




Research Article

A deep learning based approach for trajectory estimation using geographically clustered data

Aditya Shrivastava¹ · Jai Prakash V Verma¹  · Swati Jain¹ · Sanjay Garg²

Received: 5 November 2020 / Accepted: 3 April 2021

Published online: 01 May 2021

© The Author(s) 2021 

Abstract

This study presents a novel approach to predict a complete source to destination trajectory of a vehicle using a partial trajectory query. The proposed architecture is scalable to extremely large-scale data with respect to the dense road network. A deep learning model Long Short Term Memory (LSTM) has been used for analyzing the temporal data and predicting the complete trajectory. To handle a large amount of data, clustering of similar trajectory data is used that helps in reducing the search space. The clusters based on geographical locations and temporal values are used for training different LSTM models. The proposed approach is compared with the other published work on the parameters as Average distance error and one step prediction accuracy. The one-step prediction accuracy is as good as 81% and Distance error are .33 Km. Our proposed approach termed Clustered LSTM is outperforming in both the parameters when compared with other reported results. The proposed solution is a clustering-based predictive model that effectively contributes to accurately handle the large scale data. The outcome of this study leads to improvise the navigation systems, route prediction, traffic management, and location-based recommendation systems.

Keywords Long-term trajectory prediction · Recurrent neural Network · Large-scale trajectory data · Real-time prediction

1 Introduction

Technologies such as Sensors, the Global Positioning System (GPS), and the Internet of Things (IoT), has resulted in an abundant amount of mobility data. These data are generated and shared by businesses, public administrations, non-profit organizations, scientific research institutes, etc. They want to capture and analyze this data to make better decisions for future actions. The improved data mining methods are important to efficiently process and facilitate better decisions using this huge volume of data. As a consequence of churning a large volume of data, Mobility Data Analytics, or Trajectory Prediction, has become an

active ongoing research avenue [5, 10, 32]. These research outcome will lead to improved navigation [20], route prediction [28], traffic sensing [14] and location-based recommendations [3]. These types of analysis will be helpful for smart city projects to develop, deploy, and promote sustainable development practices to address growing urbanization challenges. One of the major urbanization challenges is mobility, and recommendations while navigation is an essential service in large cities. Use of mobility data for trajectory prediction will help arrive at an optimal framework for this type of application requirement.

If a person moves across the same roads a repeated number of times, his inference capabilities enable him/

✉ Jai Prakash V Verma, jaiprakash.verma@nirmauni.ac.in; Aditya Shrivastava, 17bit014@nirmauni.ac.in; Swati Jain, swati.jain@nirmauni.ac.in; Sanjay Garg, gargsv@gmail.com | ¹Institute of Technology, Nirma University, Ahmedabad, India. ²School of Computing, DIT University, Deharadun, India.



her to make smart decisions and plan the trajectory that would be shorter and better. This analogy is used for the trajectory prediction of the moving body. The process of predicting the path of the moving vehicles based on the available vehicle mobility data is termed as Trajectory Prediction (TP). Recently, several studies have been carried out on trajectory prediction using machine learning techniques [1, 29, 34] that can categorize into two types. The first type is the short-term trajectory prediction model, which predicts the route only up to a few locations and considers only a short history for prediction. These types of studies are applicable to small cities where the road networks are small compared to large cities. The second type is the long term trajectory prediction models. Long term Trajectory prediction models predicts more number of intermediate locations and hence requires more similar data for training. In large cities, these trajectories usually follow a complex road network making their mobility data ambiguous. In these cases, long term predictions are more useful and effective as cost and time of travel has a higher multiplicative factor.

It is a challenging task to carry out trajectory prediction from a large amount of trajectory data using conventional computing algorithms. The Machine Learning algorithms for Trajectory Prediction are experimented with a small set of data and have shown results. These algorithms also failed to show similar results in real-life large-scale trajectory data. It is because the complexity level increases at a much higher rate with increasing number of intersecting and overlapping trajectory scenario. There are some implementations based on the Markov model and other traditional pattern recognition algorithms such as clustering-based algorithms that rely only on the initial 2 or 3 GPS points and cannot predict long term trajectories very efficiently. Some of the modern practices are using Recurrent Neural Network (RNN) [7, 22]. Its hidden layer can very well store long-term dependencies between points. However, these systems tend to be sensitive to small changes in the future scope.

There are applications [31] that shows that the RNN and Long short term memory (LSTM) have performed better on the time series data. However, as the training data increases, the architecture of these models becomes more and more complex. Training these models in a reasonable time demands special and expensive hardware. Also, the data set as a whole has a lot of patterns and makes it difficult for any singular model to learn with this high variance data. Hence, to keep up with both the volume and complexity of the data for long-distance trajectories, we propose a robust approach that is scalable in terms of size and scalable. The scenario of the overlapping road is typical in major cities around the world. It is quite intuitive that the path taken by any vehicle will be more regulated/

motivated by the moving pattern of that vehicle in the same area in comparison to a different one. Hence, we first apply the clustering technique on the large trajectory data considering time and location. Then, corresponding RNN models are constructed and trained on the trajectories of the same cluster to learn the movement patterns within that cluster. Such clustering of data is especially beneficial for complex models such as LSTMs. Clustering will result in improved performance, as one big problem is divided into several smaller problems. Each model has the task of learning moving patterns of smaller areas. All these models are combined to solve the bigger and complex problems. Dividing the problem to sub-problem is to tackle the time and space complexity of the deep learning model in a big data environment [8, 23].

1.1 Contribution

This paper proposes a predictive model where a clustering technique is applied before LSTM to address the scaling issue and to capture the local phenomenon in the data. This way the large volume of data is grouped into smaller clusters making the LSTM training manageable without high-end resources. This helps to achieve the following research objectives

1. To overcome the issues of scalability, data are geographically clustered before applying multiple LSTM models to predict the trajectory.
2. A comparative analysis of the proposed approach (geographically cluster-LSTM) with the other two conventional approaches LSTM and cluster—HMM.

1.2 Organization

This paper is organized into eight sections. Section 2 presents the related research work in the area of machine learning-based data modeling for location-based Trajectory prediction. Section 3 is the problem formulation where the notations used in the paper are defined and the relationship of various entities is represented. Section 4 gives a glimpse of the entire architecture and the proposed approach framework. Section 5 represents the Methodology, tools, and concepts applied to achieve the proposed outcome. This section is consists of a stepwise algorithm to explain the overall methodology. Section 6 describes the steps followed for conducting the experimentation including data pre-processing. Section 7 includes results and discussion of the experimental analysis done for the proposed research work. Section 8 presents the conclusions drawn and future directions identified.

2 Related work

A considerable amount of work has been done in recent years on trajectory prediction, also termed as destination prediction or trajectory querying. Here, we briefly discuss the work done for effective trajectory prediction involving pattern recognition or machine learning frameworks. Also, Table 1 shows a summarization of the related work done in the area of Trajectory Prediction. We mainly categorize the discussion of related works into 3 categories: symbolic rule based approaches, machine learning approaches and deep learning approaches.

2.1 Symbolic rule based approaches

Initial methods for trajectory prediction involved scripting the symbolic rules to determine the movement patterns from the trajectories. Next location is predicted through association rules employed to infer from the observed patterns in the movement. Morzy et al. [19] used a hybrid of the Prefix Span algorithm and frequency pattern tree as an association rule to predict the trajectories. Similarly, Monreale et al. [18] programmed a T-pattern Tree that accounted for the frequency of the trajectory patterns for predicting the next location. The issue, with this method, is, it is computationally expensive to identify such frequent trajectory patterns. Sometimes, the future location prediction is also done by estimating the motion characteristics through custom-programmed functions that use the recent movements of the object [18]. However, since the approaches mentioned in [18] are reported to be quite inaccurate for long distance predictions, they are only recommended for short-distance trajectory predictions.

2.2 Machine learning based approaches

Later, the problems faced by using symbolic approaches were identified, and to overcome the same the research works corroborate widespread adoption of machine learning and data mining techniques in trajectory prediction. The simplest clustering methods employ a singular training algorithm such as k-means, C-SCAN with some modifications to fit on the trajectory data [25, 35] or else, parameters such as vehicle frequencies in terms of flux can also be considered [12]. To make such approaches helpful in terms of accuracy, they are often combined with prediction algorithms such as Markov Models (MM). Such methods tend to perform well on the small-scale, artificial data sets but when tested on real, large-scale trajectory data yield poor results [2, 4].

Other approaches involve prediction based on the semantic score where the semantic score is evaluated

through the extraction of geographical and other semantic features from the user trajectories [38]. However, the analysis suggests that the computation of semantic score causes computational overhead. A spatiotemporal prediction technique proposed in [16] forms the cluster of the trajectories based on entropy and then use Hidden Markov Models (HMM) for prediction.

