A Distributed Online Learning Approach for Pattern Prediction over Movement Event Streams with Apache Flink - DRAFT

Ehab Qadah
Fraunhofer IAIS
Sankt Augustin, Germany
Ehab.Qadah@iais.fraunhofer.de

Michael Mock
Fraunhofer IAIS
Sankt Augustin, Germany
Michael.Mock@iais.fraunhofer.de

Elias Alevizos
NCSR "Demokritos"
Athens, Greece
alevizos.elias@iit.demokritos.gr

Georg Fuchs
Fraunhofer IAIS
Sankt Augustin, Germany
Georg.Fuchs@iais.fraunhofer.de

ABSTRACT

We present a distributed online prediction system for user-defined patterns over multiple massive streams of events related to trajectories of movement objects, built using the general purpose stream processing framework Apache Flink. The proposed approach is based on combing probabilistic event pattern prediction models on multiple predictor nodes with a distributed online learning protocol in order to continuously learn a global prediction model to share it among the predictors in a communication-efficient way. Our approach enables the collaborative learning between the predictors (i.e., "learn from each otherâĂİ), thus the learning rate is accelerated. The underlying model provides an online predictions when a pattern (i.e., a regular expression over the event types) will be completed within each event stream. We describe the distributed architecture of the system, its implementation in Flink and present first empirical evaluation results over real-world event streams related to trajectories of moving vessels.

KEYWORDS

Big Movement Event Streams, Stream Processing,Event Pattern Prediction, Distributed Pattern Markov Chain

1 INTRODUCTION

In recent years, the technological advances have led to a growing in the availability of massive amounts of continuous streaming data (i.e., event streams) in many application domains such as social networks [16], Internet of Things (IoT) [17], and Maritime surveillance [19]. Ability to detect and predict the full matches of a pattern of interest (i.e., sequence of events within the event stream) defined by a domain expert, is important for several operational decision making tasks. An event stream is an unbounded collection of timely ordered data observations in the form of an attribute tuple that is composed of a value from finite event types along with other categorical and numerical features. In this work we deal with movement event streams, for instance, in the context of maritime surveillance, event patterns prediction over real-time tracking streams of moving vessels is useful to alert maritime operation mangers about suspicious activities (e.g., fast sailing vessels near ports or illegal fishing) before they happen, in this scenario, the event stream of a moving vessel consists of spatial-temporal and kinematic information along with the vessel's identification and its trajectory related events. However, processing real-time streaming data with low latency is challenging since data streams are large and distributed in nature and continuously keep on coming at a high rate.

In this paper, we present a design and implementation of an online, distributed and scalable pattern prediction system over multiple massive streams of events. More precisely we consider the event streams related to trajectories of moving objects (i.e., vessels and aircrafts). The proposed approach is based on a novel method that combines the distributed online prediction protocol [10, 12] with the event forecasting with Pattern Markov Chain system [6], implemented on top of the Big Data framework for stream processing Apache Flink [5]. We evaluate our proposed system over real-word data streams of moving objects, in particular, streams of events related to trajectories of aircrafts and moving vessels, which are provided in the context of the datAcron project[1].

2 RELATED WORK AND BACKGROUND

2.1 Pattern prediction over event streams

Event forecasting with pattern Markov Chains:

2.2 Distributed online learning

Distributed online prediction by mini-batch based approach has been proposed in [10]. Their approach is based on a static synchronization method, the learners periodically communicate their local models with a central coordinator unit after consuming a fixed number of input samples/events, in order to create a global model and share it between all learners. This work has been extended in [12] by introducing a dynamic synchronization scheme that reduces the required communication by monitoring the variance of the local models from a global reference point. In this

© 2018 Copyright held by the owner/author(s). Published in Proceedings of the 21st International Conference on Extending Database Technology (EDBT), March 26-29, 2018, ISBN 978-3-89318-078-3 on OpenProceedings.org.
Distribution of this paper is permitted under the terms of the Creative Commons license CC-by-nc-nd 4.0.

