilize "frequent-mobility based queries", where an application can query frequent movement patterns – instead of all the trajectories of the user – to infer some insight. Examples of frequent-mobility based queries include: (1) users who frequently pass through a given place such as a coffee shop, (2) Next-path prediction problem where user's future path is predicted based on current path and a history of user trajectories, and (3) Anomalous Trajectory Detection where outlier trajectories of a user need to filtered. Such queries are efficiently solved by a one-time computation of mobility summary. Applications can then query the summary, rather than each application querying thousands of user trajectories, to provide insights. Hence, mobility summary enables efficient and fast-lookup for many movement-based queries that rely on computing frequent trajectories of an individual.

Modeling movement summaries of individuals from the location traces has not been studied in the existing literature. Some studies have examined trajectory clustering by extending point or sub-trajectory based clustering mechanisms [8], [7]. Such schemes primarily operate on points (or sub-parts) of trajectories, and finally aggregate the point clusters to compute a trajectory cluster. However, as we show in the paper, such schemes are poor in summarizing individual's trajectories for two main reasons. First, aggregating on cluster of points or sub-trajectories does not consider the similarity between entire trajectories; this often aggregates dissimilar trajectories or fails to identify similar trips within a cluster unless a careful parameter tuning is performed for each individual user. Second, these schemes do not scale to large sets of location traces since they incur high computational time; usually they repeatedly apply clustering to sub-parts on the all the trajectories and later aggregate the results.

In this paper, we design an end-to-end system "MoSum" for constructing mobility summary for individual users. MoSum takes in a time-series of user's location traces and outputs a set of weighted representative summary trajectories of the user, which describe the user mobility pattern. Our paper has the following contributions:

- We propose abstraction of Mobility Summary to capture

important representative paths of an individual; this enables in building spatio-temporal mobility signatures of people and applications that use frequent-mobility based queries.
- We devise an efficient metric, called "Weighted LP-Norm", to compute the distance between a pair of user trajectories. We utilize this metric as a distance metric for clustering trajectories. We show that existing metrics such as Dynamic Time Warping [11] are insufficient as they are non-metrics and computationally expensive.
- We devise algorithms to determine optimal clustering of user trajectories, and determine representative summary trajectories from a set of trajectories.
- We implement Next-Path Prediction algorithm, which uses frequent-mobility query, to demonstrate that one-time computation and storage of user's mobility summary significantly reduces the complexity of applications. These applications can use fast lookup on mobility summary to derive insights instead of querying all user trajectories.

The rest of the paper ...

II. Formulation Overview and Preprocessing

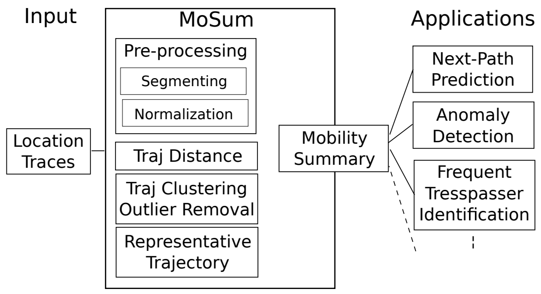

Fig. 1. MoSum Components

Figure 1 provides the different components for computing the Mobility Summary, which in turn can be used a variety of applications that rely on frequent-mobility based queries. We now describe the components of MoSum in detail

We first identify meaningful trips of a user using trajectory segmentation approaches [14]. Here we compute the distance, velocity and time-gap between consecutive set of sample points to identify if the user is mobile. We use the well-known representation of trajectory to denote a meaningful trip; it is an ordered set of 3-tuples $\langle \text{latitude}, \text{longitude}, \text{time} \rangle$.

We next normalize the user locations since small changes in latitude and longitude can result in large distances on earth. We normalize each raw latitude rlat_i in the sample into a normalized latitude lat_i in the range $[0, 1]$ as:

$$\text{lat}_i = \frac{\text{rlat}_i - \min_{\text{lat}}}{\max_{\text{lat}} - \min_{\text{lat}}} \tag{1}$$

where $\min_{\text{lat}} = \min(\text{rlat}_j, \forall j)$ is the minimum latitude observed, and $\max_{\text{lat}} = \max(\text{rlat}_j, \forall j)$. We similarly normalize raw longitude and time values into normalized longitude and time values.

III. Inter-Trajectory Distance

Computing the distance between a pair of trajectories is crucial to MoSum because of higher level clustering algorithms require such a distance measure. A large number of trajectory similarity metrics, which give an estimate of the trajectory distance, have been proposed for various types of applications. Standard LP-Norm and ERP have been shown to be metrics [3], where as a large number of other heuristic measures (such as LCSS, DTW, EDR) are non-metrics [10], [11], [4]. A taxonomy of trajectory similarity metrics is discussed in Table I.

Non-metric distance functions are clearly inappropriate in clustering. In addition, we show that blindly using the existing non-metric distance functions not only results in sub-optimal clusters but also incurs significant high cost in terms of computational time. A primary reason for the most non-metric functions (LCSS, DTW, EDR, ERP) are designed to suppress noise using techniques such as dynamic programming. However, for GPS trajectory, de-noising can be done as a preprocessing step by removing or resampling the outlier points [12], [13]. Hence, we use modified standard LP-Norm functions for distance computation, and avoid time consuming non-metric algorithms.

