Online Long-Term Trajectory Prediction Based on Mined Route Patterns

Petros Petrou[1(✉)], Panagiotis Tampakis[1], Harris Georgiou[1], Nikos Pelekis[2], and Yannis Theodoridis[1]

[1] Department of Informatics,
80 Karaoli & Dimitriou str., P.O. 18534, Piraeus, Greece
{ppetrou,ptampak,hgeorgiou,ytheod}@unipi.gr
[2] Department of Statistics and Insurance Science, University of Piraeus,
80 Karaoli & Dimitriou str., P.O. 18534, Piraeus, Greece
npelekis@unipi.gr

Abstract. In this paper, we present a Big data framework for the prediction of streaming trajectory data by exploiting mined patterns of trajectories, allowing accurate long-term predictions with low latency. In particular, to meet this goal we follow a two-step methodology. First, we efficiently identify the hidden mobility patterns in an offline manner. Subsequently, the trajectory prediction algorithm exploits these patterns in order to prolong the temporal horizon of useful predictions. The experimental study is based on real-world aviation and maritime datasets.

Keywords: Trajectory prediction · Trajectory clustering · Mobility patterns · Big data

1 Introduction

Huge amounts of tracking data are being generated on a daily basis by GPS-enabled devices which are stored for analytics purposes. These constitute a rich source for inferring mobility patterns and characteristics, which, in turn, can be valuable to a wide spectrum of novel applications and services, from mobile social networking to aviation traffic monitoring. During the last years, such data have attracted the interest of data scientists, both in industry and academia, and are used to extract knowledge and useful features on what, how and for how long the moving entities are conducting individual activities related to specific circumstances. One of the most challenging tasks is to exploit these data by means of identifying historical mobility patterns, which, in turn, can gauge the procedure of discovering what the moving entities might do in the future. As a consequence, predictive analytics over mobility data have become increasingly important and are ubiquitous in many application fields [2,30,43].

The problem of predictive analytics over mobility data finds two broad categories of application scenarios. The first scenario involves cases where the moving entities are traced in real-time to produce analytics and compute short-term

K. Tserpes et al. (Eds.): MASTER 2019, LNAI 11889, pp. 34–49, 2020.
https://doi.org/10.1007/978-3-030-38081-6_4

predictions, which are time-critical and need immediate response. The prediction includes either location- or trajectory-related tasks. Short-term location and trajectory prediction facilitates the efficient planning, management, and control procedures while assessing traffic conditions in the road, sea and air transportation field. The latter can be extremely important in domains where safety, credibility and cost are critical and a decision should be made by considering adversarial to the environment conditions to act immediately. The second scenario involves cases where long-term predictions are important to identify cases which exceed regular mobility patterns, detect outliers and determine a position or a sequence of positions at a given time interval in the future. In this case, although response time is not a critical factor, it is still crucial in order to identify correlations between historical mobility patterns and patterns which are expected to appear. Long-term location and trajectory prediction can assist to achieve cost efficiency or, when contextual information is provided (e.g., weather conditions), it can ensure public safety in different transportation modes (land, sea, air).

As the maritime and the Air Traffic Management (ATM) domains have major impact to the global economy, a constant need is to advance the capability of systems to improve safety and effectiveness of critical operations involving a large number of moving entities in large geographical areas [22]. Towards this goal, the exploitation of heterogeneous data sources, which offer vast quantities of archival and high-rate streaming data, is crucial for increasing the computations accuracy when analysing and predicting future states of moving entities. However, operational systems in these domains for predicting trajectories are still limited mostly to a short-term look-ahead time frame, while facing increased uncertainty and lack of accuracy.

Motivated by these challenges, we present a Big data solution for online trajectory prediction by exploiting mined patterns of trajectories from historical data sources. Our approach offers predictions such as 'estimated flight of an aircraft over the next 10 min' or 'predicted route of a vessel in the next hour', based on their current movement and historical motion patterns in the area. The proposed framework incorporates several innovative modules, operating in streaming mode over surveillance data, to deliver accurate long-term predictions with low latency requirements. Incoming streams of moving objects' positions are cleansed, compressed, integrated and linked with archival and contextual data by means of link discovery methods.

This paper includes three main contributions: (a) we devise a big-data methodology/algorithm that solves the *Future Location Prediction* (FLP) problem in a effective and highly scalable way; (b) the design and implementation of our algorithm on top of state-of-the-art Big data technologies (namely Spark and Kafka); (c) extensive experimental study in large real datasets from the maritime and aviation domains. To the best of our knowledge, in contrast to related state-of-the-art systems [8,10] and research approaches [7], our approach is unique as a Big data framework capable of providing long-term trajectory predictions in an online fashion.

This paper is organized as follows. Section 2 presents the related work from the field of trajectory prediction and long-term future location prediction, especially from the maritime and aviation domains. Next, Sect. 3 describes the system overview and architecture of the proposed approach, as well as how this fits into the Big data scope. Section 4 presents the mobility pattern discovery module, in the form of a novel and scalable subtrajectory clustering Big Data solution, which is the first stage of this approach. Predictive models, which is the second stage, are described in Sect. 4.2. The experimental study in Sect. 5 includes datasets from both the maritime and the aviation domain. Finally, the conclusions and future aspects of this work are described briefly in Sect. 6.

2 Background

The trajectory of a moving object is defined as: $< (p_0, t_0), (p_1, t_1), ..., (p_i, t_i), ... >$, where p_i is the location of the object in d-dimensional space (typically, $d = 2$ or 3, for a movement in plane or volume, respectively) and t_i is the time this recording was made, with $t_i < t_{i+1}$ (i.e., the sequence is chronologically ordered).

