电子科技大学
University of Electronic Science and Technology of China

Outlier Detection of Trajectory

Yuanliang Zhang

Data Mining Lab, Big Data Research Center, UESTC
Email：junmshao@uestc.edu.cn
http://staff.uestc.edu.cn/shaojunming

Outline

- Introduction
- Related Algorithms
- Conclusion & Discussion

Part 1 Introduction

Introduction

- Marathon

Introduction

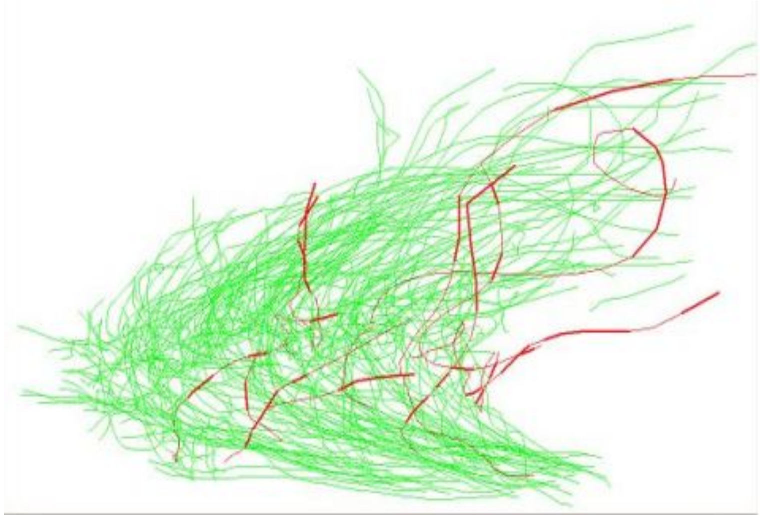
- Two significant concepts
  - Trajectory
  - Trajectory Outlier(Anomaly)

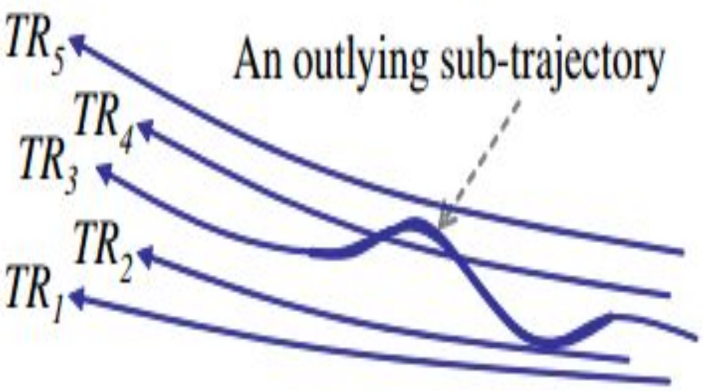
Introduction

- Trajectory:
  - A trace generated by a moving object
    - Moving objects: human, vehicles, animals, natural phenomena
    - GPS data
    - Location-based data : social network, credit card records
  - Represented by a series of chronologically ordered points
    - p1 → p2 → · · · → pn, where p = (x, y, t)

Introduction

- Trajectory outliers
  - A trajectory or <u>a segment of trajectory</u>
  - Condition 1: significantly different from other items in terms of some <u>similarity metric</u>
  - Condition 2: do not conform to <u>an expected pattern</u>

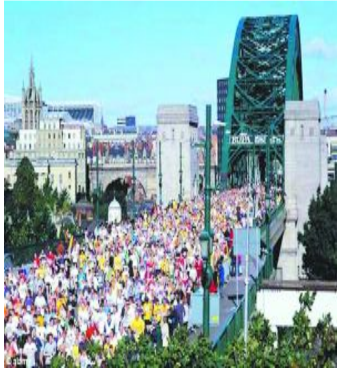




Introduction

- Marathon:
  - Which trajectories are outliers ?
  - Condition 1:

  - Condition 2:



Introduction

- Applications
  - Transportation supervision
  - social network
  - Credit card fraud
  - ……

数据挖掘实验室
Data Mining Lab

Part 2 Related Algorithms

Related Algorithm

- Distance-based
  - *Distance-based outliers: algorithms and applications （Edwin M. Knorr et al. VLDB 2000）*
- LOF
  - *LOF: Identifying Density-Based Local Outliers （Markus M. Breunig et al. SIGMOD 2000）*
- LOCI
  - *LOCI: Fast Outlier Detection Using the Local Correlation Integral (Spiros Papadimitriou et al. ICDE 2003)*

Related Algorithm

- TRAOD
  - *Trajectory Outlier Detection: A Partition-and-Detect Framework (Jae-Gil Lee et al. ICDE 2008)*
- *ROAM*
  - *ROAM: Rule- and Motif-Based Anomaly Detection in Massive Moving Object Data Sets (Xiaolei Li et al. SIAM 2007)*
- TOD-SS
  - *Trajectory Outlier Detection Algorithm Based on Structural Features （Guan YUAN et al.   Journal of Computational Information Systems 2011）*

Distance-based

数据挖掘实验室 Data Mining Lab

- Basic idea:
  - Summarize trajectory global information to a vector(point)
  - Distance function: weighted sum
  - DB(p,D)-outlier detection

- Summarization

$$P = \begin{bmatrix} P_{\text{start}} \\ P_{\text{end}} \\ P_{\text{heading}} \\ P_{\text{velocity}} \end{bmatrix}$$

where:

$$P_{\text{start}} = (x_{\text{start}}, y_{\text{start}}),$$
$$P_{\text{end}} = (x_{\text{end}}, y_{\text{end}}),$$
$$P_{\text{heading}} = (avg_{\text{heading}}, max_{\text{heading}}, min_{\text{heading}}),$$
$$P_{\text{velocity}} = (avg_{\text{velocity}}, max_{\text{velocity}}, min_{\text{velocity}}).$$