Other than being used as part of hybrid algorithms, HMMs have also been widely adopted by the researchers as an independent algorithm for trajectory prediction. Often, the map is considered as a grid, where each cell represents the specified location and HMMs are used to predict transition probability from one cell to next [13]. Also, the conventional framework involving HMM to predict the best location is sometimes extended to instead predict top-k locations [9]. This significantly boosts the prediction accuracy. However, as the number of locations to be predicted increases, the computational complexity in terms of both space and time begins to rise.

2.3 Deep learning based approaches

Most forms of MMs failed to generalize discontinuous trajectory data [24]. Moreover, since the MMs are known to employ Viterbi algorithm for finding the hidden state sequence and Baum-Welch algorithm for parameter learning, they introduce a significant computational burden when used on large-scale trajectory data. Due to the recent success of complex networks such as Recurrent Neural Networks (RNN) and Long Short Term Memory (LSTM) networks in sequence prediction, the researchers have started to replace HMMs with these architectures for trajectory prediction tasks as well. They are known to be scalable for large-scale data. Recently, a deep learning-based human mobility prediction framework was proposed by Wang et al. [33] using a LSTM network. It was trained on the user's historical trajectories. Then, the basic model of LSTM was extended to be a multi-user, region oriented framework by incorporating the sequence-to-sequence modeling. The results had shown a significantly reduced error rate accompanied by highly improved generalization abilities. In [26], RNN was used for predicting the exact coordinates of the next destination based on the taxi driver's behavior. The RNN was trained on a sequence of pick-up and drop-off points to predict future drop-off points. The framework was evaluated on the ECML/PKDD Discovery Challenge 2015 dataset. Initial benchmarking performance on the same dataset was reported by Brébisson et al. [6] using multi-layered perceptrons. Other alternative approaches such as recurrent neural networks, bidirectional recurrent neural networks, and other memory networks were also developed. Overall however,

Table 1 Summarization table of the related works

Approach	Year	Objective	Methodology	Pros	Cons
Trajectory prediction, hand-crafted rules, symbolic pattern extraction [18, 19].	2009	Fully specified rule-based algorithms are devised beforehand, based on which the trajectories are predicted.	A specific set of features is extracted using the hand-crafted rules. Later an association rule is developed that predicts based on the features extracted.	The component of bias shall be minimum as each data sample shall not participate in the framing of the algorithm.	It is quite difficult to devise accurate guidelines that can predict for the long term.
Clustering based approach, route prediction, correlation based grouping [25, 35].	2009	To predict the future route of a trajectory based on the similarity with the other trajectories.	A clustering algorithm groups the trajectory data into different clusters and later predicts the future locations using the most significant characteristics observed amongst trajectories in a cluster	Such algorithms are computationally less expensive and thus can handle large scale data without causing any high latency in prediction.	Since only a single algorithm is used for prediction and that too is classical clustering-based approaches, they do not offer high accuracy in prediction.
Hybrid machine-learning based approach, Clustering trajectories, hidden Markov models [2, 4].	2015	To predict the next 2 or 3 locations in the future by learning from the trajectories of a specific cluster with the help of HMMs.	As opposed to the mediocre clustering approach, a hidden Markov model is plugged into each one of the clusters to predict more locations in the future with high accuracy.	Since these are the hybrid architectures, they can predict with much higher accuracy than merely using clustering approaches.	Poor performance is yielded when these structures are used for long-term prediction.
Sequence Prediction approach, First-order Markov chains [9, 13]	2012	Hidden Markov Models is used to capture the patterns in the trajectory data and predict a trajectory based on the learned patterns.	The entire map of a region is framed as a grid and then a probability to transit from one place to the subsequent is predicted.	Route planning differs from one region to another and hence a single HMM can more agreeably capture the overall related to that region.	As the data increases, variance also increases, which is why HMMs are unable to epitomize the significant patterns required for predicting.
Trajectory prediction, multi-step prediction, long short-term memory, sequence-to-sequence, machine-learning [33].	2019	To provide long term prediction of human mobility using the LSTM network.	The sequence-to-sequence (Seq2Seq) learning was incorporated to provide a multi-user multi-step trajectory prediction.	A hybrid framework for trajectory prediction for both the single-user perspective and multi-user perspective was developed.	It does not consider the semantic context in the trajectory like the point of interests because of the limitation of data.
Trajectory Prediction, Long short-term memory, Recurrent Neural Network [26].	2018	Modelling taxi drivers' characteristics and his geographical information in the next location prediction in the trajectory.	Recurrent Neural Network was developed to encode taxi drivers' behavior and the semantics of visited locations by using geographical information from Location-Based Social Networks (LBSNs).	First, an approach to develop a regression approach to this task instead of framing as a multi-class classification problem.	The approach developed is not scalable to the real-world scenario where the amount of data grows extensively.
Trajectory Prediction, Multi-Layer Perceptron, Recurrent Neural Network, Memory Networks [6].	2015	To predict the destination of the trajectory given the beginning of the trajectory.	Fully automated multi-layer perceptron was developed to output the destination when the beginning of the trajectory along with embeddings such as timestamp, client ID, etc.	The approach uses very little hand-engineering compared to other approaches. It is almost fully automated and based on artificial neural networks.	Clustering-based output layer is only allowed to fall in the convex hull of clusters instead of learning clusters as parameters.

Table 1 (continued)

Approach	Year	Objective	Methodology	Pros	Cons
Urban traffic, Spatio-temporal data, Neural network, Meta-learning [21].	2019	A deep-meta-learning model-based approach to collectively predict traffic in all locations at once.	An architecture titled ST-MetaNet was employed as a sequence-to-sequence architecture, consisting of an encoder to learn historical information and a decoder to make predictions step by step.	It was capable of learning traffic-related embedding of nodes and edges from geo-graph attributes and modeling both spatial and temporal correlations.	Such heavy architectures only tend to perform well on the small or medium-sized dataset and not on the large-scale data.
Trajectory Prediction, Probabilistic Approach, Recurrent Neural Network [7].	2017	To address the long-term dependencies and data sparsity problem in predicting the destination.	The methodology involves representing trajectories as discretized features in a grid space and feeds sequences of them to the RNN model, which estimates the transition probabilities in the next timestamp.	Efficiently predicting destinations using a stochastically sampling simulation based on the RNN encoder-decoder framework.	The computational time for learning the RNN model linearly increases with the number of grid cells and making it difficult to target a huge area with fine grids.
demand prediction, deep neural network, multi-view [36].	2018	Improve the demand prediction by modeling both spatial and temporal relations simultaneously.	A Deep Multi-View Spatial-Temporal Network (DMVST-Net) framework was proposed consisting of three views: temporal view, spatial view, and semantic view.	The approach was scalable on a large scale taxi demand dataset.	Semantic information is implicitly modeled in this approach instead of modeling explicit information such as Point-of-Interest etc.

Bidirectional RNNs with the window had given the best accuracy on the custom test set.

Further, Pan et al. [21] addressed two main issues in predicting traffic. First, complexity in both spatial and temporal correlations found in urban traffic by using a deep meta-learning based approach to predict traffic. And second, issue of the diversity and variation present in such Spatio-temporal correlations by incorporating sequence to sequence modeling in the developed ST-MetaNet model.

Yao et al. [36] proposed a composite two neural network model Convolutional Neural Network (CNN) and LSTM to model both spatial and temporal relations simultaneously in predicting taxi demand in Guangzhou, China. These prior developed approaches are oriented around either predicting short-distance trajectory using the large-scale data or long-distance prediction for up to medium-sized data.

3 Problem formulation

This section represents the notations used in the paper for the entities like trajectory, clusters, partial trajectory. Initially, G is the data set of randomized trajectories represented as eq. 1.

$$G = \{T_1, T_2, T_3, \dots, T_N\} \tag{1}$$

where G is a set comprising of the total trajectories T , spanning the region's network i.e. N , for a given period of time.

Definition 1 Trajectory is the sequence of the location coordinate tuples followed by a vehicle during a trip. Each trajectory T may have different length l . The trajectory T can be defined as a set whose elements are time-ordered sequences of coordinates as shown in equation 2,c

$$T = \{C_1, C_2, C_3, \dots, C_l\} \tag{2}$$

C_1 is the starting coordinate termed as source, and last coordinate is the final location termed as destination of the trajectory. All coordinates in between, C_i , represents the intermediate location where the vehicle travelled through.

