COMP336 Assignment2 part2 Report

Q7. Dissimilarity Measure

1. Main idea

Interpret the day trajectory as a 3-D curve, calculate the flow integrals of the curve based on each dimension and form them as a 3-D vector, the dissimilarity between the trajectory is defined as the distance between the vectors.

2. Symbols

The decimal part of timestamp	t
Longitude	x
Latitude	y
Speed threshold	s
The i_{th} trajectory	T_i
Curve formula of T_i	F_i
Integral vector of T_i without weight	u_i
Weighted Integral vector of T_i	v_i
The dissimilarity between T_i, T_j	d_{ij}

3. Given Data

A set of trajectories. Each trajectory is a list of 3-D points ordered by timestamps.

4. Assumption

Users conduct uniform linear motion between two adjacent timestamps.

5. Preprocessing

- Since the trajectories are separated by days, we can only keep the decimal part of timestamps to simplify the calculation. Thus, $t \in [0,1)$.
- Mark the points with duplicate t_m in a trajectory. Keep one point that makes the average speed in $[t_{m-1}, t_m)$ closest to the average speed in $[t_{m-1}, t_{m+1})$, and drop other points.
- Go through the trajectory, calculate the average speed between any two adjacent points. If the speed > s (user-defined, by default 950km/h which is the speed of airliners), mark the latter point as an error and delete it.

6. Detailed Steps

i. Use line segments to connect any two adjacent points $< t, x, y >$ in a trajectory T_i. We can get the 3-D curve for T_i and its formula F_i (assume there are n points in T_i):

$$F_i = \begin{cases} a_1 t + b_1 x + c_1 y + b_1, & t \in [t_1, t_2) \\ a_2 t + b_2 x + c_2 y + b_2, & t \in [t_2, t_3) \\ \cdots & \\ a_{n-1} t + b_{n-1} x + c_{n-1} y + b_{n-1}, & t \in [t_{n-1}, t_n) \end{cases}$$

ii. Calculate the flow integral of the curve F_i on t, x, y respectively. We define the vector

of the flow integral as the sum of all the vectors of the flow integrals of all the line segments. Take the flow integral on x as an example:

$$\int F_i d_x = \sum_{k=1}^{n-1} \left[\int_{L_k} (a_k t + b_k x + c_k y) d_x, \qquad t \in [t_k, t_{k+1}) \right]$$

$$where\ L_k\ is\ the\ k\ th\ line\ segment\ with\ the\ direction\ from\ point\ k\ to\ point\ k+1$$

Same for the flow integral on t and y.

All the line segments have the property of one-to-one correspondence with regard to t, x, y. Hence there's no need to worry about the situation where one x or y may correspond to multiple values in F_i when computing the integral.

Now we can obtain the "integral vector" of curve F_i: $u_i = < \int F_i d_t , \int F_i d_x , \int F_i d_y >$.

Since the flow integral handles direction, $u_i \neq \mathrm{u_j}$ if trajectories T_i, T_j have the same shape but of different directions.

iii. Users can define the weight vector $w = < w_t, w_x, w_y >$ (by default $< 1,\ 1,\ 1 >$) if they need to attach different importance to t, x and y. w is the same for all the trajectories. Multiplying w and u_i element-wisely, we obtain the weighted integral vector:

$$v_i = < w_t * \int F_i d_t,\ \ w_x * \int F_i d_x,\ \ w_y * \int F_i d_y\ >$$

The dissimilarity d_{ij} between two trajectories T_i, T_j is the distance (user-defined, by default Euclidean Distance) between the two vectors v_i, v_j.

7. Analysis

(Since we not care about the exact dates of trajectories but the pattern of a trajectory of one day, t and $timestamp$ below only means the decimal part of the original timestamp in the dataset. The analysis is based on the default choice of parameters.)

- **7.1 $\boldsymbol{T_i \equiv T_j \Leftrightarrow \mathrm{F_i} = \mathrm{F_j}}$**

  Two trajectories are considered to be the same if and only if the curves they shape in the 3-D space are the same, that is, one trajectory can be reformulated by the timestamps of the other trajectory with exactly the same direction, speed and position within every single timespan as the other trajectory. For example:
  $\{(.00,0,0), (.20,2,2), (.30,4,4)\} \equiv \{(.00,0,0), (.10,1,1), (.20,2,2), (.25,3,3), (.30,4,4)\}$

- **7.2 $\boldsymbol{d_{ij} = 0 \Leftrightarrow T_i \equiv T_j}$**

  Different trajectories may shape the same curve. Thus T and F have the relationship of many-to-one correspondence. Since we conduct the integral of the curve on each dimension, F and v are one-to-one correspondence.

  The dissimilarity is the distance between v. Hence, $d_{ij} = 0 \Leftrightarrow \mathrm{F_i} = \mathrm{F_j} \Leftrightarrow T_i \equiv T_j$,

  that is, the dissimilarity between two trajectories equals zero if and only if the two trajectories are considered to be the same.

- **7.3 Symmetricity: $d_{ij} = d_{ji}$**

  This dissimilarity measure maps a trajectory (T_i) to a point (v_i) in 3-D space. The dissimilarity between trajectories is defined as the distance between the points. Since distance has the property of symmetricity (i.e., distance from point v_i to point v_j equal to the distance from point v_j to point v_i), the dissimilarity is also symmetric.