Distance-based

- Distance function:
  - Weighted sum

$$D(P_1, P_2) =$$

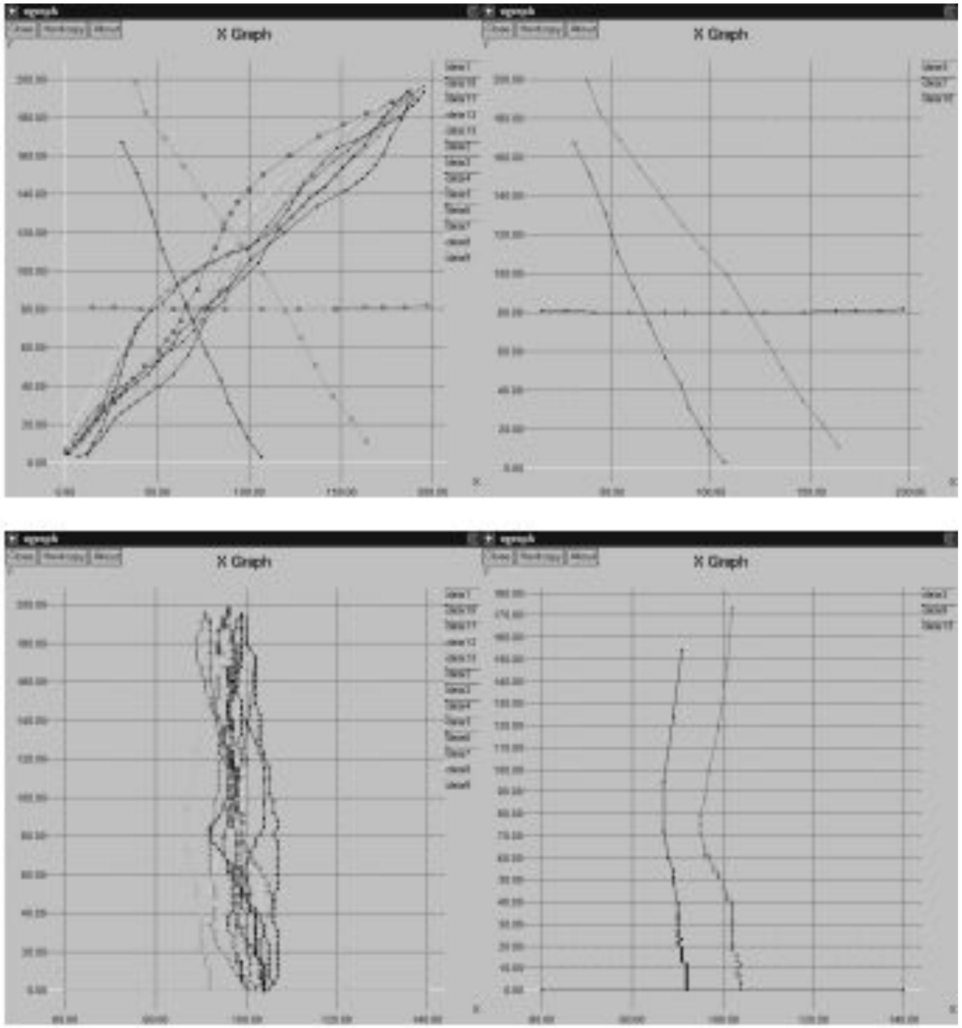
$$\begin{bmatrix} D_{\text{start}}(P_1, P_2) \\ D_{\text{end}}(P_1, P_2) \\ D_{\text{heading}}(P_1, P_2) \\ D_{\text{velocity}}(P_1, P_2) \end{bmatrix} * \begin{bmatrix} w_{\text{start}} & w_{\text{end}} & w_{\text{heading}} & w_{\text{velocity}} \end{bmatrix}$$

where $\omega_{start}, \omega_{end}, \omega_{heading}, \omega_{start}, \omega_{velocity}$ are user defined and appliication-dependent

D(P1,P2): Euclidean distance

- DB(p,D)-outlier
  - An object O in a dataset T is a DB(p, D)-outlier if at least fraction p of the objects in T lies greater than distance D from O.
    - P, D are user defined

Distance-based



Distance-based

- Drawbacks
  - Trajectory whole(global) information
    - Can't detect outlying portions
    - Doesn't consider different density distributions