Having this at hand, two main prediction-related problems can be stated for moving objects: Future Location Prediction (FLP) and Trajectory Prediction (TP) [14]. In these definitions we adopt the following terminology: symbols p and t refer to recorded or given locations and timestamps, respectively, whereas symbols p* and t* refer to (future) predicted locations and timestamps, respectively.

Problem Definition 1 **Future Location Prediction (FLP)**: Given (a) the incomplete trajectory $< (p_0, t_0), (p_1, t_1), ..., (p_{i-1}, t_{i-1}) >$ of a moving object o, consisting of its time-stamped locations recorded at past i time instances, and (b) an integer value $j \geq 1$, predict $< (p_i^*, t_i), ..., (p_{i+j-1}^*, t_{i+j-1}) >$, i.e., the objects's anticipated locations at the following j time instances.

Problem Definition 2 **Trajectory Prediction (TP)**: Given (a) the incomplete trajectory $< (p_0, t_0), (p_1, t_1), ..., (p_{i-1}, t_{i-1}) >$ of a moving object o consisting of its time-stamped locations recorded at past i time instances and (b) a target region R, predict $< (p_i^*, t_i), ..., (p^*, t^*) >$, where $p^* \in R$, i.e., the object's anticipated locations until it matches a point p^* in R (note: p^* may be never reached exactly).

Using these two baseline definitions for the FLP and TP tasks, a wide variety of algorithms can be employed to predict either sequences of future points (FLP) or the evolution of entire trajectories (TP). In the context of this work, the interest is focused specifically in TP or, complementary, to long-term FLP, i.e., with sufficiently large look-ahead time frames.

A typical example of a FLP method is presented in [38], where the authors propose TPR*-tree (index-based), which derives from TPR-tree, and exploits the characteristics of dynamic moving objects in order to retrieve only those which

will meet specific spatial criteria within the given time interval, i.e., query window, in the future. Every moving object is represented by a Minimum Bounding Rectangle (MBR) along with a Velocity Bounding Rectangle (VBR). The proposed index integrates novel insertion and deletion algorithms to enhance performance and supports predictive spatio-temporal queries by specifying a query region q_r and a future time interval q_t and retrieving the set of objects that will intersect q_r at any timestamp $t \in q_t$.

The previous method can be considered as a FLP-based approach, mostly in the context of the long-term prediction. There is also a number of TP-based approaches that address the prediction task in a similar way. In theory, every FLP method can be transformed to a full TP model, given a specific granularity upon which the same method is applied iteratively. The main difference with 'pure' TP methods is that in this case the prediction errors are accumulated with each step (e.g. via multi-step Linear Regression) along the prediction track, thus making the predicted points increasingly error-prone. In contrast, TP methods forecast the complete trajectory as a whole, thus making each predicted point equally error-prone. Regarding en route climb TP, one of the major aspects of decision support tools for ATM, Coppenbarger [8] discusses the exploitation of real-time aircraft data, such as aircraft state, aircraft performance, pilot intent and atmospheric data for improving ground-based TP. The problem of climb TP is also discussed by Thipphavong, Schultz et al. [39], as it constitutes a very important challenge in ATM. In this work, an algorithm that dynamically adjusts modeled aircraft weights is developed, exploiting the observed track data to improve the accuracy of TP for climbing flights.

In the area of **stochastic approaches**, Ayhan and Samet [4] introduce a novel stochastic approach to aircraft trajectory prediction problem, which exploits aircraft trajectories, based on Hidden Markov Models (HMM), modeled in space and time by using a set of spatio-temporal data 4-D cubes (latitude, longitude, altitude, time) enriched by weather parameters. Gong and McNally [16] proposed a methodology for automated trajectory prediction analysis, specifically for splitting the process in separated stages according to the flight phases. The purpose is to identify flights, as described by actual radar tracks, which show unpredictable modifications of their aircraft intent and can be considered outliers. In another work by Ayhan and Samet [5], the authors investigate the applicability of the HMM for TP on only one phase of a flight, specifically the climb after takeoff. Moreover, they address the problem of incorporating weather conditions in their model, as they represent a major factor of uncertainty in all TP applications.

Regression and clustering are also two main areas of interest when applying machine learning methods in TP. Neural Networks (NN) have been proposed in various works as the core regression model for the task of TP. Le Fablec and Alliot [12] have introduced NNs for the specific problem of predicting an aircraft trajectory in the vertical plane, i.e., its altitude profile with the time. Cheng, Taoya, et.al. [6] employ a data mining statistical approach on the radar tracks of aircrafts to infer the future air traffic flows using Neural Networks (NN) and

exploiting data grouped in seven 'weekday' categories for predicting the Estimated Time of Arrival (ETA) at designated fixes and airports as output. Leege, Paassen and Mulder [22] also address the specific TP task of predicting arrival routes and times via Generalized Linear Models (GLM), merging together air traffic following fixed arrival routes, meteorological data and two aircraft types.