A. *Curve Distance for Trajectories*

Instead of treating trajectory as a set of sample points, we approximate the trajectories as curves on n-dimensional vector space. This approximation is reasonable if the location traces contains finely sampled GPS points (as in our case)

For trajectory representation, the main idea is to represent trajectory as a curve in two independent dimensions latitude and longitude (in $\mathbb{R}^2$) [6]. A natural extension is to represent in three independent dimensions including time. However, in this paper, we propose to look at latitude and longitude dimensions

Let $f_i[0,1] \rightarrow \mathbb{R}^2$ be the curve for the i-th trajectory trajectory, which maps a number between 0 to 1 to the (latitude, longitude) pairs of the trajectory. The standard $\mathbb{L}^2$ norm for this curve is given $\|f_i\| = \left[\int_0^1 f_i(x)^2 \,\mathrm{d}x\right]^{\frac{1}{2}}$.

$$\text{CD}(t_i, t_j) = \left[\int_0^1 \left(f_i(x) - f_j(x)\right)^2 \,\mathrm{d}x\right]^{\frac{1}{2}}. \tag{2}$$

Numerically, we solve this by first resampling the trajectory at a large number of points for each trajectory (100 samples in our case) and then using Trapezoidal Rule [**?**] to find the curve distance.

a) Weighted Curve Distance: For human mobility, a user's meaningful trip has an associated intention (such as commuting to work or grocery shop visit). More often, each frequent trajectory are between end-points that are important to the user (such as home and work). We now propose a metric that emphasizes origin and destination of the trajectories, while computing the distance.

We first generalize the Curve Distance to Weighted Curve Distance, where all trajectories have different weights at each

Sim Measure	Is Metric	Type	Sen. to sample noise	OD Cognizant	Computational Cost
LP Norm	Yes	Sampling Sensitive	No	No	O(N)
DTW/LCSS/EDW/EDW With real sequences	No	Sampling Sensitive	Yes	No	O(n^2)
EDWP	Yes	Sampling Sensitive	Yes	No	??
LP Norm with Interpolation	Yes	Shape Sensitive	No	No	O(Num samples)
ODSim (Ours)	Yes	Shape sensitive	No	No	O(Num samples)

TABLE I
TAXONOMY OF SIMILARITY MEASURES

point. It can be shown the Weighted Curve Distance (WCD) between two trajectories t_i and t_j is given by

$$\mathrm{WCD}(t_i,t_j) = \left[\int_0^1 w(x)\left(f_i(x)-f_j(x)\right)^2 \,\mathrm{d}x\right]^{\frac{1}{2}}. \quad (3)$$

where $w(x) \rightarrow [0,1]$ is a weighting function. In the case of providing higher weights to origin and destination an Origin-Destination (OD) weighing function should be constructed such that the weights are high at the ends than at the center. We use Beta function $\mathrm{B}(\alpha,\alpha)$, where α in $[0,1]$. This provides a bimodal curve where weights at the ends are higher than the weights at the intermediate points in th curve; lower values of alpha provide very high values at ends than at the intermediate points.

Since CD is a metric and WCD is a weighted combination of CD (with positive weights), it can be shown that WCD is also a metric.

IV. TRAJECTORY CLUSTERING

We use the distance metrics to aggregate similar trajectories into one cluster. We use hierarchical agglomerative clustering since it provides the flexibility of analyzing the entire merge history of user's trajectories, and then cutting the dendogram at the right level. This enables us to personalize and automate clusters for different users. Each user has different motion patterns and – apriori – we do not have information of how often/densely a user travels along different paths. Our system automatically recognizes the right number of clusters by analyzing the dendogram.

We use "average" link clustering to measure similarity between intermediate clusters. This is to avoid bias for clustering trajectories that are associatively near (using single-link) or by concentrating on the extreme points of the merge (using complete-link).

A. Finding optimal clusters

In this step, we cut the dendogram at a level that defines meaningful mobility clusters. The main idea is to determine the optimal number of clusters for each person, and to use that knowledge to cut the dendogram. We went against determining a static similarity level to cut dendogram since different people have different forms of mobility. A user whose main travel pattern is long distance commute to two office locations has different distance thresholds than a student who is commuting mainly commuting from dormitory to classes.

Existing well-known methods, such as the elbow method, do not cut the dendogram at appropriate level to provide good movement summaries; we demonstrate this in Section . Hence, we design an algorithm that cuts the dendogram at a level where trajectories in a cluster are between nearby origin and destinations, which signify similar meaningful trips of a person.

```
Input: Dendrogram with cluster information at each level
Output: Optimal Clusters (Trip Summaries)
for k=1 to N do
    $SSW(Clus_i) = \sum_{j=1}^{|Trajs(Clus_i)|} Sim(Traj_j, Mean(Clus_i))$   (4)
    $SSW(Level_k) = \sum_{j=1}^{k} SSW(Cluster_j)$   (5)
end
Find the elbow point from the SSW Plot over all levels
Set all trajectories as unmarked for
k=elbowPoint+1 to N do
    for Each non anomalous cluster do
        if Trajs(Cluster) are unmarked &&
        isSummary($Cluster_i$) then
            Report Cluster as a final Cluster
            Mark all Trajs in Cluster
        end
    end
end
isSummary($Cluster_i$)
for All pairs of trajectories in Cluster do
    if Maximum Pointwise Distance$\leq \delta$ (kms) then
        return True
    end
end
return False
```

Algorithm 1: Algorithm for reporting final clusters from Dendrogram

The algorithm iterates down the dendrogram starting at the root; the number of clusters at this stage being $k=1$. As we proceed down each level to cut the dendrogram, the number of clusters k increases by one. We compute the possible summary clusters of the user for each k. Let $\mathbf{S}_k = \{S_{ki}, \forall i\}$ be the set of clusters for the user at level k. Let t_{kij} be the j-th trajectory in S_{kij}. Let the mean trajectory for S_{ki} be $\bar{t}_{ki}$. We examine the tightness of clustering at this level by computing the Sum of Squares Within cluster (SSW). SSW at level k is defined as $SSW_k = \sum_{S_{ki}\in\mathbf{S}_k}\sum_{t_{kij}\in S_{ki}} \mathrm{WCD}(t_{kij},\bar{t}_{ki})^2$.

