Towards A Unified Deep Model for Trajectory Analysis

Mashaal Musleh*
University of Minnesota, USA
Email: musle005@umn.edu   Advisor: Mohamed Mokbel   ACM Member Number: 2094018   SRC: Graduate

ACM Reference Format:
Mashaal Musleh. 2022. Towards A Unified Deep Model for Trajectory Analysis . In *The 30th International Conference on Advances in Geographic Information Systems (SIGSPATIAL '22), November 1–4, 2022, Seattle, WA, USA.* ACM, New York, NY, USA, 2 pages. https://doi.org/10.1145/3557915.3565529

1 INTRODUCTION

Trajectory-based applications have acquired significant attention in several areas, including transportation (e.g., mapping and routing, traffic monitoring and forecasting), location-based service (e.g., recommendations), health (e.g., contact tracing), and urban planning. However, building such applications is still cumbersome due to the lack of unified frameworks to tackle the underlying trajectory problems, including trajectory similarity search, trajectory imputation, classification, prediction, and simplification. Despite the fact that all of these problems deal with the same trajectory data, each of the proposed solutions in the literature (e.g., see [7, 10] for surveys) is entirely designed to solve one problem of interest. This makes it hard to have a unified efficient and practical framework that is capable of supporting most (if not all) trajectory problems.

Motivated by the tremendous success of the BERT [3] deep learning model (Bidirectional Encoder Representations from Transformers) in solving various NLP tasks, and inspired by the "Let's Speak Trajectories" [5] vision that aims to have a BERT-like model for a myriad of trajectory analysis operations, this paper proposes TrajBERT, a holistic framework for an efficient and practical solution for almost all fundamental trajectory analysis problems. With TrajBERT, various trajectory analysis ideas will be just about how to tune the model one way or another to support the required analysis. This model leads to a long-waited-for full-fledged trajectory data management system that does not only store and index trajectory data, but natively supports all its data analysis needs.

TrajBERT outlines the architecture to realize that vision and develops several components to address its challenges. TrajBERT changes the core of the BERT system itself to make it deal with spatial data in general and trajectory data in particular as first-class citizens. Components in TrajBERT understand that spatial data is special and support its unique characteristics. With TrajBERT, no one needs to worry again about each specific trajectory analysis operation. Whether it is trajectory imputation, similarity, clustering, or whatever, TrajBERT is the system that researchers, developers, and practitioners can deploy to get highly accurate results.

*This work is supported by the NSF, USA, under grants IIS-1907855 and IIS-2203553.

SIGSPATIAL '22, November 1–4, 2022, Seattle, WA, USA

ACM ISBN 978-1-4503-9529-8/22/11…$15.00
https://doi.org/10.1145/3557915.3565529

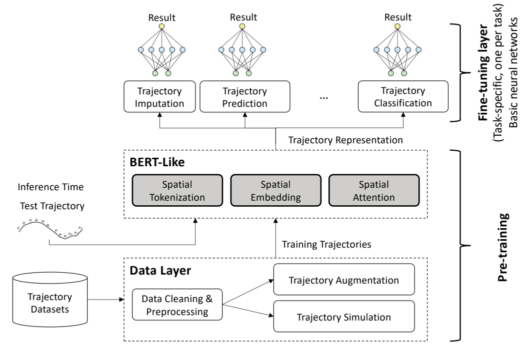

Figure 1: TrajBERT **Architecture**

2 ARCHITECTURE

Figure 1 shows the architecture of TrajBERT, which has three layers: first, the *Data Layer*, which preprocesses the input trajectories to address data quality and availability challenges. Second, the *BERT-Like Layer*, which receives the processed training trajectories from the *Data Layer* and learns a unified trajectory model. This layer is equipped with three components (highlighted in gray) specifically designed to understand spatial characteristics during the learning process, as we will explain shortly. These two layers are also referred to as the pre-training step, which trains a powerful model that represents any input trajectory as a rich numerical vector. Third, the *Fine-Tuning Layer*, which receives the numerical representations from the *BERT-Like Layer* and tunes a simple model for each analysis task (e.g., classification) to suit the given task.

Data Layer. This layer addresses the issues related to trajectory data in terms of the noise and the small ratio of available training trajectories to possible GPS points. It passes raw trajectory data through: (A) *Data Cleaning and Processing* module that uses state-of-the-art spatial data cleaning techniques [8] for noise and outlier detection, consistency verification, missing value imputation, deduplication, and other related data quality techniques. (B) *Trajectory Augmentation* module that applies basic transformations to generate new realistic and reasonable training data from existing ones, such as changing GPS sampling rates, introducing controlled noise to GPS points, etc. (C) *Trajectory Simulation* module, which learns the overall distribution of an existing dataset and simulates/generates completely new trajectories. This module uses the latest research studies [6, 9] in this area as a means to enrich our training dataset. The rich output of this layer is used by the *BERT-Like Layer*.

Fine-Tuning Layer. This layer trains one additional neural network per trajectory analysis task, such as classification, imputation, and prediction. Because most of the magic is encoded in the *BERT-Like* model, this layer contains relatively simple neural networks trained using a smaller dataset of labeled data for the needed task.

BERT-Like Layer. This layer receives the cleaned trajectories from the *Data Layer* and trains a deep neural network that can learn from large trajectory datasets. Although this layer is inspired by the general architecture of NLP BERT model, it has three crucial components that address the unique characteristics of trajectories: *Spatial Tokenization*, *Spatial Embedding*, and *Spatial Attention*.