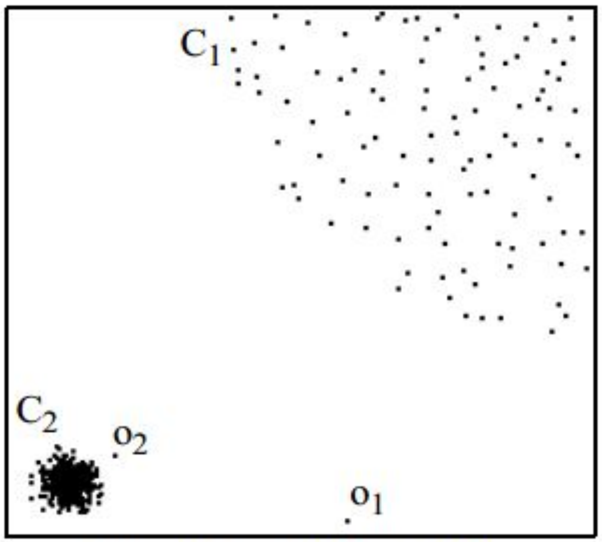


Figure 1: 2-*d* dataset DS1

LOF

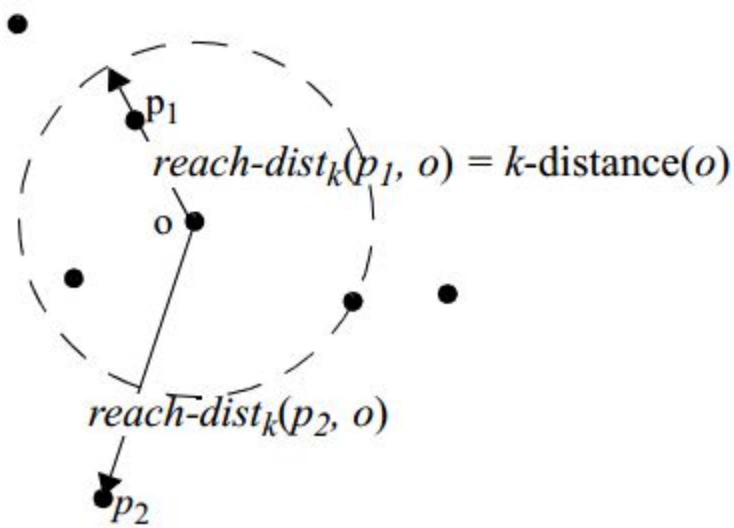
- Basic idea:
  - Regard trajectory as point
  - local outlier factor (LOF) : degree of being an outlier
  - Depends on the local density of its neighborhood
- Preliminary
  - k-distance(p):
    - for at least k objects o’∈D \ {p} it holds that d(p,o’) ≤ d(p,o)
    - for at most k-1 objects o’∈D \ {p} it holds that d(p,o’) < d(p,o)
  - k-distance neighborhood of an object p

$$N_{k-distance(p)}(p) = \{ q \in D \backslash \{p\} \mid d(p, q) \leq k\text{-distance}(p) \}$$

LOF

- Preliminary
  - Reachability distance of an object p w.r.t. object o

$$\text{reach-dist}_k(p, o) = \max\{\text{k-distance}(o), d(p, o)\}$$



LOF

- local reachability density of an object p

$$lrd_{MinPts}(p) = 1 \Bigg/ \left(\frac{\sum_{o \in N_{MinPts}(p)} reach\text{-}dist_{MinPts}(p, o)}{|N_{MinPts}(p)|} \right)$$

  - MinPts : a minimum number of objects, user defined

- (local) outlier factor of an object p

$$LOF_{MinPts}(p) = \frac{\sum_{o \in N_{MinPts}(p)} \frac{lrd_{MinPts}(o)}{lrd_{MinPts}(p)}}{|N_{MinPts}(p)|}$$

  - For objects inside a cluster: LOF value is approximately 1
  - For other objects: give tight lower and upper bounds on the LOF value

LOF

- Discussion
  - Assign to each object a degree of being an outlier
  - Density based
  - MinPts is non-trivial

LOCI

- Basic Idea
  - Density based
  - Multi-granularity deviation factor(MDEF)
  - Outlier detect:
    - MDEF value deviates significantly (more than three standard deviations) from the local averages

LOCI

- Preliminary
  - **N(pi, r):** the set of r-neighbors of pi

    N(pi, r) ≡ {p ∈ P | d(p, pi) ≤ r}
  - **n(pi, r):** the number of r-neighbors of pi

    n(pi, r) ≡ |N(pi, r)|
  - **n ˆ(pi, r, α):** average of n(p, αr) over the set of r neighbors of pi

$$\hat{n}(p_i, r, \alpha) \equiv \frac{\sum_{p \in N(p_i, r)} n(p, \alpha r)}{n(p_i, r)}$$

  - **σn ˆ(pi, r, α):**standard deviation of n(p, αr) over the set of r-neighbors

$$\sigma_{\hat{n}}(p_i, r, \alpha) \equiv \sqrt{\frac{\sum_{p \in N(p_i, r)} (n(p, \alpha r) - \hat{n}(p_i, r, \alpha))^2}{n(p_i, r)}}$$

LOCI

- MDEF (pi, r, α):Multi-granularity deviation factor for point pi at radius (or scale) r

$$MDEF\ (p_i,\ r,\ \alpha) = \frac{\hat{n}(p_i, r, \alpha) - n(p_i, \alpha r)}{\hat{n}(p_i, r, \alpha)} = 1 - \frac{n(p_i, \alpha r)}{\hat{n}(p_i, r, \alpha)}$$

- Normalized deviation

$$\sigma_{MDEF}\ (p_i,\ r,\ \alpha) = \frac{\sigma_{\hat{n}}(p_i, r, \alpha)}{\hat{n}(p_i, r, \alpha)}$$

LOCI

- LOCI outlier detection

$$MDEF(p_i, r, \alpha) > k_\sigma \sigma_{MDEF}(p_i, r, \alpha), \text{ where } k_\sigma = 3$$

```
// Pre-processing
Foreach p_i ∈ ℙ:
    Perform a range-search
      for N_i = {p ∈ ℙ | d(p_i, p) ≤ r_max}
    From N_i, construct a sorted list D_i
      of the critical and α-critical distances of p_i
// Post-processing
Foreach p_i ∈ ℙ:
    For each radii r ∈ D_i (ascending):
        Update n(p_i, αr) and n̂(p_i, r, α)
        From n and n̂, compute
          MDEF(p_i, r, α) and σ_MDEF(p_i, r, α)
        If MDEF(p_i, r, α) > 3σ_MDEF(p_i, r, α),
          flag p_i
```

TRAOD

- Basic idea
  - Partition-and-Detect Framework
  - hybrid of the distance-based and density-based approaches