A well-known metric is to accept the elbow point of the

SSW_k vs. k curve (say, at $k = k_e$) as the optimal number of clusters. However, as we show in Section **??**, the trajectories in clusters at the elbow point k_e may contain multiple type of short trips in a small region; such trips can be split further into different cluster. Hence, we iterate from the elbow point towards greater k, and each k we examine the resulting clusters to see if the trajectories from different types of meaningful trips are contained in a single cluster. We declare that different type of meaningful trips are present in a cluster if the physical distance between any point of any pair of trajectory curves in the cluster is greater than a intra-cluster separation distance T_{intra} (which is 1.5 km in our case)

We finally represent the "Mobility Summary" of a user as the clusters with number of trajectories greater than a certain threshold (5% of user's trajectories)

B. *Representative trajectories*

We now find a representative trajectory for each of the cluster that can be used as a proxy for a summary cluster. One approach is to consider the mean trajectory curve $\bar{t}_i$ as a representative for the cluster S_i. However, since the mean of multiple curves may not fall on any of the individual trajectories (and hence the roads on which the user went), it is not recommended as a representative. Hence, we follow a well-known method of computing piece-wise median trajectories [**?**]. Here, at an interval of δ number of points on interpolated mean trajectory, one point on any of the trajectories in the cluster closest to the mean is added to the representative trajectory.

V. Evaluation and Analysis

A. *Datasets*

We work on a dataset consisting the trajectories of people in Beijing. (We also plan to work on tht following datasetss-)

- Movebank/Starkey
- Singapore

B. *Clustering Effectiveness*

1) Silhouette Coefficient: We show that the proposed algorithm clusters significantly better by using Silhouette Coefficient (SC); a standard metric that shows the effectiveness of clustering. SC is based on the cohesion and the separation of clusters formed. The cohesion (*a(x)*) is defined as the average distance of x to all other vectors in the same cluster. The separation (*b(x)*) is defined as the minimum of the average distances of x to the vectors in other clusters. Further, the silhouette coefficient of a data point is defined as

$$s(x) = \frac{b(x) - a(x)}{max(a(x), b(x))} \tag{6}$$

The total silhouette coefficient of the dataset is the average over all the points given by

$$SC = \frac{1}{N}\sum_{i=1}^{N} s(x) \tag{7}$$

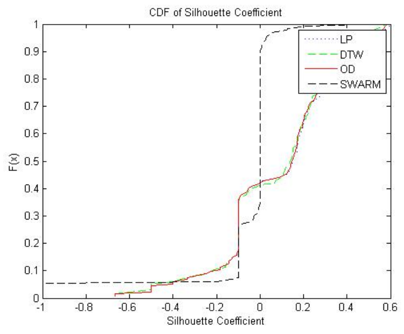

Fig. 2. Comparison of silhouette coefficients for different schemes: OD- and LP-based clustering outperform existing mechanism.

Ideally, SC is between [-1,1], where values closer to 1 representing better formed clusters.

The CDF plot shows that DTW is very similar to our approach in terms of clustering effectiveness, but SWARM does a very poor job.

2) Average Trajectories Per Cluster: Number of clusters with large number of trajectories in it is better (Figure 3 and 4).

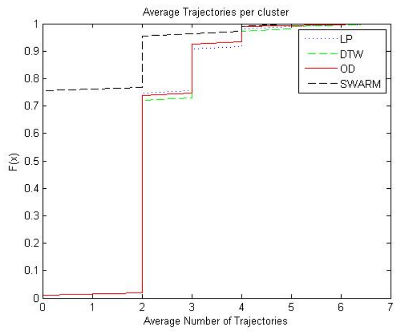

Fig. 3. CDF of Average trajectories per cluster for LP-DTW-OD

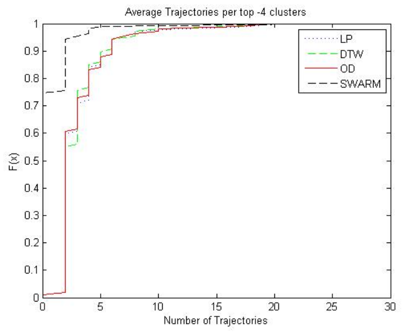

Fig. 4. CDF of Average trajectories per top-k clusters for LP-DTW-OD

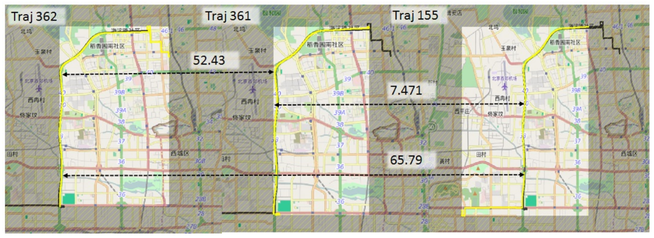

Fig. 6. Example of Triangle Inequality Violation using DTW

C. Computation Time

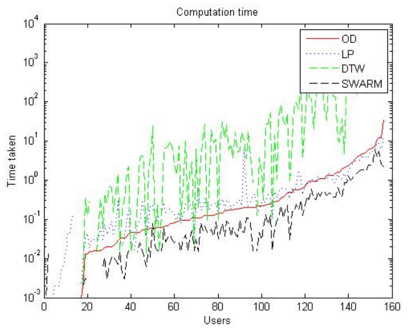

Fig. 5. Computation time for LP-DTW-OD

D. Existing Methods and comparisons

We have compared our proposed method with 3 main approaches. The first comparison is made using Dynamic Time Warping as the similarity metric, and clustering based on that matrix. The next is with SWARM, which takes identifies moving clusters. Finally, we compare with TrajClus which is based on partitioning the trajectories into segments, and then clustering over all the partitions.

