EE226 期末大作业选题2: 基于相似度的轨迹嵌入表征

本选题不在Kaggle平台上开展，但各时间点与Kaggle同步。

1. 项目叙述

此项目旨在通过给定的地理轨迹和轨迹间的相似度（距离）信息（由距离函数计算得出，一般来说，距离越大，相似度越小），使用数据挖掘方法对原轨迹进行嵌入表征。地理轨迹定义如下：

$$T = \{(P_1, t_1), (P_2, t_2), \cdots, (P_l, t_l)\}$$

其中$P_i, 1 \leq i \leq l$表示第i个位置点，一般包含经纬度坐标，即$P_i = (x_i, y_i)$；t_i表示第i个位置点的时间戳。为简化问题，本项目中轨迹的时间戳信息已被删去，故T仅为经纬点序列。

轨迹间的相似度信息由距离矩阵$D = (d_{ij})_{n \times n}$表示，其中$n$为轨迹总数。$d_{ij} = dist(T_i, T_j)$，其中$dist(\cdot, \cdot)$为距离函数，本项目涉及四种距离函数：离散[弗雷歇距离]（Fréchet Distance）(https://zhuanlan.zhihu.com/p/20159963)、豪斯多夫距离（Hausdorff Distance）、动态时间规整（Dynamic Time Wrapping, DTW）和最长公共子序列（Longest Common SubSequence, LCSS）。

由于轨迹本身的长度任意，且信息密度较低（对于GPS轨迹，采样间隔为3秒左右，故相邻位置点所带的信息重复性大），为了聚合轨迹中隐含的特征信息，我们一般需要对其进行降维，也即嵌入(Embedding)。简单来说，对轨迹的嵌入即为寻找一个d维（一般远小于轨迹平均长度）的嵌入空间以及映射$T \to e \in \mathbb{R}^d$，并且使嵌入向量最大限度地保留原轨迹的信息。在本项目中，需要保留的信息即为轨迹间的相似度（距离），一般思路是使嵌入向量间的欧氏距离、内积等低计算复杂度的相似度（距离）函数近似原轨迹间的相似度，以满足下游任务快速匹配相似轨迹的需求。

在本项目中，10,000条去除时间信息的地理轨迹以及四种距离的距离矩阵（均为对称矩阵）已经给出，请你发挥自己的聪明才智，搜索参考资料，编写代码进行轨迹的嵌入表征。

2. 数据说明

本项目的数据包含五个文件，均使用Python pickle包进行读写，分别为：

- traj_coord - 包含10,000条不等长轨迹，其中每条轨迹均为经纬点序列，以列表形式存储。地理范围 $-8.735152^\circ \leq x \leq -8.156309^\circ, 40.953673^\circ \leq y \leq 41.307945^\circ$.
- traj_discret_frechet_distance - 10000×10000的轨迹离散弗雷歇距离矩阵，距离范围$[0, 1]$，以nparray形式存储。
- traj_dtw_distance - 10000×10000的轨迹DTW距离矩阵，距离范围[0, 100]，以nparray形式存储。
- traj_hausdorff_distance - 10000×10000的轨迹豪斯多夫距离矩阵，距离范围[0, 1]，以nparray形式存储。
- traj_lcss_distance - 10000×10000的轨迹LCSS距离矩阵，距离范围[0, 1]，以nparray形式存储。

3. 测试方法

每条轨迹按在文件traj_coord中的先后顺序从0编号到9999，所有距离矩阵nparray的行列索引均与该编号一致，例如，$distmartix[1000][1001]$即表示编号为1000的轨迹与编号为1001的轨迹的距离。