In a very recent work of TP in aviation, Georgiou et al. [15] introduce flight plans, localized weather and aircraft properties as trajectory annotations that enable modelling in a space higher than the typical 4-D spatio-temporal. A multi-stage hybrid approach is employed for a new variation of the core TP task, the so called *Future Semantic Trajectory Prediction* (FSTP), including clustering the enriched trajectory data using a semantic-aware similarity function as distance metric. Subsequently, a separate predictive model is trained for each cluster, using a non-uniform graph-based grid that is formed by the waypoints of each flight plan. In practice, flight plans constitute a constrained-based training of each predictive model, one for each waypoint, independently. Various types of predictive models are tested, including HMM, linear regressors, regression trees and feed-forward NNs. The results show very narrow confidence intervals for the per-waypoint TP errors in HMM, while the more efficient linear and non-linear regressors exhibit 3-D spatial accuracy much lower than the current state-of-the-art, up to a factor of five compared to 'blind' TP for complete flights, in the order of 2–3 km compared to the actual flight routes.

Concerning mobility pattern discovery, the aim is to identify several types of collective behavior patterns among moving objects like the so-called flock pattern [20,41] and the notion of moving clusters [19]. A number of research efforts that emerged from the above ideas are the approaches of convoys [18,28], platoons [23], swarms [24], gathering pattern [42] and traveling companion [37]. Trasarti et al. [40] introduced "individual mobility patterns" in order to extract the most representative trips of a specific moving object, so that they can predict object's future locations. However, all of the aforementioned approaches are centralized and cannot scale to massive datasets. Towards this, the problem of convoy discovery in a distributed environment by employing the MapReduce programming model was studied both in [27]. An approach that defines a new generalized mobility pattern which models various co-movement patterns in a unified way and is deployed on a modern distributed platform (i.e., Apache Spark) to tackle the scalability issue is presented in [13].

Another line of research, tries to discover groups of either entire or portions of trajectories considering their routes. A typical strategy is to transform trajectories to a multi-dimensional space and then apply well-known clustering algorithms such as OPTICS [3] and DBSCAN [11]. Another approach is to define an appropriate similarity function and embed it to an extensible clustering algorithm [26]. Nevertheless, trajectory clustering is an "expensive" operation and centralized solutions cannot scale to massive datasets. Furthermore, [34] proposes a MapReduce approach that aims to identify frequent movement patterns from the trajectories of moving objects. In [17] the authors tackle the problem of parallel trajectory clustering by utilizing the MapReduce programming model

and Hadoop. They adopt an iterative approach similar to k-Means in order to identify a user-defined number of clusters, which leads to a large number of MapReduce jobs.

However, discovering clusters of complete trajectories can overlook significant patterns that might exist only for portions of their lifespan. To deal with this, the authors of [21] propose TraClus, a partition-and-group framework for clustering 2-D moving objects which segments the trajectories based on their geometric features, and then clusters them by ignoring the temporal dimension. A more recent approach to the problem of subtrajectory clustering, is S^2T-Clustering [32], where the authors take into account the temporal dimension, and the segmentation of a trajectory takes place whenever the density of its spatiotemporal 'neighborhood' changes significantly. The segmentation phase is followed by a sampling phase, where the most representative subtrajectories are selected and finally the clusters are built "around" these representatives. A similar approach is adopted in [1], where the authors aim at identifying common portions between trajectories, with respect to some constraints and/or objectives, by taking into account the "neighborhood" of each trajectory. These common subtrajectories are then clustered and each cluster is represented by a pathlet, which is a point sequence that is not necessarily a subsequence of an actual trajectory. A different approach is presented in QuT-Clustering [31] and [35], where the goal is, given a temporal period of interest W, to efficiently retrieve already clustered subtrajectories, that temporally intersect W. To achieve this, a hierarchical structure, called ReTraTree (Representative Trajectory Tree) that effectively indexes a dataset for subtrajectory clustering purposes, is built and utilized.

The approach presented in this paper combines several aspects and ideas from the methods cited above, in order to develop a highly adaptive, long-term, Big data framework for FLP which is experimentally evaluated with datasets from both the maritime and the aviation domain. More specifically, this two-stage approach includes: (a) mobility pattern discovery from the historical movement of the moving objects; and (b) employ optimal estimations of FLP in the sense of maximum likelihood, as they are dictated by the identified patterns. Furthermore, some promising experimental results are presented for real datasets from both domains, as well as performance indicators for deployment in a Big data platform.

3 Overview of the Approach

In this section we describe the architecture of our proposed framework, which follows a typical lambda architecture [25] that combine streaming and batch layers to implement an end-to-end big data prediction solution. The proposed framework, as depicted in Fig. 2, consists of two main modules, namely, Pattern Extraction and Future Location Prediction (FLP). All modules are build on top of big data engines, so that they can be scalable and offer low latency. Kafka is used as an integration network for online toolboxes and a shared storage (i.e. Apache Hadoop HDFS) is used in order to update existing patterns or add new

ones. Subsequently, the FLP module can "read" these patterns and execute the prediction pipeline.

At first, each moving object sends its location via traditional network protocols and then a Kafka producer collects all positions and pushes them to a Kafka topic. The Pattern Extraction module identifies "typical routes", in an offline manner. Finally, these "typical routes" are broadcast among all slaves and the FLP module combines them with the live incoming stream of data in order to predict the future location for each object (Fig. 1).