Definition 2 Clusters The entire set of trajectories G is divided into clusters. Each trajectory is associated to some clusters. S_j is the j th cluster, T_i^j represents the i th trajectory associated to the j th cluster.

$$S_j = \{T_1^j, T_2^j, T_3^j, \dots, T_M^j\} \tag{3}$$

The above equation 3 represents the schema of each cluster. There are M trajectories present in the cluster. When

these clusters are combined, they result into a complete trajectory set i.e (equation 4).

$$S = \{S_1, S_2, S_3, \dots, S_k\} \quad (4)$$

Definition 3 Partial Trajectory T_p is defined as part of the complete trajectory. Every thing else is same except C_1 may not necessarily be the source and C_t may not necessarily be the destination equation 5).

$$T_p = \{C_1, C_2, C_3, \dots, C_t\} \quad (5)$$

where C_i denotes coordinates indexed at specific instant i . To mention, the partial trajectory T_p has all of the characteristics identical to trajectory T but that it is partial in nature and is not complete.

Definition 4 Statement If a set of historical trajectories is given as $G = \{T_1, T_2, T_3, \dots, T_N\}$ then, for queried partial trajectory $T_p = \{C_1, C_2, C_3, \dots, C_t\}$ the goal is to predict the next location C_{t+1} .

4 Proposed approach

This section briefly presents the proposed work for predicting the next location in the trajectories. We consider the scenario of major cities, where there is a large area of the road network which increases the trajectory data and also leads various options for the taxis. Sometimes these taxis follow unusual paths. So, to mine the routing pattern of the moving objects, we propose to cluster the historical trajectories and thereby segregate the data into several groups based on the similarities between the trajectories. Further, for quickly clustering the trajectories through Nearest Prototype Rule (NPR), we introduce cluster head as Representative Trajectories (RT) for each cluster. The clustering algorithm partitions all of the trajectories into a different cluster based on the paths followed and the distance between these trajectories. After clustering, a deep learning architecture named LSTM shall be trained separately for each cluster using only the trajectories of that cluster. The LSTM model performs the task of predicting the future location given the partial trajectory, which results in improved performance for prediction of long-term trajectory.

The entire pipeline for our trajectory prediction model is described hereby a concise way. Figure 1 presents the streaming of the data in our model. The model shall record the data that is collected at the secondary storage from the local storage through the streaming process. Here we use the Big Data Environment as the distributed storage environment because the size of the real-world data can easily outgrow the storage space available in the single

system. Initially, as the data is streamed into a distributed environment and stored across various nodes. Then, all of the trajectory data shall be grouped using our hybrid clustering approach and the RT is generated as a cluster head for each of the clusters.

The geographical grouping using KNN based clustering technique is performed as the pre-processing task on the trajectory data. A corresponding LSTM architecture is developed separately for each of the clusters on the front-end and trained with the member trajectories as input. Once the LSTM architectures are trained for different clusters, all their parameters are saved in the system. Now, whenever a partial trajectory is queried in real-time and its future location is required to be predicted, it is first parsed through an NPR (Nearest Prototype Rule) classifier. The role of the NPR classifier in the prediction model is to evaluate which cluster's RT showcases the highest *SimilarValue* with the queried trajectory and thereby feed the trajectory as an input to the LSTM model of that cluster. As the LSTM model predicts the next location, the queried trajectory is updated by the addition of the new predicted location and iterative inputted to an NPR classifier for prediction. In this way, the process is repeated until the end of the trajectory.

5 Methodology

This section describes the methodology used to achieve the proposed objectives in this paper. It is explained in two phases- *training* followed by *prediction*.

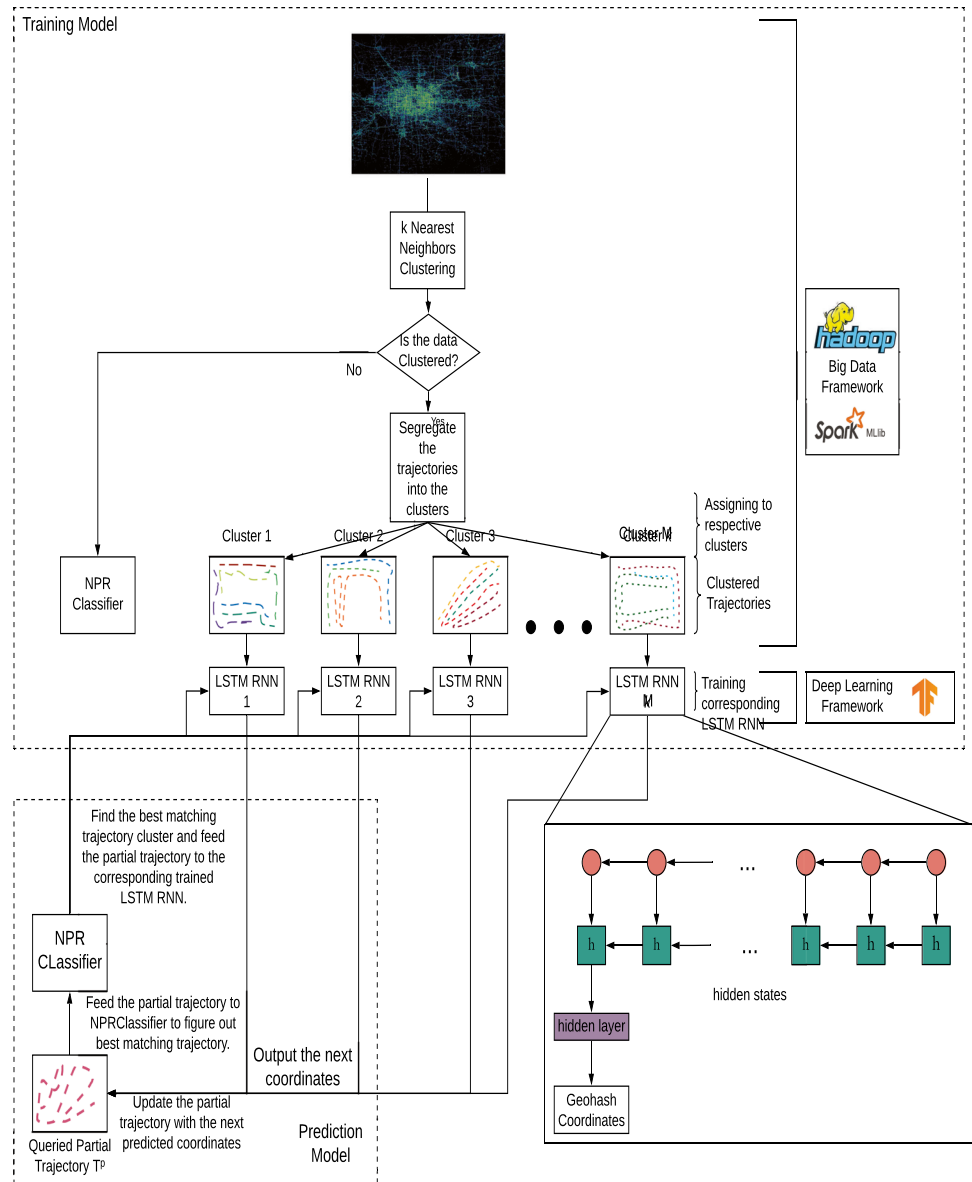
5.1 Training

The training processes is divided into two parts, clustering the trajectory data based on the geographical locations and then training the model on the clustered data.

5.1.1 Clustering the data

The first step of our training algorithm is to group the entire set of trajectory data into appropriate clusters based on similarities between the trajectories of a specific area during a specific time interval. For this, we use a modification of the k-nearest neighbor algorithm by considering the trajectories instead of points. Note that for our algorithm in this paper, by mentioning the term nearest, we mean the trajectory that is in the highest similarity with another trajectory. We start by considering a trajectory and the value k indicating the number of nearest neighbors (in this case nearest trajectories) to be grouped. Thus, the total of $(k + 1)$ trajectories shall be grouped into each of the clusters. This means that each cluster, during the start

Fig. 1 Schematic diagram of the proposed approach



shall be having approximately $k+1$ trajectories. Thus, the number of clusters M can be determined as

$$M = \frac{\text{Total no. of trajectories } (N)}{(k + 1)} \tag{6}$$

Now, once the number of clusters to be formed is known i.e. M , we proceed further to form those M clusters. For this, we first determine the M most distinguished trajectories from the entire dataset. We use the Maximin Random Sampling (MMRS) algorithm described in [11] is used to find such trajectories. It ensures that that the selected set of trajectories has maximum separability and not more than one trajectory is selected from the same route. Upon initially performing MMRS algorithm on the entire trajectory data, arbitrary no. of trajectories, let us say M' are selected

from each of the routes. Therefore, M trajectories out of a total of M' trajectories are selected having the maximum separability in terms of distance measure. These M trajectories form the base trajectories of their respective cluster.