Among all the methods, DTW is very close to our method considering the clustering effectiveness, but there are cases where it misses out trajectories that are a part of a meaningful trip summary. On the basis of computation time, our approach is way faster than DTW, because DTW heavily depends on the number of sample points. As the number of sample points increase, the time starts to blow up. SWARM is very poor at identifying meaningful summaries, and misses out even very huge clusters for many users.

a) Dynamic Time Warping: Problems with DTW

- DTW is not a metric as it violates triangle inequality. This can lead to issues during clustering.Any distance metric d follows triangle inequality if, for any three points, x,y,and z: d(x, z) d(x, y) + d(y, z). Figure 6 shows an example of the violation of triangle inequality using DTW similarity.
- The biggest concern about DTW is the computation time. When the sample points are very large, it can get to as much as 400 times slower than the proposed approach. If the points are resampled and DTW similarity is computed, it would reduce to the same as pointwise Eucledian distance, and would still be computationally more expensive. Fig 7 shows the computation time differences over all the users for clustering using DTW and OD similarity measures over all the users
- As we decrease the number of sample points, the effectiveness of both our proposed method and DTW go down. But the goodness of the clusters returned by DTW decreases more than that of the proposed method. We reduced the number of sample points in each of the trajectories to 90%,95%, and 97% and plotted the silhouette coefficient values using DTW,LP and OD.

b) SWARM: Problems with SWARM

- Does not report all the big clusters
- Dependent a lot on the sample points. If we resample,

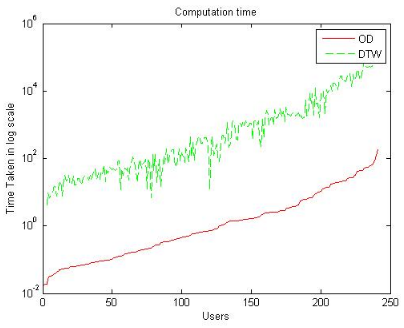

Fig. 7. Computation time comparison of DTW and OD

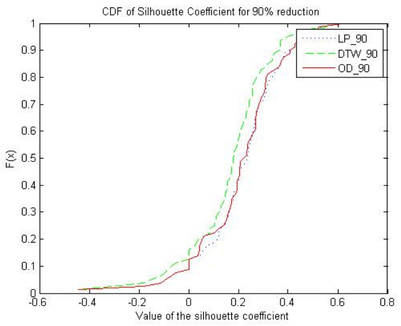

Fig. 8. CDF of the silhouette coefficient for 90% reduction of sample points

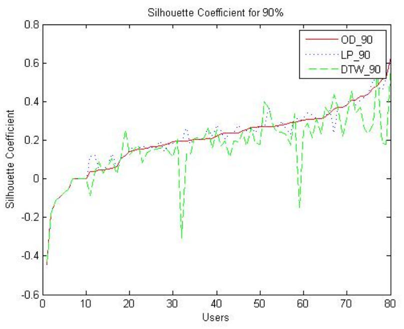

Fig. 9. Plot of the silhouette coefficient for 90% reduction of sample points

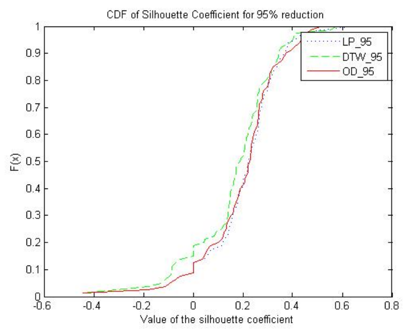

Fig. 10. CDF of the silhouette coefficient for 95% reduction of sample points

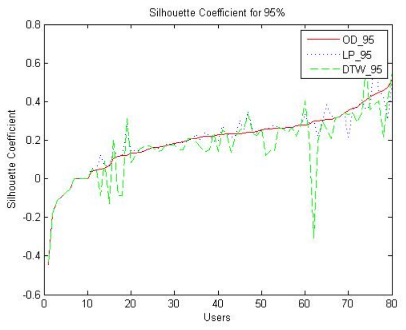

Fig. 11. Plot of the silhouette coefficient for 95% reduction of sample points

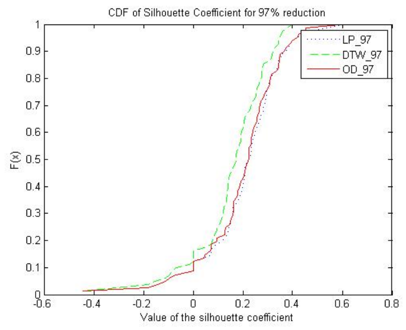

Fig. 12. CDF of the silhouette coefficient for 97% reduction of sample points

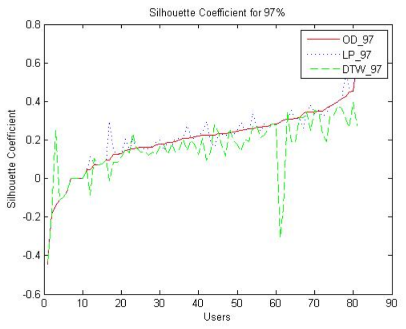

Fig. 13. Plot of the silhouette coefficient for 97% reduction of sample points

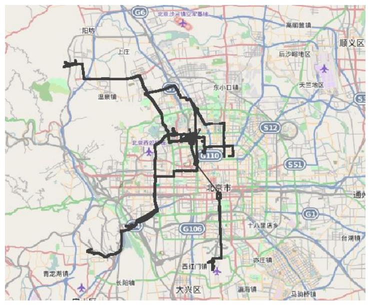

Fig. 14. All the trajectories of an example user

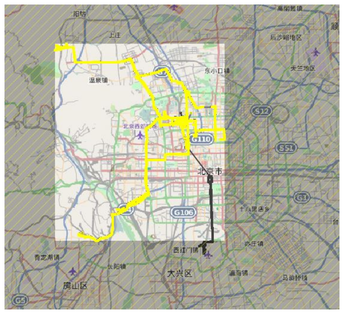

Fig. 15. Cluster reported by TrajClus

it would be computationally more expensive than our approach.