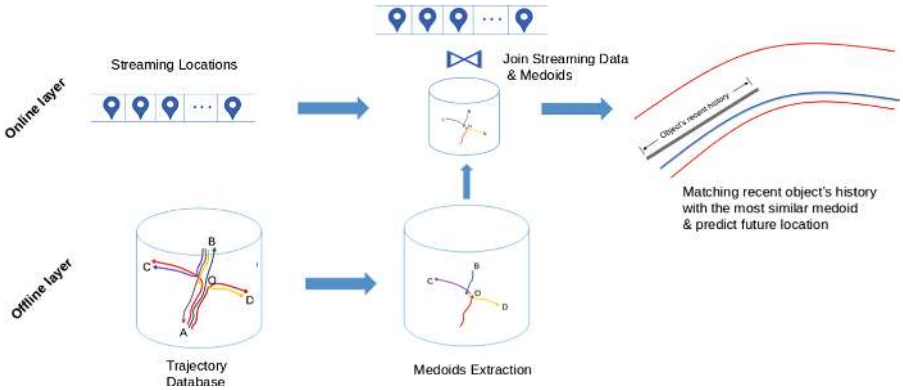

Fig. 1. Data workflow of the proposed framework

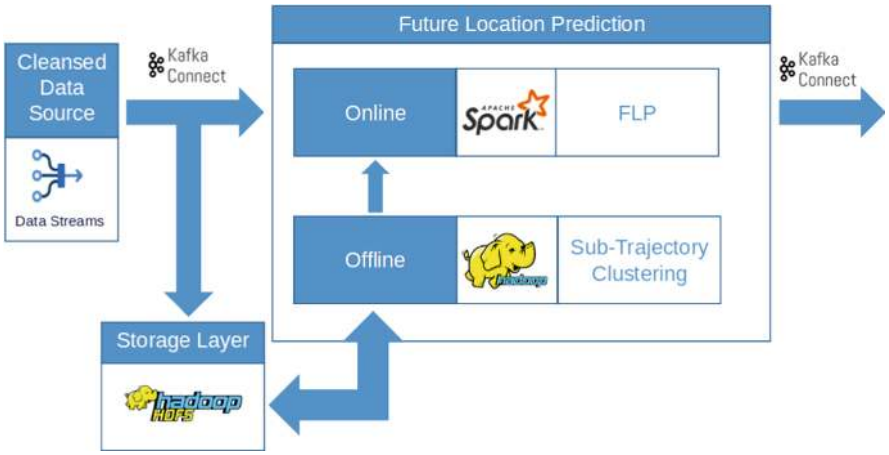

Fig. 2. Architecture of the proposed framework

4 Methodology

4.1 Offline Step: Mobility Pattern Extraction Based on Sub-trajectory Clustering

The goal of this module is to identify frequent patterns of movement that will assist the FLP module to increase the accuracy of the predictions. The research so far has focused mainly in methods that aim to identify specific collective behavior patterns among moving objects, such as flocks, convoys and swarms [44], or methods that try o identify patterns that are valid for the entire lifespan of the moving objects [9,26]. However, discovering clusters of entire trajectories can overlook significant patterns that might exist only for small portions of their lifespan. Furthermore, most of the approaches either operate at specific predefined temporal "snapshots" of the dataset and ignore the movement between these "snapshots" and/or ignore the temporal dimension and perform spatial-only clustering and/or assume that the length (number of samples) of the trajectories and the sampling rate is fixed, which is unrealistic. Another thing that should be taken into account when designing a prediction-oriented trajectory clustering algorithm, is that the resulting clusters should have a small extent in order for the predictions to be more accurate. Obviously, this, rules out a large number of approaches that perform density-based clustering which might lead to spatially extended clusters through expansion.

For the above reasons, the desired specifications that such a trajectory clustering algorithm should hold, in order to be able to predict the movement of future trajectories, are the following:

- Discovering of clusters of subtrajectories, instead of whole trajectories.
- Spatio-temporal clustering, instead of spatial only.
- Support of trajectories with variable sampling rate, length and with temporal displacement.
- Distance-based clustering.

There have been some approaches to deal with the problem of subtrajectory clustering in a centralized way [1,21,32], however, all the above subtrajectory clustering approaches are centralized and do not scale with the size of today's trajectory data, thus calling for parallel and distributed algorithms. For this reason, we utilize the work presented in [36], coined *DSC*, which introduces an efficient and highly scalable approach to deal with the problem of *Distributed Subtrajectory Clustering*, by means of MapReduce. More specifically, the authors of [36] split the original problem to three sub-problems, namely *Subtrajectory Join*, *Trajectory Segmentation* and *Clustering and Outlier Detection*, and deal with each one in a distributed fashion by utilizing the MapReduce programming model.

To elaborate more, the *Subtrajectory Join* step aims at retrieving for each trajectory $r \in D$, all the moving objects, with their respective portion of movement, that moved close enough in space and time with r, for at least some time duration. Subsequently, the *Trajectory Segmentation* step takes as input the result

of the *Subtrajectory Join* step, which is actually a trajectory and its neighboring trajectories and targets at segmenting each trajectory $r \in D$ into a set of subtrajectories in a neighbourhood-aware fashion, meaning that a trajectory will be segmented whenever its neighbourhood changes significantly. Finally, the third step takes as input the output of the first two steps and the goal is to create clusters of similar subtrajectories and at the same time identify subtrajectories that are significantly dissimilar from the others (outliers).

For more details about the algorithms involved in *DSC* and an extensive experimental study, please refer to [36].

4.2 Online Step: On Long-Term Future Location Prediction

In this section, we describe how the FLP module takes advantage of an individual's typical movement (medoids from now on), based on the observation that moving objects often follow the same route patterns. This observation fits exactly in the maritime and aviation domain where vessels or airplanes have very strict routes between ports and airports, either implied due to route optimization (e.g. ship's fuel consumption) or explicitly required as official regulation (flight plans). The Future Location Prediction (FLP) module aims to make an accurate estimation of the next movement of a moving object within a specific look-ahead time frame.