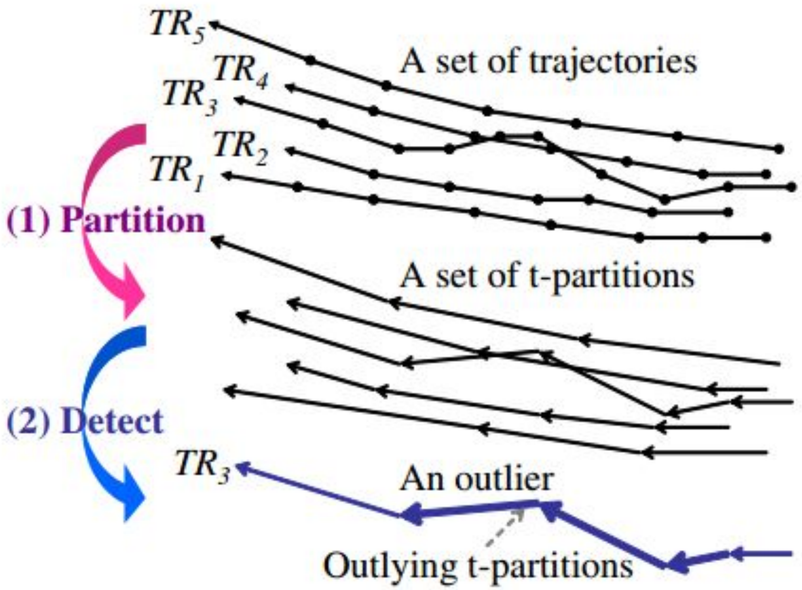


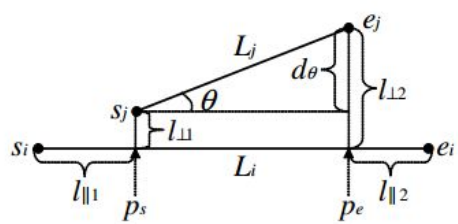
Fig. 2. An example of trajectory outlier detection in the partition-and-detect framework.

TRAOD

- Pre-definition
  - Trajectory
    - $TR_i = P_1 P_2 P_3 \cdots P_j \cdots P_{len_i}$, where P_j ($1 \le j \le len_i$) is a d-dimensional point
  - sub-trajectory
    - $P_{c_1} P_{c_2} \cdots P_{c_k}$ ($1 \le c_1 < c_2 < \cdots < c_k \le len_i$)
  - trajectory partition
    - line segment $p_i p_j$ ($i < j$)
    - t-partition
  - outlying trajectory partitions
    - t-partition does not have “enough” similar neighbors (i.e., close trajectories)

TRAOD

- Partition
  - MDL Principle
  - Distance between T-Partitions(distance function)
    - the perpendicular distance (d⊥)
    - the parallel distance (d//)
    - the angle distance (dθ)



$$d_\perp = \frac{l_{\perp 1}^2 + l_{\perp 2}^2}{l_{\perp 1} + l_{\perp 2}}$$

$$d_\| = \mathrm{MIN}(l_{\|1}, l_{\|2})$$

$$d_\theta = \|L_j\| \times \sin(\theta)$$

Fig. 4. Three components of the distance function for line segments.

$$dist(L_i, L_j) = w_\perp \cdot d_\perp(L_i, L_j) + w_\| \cdot d_\|(L_i, L_j) + w_\theta \cdot d_\theta(L_i, L_j)$$

TRAOD

- Trajectory outlier detection
  - Close

A trajectory TR_i is *close* to a t-partition $L_j \in P(TR_j)$ $(TR_i \neq TR_j)$ if

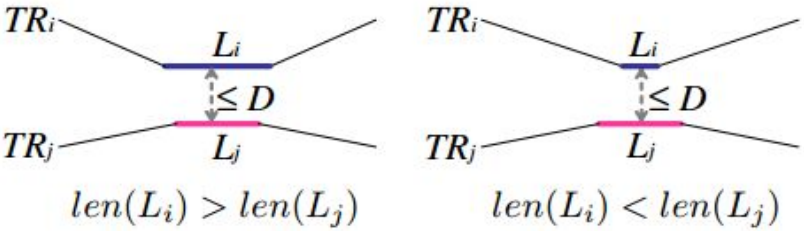
$$\sum_{L_i \in CP(TR_i, L_j, D)} len(Li) \geq len(L_j)$$

D is a parameter given by a user.

$P(TR_i): the\ set\ of\ all\ t-partitions\ of\ TR_i$

$CP(TR_i, L_j, D): the\ set\ of\ TR_i's\ t-partitions\ within$ the $dis\tan ce\ D$

$from\ L_j \in P(TR_j)\ (TR_i \neq TR_j)$



(a) TR_i is close to L_j. (b) TR_i is *not* close to L_j.

Fig. 3. The concept of the close trajectory.

TRAOD

- Trajectory outlier detection
  - Outlying

    A t-partition $L_i \in P(TRi)$ is *outlying* if

    $$|CTR(L_i, D)| \leq \lceil (1-p)|I| \rceil$$

    $|I|$ indicates the total number of trajectories.

    p is a parameter given by a user.