c) TrajClus: TrajClus looks at the sub-trajectory level, and clusters trajectories on the basis of the similarities between those sub-trajectories. In the first phase, it partitions the trajectories into segments, and in the second phase, it clusters the partitions together using a DBSCAN-like technique.The problems with TrajClus are

- The similarity measure between any two partitions is defined as a weighted sum of the perpendicular distance, parallel distance and the angular distance between the partitions. Here, three quantities with different units are being merged together, so when two partitions are similar, it is difficult to say which distance contributed to the similarity. Also, this creates a problem in arriving at the neighbourhood parameters.
- There are two parameters used in this algorithm, *epsilon* and *minLns*. *epsilon* defines the neighourhood reach of each of the partition, and minLns is the minimum number of Partitions required in the neighbourhood for it to be considered as a cluster. The authors suggest a simulated annealing technique to arrive at the value of epsilon , and from that value, further calculate the value of minLns. But, because the algorithm is highly associative, nearly all the partitions end up in one cluster, thus not identifying the correct movement summaries.
- TrajClus does not give enough weightage to the direction of the line segment. In cases like animal movement or hurricane movement, this makes sense, because there wont be many cases of trajectories in different directions

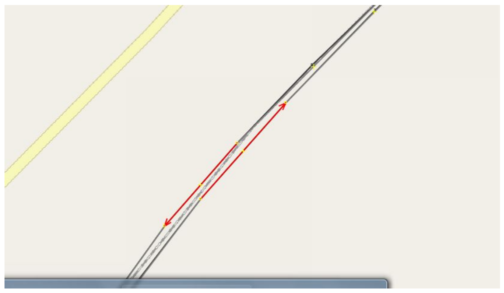

Fig. 16. Trajectories with different directions clustered together by TrajClus

in a flock or cluster. But when it comes to human movement pattern, directions play a very important role in determining the movement summaries of a person. TrajClus overlooks it and clusters two trajectories in different directions in the same cluster.

E. Next Location Prediction

Another way to test the summarization of the movement patterns is to test a query trajectory and plot its predicted next location/destination as predicted by all the methods.

The next location prediction is done by the following algorithm:

a) Explanation of the algo: For any query trajectory, resample, and compute similarity with the median trajectories of all summary clusters. Report the one with the maximum similarity.

Let $g(i)$ be the probability of the summary i. Given an input traj t_{in}, compute the distance (in meters or so) $d(i, t_{\text{in}})$ between summary i and t_{in}. Now the probability that this sub-trajectory lies within summary i is given by

$$p(i, t_{\text{in}}) = \frac{1}{\sqrt{2\pi}\sigma_t} \mathrm{e}^{-0.5\left(\frac{d(i,t_{\text{in}})}{\sigma_t}\right)} \tag{8}$$

Here we assume that the input trajectory is a noisy input from GPS samples. σ_t is the standard deviation of the sub-trajectory distance. For now take, $\sigma_t = \sigma_p$, where σ_p is the standard deviation of the GPS sampling a location (value is 15.61, which is the 95-th percentile of GPS considering 30 m error). It should ideally be standard deviation introduced when we compute distance between 100 points of a path

For each of the methods compared, we plot the CDF of the error of the predicted destination for the Top-3 Closest clusters to each of the query trajectories. Each of the graphs contain 4 plots, each one varying the number of sample points given to the query trajectory. We have plotted the errors for query trajectories with 10%, 25 %, 50% and 90% of the sample points.

Some of the observations from the next location prediction results are as follows:

Our Approach and LP predict the destination of *82%* of the trajectories with less than 10km error for closest cluster. DTW also is very close with prediction *80%* of the trajectories, but SWARM fares very badly with just *5%* of the trajectories'