Most approaches do not take advantage of any other historic data available, either from the object itself or other "similar" objects moving within the same area and context, making it susceptible to errors associated to noise, artifacts or outliers in the input. This results in inaccurate predictions and only with a short horizon (seconds or few minutes). A very different approach for the FLP problem is making the associated predictive models less adaptive but more reliable, by introducing specific "memory" based on historic data of an entire fleet of objects relevant to the context at hand. On the other hand, this requires a combination of historical and streaming data which is not a trivial task. A big challenge of our proposed framework is how to handle thousands of records efficiently in the context of online streaming data, join each object with the appropriate medoids and finally do all the necessary model calculations to produce predictions for the future locations of an object. In practice, several such medoids are pre-computed and stored in an efficient way (partitioned by object identifier), so that they can be retrieved on demand or even kept in-memory for several thousands of objects, making long-term FLP feasible in a large scale. This task is addressed by employing a Big data engine that is designed to conduct fast joins between streaming data and historical data. Spark module (SQL or Streaming) can efficient join historical and streaming data. Either with map-side-join (a.k.a broadcast join) or using Dataset (Spark structure) metadata to achieve extra optimizations. For example if the medoids can be sent to all workers (broadcast) at the initial phase, it is recommended to replicate medoids (create a local variable) in each worker and for each object in Map-Reduce phase we select its medoids to perform prediction. On the other hand, if the medoids' size cannot stored in each workers' memory, we partition the medoids by objects' identifier in order to have quick

access for a specific object and create spark distributed structures that can be easily joined with Streaming data via Sparks SQL API.

Medoid matching: The first step tries to match the object's recent history with the medoids. More specifically, for all the medoids, we find the closest to the object's current trajectory. Algorithm 1 uses a spatiotemporal similarity function in order to find the best match. Prediction: The algorithm has already identified the last point from the best-matched, according to the previous stage. Then, it follows the medoid's points one by one until it reaches the prediction horizon.

The FLP-L approach described in brief above is inherently intuitive and self-explanatory. It relies on past routes of the same or similar objects in order to forecast how a specific object will move while it is already residing on a specific frequently-traversed route. The weighted similarity function between two spatiotemporal points $d(p, p') = \sqrt{w_1 \cdot (x - x')^2 + w_1 \cdot (y - y')^2 + w_2 \cdot (t - t')^2}$, was proposed in [29] and in our algorithm weights ratio is estimated by mean speed.

Algorithm 1 describes the prediction step in a more technical $\varsigma\alpha v$. Actually, these steps is the Spark's map function after collecting streaming data in a certain (user-defined time window).

Algorithm 1: FLP-L Algorithm

```
Input  : current state (object's recent history), object's network, horizon,
         distance_threshold
Output: prediction path
min_dist ← Double.MaxValue;
best_match ← null;
foreach trajectory ∈ medoids do
    traj_medoid_distance ← SpatioTemporal Distance(current_state, trajectory);
    if traj_medoid_distance <min_dist AND traj_medoid_distance
    <distance_threshold then
        best_match ← trajectory;
        min_dist ← traj_medoid_distance;
    end
end
if best_match is not null then
    while prediction_path.getLast.getTimestamp <horizon AND
    best_match.hasNext do
        prediction_path.add(best_match.next());
    end
end
return prediction_path;
```

The above algorithm could be implemented in Spark Map-Reduce API as follows:

1. Receiving and parsing messages from input Kafka topic (map)
2. Reduce by object identifier over a window period
3. Join objects streaming data with the proper medoids.
4. Map partition (process each object for the current window) in order to perform prediction.

Step 3 is required only for the Dataset Join, otherwise (broadcast join) step 3 is performed inside step 4. Figure 3 illustrates an example of the FLP-L approach over a flight between Madrid and Barcelona, where the red points are the actual data and the blue points are the predictions.

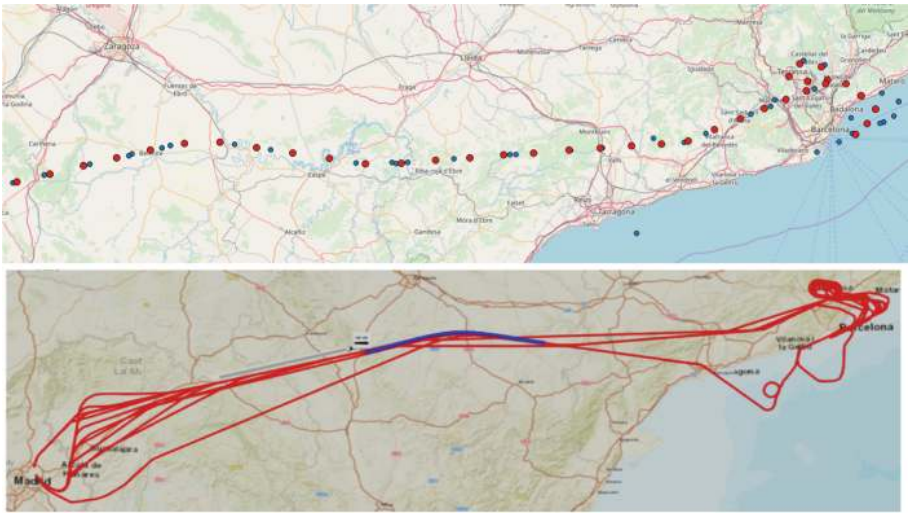

Fig. 3. Madrid - Barcelona flight example of the FLP-L approach. In the frst figure red points are real data and blue points are the predictions. In the second image red lines are medoids, gray line is the current window of a flight and the blue line is the predicted path (Color figure online)