我们使用K近邻搜索来测试嵌入的性能。我们根据选取的距离矩阵，对每一条轨迹，取与其相似度最高（距离最小）的$m (m \leq K)$条轨迹（不包括自身）作为测试集。为了测试嵌入的性能，我们逐对计算轨迹对应的嵌入向量的距离（请同学们自己定义），并据其对每条轨迹进行K近邻搜索，并建立Top-K列表，最终使用命中率（Hit Rate, HR）$HR@K$进行测试。

$$HR@K = \frac{hits}{trajs}$$

其中$hits$表示测试集中所有相似轨迹均出现在Top-K列表中的查询轨迹数量，$trajs$表示查询轨迹总数，即$10,000$。本项目中，取$K = 50$。同学们可以在四种距离矩阵中选取一种，自行选取m（建议取值不要小于10）建立测试集，并进行对照实验测量所用方法的性能。

此外，嵌入向量的维度数d也是参考的重要指标，同等性能下，我们希望维度数尽可能低，且另一方面，较大的维数（例如，大于轨迹最大长度）反而违背了嵌入表征"降维"的目标。因此，建议的维度数选择范围为$d \in [2, 64]$，并且同等性能下，维度数较低的嵌入会获得更高的参考分数。

附加题：你能否结合四种距离矩阵和轨迹信息，得到同时保持四种度量下相似度的轨迹嵌入？

若选择参加附加题的测试，我们将会在原有的测试上，增加新的测试：对每一条轨迹，在四种距离矩阵中分别选取$m/4$条轨迹作为测试集，此后环节与上述类似，取$K = 50$。

4. 文件提交

选择本项目后，你需要注意以下时间点（与Kaggle平台一致）：

1) **4.14（第八周周三）**前，确定分组和选题，如对项目有问题，可在此时间段内反馈。

2) **4.28 （第十周周三）**前，提交中期报告，包含introduction, related work, research plan, expected outcome, options.

3) **6.9（第十六周周三）**前，提交以下材料：

- [ ] 代码输出的10,000条轨迹的d维嵌入向量，即$d \times 10000$的矩阵，建议使用nparray进行存储，同时请标明嵌入向量的何种距离（必须是欧氏距离、余弦距离、L1范数等线性复杂度的距离函数）近似了原距离。
- [ ] 完整的程序代码，如果是基于神经网络的方法，最好提供Checkpoint，我们将会运行程序，对照生成的嵌入和提交的嵌入。
- [ ] 英文项目报告，以论文形式撰写，包含introduction， related work， problem definition，algorithm， results， conclusion。
- [ ] （可选）根据嵌入自建的K近邻列表（$K = 50$）。

  提交时请注明是否参加附加题的测试，如不参加，请注明使用何种距离矩阵。

5. 参考思路

以下是参考的技术路线：

1) 矩阵降维方法。此类方法不涉及对轨迹的处理，仅对距离矩阵进行线性降维，性能比较一般，可以作为性能基线（Baseline）验证自建方法的性能。参考方法：主成分分析（Principal Component Analysis, PCA）、多维缩放（Multi-Dimension Scaling, MDS）。

2) 基于流形学习（Manifold Learning）的方法。此种方法为非线性降维方法，但依然只利用了距离矩阵的信息。参考方法：t-SNE、LLE。

3) 基于深度学习（Deep Learning）和递归神经网络（Recurrent Neural Networks, RNNs）的方法。此方法使用递归神经网络对轨迹进行处理，同时部分方法还使用了度量学习（Metric Learning）的思想，使用距离矩阵作为构建损失函数的指导。参考方法：自动编码器（Autoencoders）、t2vec（ICDE2018）、NeuTraj（ICDE2019）、Traj2SimVec（IJCAI2020）。

4) 基于深度学习和图神经网络的方法。由于距离矩阵和图的邻接矩阵的相似性，可以根据距离矩阵，以轨迹为节点、轨迹间的相似关系为边建立轨迹相似度图，并使用图神经网络进行节点嵌入的学习。此方法的核心之一即轨迹相似度图的构建，由于轨迹间的相似关系并不明确，因此需要自主确定邻接矩阵的构建原则。参考方法：DeepWalk（KDD2014）、Node2Vec（KDD2016）、GCN（ICLR2017）、

GAT（ICLR2018）。