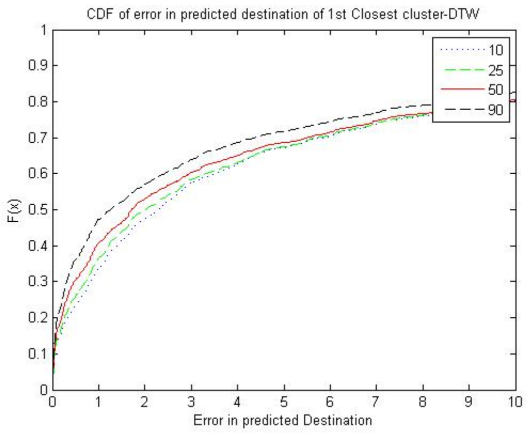

Fig. 17. CDF of error in predicted destination for 1st closest cluster using DTW

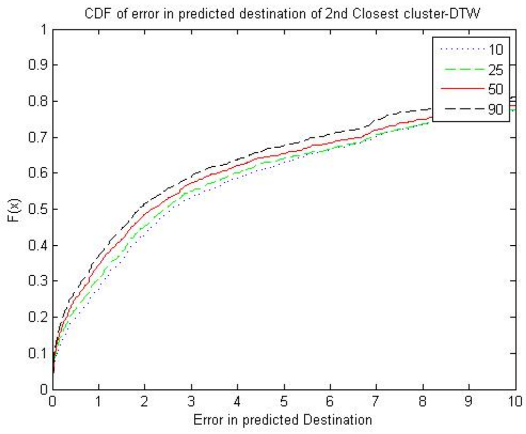

Fig. 18. CDF of error in predicted destination for 2nd closest cluster using DTW

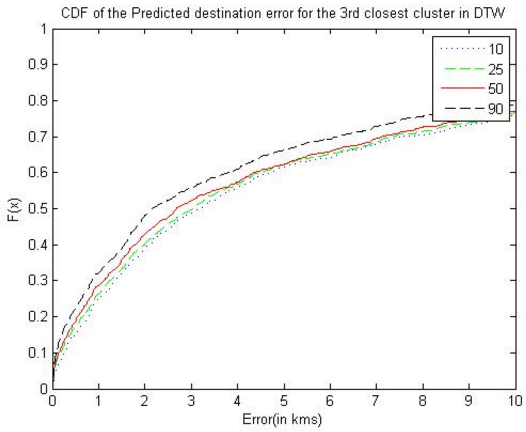

Fig. 19. CDF of error in predicted destination for 3rd closest cluster using DTW

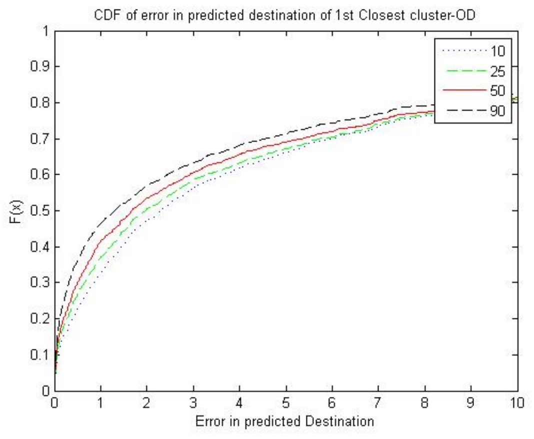

Fig. 20. CDF of error in predicted destination for 1st closest cluster using OD

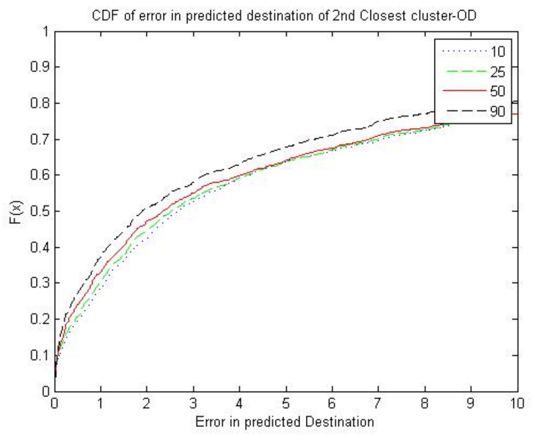

Fig. 21. CDF of error in predicted destination for 2nd closest cluster using OD

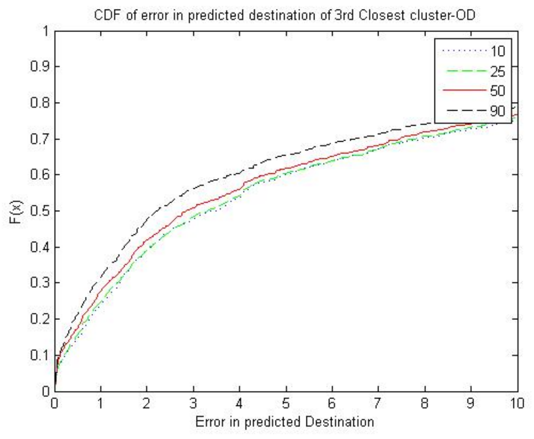

Fig. 22. CDF of error in predicted destination for 3rd closest cluster using OD

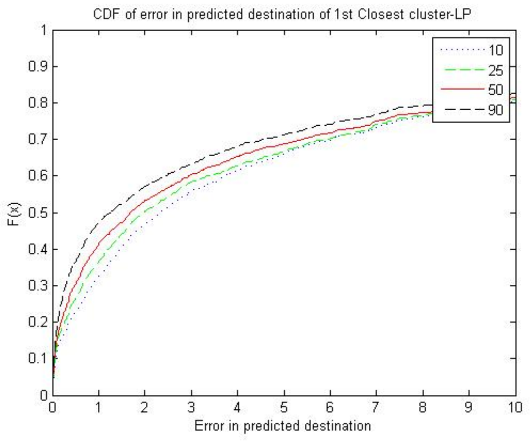

Fig. 23. CDF of error in predicted destination for 1st closest cluster using LP

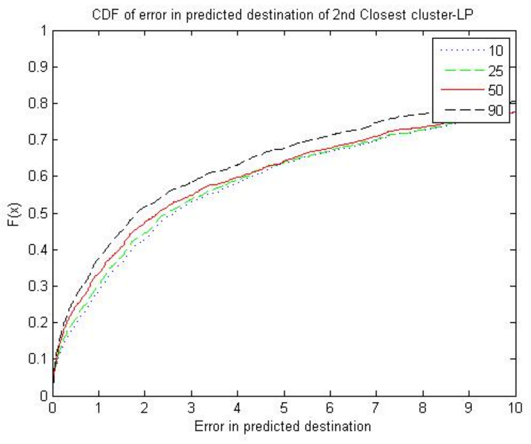

Fig. 24. CDF of error in predicted destination for 2nd closest cluster using LP

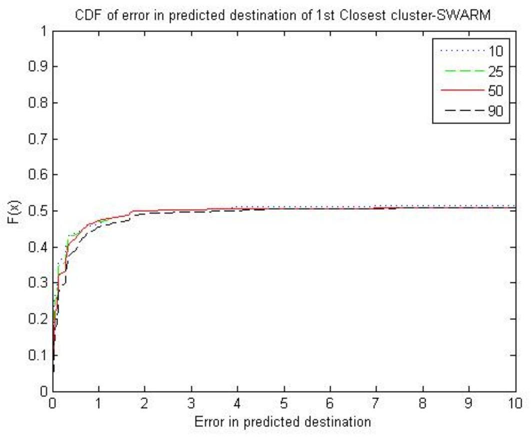

Fig. 25. CDF of error in predicted destination for 1st closest cluster using SWARM