    $CTR(L_i, D): the\ set\ of\ trajectories\ close\ to\ L_i$

  - Outlier

    $$Ofrac(TR_i) = \frac{\sum_{L_i \in OP(TR_i, D, p)} len(L_i)}{\sum_{M_i \in P(TR_i)} len(M_i)} \geq F$$

    - Where F is a parameter given by a user

TRAOD

- Incorporation of Density: adjusting coefficient
  - Density

$$density(L_i) = \left| \bigcup_{TR_j \in I} CP(TR_j, L_i, \sigma) \right|$$

σ is the standard deviation of pairwise distances between t – partitions

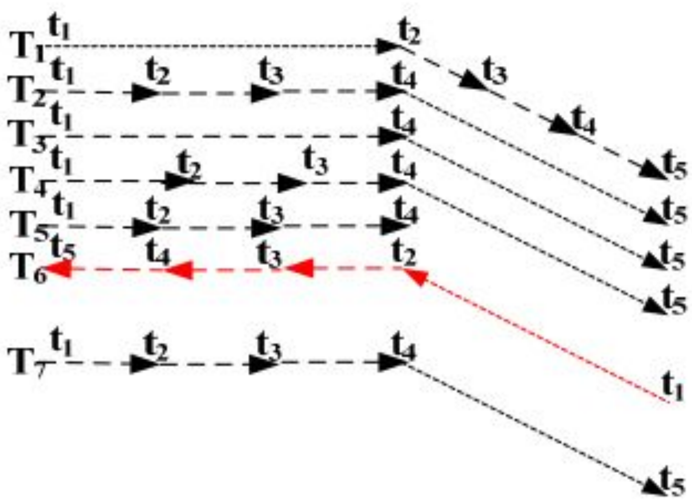
  - adjusting coefficient

$$adj(L_i) = \frac{\sum_{L_j \in L} density(L_j) \; / \; |L|}{density(L_j)}, \text{where } L = \bigcup_{TR_k \in I} P(TR_k)$$

$$adj(L_i) * |CTR(L_i, D)|$$

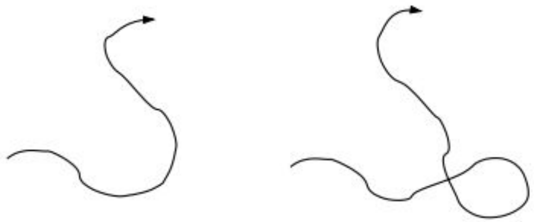
TRAOD

- Discussion
  - Detect outlying sub-trajectories
  - Without the temporal information

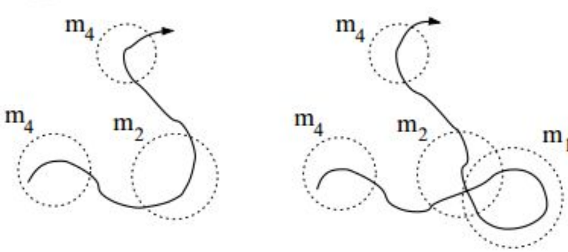


ROAM

- ROAM (Rule- and Motif-based Anomaly Detection in Moving Objects)
- Basic idea



(a) Two similar trajectories. The loop in the right trajectory is difficult to handle in holistic approaches.



(b) Same two trajectories after motif extraction. The right trajectory has an extra m_1.

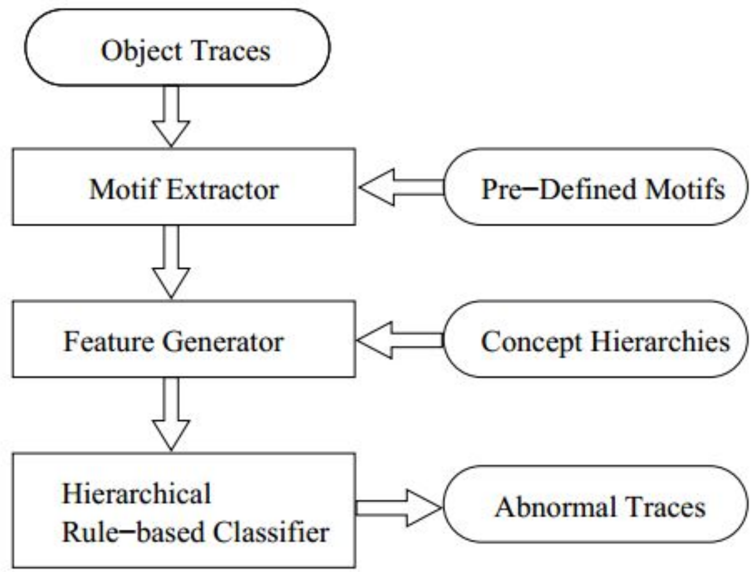


Figure 2: ROAM Framework

ROAM

- Motif Extractor
  - Sliding window
    - Window length w: user defined
    - All windows are overlaid on top of each other
  - Clustering
    - group trajectory segments into representative sets

$$(\mathrm{m}_i,\ \mathrm{t}_{\mathrm{start}},\ \mathrm{t}_{\mathrm{end}},\ \mathrm{l}_{\mathrm{start}},\ \mathrm{l}_{\mathrm{end}})$$

  - Motif expressions
    - duration, top speed, avg speed, radius, and general location

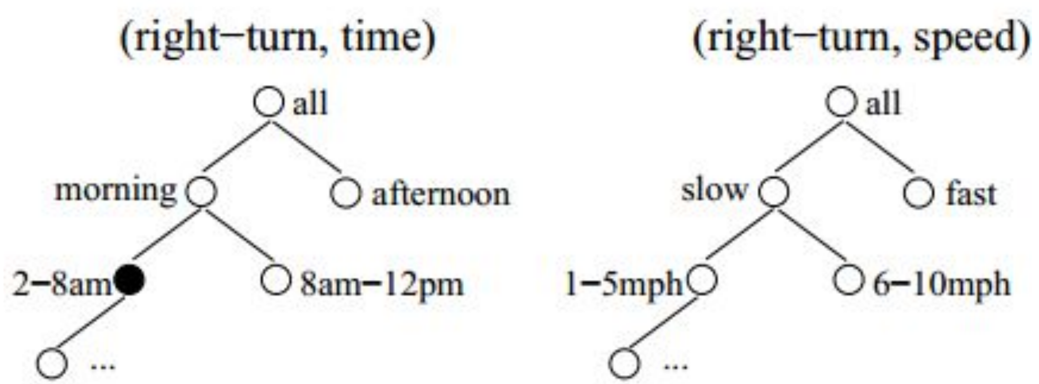
$$(\mathrm{m}_i,\ V_1,\ V_2 \ldots V_A)$$