Once the base trajectories are formed, k trajectories are to be selected for each of the base trajectories such that they are nearest and in the highest similarity with the base trajectory of the corresponding cluster. Now, as mentioned earlier, we use the concept of the kNN algorithm with some modifications to cluster the trajectories. Here, the amount of proximity or similarity can not be measured in quite the same way as in the original kNN algorithm. The reason being that kNN usually uses Euclidean Distance as a metric to calculate the nearness of the two points, which cannot be seen as a suitable metric for measuring

nearness in trajectories. Hence, we measure the Hausdorff Distance between the trajectories and take the k trajectories having the least Hausdorff Distance measure with the base trajectories of the corresponding clusters. For any two trajectories represented by A and B , their Hausdorff distance can be calculated as,

$$H(A, B) = \max(h(A, B), h(B, A)) \quad (7)$$

$h(A, B)$ ranks each co-ordinate of A based on its nearest point of B .

$$h(A, B) = \max_{a \in A} \min_{b \in B} ||a - b|| \quad (8)$$

Thus, with the selection of M base trajectories corresponding to each of the clusters, the algorithm can be said to be partially completed.

5.1.2 Nearest prototype rule

Still, there shall be plenty of trajectories in the dataset that weren't assigned to any of the M clusters as k NN would only group $k+1$ trajectories into a single cluster. The Nearest Prototype Rule (NPR) is implemented to cluster such non-sampled trajectories. For the efficient and accurate implementation of NPR, the Representative Trajectory (RT) is required to be computed. Such trajectories are interpreted as the best possible representation of the trajectories of that cluster. It can be thought to be analogous to the centroid in the polygon except that to compute the centroid of clusters of trajectories requires more intricate and composite methods. There are a bunch of methods proposed to compute the representative trajectory of the cluster computing the mean trajectory using the mean of the GPS coordinates of all trajectories for each timestamp [11, 30]. However, there are a couple of problems in implementing this method. First, merely computing a trajectory by a statistical mean of all trajectories in the cluster does

not always mean that there shall be a real pathway on the roadmap through which the trajectory can be routed. Second, the trajectories are all of the different lengths, and therefore, computing the mean would introduce bias as all the trajectories shall not be able to participate in average estimation at all time-stamps. Next, it is also proposed to sample a random trajectory from the cluster and denote it as an RT. It is not usually recommended, as sometimes a huge deviation is observed between the random trajectory selection and the actual best RT. Therefore, we choose the approach presented in [15, 37] to nominate a trajectory that has minimum dissimilarity among the trajectories of the same cluster. For every trajectory T_i , we estimate the *SimilarValue* metric that quantifies the similarity between that trajectory and the rest of the trajectories in that cluster.

$$\text{Similar Value}(T_i) = \frac{\sum_{j=1}^N \text{Similarity}(T_i, T_j)}{N} \quad (9)$$

After the *SimilarValue* is computed for each trajectory, the trajectory with the highest *SimilarValue* is assumed to be the representative trajectory.

Now that the RT is computed for each cluster, we proceed to accommodate the most distinguished and unusual trajectories that were not classified in the initial attempt, into appropriate clusters through Nearest Prototype Rule (NPR). In this, we again employ the algorithm used for computing the RT, but this time with one to one mapping. This means that we compute the *SimilarValue* metric for the unassigned trajectory with the RT of each cluster. Then, the cluster whose RT has the highest *SimilarValue* with the trajectory is assigned the trajectory of that cluster. The same process is repeated for all the unclustered trajectories. The procedure is represented by the name *NPRClassifier* in Fig. 1 and Algorithm 24.

5.1.3 Training the architecture

Algorithm 1: Training

Input : Initial set of randomized trajectory data— $G = \{T_1, T_2, T_3, T_4, \dots, T_N\}$, number of nearest (similar) trajectories w.r.t. each of base trajectories to be grouped into a single cluster— k .

Output: The LSTM-RNN model trained on the M cluster segregated data.

- 1 Calculate $M = \text{Total no. of trajectories } (N)/(k+1)$;
 - 2 Fetch out a set of M' trajectories having maximum separability, $G_{M'} = MMRS(G)$;
 - 3 Select G_M , a set of top M trajectories to be represented as base trajectories from $G_{M'}$ such that $G_M \subseteq G_{M'} \wedge M \leq M'$;
 - 4 Update G by removing the base trajectories as they have been represented by a different set G_M , $G = G - G_M$;
 - 5 Create a separate set S_j corresponding to each of the base trajectories T_j of G_M ;
 - 6 **foreach** trajectory $T_i \in \mathcal{G}$ **do**
 - 7 **foreach** trajectory $T_j \in \mathcal{G}_M$ **do**
 - 8 Calculate Hausdorff distance with T_i , $H(T_i, T_j)$;
 - 9 Select the trajectory T_j with minimum Hausdorff distance for T_i ;
 - 10 **if** $|S_j| \leq k + 1$ **then**
 - 11 add trajectory T_i to the cluster S_j represented by it, $S_j = S_j \cup \{T_i\}$;
 - 12 Update the set G by removing trajectories that were assigned to a cluster, $G = G - S$;
 - 13 **Procedure** NPRClassifier()
 - 14 **foreach** cluster $S_j \in \mathcal{S}$ **do**
 - 15 **foreach** cluster $T_j \in \mathcal{S}_j$ **do**
 - 16 Compute $SimilarValue(T_i)$;
 - 17 Declare the trajectory with highest SimilarValue as the Representative Trajectory (RT) of the cluster S_j ;
 - 18 **foreach** trajectory $T_i \in \mathcal{G}$ **do**
 - 19 Compute $SimilarValue$ with RT of each cluster;
 - 20 add trajectory T_i to the cluster S_j having highest $SimilarValue$ with its RT;
 - 21 **foreach** cluster $S_j \in \mathcal{S}$ **do**
 - 22 Create dataset of all the trajectories belonging to the cluster;
 - 23 Train an LSTM-RNN model on the trajectories of the dataset;
 - 24 Save the model state (weights and biases) for the respective cluster;
-

Once all of the trajectory data is filtered and appropriately segregated into discrete clusters, we build a separate Long Short Term Memory (LSTM) network for each of the clusters. An LSTM neuron is composed of three gates, the

input-output gates through which the information flows in and out and a forget gate that remembers values of an arbitrary time interval. Further, there are weighted connections associated with these gates, are trained using

backpropagation that determines how the gates operate. Sometimes, in trajectory planning, there is a need to remember the coordinates for a longer period i.e. the more distant historical coordinates also contribute sometimes in predicting the future coordinates. And the conventional prediction models such as Markov Chains and even the basic RNNs fail to characterize such long-term dependencies. Therefore, LSTM architecture is implemented as it can store such pertinent information more effectively.

is over data is aggregated and extracted through the data streaming process and supplied to the Big Data Storage Centers. In between, the flow control process ensures the integrity in the data supply and that it does not suffer from any problems such as data loss, congestion, etc. In reality, high frequency sensors output several thousands data points/second. Since, it is impractical to store this on a single machine, the data is distributed across several nodes. In this scenario, the local node or system performs

Algorithm 2: Prediction

Input : A partial $T^p = \{T_1, T_2, T_3, T_4, \dots, T_t\}$.

Output: Next location T_{t+1} .

- 1 **repeat**
 - 2 Compute the cluster by partially executing procedure NPRClassifier, $NPRClassifier(T^p)$;
 - 3 Predict the next coordinate using the LSTMRNN model with the state (weights and biases) belonging to the identified cluster, $T_{t+1} \leftarrow LSTMRNN_{clusterid}(T^p)$;
 - 4 Update the partial trajectory with the predicted coordinate to predict further coordinates in the same way, $T^p = T^p \cup \{T_{t+1}\}$;
 - 5 **until** *end*;
-

5.2 Prediction

For each partial trajectory $T^p = \{C_1, C_2, C_3, \dots, C_t\}$ queried, the best suitable representative cluster is estimated using our previously defined NPR algorithm. Then, the corresponding LSTM model is chosen in which the partial trajectory T^p is fed and the next location, C_{t+1} is predicted. The T^p is updated by appending the next predicted location C_{t+1} . Then, the best matching cluster is again estimated for the updated T^p , and the corresponding LSTM model is used to predict the next location. Hence, the full trajectory is predicted by sequentially computing the next locations using these steps.