- **7.4 Robustness**
  - **Different timestamps and different number of observations**

    Since this measure uses integral of the curve, the difference in values of t or the number of points won't cause problems. A shorter timespan or path length will lead to a smaller v_i, and it can be used to calculate correctly the dissimilarity with v_j which may stand for a trajectory with a longer timespan or path length.

    Furthermore, any trajectories with a standard format should be able to fit into this measure.

  - **Duplicated timepoints**

    Redundant points with duplicated timestamps are handled in the second step in preprocessing.

  - **Adding timepoints in the middle that do not really change the trajectory**

    As stated in 7.1, trajectories with a different number of points but actually are the same form the same curve F and further the same v. Thus dissimilarity between these trajectories is zero and they have the same dissimilarity with any other trajectory. Hence, adding timepoints that do not really change the trajectory will not cause problems.

  - **Wrong record due to GPS error**

    As stated in the third point in preprocessing, this measure could identify mistakes in observations, such as a user traveling too fast (500m/s), and delete the relevant points.

- 7.5 **Time & Space Complexity of measuring the dissimilarity between two trajectories**

  Assume that the maximum number of points of a trajectory is m, computing the Euclidean Distance between 2 points in 2-D space needs time $\mathrm{O}(\beta_2)$ and $\mathrm{O}(\beta_3)$ in 3-D space, computing the integral of a line segment needs time $\mathrm{O}(\alpha)$, and it takes constant time for other basic operations.
  - Calculating the speed in preprocessing: $\mathrm{O}(m\,\beta_2)$
  - Calculating the integral of a curve: $\mathrm{O}(\mathrm{m}\alpha)$
  - Calculating the distance between vectors: $\mathrm{O}(\beta_3)$

  Hence, the time complexity for measuring the distance between two trajectories: $\mathrm{O}(\beta_3 + m\,(\beta_2 + \alpha))$.

  Overall, we need 4 lists to store the 2 trajectories and their curve functions. Assume it takes constant space to calculate integral, the space complexity should be proportional to m.

- 7.6 **Suitability**
  - In general, this measurement can calculate the dissimilarity between any two trajectories.
  - This measurement has the assumption that users conduct uniform linear motion between any two adjacent timestamps. Therefore, it needs a small timespan between two adjacent timestamps. Otherwise, if the timespan is one hour and the user first travels by car then by walking, this measurement will fail to capture the difference of features between trajectories. Fortunately, most adjacent timestamp in this dataset has a small timespan of 2 or 3 seconds.
  - As shown in 7.5, the measurement has the time and space complexity are proportional to the number of points in the trajectory. Thus if an efficient way of computing flow integrals can be found, the measurement can also have high efficiency.

Q8. Satisfying the triangle equality

As stated in 7.3, this measurement maps a trajectory to a vector in 3-D space and uses the distance between vectors to demonstrate the dissimilarity between trajectories. If the distances between vectors satisfy the triangle equality, e.g., Euclidean Distance, the dissimilarities between trajectories satisfy the triangle equality as well.

Q9. Clustering algorithms

Clustering trajectories under this measure is the same as clustering the corresponding three-dimensional vectors of the trajectories. DBSCAN is recommended to be applied to group daily trajectories with similar patterns for each user.

- **Hierarchical Agglomerative Clustering (HAC)**
  - Single linkage performs well on nonglobular data and is sensitive to noise.
    Average and complete linkage perform well on cleanly separated globular clusters.
  - HAC has a high time complexity of $\ominus(N^3)$ or $\ominus(N^2 logN)$, and high space complexity of $\ominus(N^2)$.

  HAC may work well on users with a small number of daily trajectories, if vectors in the space are nonglobular, or globular and cleanly separated. Since we do not know how the vectors are distributed in the space and the dataset is quite large with some users having nearly 1000 trajectories, HAC may not be suitable for this case.

- **K-Means Clustering**
  - May need to find the best fit value of K (by plotting of k versus total WCV) for each user in the dataset.
  - Only support Euclidian Distance to measure the distance between the vectors, and thus not very robust towards outliers.
  - Can only create boundaries that are linear and equidistant between centroids.
  - Perform poorly on interlocking clusters, nested clusters etc.

  Though K-Means Clustering has a low time complexity linear to N, we may struggle with

choosing different values of K for different users. The boundaries of clusters of the vectors in 3-D space probably won't be linear. It also restricts the choice of distance measures. Therefore, K-Means Clustering may not be the best choice for this dissimilarity measure.

- **DBSCAN**
  - Can discover clusters of arbitrary shape, but still can fail in case of a neck type of dataset (e.g., clusters with the shape of "8")
  - Able to identify noise data while clustering.
  - Advanced implementations can achieve low time complexity of $\mathrm{O}(logN)$ and low space complexity of $\ominus(N)$.
  - We may be able to find a best-fit value of the parameter Eps for all the users by randomly selecting sample users, computing their best Eps and taking their average value as the final Eps for all the users.

Given the efficiency of algorithms and the support of cluster shapes, DBSCAN should be most suitable for the task of grouping similar trajectories for each user.