Identifying and predicting social lifestyles in

people's trajectories by neural networks

In this research, we exploit repeated parts in daily trajectories in people's movements, which we refer to as mobility patterns, to train models to identify and predict a person's lifestyles. We use cellular data of a group ("society") of people and represent a person's daily trajectory using semantic labels (e.g., "home", "work", and "gym") given to the main places of interest (POI) he has visited during the day, as determined collectively based on interviewing all people of the group. First, in an unsupervised manner using a neural network (NN), we embed POI-based daily trajectories that always appear together with others in consecutive weeks and identify the result of this embedding with social lifestyles. Second, using these lifestyles as labels for lifestyle prediction, user POI-based daily trajectories are used to train a convolutional NN to extract mobility

patterns in the trajectories and a dynamic NN with exible memory to assemble these patterns to predict a lifestyle for a trajectory never-seen-before. The two-stage algorithm shows model accuracy and generalizability in lifestyle identi\_cation and prediction (both for a novel trajectory and a novel user) that

are superior to those shown by state-of-the-art algorithms.

In this paper, we explain the code for this research.

The original raw data sets are JSON files of Google History Location from 38 subjects. From privacy concern, we only supply processed data of these subjects in folder ./data/npData.

There are three group of packages in this code:

* The preprocessed package, which includes:
  + prepare.py: The pipeline for creating trajectories from JSON files
  + util.py: Users and their POIs and the meaning of these POIs (in numbers).
* Identify lifestyle package:
  + w2v\_optimum: Identify the lifestyles for these trajectories using word2vec.
* Predict lifestyle package: For each model, we create a separate package:
  + benchmark.py, include SVM and all its type, and Naïve Bayes
  + hmm\_optimum.py, (also include the files: my\_hmm.py and my\_base.py) include HMM model.
  + rnn\_optimum.py, include the RNN model
  + cnn\_optimum.py, include the CNN model
  + lstm\_optimum.py, include the LSTM model
  + blstm\_optimum.py, include the BLSTM model
  + cnnlstm\_optimum.py, include the LSTM with the CNN model
  + cnnblstm\_optimum.py, include the BLSTM with the CNN model