5.3 Data extraction and storage

Figure 2 shows a brief conceptualization of the realistic scenario on the extraction and loading of the trajectory data into the systems. The data is first generated and stored in local storage systems like mobile devices, IoT sensors, cell towers, remote servers, etc. And all these data are in heterogeneous format since they are produced in distinct formats and data types. Next, the data ingestion process is performed on all these heterogeneous data present in the local storage. Once the ingestion process

the pre-processing on the data stored locally and this pre-processing is replicated synchronously across all the devices. This also helps in making the data to be sent to the main node for secondary stage processing to be all the more less redundant and much cleaner through aggregation, summarization etc. In a nutshell the total amount of data that is forwarded to the master node from the local

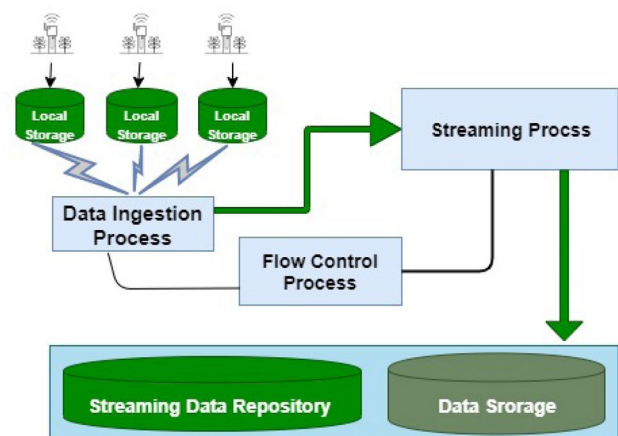
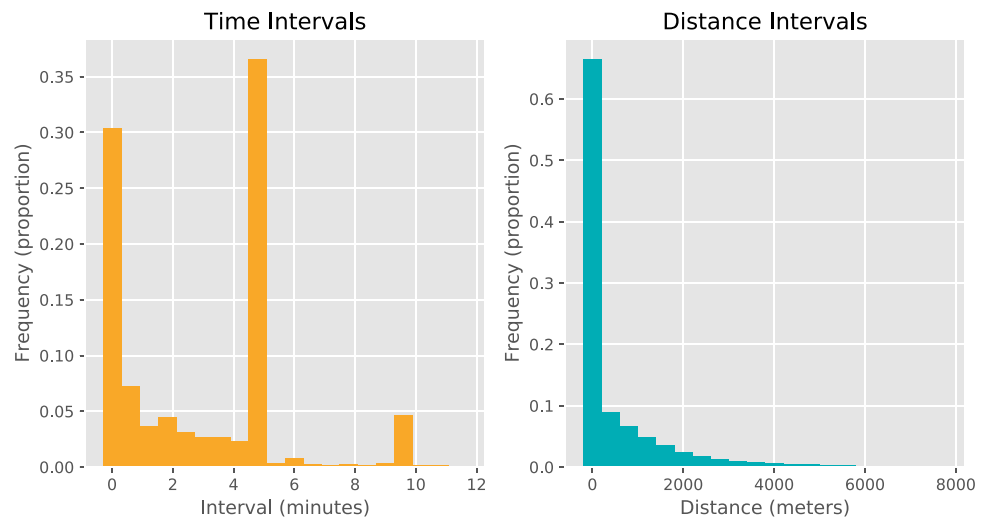


Fig. 2 Data extraction and loading

Fig. 3 Histograms of time interval and distance between two consecutive points. **a** Time Intervals **b** Distance Intervals



node is minimized significantly leading to quick response times and diminishing transmission costs.

6 Experimental settings and environment

In this section, we provide an detailed empirical analysis to evaluate the performance of our proposed framework.

6.1 Dataset

Experiments are conducted against the real dataset to evaluate the performance of the approach. The dataset¹ was obtained from the T-drive project [39, 40] which had contained one-week worth of continuously running non-stop GPS trajectories of a total of 10,357 taxis from Feb. 2 to Feb. 8, 2008, within Beijing, China. As per [40], the dataset represents 106,579 road nodes and 141,380 road segments of the road network of Beijing.

Each data file of the dataset basically contained 4 fields: (i) The TAXI ID of the related taxi. The attribute (ii) TIMESTAMP is the date/time in seconds of when the position was sampled and the attributes (iii) and (iv) refers to the LONGITUDE and the LATITUDE corresponding to the acquired GPS position. There were a total of 17.7 million data points that accounted for 9 million kilometers of distance. Figure 3 plots the distribution of time interval and distance interval between two consecutive points.

And as shown in the figure (b), because of being sampled very frequently, most of the euclidean distances between two data points across the trajectories are tending to 0. And with respect to this, the average euclidean

distance between consecutively sampled data points which is equivalent to 707 meters, is quite large for a city traffic environment. On the other hand, the time stamp sampling interval is more distributed. This is because of variance in the traffic across different segments of the road network. The average sampling interval was of 214 seconds.

In the above two plots are two-dimensional histograms, density is plotted as a function of latitude and longitude. Not surprisingly, as per the figure 4(a), the data mostly lie on major roads and highways. Figure 4(b) further details into the 5th Ring Road in Beijing. Cities in China use concentric ring roads centered on the city center.

6.1.1 Pre-processing step

Then, to prepare training and validation sets, we first divide the trajectories between weekdays and weekends. The 80% of the trajectories are randomly sampled from the weekdays and weekends for the training and the rest are included for validation. Hence, a total of 14.16 million rows were considered for training while the rest 3.54 million rows were considered for validation. Then, GPS coordinates in the form of latitude-longitude tuples were converted to their corresponding GeoHashes using GeoHash mapping. After the data was filtered and completely preprocessed, all of the data was stored on the system using Apache Hadoop (HDFS) framework [27]. Since the data storage capacity available on our machine was sufficient to store the entire dataset. We build the data storage architecture of just one node. Hence, there is no formation of hadoop cluster since we use only one computer for storage. Hence, also note that the logical cluster formed as per our algorithm are disparate than the clusters that are normally thought of to be in big data reference. Apache Spark [41] framework was adapted to update existing patterns,

¹ The dataset is open-source and can be accessed from <https://www.microsoft.com/en-us/research/publication/t-drive-trajectory-data-sample/>

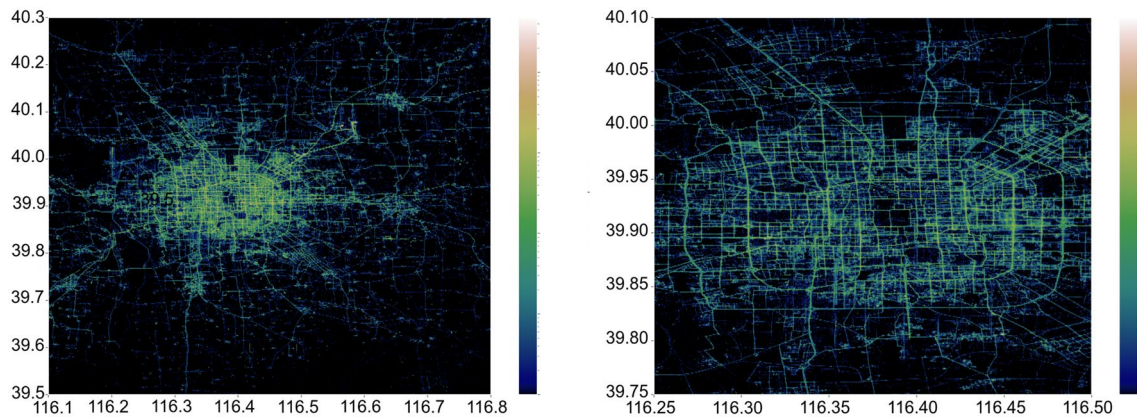


Fig. 4 Distribution of GPS points, where the color indicates the density of the points. Dark colour indicates low density, light colour high density. The X-axis represents geographical longitude (degree

E) and the Y-axis geographical latitude (degree N). **a** Data overview in Beijing **b** Within the 5th Ring Road of Beijing

add new ones, and all of the other in-memory processing tasks.

6.2 Evaluation protocols

We chose four performance metrics, namely average prediction accuracy, one-step prediction accuracy, average distance error and one-step distance error to test our results:

6.2.1 Prediction accuracy

It is the percentage of GPS coordinates correctly predicted out of total number of steps till a specific step. Suppose a predicted trajectory is $T_{pred} = \{P_1, P_2, P_3, P_4, P_5, \dots, P_n\}$ and true trajectory is $T_{true} = \{T_1, T_2, T_3, T_4, T_5, \dots, T_n\}$ prediction accuracy is given by,

$$PA = 1/n \sum_{j=1}^n Q(P_j, T_j)$$

Where j is the step and function $Q(P_j, T_j)$ returns 1 if $P_j = T_j$ else 0.

6.2.2 One-step prediction accuracy (OA)

This metric is defined as the ratio of the one step locations that are correctly predicted to the total number of one step predictions made for every queried trajectory in the test set.

6.2.3 Average distance error

At a specific step of prediction, Distance Error is defined as the average spatial distance between the predicted

coordinates and actual coordinates. Let predicted coordinates of the jth step be (x_j^p, y_j^p) and true coordinates be (x_j, y_j) . Then, the distance error at that step is given by,

$$DE = \sqrt{(x_j^p - x_j)^2 + (y_j^p - y_j)^2}$$

6.2.4 One-step distance error (ODE)

Similar to the OA metric, the ODE metric is defined as the average distance error for just the next location prediction.