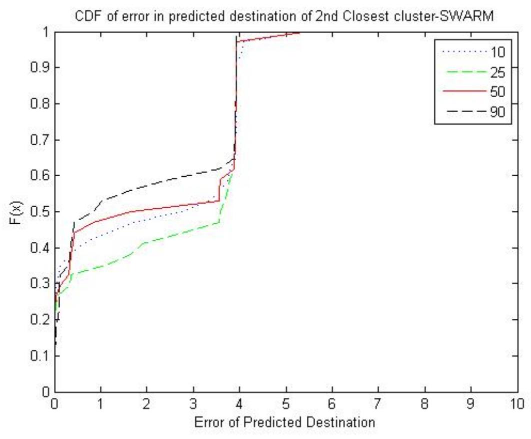

Fig. 26. CDF of error in predicted destination for 2nd closest cluster using SWARM

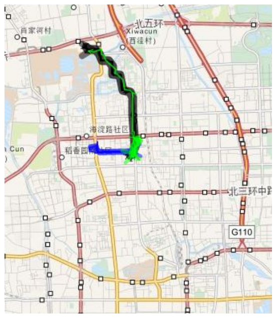

Fig. 28. Snapshot of clusters reported by SWARM

destination predicted. SWARM cannot detect the third closest cluster for any of the input trajectories.

F. Visualization at various granularity

G. Case Study: Reported Final Clusters from each method

We take up a sample case, and show the snapshots of the final clusters as reported by all the different methods. DTW and OD have similar final clusters whereas SWARM reports only final clusters. s

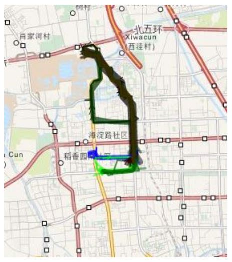

Fig. 29. Snapshot of clusters reported by OD

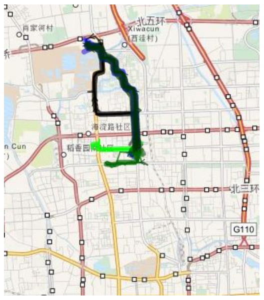

Fig. 30. Snapshot of clusters reported by DTW

VI. Conclusion

VII. Motivation

A. Trajectory Similarity

a) Motivation behind giving weight age to OD : One of the contributions of this paper, is the similarity metric that we use to define the similarity between two trajectories. The intuition behind the similarity measure is that whenever humans move, there is an intent behind the trip. Thus, the origin and destination have an important role to play. If someone is making a trip, the destination has to be of some importance to the person, and the origin should also mean something to him. Following this intuition, we have given more weightage to the points closer to the origin and destination. Another reason behind doing this is that we want to overlook tiny diversions in the route taken from a set pair of origin-destination. For example, for a user, if the trips he make from the office to his home are considered as one summary, and on some day if he takes a tiny diversion in the form of a by-lane rather than the main road, it should still be considered in the same trip summary. Thus, the points closer to the origin and destination are given more importance than those in the middle.

b) Problems with existing metrics: There are various existing metrics for computing similarity between trajectories like Dynamic Time Warping(DTW), Edit Distance on Real Sequences (EDR), Longest Common Sub-sequence(LCSS), etc. But the major problem with most of them is that they are not mathematical metrics and thus, do not follow triangle inequality. This might leads to inconsistent results in various situations, and mainly affect the clustering results. For example

- Toy scenario to show DTW/LCSS gives wrong measurement (triangular inequality in a 3 traj clustering)

c) Problems with defining similarity when both spatial and temporal domain come into the picture: Moving into the the domain of defining a spatio-temporal similarity between two trajectories brings in various questions of ambiguity. Are simiarlities between two trajectories which go on the same path 10 mins part same as that of two trajectories which go 1 km apart at the same time? Hence, a hierarchical approach for solving this problem is suggested so as to decouple the spatial and temporal similarities. We first come up with the trip summaries by looking only at the spatial values, and then

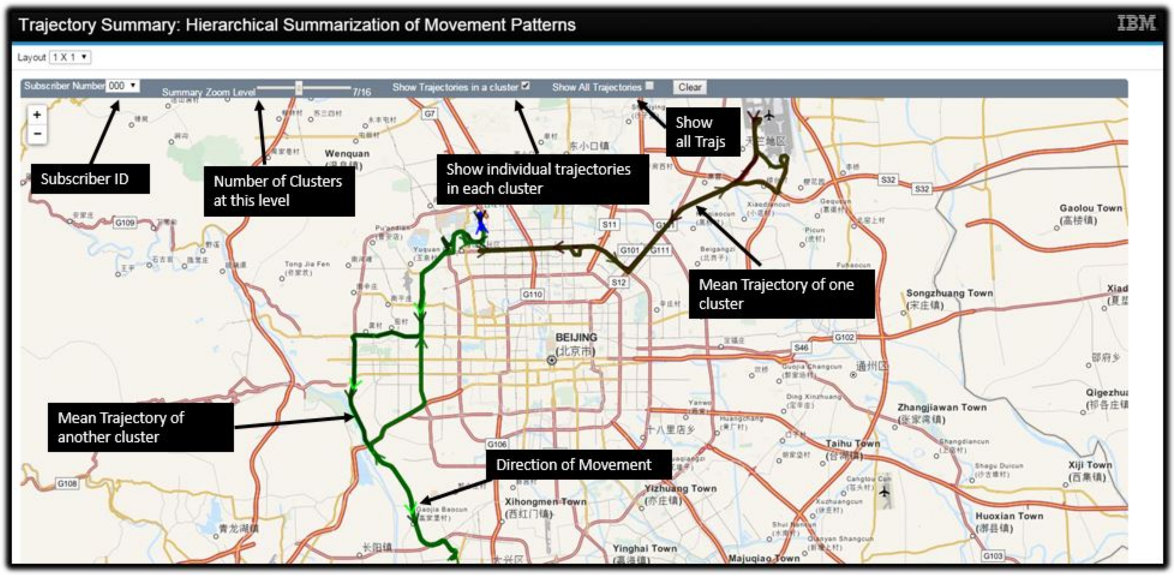
Trajectory Summary: Hierarchical Summarization of Movement Patterns
IBM
Layout 1 X 1
Subscriber Number 000
Summary Zoom Level 7/16
Show Trajectories in a cluster
Show All Trajectories
Clear
Subscriber ID
Number of Clusters at this level
Show individual trajectories in each cluster
Show all Trajs
Mean Trajectory of one cluster
Mean Trajectory of another cluster
Direction of Movement
BEIJING