ROAM

- Feature Generator

$$(m_i, a_j, v_k) \leftrightarrow f_x$$

  - Feature generalization
    - tree-based data structure



ROAM

- Classification
  - CHIP (Classification using Hierarchical Prediction Rules)
    - Iteratively and greedily searches for the best available rule until all positive examples are covered
    - Tries to use high-level features
  - A single rule r：
    - $l1 \wedge l2 \wedge ... \wedge ln \rightarrow C$
      - li : a literal (or predicate) of the form(feature = value)
      - C : the class label
    - An example is "covered" by r if all the literals in r are satisfied in the example

ROAM

- Classification
  - Foil Gain(l, r)
    - A rule is learned one literal at a time. Literals are selected according to a weighted version of Foil Gain
    - Based on the positive and negative coverage of the rule before and after adding the literal

$$p_1 \left(\log_2 \frac{p_1}{p_1 + n_1} - \log_2 \frac{p_0}{p_0 + n_0} \right)$$

  - p0 , n0 : the number of positive and negative examples covered by rule r without literal l
  - p1 , n1: the number of positive and negative examples covered by rule r ∧ l

ROAM

- Classification
  - Exp_Gain(f, r)
    - The maximum Foil Gain achieved by any literal in any of its child features

$$Exp\_Gain(f, r) = \max_{(l, f_i) \forall l, f_i \in Exp(f)} Foil\_Gain(l, r)$$

    - Exp(f): the set of f's children in F''s hierarchy
    - f is a feature and r is a rule

ROAM

Input: (1) Training set $\mathcal{D} = P \cup N$, where P and N are the positive and negative examples. (2) Initial feature set $\mathcal{F}_C \in \mathcal{F}'$.

Output: Set of classification rules R.

Method:

```
while not all of P is covered
    initialize new rule r
    while true
        find literal l with highest Foil_Gain(l, r)
        find feature f with highest Exp_Gain(f, r)
        if both gains < min_gain then break
        if Foil_Gain(l, r) > β · Exp_Gain(f, r) then
            add l to r
        else
            add feature f to F_C
    add r to R
return R
```

ROAM

数据挖掘实验室
Data Mining Lab

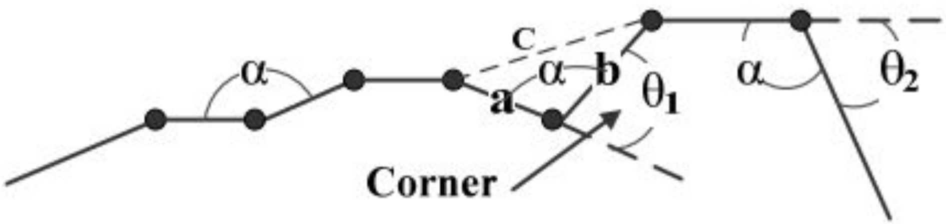
- Discussion
  - Detect outliers with multiple granularity of spatiotemporal features
  - Training set

TOD-SS

- Trajectory Outlier Detection algorithm based on Structural Similarity (TOD-SS)
- Basic idea
  - Partition trajectory : corner threshold
  - Abstract trajectory structure features
  - Outlier detection:
    - Segment outlier : SSIM(Structural Similarity) matrix
    - Trajectory outliers： the proportion of its segment outliers

TOD-SS

- The Partition of Trajectory



$$\alpha = \arccos\left((a^2 + b^2 - c^2)/2ab \right) \qquad \theta = \begin{cases} \pi - \alpha, if\,(\bar{a} \times \bar{b} \geq 0) \\ \alpha - \pi, if\,(\bar{a} \times \bar{b} < 0) \end{cases}$$

  – Partition point: |θ|>ω, ω is user defined

TOD-SS

数据挖掘实验室
Data Mining Lab

- Trajectory Structure

  Trajectory Structure <Direction, Speed, Angle, Location>

  – Direction

$$DirDist(L_i, L_j) = \begin{cases} \min(\| L_i \|, \| L_j \|) \times \sin(\varphi), if\,(0 \le \varphi \le 90) \\ \min(\| L_i \|, \| L_j \|), if\,(90 \le \varphi \le 180) \end{cases}$$

$\| L_j \|$ The length of L_j, and $j \in N \wedge 1 \le j \le |TS|$

  – Speed:

$$SpeedDist(L_i, L_j) = \frac{1}{3}\left(S_{\max}(L_i, L_j) + S_{avg}(L_i, L_j) + S_{\min}(L_i, L_j)\right)$$

  - Smax(Li, Lj)=|Vmax(Li)-Vmax(Lj)|
  - Savg and Smin： the absolute divergence of average and minimum speed.