6.3 Compared frameworks

Since our novel framework mainly focuses on combining the effective clustering of trajectories with that of deep learning methods for prediction to improve the long term prediction performance, we primarily compare our framework to the previous state-of-the-art clustering based but non-deep learning methods and non-clustering deep learning methods.

6.3.1 LSTM RNN [6]

We compare our approach with the award winning LSTM RNN network [4] that reads the trajectory one GPS point at a time from the beginning to the end of each input prefix. LSTMs, in general are considered to be an upliftment over conventional recurrent architectures such as RNNs as they significantly ameliorate the problem of vanishing and exploding gradients. As an improvement, the architecture is adopted where a consecutive varying range of GPS points are used to predict the next location unlike

the previous case where only a single previous location is used to prediction.

6.3.2 objectTra-MM [4]

objectTra-MM is built up of a combination of two novel models exploiting the similarity between objects and the similarity between trajectories. The first model clusters similar objects based on their spatial localities and then adjoins this entire architecture to a variable order Markov model. It is named object-clustered Markov model (object-MM). The second model carries out the identical process but using the trajectories and similarities between them for clustering. Effective integration of this leads to the final next location predictor named objectTra-MM.

6.4 Execution

After all of the preprocessing, the k-nearest neighbor clustering algorithm is performed at the node to cluster the entire dataset into 500 clusters in the standard parameters setting using Apache Spark MLlib's [17] pyspark. Note that the optimal cluster number of 500 is chosen after exploring the performances of several total cluster numbers and corresponding cluster sizes. We shall also be briefly demonstrating the performance achieved with different total cluster numbers and hence also different cluster sizes. Apache Spark's in-memory processing is highly efficient as it is performed directly at the node. After that, the pyspark scripted Nearest Prototype Rule (NPR) is applied to the remaining uncluttered trajectories for clustering.

As per figure 1 once the trajectories are completely clustered, a corresponding LSTM RNN model having the same architecture is trained for each of the clusters. Each of the models was trained for 25 epochs and a batch size of

64. The LSTM RNN model was implemented using TensorFlow 2.0. After the completion, a partial trajectory is queried to Apache's HDFS using Spark in real-time. Apache Spark returns the cluster id to which the partial trajectory belongs using Nearest Prototype Rule (NPR). This queried partial trajectory is then fed to the LSTM RNN of the corresponding cluster and as soon as the next location is predicted, the partial trajectory is updated by appending the predicted location to it and queried using Apache Spark. This process is repeated until the end of the trajectory. In the rest of the paper, we refer to our proposed approach as Clustering-based LSTM.

When we compared our results, with two existing, most commonly referred approaches, namely LSTM RNN [6] and objectTra-MM [4]. Each of these models were trained using the training set as used for the proposed approach. The trajectories of the validation set were then used to analyze the performance. As the Clustering-based LSTM (Proposed) and objectTra-MM (Existing) models are having more than one predicting models hence average accuracy achieved by the corresponding LSTM and HMM model of each cluster is used for comparison.

7 Results and discussion

7.1 Comparison of LSTM RNN, objectTra-MM, and cluster LSTM for long-term predictions

Figure 5 shows the performance in terms of prediction accuracy w.r.t. the number of steps involved in all the three approaches. The Cluster LSTM approach outperformed the LSTM RNN as well as objectTra-MM in terms of accuracy. In the proposed approach 8th step accuracy is better than the 5th step accuracy of the other two approaches. Also,

Fig. 5 The above graphs shows the comparison between N steps Prediction Accuracy between the three approaches

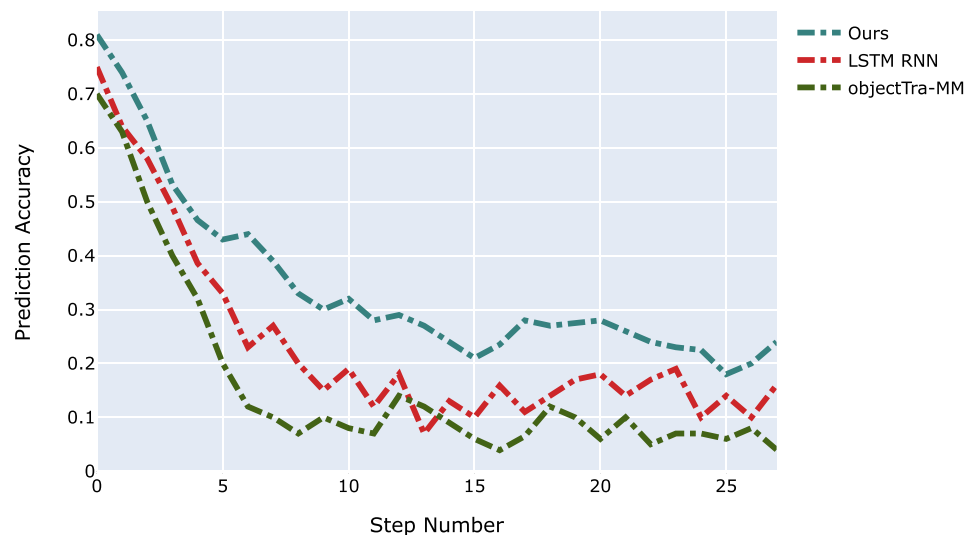


Fig. 6 The above graphs show the comparison between N steps Distance Error between the three approaches

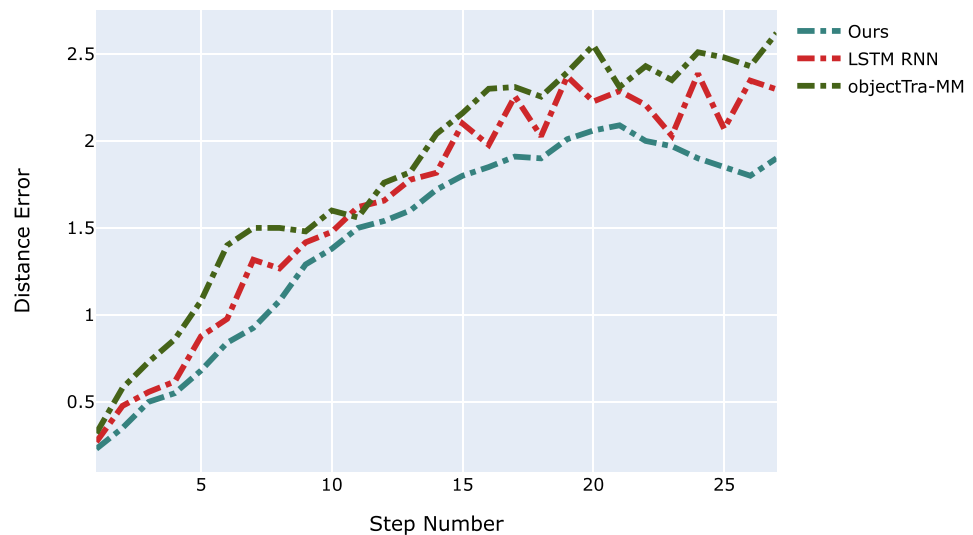


Table 2 Comparison of average prediction accuracy and average distance error

Model	Average PA	Average DE (km)
Cluster LSTM	0.639	0.332
LSTM RNN	0.569	0.424
objectTra-MM	0.524	0.641

Table 3 Comparison of Next location Prediction Accuracy between all three models and Next location Distance Error between all three models

Model	OA	ODE (km)
Cluster LSTM	0.8	0.23
LSTM RNN	0.75	0.29
objectTra-MM	0.32	0.32

as the number of steps increases the proposed approach outperforms the other two approaches with the bigger margins. This implies that, the proposed approach is an even better choice for long distance trajectory predictions. These favorable results are achieved through the proposed approach (Cluster-LSTM) by leveraging the variance reduction capacity from the clustering technique and fusing it with the sequence prediction algorithm of LSTM which increases reliability for the long-term prediction. This can also be the reason for another interesting observation w.r.t. performance of Cluster-LSTM which shows less fluctuation on the graph as compared to the other two approaches. However with an increasing number of steps the prediction accuracy of all three approaches declines. As fewer instances are available for long distance trajectories hence enough information is not gathered to accurately predict future locations for long distances. Particularly, objectTra-MM’s accuracy is seen to have dropped drastically after just 5 to 6 prediction step. This is because it cannot capture the sequence correlation for larger sequences, hence are often not suitable for long distance prediction. This further establishes the strength of LSTM in dealing with time-series data and capture the long sequence correlation in the data.