5 Experimental Evaluation

5.1 Experimental Setup

In this section, we present the results of our experimental study. Our cluster consists of 10 nodes (1 master, 9 workers) with 5 executor cores per worker and 4 GB memory per worker. Input streams are provided by a Kafka topic and FLP-L is implemented on top of Spark SQL Streaming engine and Apache Yarn used as a resource manager. Spark SQL streaming tasks are processed using a micro-batch processing engine, which processes data streams as a series of small batch jobs thereby achieving low latency and exactly-once guarantees. Spark-Kafka integration is provided by Spark, but Spark tuning depends on

parallelism, namely data partitioning and park Streaming integration for Kafka in our architecture provides simple parallelism and 1:1 correspondence between Kafka partitions and Spark partitions. His means that if we want the higher performance, we have to configure Spark to create the same partitions as Kafka and Kafka to have as many partitions as possible. For example, if input Kafka topic has 60 partitions, then the cluster must have at least 60 cores for the query to make progress and achieve the best performance. In our experiments we used one Kafka topic for each domain (aviation, maritime) with 60 partitions.

We conducted experiments against real datasets (IFS messages and AIS messages [33]). Table 1 summarizes some basic statistics about the input dataset.

Table 1. Dataset description

	Aviation	Maritime
Number of point	455000	16000000
Number of objects	680 flights	5055 MMSI
Spatial coverage	Spain (Madrid - Barcelona flights)	Brest area
Time span	April 2016 (one week)	6 months

5.2 Results

Based on the optimal Spark/Kafka configuration described in Fig. 4, the total delay originates almost entirely from the processing time, which asymptotically stabilizes at around 5 s. This essentially translates to 60,000 Kafka messages (points) per 10 s or 6,000 points/second, which corresponds to 8-min look-ahead window. In other words, with an average sampling rate of 5 s for each moving object, this system configuration of the FLP module can accommodate up to 30,000 moving objects with 5-s update and 8-min look-ahead predictions. It is also important to notice that scheduling time in Fig. 4, which is related with Spark-Kafka integration. Scheduling time with three workers overcome processing time because there are not enough resources (cores) in the Spark cluster in order to process input messages and Kafka input partitions. On the other hand, with six workers and above scheduler has enough resources to assign the planned tasks. This behaviour occurs because there are enough resources (cores) for executing Spark Tasks. On the other hand, with three workers there are not enough resources for the input messages for scheduling and the algorithm breaks. As described above, in this option a FLP approach is employed for exploiting the cluster medoids as "guidelines" for providing online predictions, e.g. as the actual flight evolves in real time. The general clustering method in this case is the same as described in Sect. 4. We use up to 14 clusters in order to perform future location prediction. The FLP module, uses sliding windows of 2 min of past positions in order to optimally match the most recent segment of the current trajectory to one of the available medoids, using a custom spatio-temporal

similarity function. Then, the best-matched medoid is used as the maximum-likelihood trajectory evolution and the predicted positions are taken along its path for a specific (user-defined) look-ahead step.

Figure 5 illustrates the histogram of the horizontal error, i.e., the distribution of errors, for all the trajectories in the Aviation (Madrid/Barcelona) and Maritime (Brest Area) dataset and with spatial-only comparison (point-wise Euclidean distance). Specifically, they illustrate the boxplots of the per-complete-trajectory mean error for multiple look-ahead steps (1, 2, 4, 8, 16, 32 min). Additionally, the notation of the boxplot provides hints of the underlying error distributions, i.e., means, medians, upper/lower quartiles, non-outlier ranges, etc. These verify that the prediction errors are indeed in accordance with the expected shape of the distribution, i.e., a typical Extreme Value (EV) with medium/low skewness (Gaussian-like) towards the lower limit and an asymptotically decreasing right tail, i.e., accumulate and expand exponentially as the look-ahead span doubles.

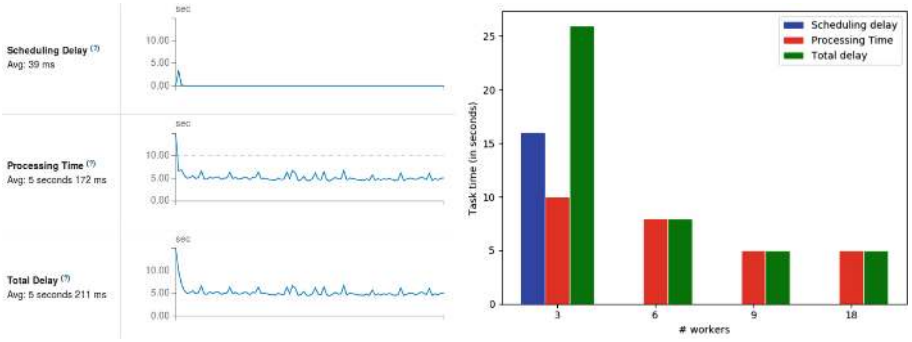

Fig. 4. Performance metrics for $16 \cdot 10^6$ points, $6 \cdot 10^3$ points/second, batch interval 10 s, 9 workers and 60 partitions.