TOD-SS

- Trajectory Structure
  - Angle:

$$AngleDist(L_i, L_j) = \frac{\sum_{1,1}^{P(L_j),P(L_j)} ((|\theta_i - \theta_j|)/(|\theta_i| + |\theta_j|))}{P(L_i) + P(L_j)}$$

$P(x)$ The number points of a trajectory or a segment, and $x \in \{TD\} | x \in \{TS\}$

  - Location

$$LocDist(L_i,\ L_j) = \max(\mathrm{h}(L_i,\ L_j),\ \mathrm{h}(L_i,\ L_j))$$

- $h(L_i, L_j) = \max_{a \in L_i}(\min_{b \in L_j}(dist(a,b)))$： Hausdorff distance

TOD-SS

- Structural Similarity
  - Feature weights : <u>user defined</u>
    - W= {WD, WS, WA, WL}
    - WD ≥ 0, WS ≥ 0, WA ≥ 0, WL ≥ 0;   WD+WS+WA+WL = 1.
  - Structural distance (SDIST)

$$SDIST(L_i,\ L_j) = (DirDist \times W_D + SpeedDist \times W_S + AngleDist \times W_A + LocDist \times W_L$$

$$SSIM(L_i,\ L_j) = 1\text{-}N(SDIST(L_i,\ L_j))$$

  - Where N(…) is a function of distance normalization

TOD-SS

- Structural Similarity
  - SSIM of a whole trajectory

$$MSSIM(TR_i, TR_j) = \frac{1}{\min(|L_i|, |L_j|)} \sum_{i=0, j=0}^{\min(|L_i|, |L_j|)} SSIM(L_i, L_j)$$

- Trajectory outlier detect
  - **ε-Neighbor:**
    - SSIM(Li, Lj)≥ε, Lj∈Nε(Li), ε is user defined
  - **Segment Outlier:**
    - ε-neighbor count |Nε(Li)| < σ,
    - σ is the neighbor threshold specified by users.

TOD-SS

- Trajectory outlier detect
  - **Trajectory Outlier:**
    - The proportion of outlier segments in TRi >=given proportion threshold ξ
    - The total similarity (MSSIM) <=threshold F

Table 1 Basic Symbols Notation.

Parameters	Description
TD, TS	Trajectory and Segment set
$\lVert TR_i \rVert$	The length of TR_i, and $(i \in N) \wedge (1 \le i \le \lvert TD \rvert)$
$\lVert L_j \rVert$	The length of L_j, and $j \in N \wedge 1 \le j \le \lvert TS \rvert$
$P(x)$	The number points of a trajectory or a segment, and $x \in \{TD\} \lvert x \in \{TS\}$
$V_{flag}(L)$	Some kinds speed of L, where $flag$ is s, e, max, min, avg, and represents start, end, maximum, minimum, average speed respectively.
$SO_{\varepsilon,\sigma}(L)$	L is the ε, σ related segment outlier
$TO_{\xi,F}(TR)$	TR is the ξ, F related trajectory outlier
θ, ω, ε, σ	Corner, corner threshold, SSIM threshold, and neighbor threshold respectively.
ξ, F	Proportion threshold and structural similarity threshold

TOD-SS

Algorithm: Similarity Computing

Input: Trajectory set $I=\{TR_1, TR_2, \ldots, TR_n\}$, ω (corner threshold), W (feature weights)

Output: Similarity matrixes M_L (Segments) and M (Trajectories)

/*First phase: Trajectory Partition*/

01: for each $TR_i \in I$ **do**

02: $TS \leftarrow$ Partition TR_i according to ω; /*Sencond phase: Computation of Structural Similarity */

03: for each $L_i, L_j \in TS \wedge i \neq j \wedge L_i.TR \neq L_j.TR$ **do**

04: Initialize SSIM(L_i, L_j);

05: SSIM_Computer(L_i, L_j)* $\{W\}$; /* multiply by weighs */

end for

06: Generate Segment Matrix M_L;

07: for each $TR_i, TR_j \in I \wedge i \neq j$ **do** /* compute similarity matrix*/

08: Initialize MSSIM(TR_i, TR_j);

09: Calculate MSSIM(TR_i, TR_j);

end for

10: Generate Trajectory Matrix M;

End.

Algorithm: Outlier Detection

Input: Similarity matrixes M_L (Segments), M (Trajectories), ε (neighbor threshold), σ, ξ, F

Output: Segment Outliers, Trajectory Outliers

01: for each *Segment Pair*(L_i, L_j) $\in M_L$ **do**

02: if $SSIM(L_i, L_j) \geq \varepsilon$ **then**

03: $N_\varepsilon(L_i) \leftarrow L_j$; /*set the ε-neighbor of L_i */

04: for each $L_i \in TS$ **do**

05: if $|N_\varepsilon(L_i)| \leq \sigma$ **then**

06: Set $SO_{\varepsilon,\sigma}(L_i) \leftarrow L_i$; /*mark L_i as an outlier*/

07: for each $TR_i \in I$ **do**

08: if Count(L is an outlier $\wedge$ $L \in TR_i$)/ TR_i.SegmentCount $> \xi \wedge$ MSSIM(TR_i) $< F$ **then**

