CS 513 Assignment 2

Map Matching & Road Slope

By: Theo Guidroz, A20426895
Hunter Negron, A20417545
Zhengcong Xiao, A20453141

Contents

Introduction

Problem Statement

- Understanding what Probe Data is
- Understanding what Link Data is
- We need to map match probe points to road links
- We need to derive road slope for each road link
- We need to evaluate the derived road slope with the surveyed road slope in the link data file

Probe Data

- This is data generated by monitoring the spatial and temporal position of a moving object. These moving objects can be used for information collection by special vehicles.
- A single probe point consists of probe ID, latitude, longitude, altitude, date time, source code, speed, heading and so on.
- If we need to reinforce the accuracy of the current road network, the incoming probe data must correspond to the correct link in the road network.

Link Data

- Link data is used to describe ways to expose and connect data from different sources on the Web.
- "Link" is a set of GPS coordinates that indicate a road section or constitute a road curve.
- It is the information of the road interval. This includes data for links that match the probe points.
- It has different attributes, such as linkPVID, length, directionOfTravel and so on.

Introduction

Goals

- We have to map matching probe points to road links.
- Then, the slope of each road section should be derived.
- Finally, we evaluate the derived road slope based on the road slope measured in the linked data file

Introduction

Provided Data

- We were given two CSV files, one representing probe point data collection from GPS and other sensors on cars and cellphones, and one representing the information describing the links in a road network.
- The probe points, due to natural error in the the data collection systems, is not perfecting aligned with the road network. Therefore, the first step should be to match the probe points with a the most likely link in the road network from where that point is collected.
- Once the probe points are matched, we can then use probe elevation data to determine the slope of the road links at the given points.

Background & Motivation

Map Matching

- Latitude and longitude are both showed on Probe data and Link data.
- For map matching, we use the latitude and longitude information of the two files and compare them to the other files to find the pair with the smallest distance between them.
- The pair is the result of the map matching link being consistent with the probe data points.

Background & Motivation

Road Slope

- Using the matched point in the previous step, then two consecutive points is calculated.
- Two methods to calculate the slope:
  - The slope between two consecutive points (P1, P2) is = $RISE/RUN\ \ i.e.\ \ (Y2-Y1)/(X2-X1)$, where p1(x1,y1) p2(x2,y2) are two points.
  - Feeding a model with the difference in distance and height, and the length of the link. Then train the model.
- The second method was chosen to counteract the presence of inaccurate data.

Approach #1: Brute-Force Proximity

Key Idea

- Since the basic idea of map matching involves associating each probe point with a particular link in the road network, the most natural approach is to use a simple proximity-based algorithm. This can easily be implemented using the formula for finding the distance from a point (in our case, probe point) to a line segment (in our case, a link defined by its reference and non-reference nodes).

Approach #1: Brute-Force Proximity

Implementation

- See “cs 513 hw 2 Notebook 1.ipynb” for the code
- Followed the simplest map-matching algorithm presented in class
- Proximity is based on a minimum perpendicular distance between a point and a line segment formula
- Because every probe point was being checked against every road link, this method is extremely inefficient, resulting in a prohibitively long runtime.

Approach #2: Improved Proximity

Key Idea

- After discussion regarding class lectures and reviewing our notes, and much time spent researching existing map matching algorithms online, we determined that our current simple approach had much room for improvement.
- Not every probe point has to be checked against every road link.
- An obvious simplifying assumption is that a single car/phone/probe on the road moves in a consistent manner; that is, it does not randomly jump between disparate links in the network

Approach #2: Improved Proximity

Key Idea Continued

- A car/phone/probe can only travel to its current link (trivially) or to links that are directly connected (incident) to its current one.
- Therefore, first determine which road links are incident to which other road links, then when iterating, a probe point only needs to be checked for proximity to those links that are incident to the link that was matched to the previous probe point.
- Probe points must be checked in order sorted by timestamp.

Approach #2: Improved Proximity

Implementation: Pre-processing

- Note, there are multiple revisions of this approach and they can be found in “cs 513 hw 2 Notebook 1.ipynb” and “cs 513 hw 2 Notebook 2.ipynb”
- First, find all of the unique trajectories (indicated by the “sampleID” column in the probe points table) and sort all of the probe points by timestamp. A trajectory here is considered a collection of probe points collected in sequence from a single car/phone/probe traveling on the road network.

Approach #2: Improved Proximity

Implementation: More link data

- Since the key to this approach is only checking for proximity against the links incident to the nearest link of the previous probe point, the link data must be analyzed to find which links are incident to which others.
- Links are defined by a reference node and a non-reference node. Therefore, incident links will be sharing at least one of these, so iterate over the link data table, querying for other links that have the same reference or non-reference node and store all of their linkPVIDs in a list in a new column “incident” in the dataframe.

Approach #2: Improved Proximity

Implementation Continued

- For each trajectory, match the first point against the entire road network. A probe can start at any point all must be checked for proximity. Keep track of which link nearest.
- For each subsequent probe point of the same trajectory, get the list of incident links from the previous nearest link and only check those for proximity. This reduces the number of checks from hundreds of thousands to only a handful.
- Performance is drastically improved; instead of running one *probe point* per second, this new approach runs roughly one *trajectory* per second.