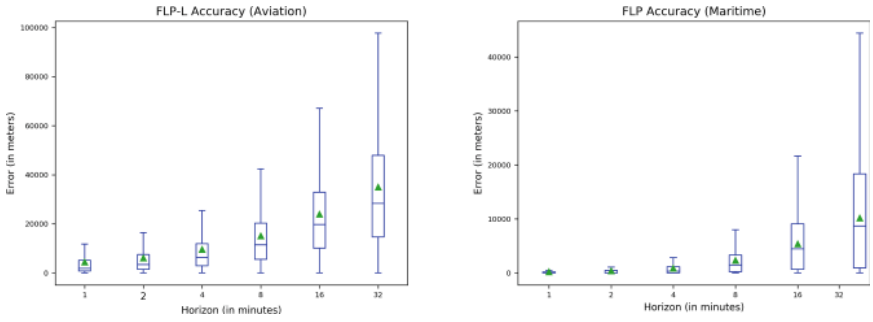

Fig. 5. Mean error for multiple look-ahead steps (1, 2, 4, 8, 16, 32 min), with custom spatio-temporal similarity function and with 90%-threshold outliers removed.

6 Conclusion

In this work, a novel approach was introduced for the long-term FLP problem (FLP-L). Our approach is based on purely data-driven extraction of mobility patterns, i.e. subtrajectory cluster medoids. This approach is generic enough to be applicable to various domain, such as in the aviation and maritime domain. It is important to emphasize that the proposed framework relies end-to-end in big data technologies The experimental results included here are focused primarily on the maritime domain, since the aviation is considered a more 'constrained' problem due to the fact that all flights are legally bounded to file and closely follow specific flight plans, i.e., the 'intended path' is much more specific and mandatory. Nevertheless, this framework is directly applicable and valid in the aviation domain too, especially since the medoids discovery is based upon some form of clustering to discover groups and common motion patterns, either with or without considering flight plans as input in the predictive models. The accuracy in both domains, as well as the performance results, prove that it is a very efficient and scalable Big data solution for real-world applications, easily adaptable to various other domains.

Acknowledgements. This work was partially supported by projects datACRON (grant agreement No 687591), Track&Know (grant agreement No 780754) and MASTER (Marie Sklowdoska-Curie agreement N. 777695), which have received funding from the EU Horizon 2020 R&I Programme. This research has also been co-financed by the European Regional Development Fund of the European Union and Greek national funds through the Operational Program Competitiveness, Entrepreneurship and Innovation, under the call RESEARCH-CREATE-INNOVATE (project code: T1EDK-3268).

References

1. Agarwal, P.K., Fox, K., Munagala, K., Nath, A., Pan, J., Taylor, E.: Subtrajectory clustering: models and algorithms. In: PODS, pp. 75–87 (2018)
2. Andrienko, G., Andrienko, N., Bak, P., Keim, D., Wrobel, S.: Visual Analytics of Movement. Springer, Heidelberg (2013). https://doi.org/10.1007/978-3-642-37583-5
3. Ankerst, M., Breunig, M.M., Kriegel, H., Sander, J.: OPTICS: ordering points to identify the clustering structure. In: SIGMOD, pp. 49–60 (1999)
4. Ayhan, S., Samet, H.: Aircraft trajectory prediction made easy with predictive analytics. In: Proceedings of the ACM SIGKDD 2016 (2016)
5. Ayhan, S., Samet, H.: Time series clustering of weather observations in predicting climb phase of aircraft trajectories. In: Proceedings of the IWCTS 2016 (2016)
6. Cheng, T., Cui, D., Cheng, P.: Data mining for air traffic flow forecasting: a hybrid model of neural network and statistical analysis. In: Proceedings of the ITSC 2003 (2003)
7. Ciccio, C.D., var der Aa, H., Cabanillas, C., et al.: Detecting flight trajectory anomalies and predicting diversions in freight transportation. Decis. Support. Syst. **88**, 1–17 (2016)