Figure 6 shows the Distance Error against the number of steps. The graph demonstrates that the proposed model

has a minimum Distance Error in comparison to the other two approaches. The experiment was conducted for the number of steps as high as 25 and in all the cases the proposed method demonstrates the minimum Distance Error.

It is observed through experiments that the clusters have a trajectory pattern with an average length of 5. Hence, the prediction accuracy and distance error are calculated considering the average of 5 steps prediction. Table 2 shows the comparative analysis of the proposed approach with the compared two approaches. The average prediction accuracy of the proposed approach Clustered-LSTM is approx 64% while the other two models show the average accuracy of below 60%. Also, the average distance error of the proposed approach is the lowest of all the three models and almost half of the distance error of the objectTra-MM.

7.2 Next location prediction

In this experiment, we compare the cluster-LSTM to the other two approaches for predicting upcoming consecutive locations. Given a particular taxi’s current location, the task is to forecast the next location where the taxi may be going. Table 3 shows one-step accuracy (OA) and one-step distance error (ODE) on the experimented T-drive dataset. The cluster LSTM approach predicts the next location with distance error of less than a quarter kilometer and with approximately 80% accuracy. Even though the

one-step distance error for cluster LSTM is about 0.23 kms, its Average Distance Error is 0.332 kms. Whereas, for LSTM RNN and objectTra-MM, the difference between average DE and ODE is much larger. This suggests better performance robustness and retention capacity of Cluster LSTM for long-term prediction tasks. One another important observation is that the average sampling interval for the considered T-Drive dataset is 707 meters. Had the sampling interval been more frequent, we suggest that this could improve the performance of the cluster LSTM model. Summarizing, we would suggest that Cluster LSTM outperforms both LSTM RNN and objectTra-MM for next location prediction task.

ancewithethicalstan

7.3 Considering only latest locations of partial trajectory for prediction

The conventional procedure is to find the most suitable cluster for the queried partial trajectory $T_p = \{C_1, C_2, C_3, \dots, C_t\}$. For a T_p , once the best cluster is chosen as per the procedure, the length of the queried partial trajectory T_p consecutively increases as every prediction is appended to the T_p . Further, this updated T_p is queried for next location prediction and then again updated by appending the predicted location and so on.

Now we modify this basic conventional procedure. Instead of feeding in fully known partial trajectory T_p , we only consider the latest n number of output predictions by the cluster LSTM to choose the best matching cluster and then further predict the next step. Therefore, we choose increasing number of latest location prediction by the Cluster LSTM with known partial trajectories until prediction and then investigate its performance.

The Average Distance Error for a corresponding number of latest locations considered of known partial trajectories is shown in Fig. 7. It can be seen from the figure that the performance doesn't drop until two or three locations of the queried partial trajectory. The average distance error is seen to be consistently increasing after the three latest locations. This happens because as the length of T_p increases, its path probability exponentially decreases, which means both the chance of identifying and being assigned to the correct cluster as well as predicting correct coordinates starts degrading.

7.4 Effect of number of clusters

In this experiment, we study the function of performance of Cluster LSTM as a function of the number of clusters K . Figure 8 shows Average DE for different No. of Clusters. The higher the number of clusters, the tighter the cluster boundaries. The Average Distance Error is seen to

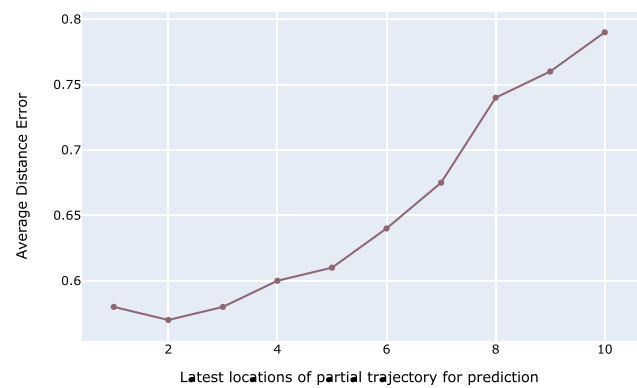


Fig. 7 The graph plot of Average Distance Error against latest locations of partial trajectory used to select best cluster in the NPR Procedure

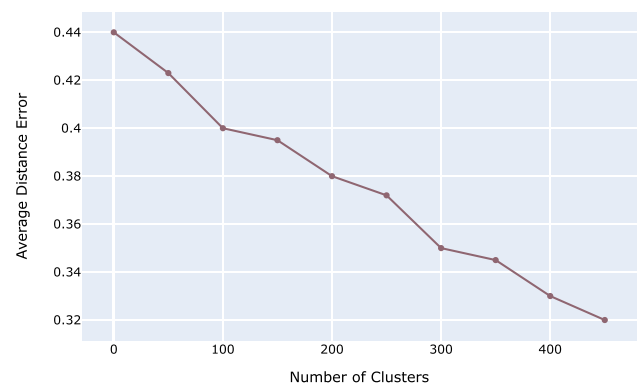


Fig. 8 Effect of number of clusters

decrease as the number of cluster increases. This is because the larger number of clusters corresponds to more intricately diversified clusters. Figure 8 shows that the cluster LSTM performance improves with the higher number of clusters. However, with higher number of clusters, more LSTMs need to be trained, and hence, system complexity increases. Moreover, the performance of Cluster LSTM does not improve significantly above a certain value corresponding to the number of clusters. For our experiments we note the highest number of cluster value to be 500, after which the accuracy does not improve significantly. Rather, the model complexity factor becomes more concerning, as the complexity starts to elevate rapidly because of increase in the number of LSTM model corresponding to each cluster.

7.5 Time performance analysis

The latency in prediction is another important criterion for the real-time trajectory prediction besides prediction accuracy and distance error. The average prediction time

Table 4 Average prediction time of all the three approaches (in seconds)

	Prediction time
Cluster LSTM	0.96
LSTM RNN	0.79
objectTra-MM	0.51

for all three approaches is presented in Table 4. Prediction time observed is the highest for the proposed approach i.e. Clustered-LSTM. The added latency is obvious due to the increased architectural complexity. However, this prediction time still qualifies for the real-time requirement as this is much smaller than the travel time. From the data set it is observed that it takes an average time interval of 214 seconds to cover the average distance of 707 meters between any specific sampled location and its next sampled location. So, the latency of 0.96 seconds is much smaller than 214 seconds, and an increase in latency in comparison to other methods can be considered insignificant. Therefore, when it comes to predicting long distance trajectories, the overall results of the proposed approach have demonstrated significantly improved accuracy and distance metrics as well as the outstanding capability of predicting in real-time.

8 Conclusion and future directions

In this paper, we proposed a novel method to predict a complete long distance trajectory from the queried partial trajectory in extremely large-scale data of the dense road network. The proposed approach outperformed the existing approaches in Prediction accuracy and Distance error with substantial gain and particularly for long distance trajectory predictions. Moreover the deep learning model is being trained without high-end computing resources. The clustered approach made a large volume of data manageable for LSTM training. The proposed approach can be seen as a step towards bridging the gap for long-distance trajectory prediction on the large-scale data by clustering data into several groups and applying deep learning techniques on it. The conducted experiments establish that clustered data improves the predictive performance of the model as compared to other approaches where the historical trajectories of vehicles of the entire region are used as it is. It helped in reducing the data variance that enabled better learning through reduced overfitting and resulted in faster convergence in LSTM architecture. Also, our approach helps in satisfying the requirement for real-time predictions, and the use of a big data environment framework to accommodate the proposed model, makes it scalable.

We further see opportunities for improvement in the proposed model in terms of improved algorithms to learn appropriate limits for members in each cluster on its own. Deploying this model over the cloud can be an obvious next step. The presented work, being straightforward and intuitive yet highly scalable and robust, may also lead to form a base framework for the development of more modular and extensive trajectory prediction systems.

Declarations

Conflict of interest The authors declare that they have no conflict of interest.

Open Access This article is licensed under a Creative Commons Attribution 4.0 International License, which permits use, sharing, adaptation, distribution and reproduction in any medium or format, as long as you give appropriate credit to the original author(s) and the source, provide a link to the Creative Commons licence, and indicate if changes were made. The images or other third party material in this article are included in the article's Creative Commons licence, unless indicated otherwise in a credit line to the material. If material is not included in the article's Creative Commons licence and your intended use is not permitted by statutory regulation or exceeds the permitted use, you will need to obtain permission directly from the copyright holder. To view a copy of this licence, visit <http://creativecommons.org/licenses/by/4.0/>.