2.1 Spatial Tokenization

This module encodes each trajectory as a sequence of tokens so the BERT model can learn their relationships. This module should satisfy two objectives: (I) minimzes the number of possible tokens, and (II) accurately represents any original GPS point by a composition of one or more tokens. These two objectives mitigate the issue of limited availability of training data by minimizing the number of tokens while also allowing multiple GPS points to contribute to multiple tokens. We explore several ideas for this module such as partitioning the space into a set of fine-grained hexagons, using Uber's H3 Hexagonal Hierarchical Spatial Index [2], and then representing all points within the same hexagon by its centroid or a combination of a hexagon and a displacement. Alternatively, we can have a hierarchical scheme or even an adaptive partitioning by varying the size of the hexagons inversely proportional to the GPS density, where small hexagons are used in high-density regions and vice versa. Empirically, we used hexagon partitioning in one of our experiments and found that it helped reduce the number of possible tokens by about three orders of magnitude while still preserving the accuracy due to the fine-grained nature of the hexagons.

2.2 Spatial Embedding

This module learns a numerical representation for each token and propagates it through the remaining layers of the system. In addition to using BERT embedding module that pays attention to the surrounding words (which applies to trajectories as well), we customize it to utilize the spatial properties exclusively available to spatial data but not words. For example, embeddings for two words closely used with each other are is affected only by the example sentences they appear in, and there is no other way to tell how these two words relate to each other except by their usage in the training data. However, embeddings of two spatial tokens is affected by (I) the trajectories they appear in (similar to statements), and in addition, (II) their spatial attributes such as their proximity to each other/roads/and other geospatial features. These techniques address the spatio-temporal constraints while also helping accelerate embedding learning and overcoming the challenges of the small ratio of available training trajectories to possible GPS points.

2.3 Spatial Attention

This module learns how much each GPS point affects each other, not only in terms of spatial proximity but also through the relationship between key points in the trajectories. This help overcomes the issue of long and unrelated consecutive trajectories and addresses the spatial and temporal constraints.

3 DEPLOYMENT AND OPERATIONS

This section provides three examples of trajectory analysis tasks that can be done using TrajBERT:

Trajectory Imputation is the process of densifying sparse trajectories by inferring additional points between consecutive ones, which is analogous to the *"finding the missing word"* problem in NLP. Given a statement like, "My husky dog was — loudly", where "—" represents a missing word, BERT can understand the context and accurately find out that the missing word is "barking". Similarly, TrajBERT is trained to accurately understand the trajectory and find out a missing point between two consecutive points.

Trajectory Prediction is the task of predicting the next few points of the current trajectory. Trajectory prediction can be seen as analogous to the *"next sentence prediction"* problem in NLP. Given a sentence, a BERT model can find the most likely sentence that naturally follows the given one. Similarly, TrajBERT is trained and used to predict the next trajectory (next few points) for a given one.

Trajectory Classification is the process of associating a trajectory with one class from a predefined set of classes, e.g., modalities such as biking, train, or car, which is analogous to the *"text classification"* problem in NLP. Given a social media post (e.g., tweet) and a set of categories (e.g., sports, politics), a BERT model can classify the given tweet to fall in one of the given categories. Similarly, TrajBERT is trained and fine-tuned to find the modality for any given trajectory.

4 PRELIMINARY RESULTS

To evaluate TrajBERT, we ran an initial experiment for trajectory imputation of the GISCUP'17 dataset [1], which includes 5M GPS points in San Francisco. We used a partial TrajBERT system that only implements the *Spatial Tokenizer* module (see Figure 1), which grouped the GPS points into 18K hexagons with 66 meters edge-length. We train the *BERT-Like Layer* on 80% of the points and keep 20% for testing, in which we down-sample the trajectories by dropping three-quarters of the points of each trajectory and then run TrajBERT to fill the gaps by imputing the missing points. Since we know the ground truth trajectories, we measure the error by computing the shortest Euclidean distance between the imputed points and the actual trajectories, which is similar to what other studies have used [4]. The mean and median distances were 37.9 and 38.9 meters, which represent a promising accuracy. These initial results are strong indications of the system's capability to achieve higher accuracies for a myriad of trajectory analysis tasks.

REFERENCES

[1] ACM SIGSPATIAL CUP 2017. http://sigspatial2017.sigspatial.org/giscup2017.
[2] I. Brodsky. H3: Uber's Hexagonal Hierarchical Spatial Index. https://eng.uber.com/h3/.
[3] J. Devlin, M. Chang, K. Lee, and K. Toutanova. BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding. *CoRR*, abs/1810.04805, 2018.
[4] Y. Li, Y. Li, D. Gunopulos, and L. J. Guibas. Knowledge-based Trajectory Completion from Sparse GPS Samples. In *SIGSPATIAL*, 2016.
[5] M. Musleh, , M. Mokbel, and S. Abbar. Let's Speak Trajectories. In *SIGSPATIAL*, 2022. To Appear.
[6] J. Rao, S. Gao, and X. Zhu. VTSV: A Privacy-Preserving Vehicle Trajectory Simulation and Visualization Platform Using Deep Reinforcement Learning. In *ACM SIGSPATIAL Workshop on AI for Geo Knowledge Discovery, GeoAI*, 2021.
[7] S. Wang, Z. Bao, J. S. Culpepper, and G. Cong. A Survey on Trajectory Data Management, Analytics, and Learning. *ACM Comput. Surv.*, 54(2), 2021.
[8] A. Zhang, S. Song, J. Wang, and P. S. Yu. Time Series Data Cleaning: From Anomaly Detection to Anomaly Repairing. *PVLDB*, 10(10), 2017.
[9] G. Zheng, H. Liu, K. Xu, and Z. Li. Learning to Simulate Vehicle Trajectories from Demonstrations. In *ICDE*, 2020.
[10] Y. Zheng. Trajectory Data Mining: An Overview. *ACM TIST*, 6(3), 2015.