Approach #3: Estimating the slope

See SlopeNeuralNetwork.ipynb for code

Train a neural network to estimate the slope

- Using the information below from the Partition6467MatchedPoints.csv file:
  - Speed: Find the difference in speed between two consecutive probe data
  - Altitude: Find the difference in altitude between two consecutive probe data
- Using the information below from the Partition6467LinkData.csv file:
  - Length: The length of the link
- Discard all the rows with no information for the slope
- Normalize the data used to train the model
- Split the data into training and test sets
- Train the model
- Evaluate the model and change the parameters to increase accuracy

Results

Map Matching

- The results for the map matching stage of the project are found in "Partition6467MatchedPoints.csv". Due to the limitations of the Google Colaboratory environment, this CSV only represents a fraction of the probe points. Had runtime not been a constraint imposed by the python notebook execution server, we would have been able to easily return the full dataset. The completeness of the resulting CSV notwithstanding, these points are an accurate reflection of the correctness algorithm.

Results

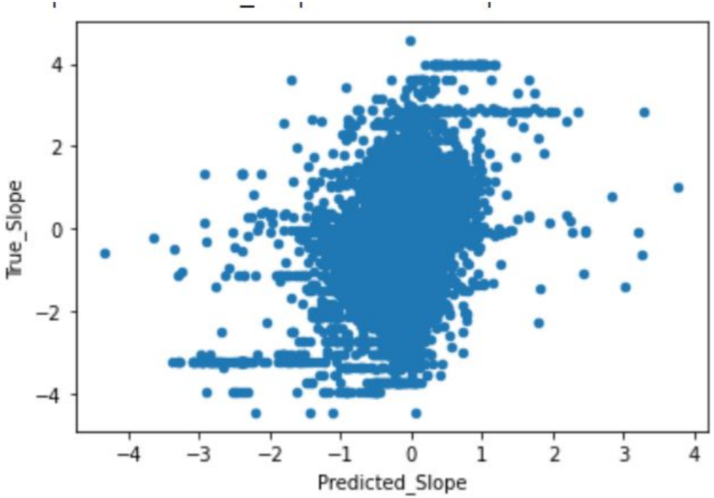
Map Matching Shortcomings

- While the final results provided here are the product of much time and effort to revise, edit, and reinvent our algorithms, we should acknowledge some pitfalls that will be found in the output dataset. Firstly, the “distFromLink“ column contains quite a few “-1”, which is postulated to be the result of an undetected error in the distance functions due to lack of rigorous testing of that specific module.

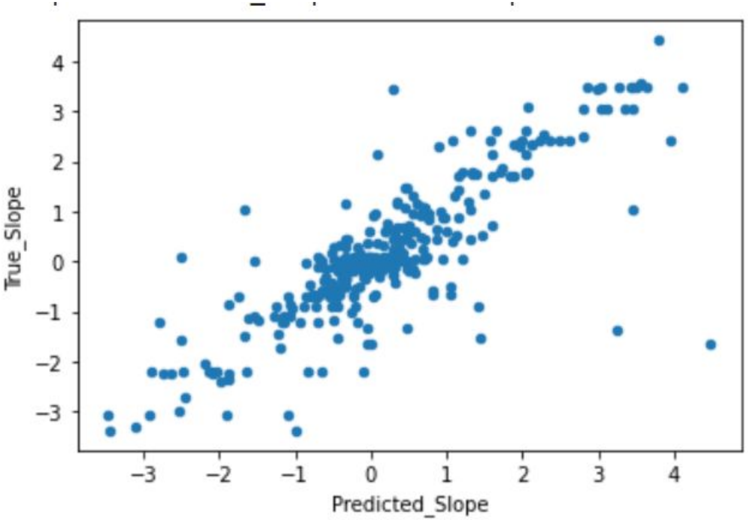
Results

Slope

While a portion of the match mapping data is accurate, some inaccuracy in the Matched csv file made the it hard to output predictions close to the true slope. The two graphs below show scatter plots of true slope vs predicted slope. Note that ideally, the graph would look similar to the line y = x



On the entire data set, the model struggles to output accurate slope estimation.



On a small amount of curated data, the graph looks similar to the line y =x.

Conclusion

The map matching calculations were accurate with some flaws. This lead to slightly less accurate slope prediction by the model. Unfortunately, the group did not possess enough computational power to run the entire data provided which caused the csv output files to be partially filled. Optimizing the algorithms would help solving this issue. Overall, the results produced by the algorithms were satisfactory but there is definitely room for improvement.

Conclusion

Further Steps:

- The model used to calculate the slope would be more accurate if more link data had slope information. To remediate to the lack of slope information, k-fold validation could be used to train the model instead of dividing the set into two.
- The final algorithm that we decided to use in order to map match the probe points is admittedly quite primitive, although it closely aligns with methods discussed in lecture. Beyond this assignment, it would be worthwhile to research other, more intelligent algorithms to produce more accurate map matched results that can be tailored to our specific dataset. Not only would this improve runtime, but also the accuracy so that there is a better chance that the slope algorithm will produce accurate results.

References

https://pandas.pydata.org/

http://www.vldb.org/conf/2005/papers/p853-brakatsoulas.pdf

“Map matching when the map is wrong: Efficient on/off roadvehicle tracking and map learning” (Paper on Blackboard)

https://www.geeksforgeeks.org/minimum-distance-from-a-point-to-the-line-segment-using-vectors/