References

1. Altmé F, de La Fortelle A (2017) An lstm network for highway trajectory prediction. In: 2017 IEEE 20th international conference on intelligent transportation systems (ITSC), IEEE, pp 353–359
2. Ashbrook Daniel, Starner Thad (2003) Using gps to learn significant locations and predict movement across multiple users. *Personal Ubiquitous Comput* 7(5):275–286
3. Jie Bao Yu, Zheng David Wilkie, Mokbel Mohamed (2015) Recommendations in location-based social networks: a survey. *Geoinformatica* 19:07
4. Chen M, Liu Y, Yu X (2015) Predicting next locations with object clustering and trajectory clustering. In: *Pacific-Asia Conference on Knowledge Discovery and Data Mining*. Springer, pp 344–356
5. Chon Y, Shin H, Talipov E, Cha H (2012) Evaluating mobility models for temporal prediction with high-granularity mobility data. In: 2012 IEEE international conference on pervasive computing and communications, IEEE, pp 206–212
6. De Brébisson A, Simon É, Auvolat A, Vincent P, Bengio Y (2015) Artificial neural networks applied to taxi destination prediction. *arXiv preprint arXiv:1508.00021*
7. Endo Y, Nishida K, Toda H, Sawada H (2017) Predicting destinations from partial trajectories using recurrent neural network. In: *Pacific-Asia conference on knowledge discovery and data mining*. Springer, pp 160–172
8. de Castro AG, Talavera-Llames R, Troncoso A, Koprinska I, Martínez-Álvarez F (2018) Multi-step forecasting for big data time series based on ensemble learning. *Knowledge-Based Systems*, p 10

9. Gambs S, Killijian M-O, del Prado Cortez MN (2012) Next place prediction using mobility markov chains. In: Proceedings of the first workshop on measurement, privacy, and mobility. ACM, p 3
10. Giannotti Fosca, Nanni Mirco, Pedreschi Dino, Pinelli Fabio, Renso Chiara, Rinzivillo Salvatore, Trasarti Roberto (2011) Unveiling the complexity of human mobility by querying and mining massive trajectory data. *VLDB J Int J Very Large Data Bases* 20(5):695–719
11. Hall LO (2018) A primer on cluster analysis by James C. Bezdek (by the book). *IEEE Syst Man Cybern Mag* 4(1):48–50
12. Han Binh, Liu Ling, Omiecinski Edward (2013) Road-network aware trajectory clustering: integrating locality, flow, and density. *IEEE Trans Mobile Comput* 14(2):416–429
13. Ishikawa Y, Tsukamoto Y, Kitagawa H (2004) Extracting mobility statistics from indexed spatio-temporal datasets. In: *STDBM*, pp 9–16
14. Liebig Thomas, Piatkowski Nico, Bockermann Christian, Morik Katharina (2017) Dynamic route planning with real-time traffic predictions. *Inf Syst* 64:258–265
15. Lu H-C, Tseng V, Yu P (2011) Mining cluster-based mobile sequential patterns in location-based service environments. *IEEE Trans Knowl Data Eng* 23:914–927
16. Lv Qiujian, Qiao Yuanyuan, Ansari Nirwan, Liu Jun, Yang Jie (2016) Big data driven hidden markov model based individual mobility prediction at points of interest. *IEEE Tran Veh Technol* 66(6):5204–5216
17. Meng X, Bradley J, Yavuz B, Sparks E, Venkataraman S, Liu D, Freeman J, Tsai DB, Amde M, Owen S, Xin D, Xin R, Franklin MJ, Zadeh R, Zaharia M, Talwalkar A (2016) Mllib: Machine learning in apache spark. *J Mach Learn Res* 17(34):1–7
18. Monreale A, Pinelli F, Trasarti R, Giannotti F (2009) Wherenext: a location predictor on trajectory pattern mining. In: Proceedings of the 15th ACM SIGKDD international conference on Knowledge discovery and data mining. ACM, pp 637–646
19. Morzy M (2007) Mining frequent trajectories of moving objects for location prediction. In: International workshop on machine learning and data mining in pattern recognition. Springer, pp 667–680
20. Moshfeghi M (2016) Navigation system and methods for route navigation, US Patent 9,360,337
21. Pan Z, Liang Y, Wang W, Yu Y, Zheng Y, Zhang J (2019) Urban traffic prediction from spatio-temporal data using deep meta learning. In: Proceedings of the 25th ACM SIGKDD international conference on knowledge discovery and data mining. Association for Computing Machinery, New York, pp 1720–1730
22. Park SH, Kim B, Kang CM, Chung CC, Choi JW (2018) Sequence-to-sequence prediction of vehicle trajectory via lstm encoder-decoder architecture. In: 2018 IEEE intelligent vehicles symposium (IV), IEEE, pp 1672–1678
23. Petrou P, Nikitopoulos P, Tampakis P, Glenis A, Koutroumanis N, Santipantakis GM, Patroumpas K, Vlachou A, Georgiou H, Chondrodima E et al. (2019) Argo: a big data framework for online trajectory prediction. In: Proceedings of the 16th international symposium on spatial and temporal databases, pp 194–197
24. Qiao S, Han N, Wang J, Li R-H, Gutierrez LA, Wu X (2017) Predicting long-term trajectories of connected vehicles via the prefix-projection technique. *IEEE Trans Intell Transp Syst* 19(7):2305–2315
25. Roh G-P, Hwang S-w (2010) Nncluster: An efficient clustering algorithm for road network trajectories. In: International conference on database systems for advanced applications. Springer, pp 47–61
26. Rossi Alberto, Barlacchi Gianni, Bianchini Monica, Lepri Bruno (2019) Modelling taxi drivers' behaviour for the next destination prediction. *IEEE Trans Intell Transp Syst*
27. Shvachko K, Kuang H, Radia S, Chansler R (2010) The hadoop distributed file system. In: Proceedings of the 2010 IEEE 26th symposium on mass storage systems and technologies (MSST), MSST '10. IEEE Computer Society, USA, pp 1–10
28. Song Chaoming, Zehui Qu, Blumm Nicholas, Barabási Albert-László (2010) Limits of predictability in human mobility. *Science* 327(5968):1018–1021
29. Sung C, Feldman D, Rus D (2012) Trajectory clustering for motion prediction. In: 2012 IEEE/RSJ international conference on intelligent robots and systems, pp 1547–1552
30. Sung C, Feldman D, Rus D (2012) Trajectory clustering for motion prediction. In 2012 IEEE/RSJ international conference on intelligent robots and systems, IEEE, pp 1547–1552
31. Torres JF, Fernández AM, Lora AT, Martínez-Álvarez F (2017) Deep learning-based approach for time series forecasting with application to electricity load. In *IWINAC*
32. Verma JP, Mankad SH, Garg S (2018) A graph based analysis of user mobility for a smart city project. In: International conference on next generation computing technologies. Springer, pp 140–151
33. Wang C, Ma L, Li R, Durrani TS, Zhang H (2019) Exploring trajectory prediction through machine learning methods. *IEEE Access* 7:101441–101452
34. Wiest J, Höffken M, Kreßel U, Dietmayer K (2012) Probabilistic trajectory prediction with gaussian mixture models. In: 2012 IEEE intelligent vehicles symposium, IEEE, pp 141–146
35. Won J-I, Kim S-W, Baek J-H, Lee J (2009) Trajectory clustering in road network environment. In: 2009 IEEE symposium on computational intelligence and data mining, IEEE, pp 299–305
36. Yao H, Wu F, Ke J, Tang X, Jia Y, Lu S, Gong P, Ye J, Li Z (2018) Deep multi-view spatial-temporal network for taxi demand prediction. In: *CoRR*, abs/1802.08714
37. Yavaş Gökhan, Katsaros Dimitrios, Ulusoy Özgür, Manolopoulos Yannis (2005) A data mining approach for location prediction in mobile environments. *Data Knowl Eng* 54(2):121–146
38. Ying JJ-C, Lee W-C, Weng T-C, Tseng VS (2011) Semantic trajectory mining for location prediction. In: Proceedings of the 19th ACM SIGSPATIAL international conference on advances in geographic information systems. ACM, pp 34–43
39. Yuan J, Zheng Y, Xie X, Sun G (2011) Driving with knowledge from the physical world. In: Proceedings of the 17th ACM SIGKDD international conference on Knowledge discovery and data mining, pp 316–324
40. Yuan J, Zheng Y, Zhang C, Xie W, Xie X, Sun G, Huang Y (2010) T-drive: driving directions based on taxi trajectories. In Proceedings of the 18th SIGSPATIAL international conference on advances in geographic information systems. ACM, pp 99–108
41. Zaharia M, Chowdhury M, Franklin MJ, Shenker S, Stoica I (2010) Spark: Cluster computing with working sets. In: Proceedings of the 2nd USENIX conference on hot topics in cloud computing, hotcloud'10. USENIX Association, USA, pp 10

Publisher's Note Springer Nature remains neutral with regard to jurisdictional claims in published maps and institutional affiliations.