09: Set $TO_{\xi,F}(TR_i) \leftarrow TR_i$; /*mark TR_i as an outlier*/

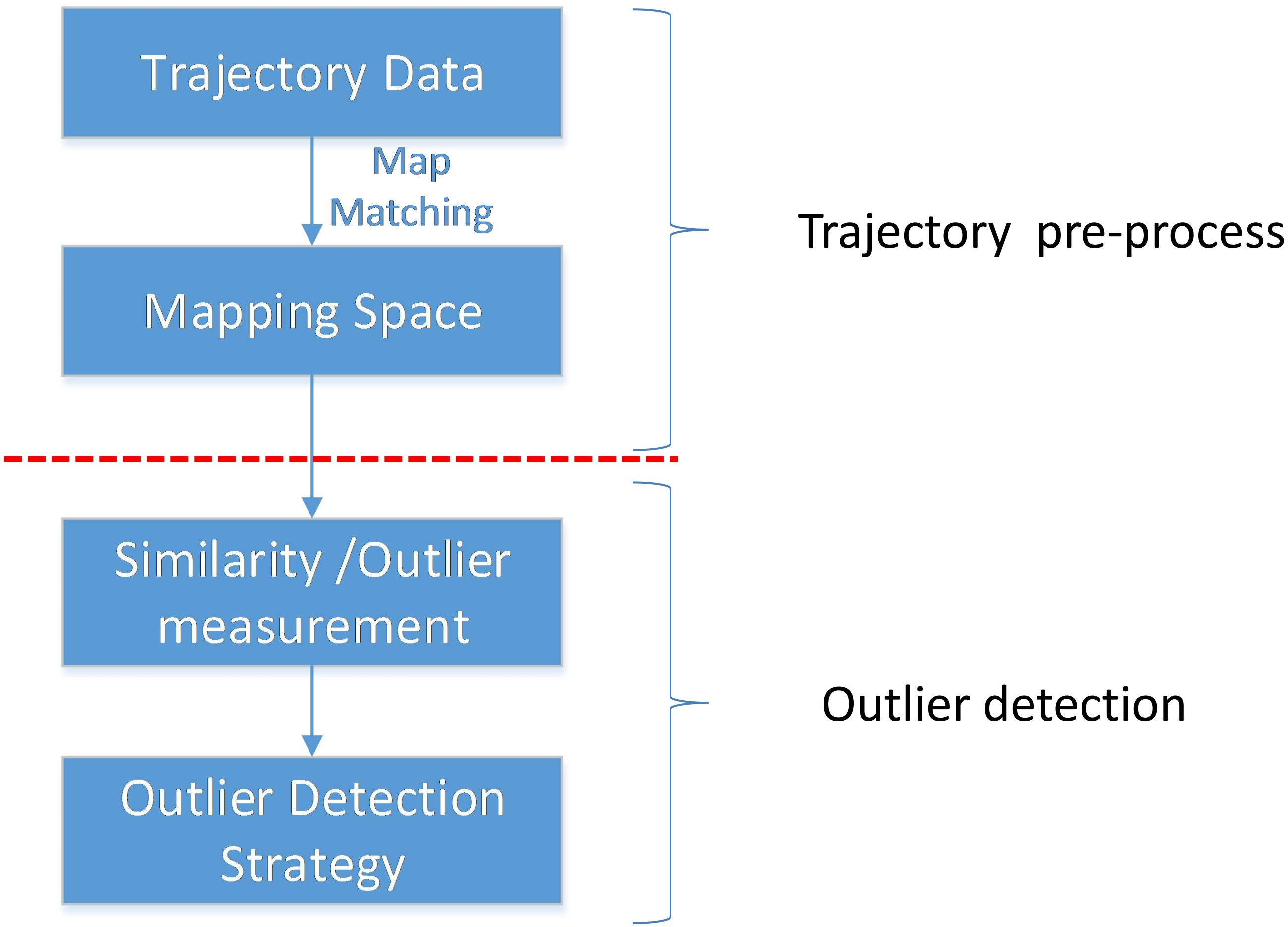
End.

Part 3 Conclusion & Discussion

Conclusion & Discussion

数据挖掘实验室
Data Mining Lab

- Basic Framework



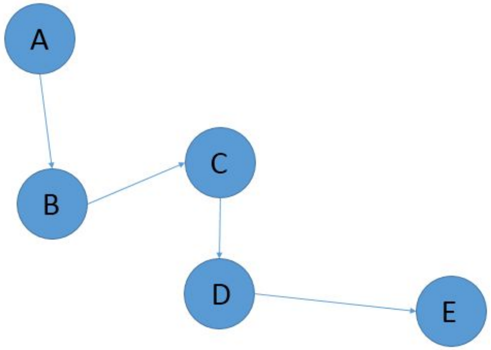
Conclusion & Discussion

- Trajectory pre-process
  - Raw data： $P(L,t)$
  - Space mapping：
    - Trajectory → Vector(Point)
      - Merit： can utilize common anomaly detection approaches
      - Defect： may exist information loss
    - Trajectory →Line segment
      - Merit： original space; intuitive
      - Defect： partition strategy ; temporal property

Conclusion & Discussion

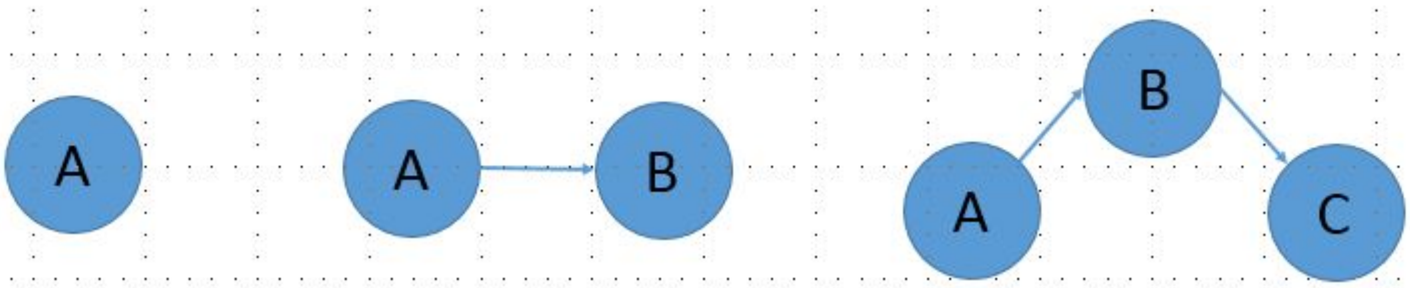
- Trajectory outlier detection
  - Doesn't have a uniform definition
  - Personal idea：
    - Location distribution
    - Temporal information
    - Semantic meanings
    - Context information

- Future work
  - Trajectory → Region of interest
  - Pattern mining



Conclusion & Discussion

- Future work
  - Pattern mining



  - Global temporal distribution
  - Similarity measurement
  - Outlier detection