8. Coppenbarger, R.: En route climb trajectory prediction enhancement using airplane flight-planning information. American Institute of Aeronautics and Astronautics (AIAA-99-4147) (1999)
9. Deng, Z., Hu, Y., Zhu, M., Huang, X., Du, B.: A scalable and fast OPTICS for clustering trajectory big data. Clust. Comput. **18**(2), 549–562 (2015)
10. Enea, G., Poretta, M.: A comparison of 4D-trajectory operations envisioned for Nextgen and SESAR. In: Proceedings of the ICAS 2012 (2012)
11. Ester, M., Kriegel, H., Sander, J., Xu, X.: A density-based algorithm for discovering clusters in large spatial databases with noise. In: KDD, pp. 226–231 (1996)
12. Fablec, Y.L., Alliot, J.: Using neural networks to predict aircraft trajectories. In: Proceedings of the ICIS 1999 (1999)
13. Fan, Q., Zhang, D., Wu, H., Tan, K.: A general and parallel platform for mining co-movement patterns over large-scale trajectories. PVLDB **10**(4), 313–324 (2016)
14. Georgiou, H., et al.: Moving objects analytics: survey on future location & trajectory prediction methods (2018)
15. Georgiou, H., Pelekis, N., Sideridis, S., Scarlatti, D., Theodoridis, Y.: Semantic-aware aircraft trajectory prediction using flight plans. Int. J. Data Sci. Anal., 1–14 (2019). https://doi.org/10.1007/s41060-019-00182-4
16. Gong, C., McNally, D.: A methodology for automated trajectory prediction analysis (2004)
17. Hu, C., Kang, X., Luo, N., Zhao, Q.: Parallel clustering of big data of spatio-temporal trajectory. In: ICNC, pp. 769–774 (2015)
18. Jeung, H., Yiu, M.L., Zhou, X., Jensen, C.S., Shen, H.T.: Discovery of convoys in trajectory databases. PVLDB **1**(1), 1068–1080 (2008)
19. Kalnis, P., Mamoulis, N., Bakiras, S.: On discovering moving clusters in spatio-temporal data. In: Bauzer Medeiros, C., Egenhofer, M.J., Bertino, E. (eds.) SSTD 2005. LNCS, vol. 3633, pp. 364–381. Springer, Heidelberg (2005). https://doi.org/10.1007/11535331_21
20. Laube, P., Imfeld, S., Weibel, R.: Discovering relative motion patterns in groups of moving point objects. IJGIS **19**(6), 639–668 (2005)
21. Lee, J., Han, J., Whang, K.: Trajectory clustering: a partition-and-group framework. In: SIGMOD, pp. 593–604 (2007)
22. de Leege, A., Paassen, M.V., Mulder, M.: A machine learning approach to trajectory prediction. In: Proceedings of the AIAA GNC 2013 (2013)
23. Li, Y., Bailey, J., Kulik, L.: Efficient mining of platoon patterns in trajectory databases. Data Knowl. Eng. **100**, 167–187 (2015)
24. Li, Z., Ding, B., Han, J., Kays, R.: Swarm: mining relaxed temporal moving object clusters. PVLDB **3**(1), 723–734 (2010)
25. Marz, N., Warren, J.: Big Data: Principles and Best Practices of Scalable Real-Time Data Systems. Manning Publications Co., New York (2015)
26. Nanni, M., Pedreschi, D.: Time-focused clustering of trajectories of moving objects. J. Intell. Inf. Syst. **27**(3), 267–289 (2006)
27. Orakzai, F., Calders, T., Pedersen, T.B.: Distributed convoy pattern mining. In: IEEE MDM, pp. 122–131 (2016)
28. Orakzai, F., Calders, T., Pedersen, T.B.: k/2-hop: fast mining of convoy patterns with effective pruning. PVLDB **12**(9), 948–960 (2019)
29. Panagiotakis, C., Pelekis, N., Kopanakis, I.: Trajectory voting and classification based on spatiotemporal similarity in moving object databases. In: Adams, N.M., Robardet, C., Siebes, A., Boulicaut, J.-F. (eds.) IDA 2009. LNCS, vol. 5772, pp. 131–142. Springer, Heidelberg (2009). https://doi.org/10.1007/978-3-642-03915-7_12

30. Pelekis, N., Theodoridis, Y.: Mobility Data Management and Exploration. Springer, New York (2014). https://doi.org/10.1007/978-1-4939-0392-4
31. Pelekis, N., Tampakis, P., Vodas, M., Doulkeridis, C., Theodoridis, Y.: On temporal-constrained sub-trajectory cluster analysis. Data Min. Knowl. Discov. **31**(5), 1294–1330 (2017)
32. Pelekis, N., Tampakis, P., Vodas, M., Panagiotakis, C., Theodoridis, Y.: In-DBMS sampling-based sub-trajectory clustering. In: EDBT, pp. 632–643 (2017)
33. Ray, C., Dréo, R., Camossi, E., Jousselme, A.L.: Heterogeneous integrated dataset for maritime intelligence, surveillance, and reconnaissance (2018). https://doi.org/10.5281/zenodo.1167595
34. Seki, K., Jinno, R., Uehara, K.: Parallel distributed trajectory pattern mining using hierarchical grid with mapreduce. IJGHPC **5**(4), 79–96 (2013)
35. Tampakis, P., Pelekis, N., Andrienko, N.V., Andrienko, G.L., Fuchs, G., Theodoridis, Y.: Time-aware sub-trajectory clustering in Hermes@ PostgreSQL. In: ICDE, pp. 1581–1584 (2018)
36. Tampakis, P., Pelekis, N., Doulkeridis, C., Theodoridis, Y.: Scalable distributed subtrajectory clustering (2019). http://arxiv.org/abs/1906.06956
37. Tang, L.A., et al.: On discovery of traveling companions from streaming trajectories. In: ICDE, pp. 186–197 (2012)
38. Tao, Y., Faloutsos, C., Papadias, D., Liu, B.: Prediction and indexing of moving objects with unknown motion patterns. In: Proceedings of the ACM SIGMOD 2004 (2004)
39. Thipphavong, D., Schultz, C., et al.: Adaptive algorithm to improve trajectory prediction accuracy of climbing aircraft. J. Guid. Control. Dyn. (JGCD) **36**(1), 15–24 (2013)
40. Trasarti, R., Guidotti, R., Monreale, A., Giannotti, F.: MyWay: location prediction via mobility profiling. Inf. Syst. **64**, 350–367 (2017)
41. Vieira, M.R., Bakalov, P., Tsotras, V.J.: On-line discovery of flock patterns in spatio-temporal data. In: ACM SIGSPATIAL, pp. 286–295 (2009)
42. Zheng, K., Zheng, Y., Yuan, N.J., Shang, S.: On discovery of gathering patterns from trajectories. In: ICDE, pp. 242–253 (2013)
43. Zheng, Y.: Trajectory data mining: an overview. Trans. Intell. Syst. Technol. **6**(3), 1–41 (2015)
44. Zheng, Y.: Trajectory data mining: an overview. ACM TIST **6**(3), 29:1–29:41 (2015)