Fig. 27. Visualization at various zoom levels

in the next stage , bring in the temporal (time of the day aspect) aspect to gain further insights.

d) Denoising over using similarity measures resilient to noise: For human trip movement, denoising can be done prior to trajectory processing, instead of making the similarity measures resilient to noise. Such similarity measures are computationally expensive, and do not yield accurate results in all cases.

B. Trajectory Clustering

Why hierarchical: We really dont know the number of clusters. We need to iterate over each k and then find out the optimal k. The time complexity of running k-means for 100's of ks and then finding out the optimal k is much more expensive than doing 1-shot hierarchical clustering and finding a good point to cut.

C. Trajectory Summarization

Use cases for trajectory summarization

Computing the trajectory or trip summaries of a person can answer various queries about the person. Some of them include

- Customer Profile: Give summary of a person's trips in a region – both in space or time
- Next-Location Profile: What are the most probable trajectories to find a person between time x to time y (optional: given that the person is at location L at time t)
- Alerts: Alert when a customer is moving in an anomalous way

Trip summaries can also be used for

- Insurance Usecase: Give summary of a person's trips that have good or bad speed profiles – immaterial of the space or time

D. Storing the summaries of a person

In this paper, we also propose a method of storing the trip summaries of a person in a hierarchical fashion. By doing this, we can get an idea about how the person moves at various levels of granularity. We can also set the number of clusters to a particular value, and query his movement pattern for the top k prominent trips. This kind of storage also supports zooming into a particular summary and breaking it down further for other analytics.

VIII. Related Work

References

[1] Waze. https://www.waze.com/.

[2] 3GPP TS 25.413. UTRAN Iu interface RANAP signalling. http://www.3gpp.org/DynaReport/25413.htm.

[3] L. Chen and R. Ng. On the marriage of lp-norms and edit distance. In *Proceedings of the Thirtieth International Conference on Very Large Data Bases - Volume 30*, VLDB '04, pages 792–803. VLDB Endowment, 2004.

[4] L. Chen, M. T. Özsu, and V. Oria. Robust and fast similarity search for moving object trajectories. In *Proceedings of the 2005 ACM SIGMOD International Conference on Management of Data*, SIGMOD '05, pages 491–502, New York, NY, USA, 2005. ACM.

[5] T. M. T. Do and D. Gatica-Perez. The places of our lives: Visiting patterns and automatic labeling from longitudinal smartphone data. *Mobile Computing, IEEE Transactions on*, 13(3):638–648, March 2014.

[6] S. Kurtek, A. Srivastava, E. Klassen, and Z. Ding. Statistical modeling of curves using shapes and related features. *Journal of the American Statistical Association*, 107(499):1152–1165, 2012.

[7] J.-G. Lee, J. Han, and K.-Y. Whang. Trajectory clustering: A partition-and-group framework. In *Proceedings of the 2007 ACM SIGMOD International Conference on Management of Data*, SIGMOD '07, pages 593–604, New York, NY, USA, 2007. ACM.

[8] Z. Li, B. Ding, J. Han, and R. Kays. Swarm: Mining relaxed temporal moving object clusters. *Proc. VLDB Endow.*, 3(1-2):723–734, Sept. 2010.

[9] A. J. Richardson, E. S. Ampt, and A. H. Meyburg. *Survey Methods for Transport Planning*. Eucalyptus Press, 1995.

[10] M. Vlachos, G. Kollios, and D. Gunopulos. Discovering similar multidimensional trajectories. In *Data Engineering, 2002. Proceedings. 18th International Conference on*, pages 673–684, 2002.

[11] B.-K. Yi, H. Jagadish, and C. Faloutsos. Efficient retrieval of similar time sequences under time warping. In *Data Engineering, 1998. Proceedings., 14th International Conference on*, pages 201–208, Feb 1998.

[12] J. Yuan, Y. Zheng, X. Xie, and G. Sun. T-drive: Enhancing driving directions with taxi drivers' intelligence. *Knowledge and Data Engineering, IEEE Transactions on*, 25(1):220–232, Jan 2013.

[13] Y. Zheng, Y. Chen, X. Xie, and W.-Y. Ma. Geolife2.0: A location-based social networking service. In *Mobile Data Management: Systems, Services and Middleware, 2009. MDM '09. Tenth International Conference on*, pages 357–358, May 2009.

[14] Y. Zheng, Q. Li, Y. Chen, X. Xie, and W.-Y. Ma. Understanding mobility based on gps data. In *Proceedings of the 10th International Conference on Ubiquitous Computing*, UbiComp '08, pages 312–321, New York, NY, USA, 2008. ACM.