[1] http://www.datacron-project.eu/

paper, we address to use the communication-efficient distributed online learning framework for event patterns prediction, which is internally based on the Pattern Markov Chain (PMC) predictors.

Communication-efficient distributed online prediction by dynamic model synchronization: The pseudo-code in Algorithm **??** presents the algorithm of the distributed pattern prediction using the communication-efficient distributed online protocol on both the predictor and coordinator nodes. When a predictor $f \in [k]$ observes an event e_i before synchronization phase t, it revises its internal state . Note that m_t is the aggregated model received from the coordinator at synchronization phase t and m_{t+i} is the updated local model after observing the i^{th} event in the new batch.

Algorithm 1: communication-efficient distributed online prediction protocol

Predictor f: at observing event e_i
  update model $m_{t+i,f}$ and predict I
  if $i \mod b = 0 and \|m_i - r\|^2 > \triangle$ **then**
    send $m_{t+i,f}$ to Coordinator
Coordinator at synchronization:
  Receive local models $\{m_{t,f}\}_{f=1}^{k}$
  compute the global model m
  send m to all the predictors $[k]$ and set $m_{f,1} \ldots, m_{f,k} = m$

Moreover, the communication-efficient distributed online learning framework has been extended to handle kernelized online learning models [13].

2.3 Technological background

In the last years, many systems for large-scale and distributed stream processing have been proposed, including Spark Streaming [3], Apache Storm [4] and Apache Flink [5]. These framework allow to ingest real-time data streams from different sources such as Aws Kinesis [2] and Apache Kafka [1]. We implemented the proposed system over Apache Flink which provides the distributed stream processing components of the distributed predictors, alongside Apache kafka for streaming the input event streams, and as a messaging platform to enable the distributed online learning functionalities.

In the datAcron project, the Flink streaming engine has been chosen as a primary platform for supporting the streaming operations, based on an internal comparative evaluation of several streaming platforms. While The distributed online learning framework has already been implemented in the FERARI [11] based on Storm. In Storm, a distributed application is expressed as a so-called "topology", in which the individual processing steps called "Bolts" are connected in a data workflow. This means, that each Bolt can is sending and receiving data streams from other Bolts, for example, there are bolts generating the local models for each incoming data streams and there is a Bolt representing the "Coordinator" for executing the synchronization protocol between the local models. As the synchronization protocol includes the steps of sending the local models to the coordinator (for merging the models) and of sending the merged model back, it results in a cyclic workflow structure, which is supported in Storm.

Apache Flink: Apache Flink is an open source project that provides a large-scale, distributed and stateful stream processing platform [9]. Flink is one of the recent and common big data processing frameworks, it employees data-stream processing model for streaming and batch data, the batch processing is treated as a special case of streaming applications (i.e., finite stream). The Flink's software stack includes the *DataStream* and *DataSet* APIs for processing infinite and finite data, respectively. These two core APIs built on the top of the Flink's core distributed streaming dataflow engine. Additionally, Flink provides libraries such as Complex event processing for Flink (Flink-CEP), Machine Learning for Flink (FlinkML) and Flink Graph API (Gelly) [9].

The main data abstractions of Flink are *DataStream* and *DataSet* that represent read-only collection of data elements. The list of elements is bounded (i.e., finite) in *DataSet*, while it is unbounded (i.e., infinite) in the case of *DataStream*. The Flink's core is a distributed streaming dataflow engine, with each Flink program is represented by a data-flow graph (i.e., directed acyclic graph - DAG) that executed by the Flink's engine [9]. The data flow graphs are composed of stateful (state is maintained per partition), parallel operations and intermediate data stream partitions.

Apache Kafka: Apache Kafka is scalable, fault-tolerant and distributed streaming framework [1]. It allows to publish and subscribe to arbitrary data streams. Kafka manages the stream records in different categories (i.e., topics) that are partitioned and distributed over the Kafka servers. It provides the ability to publish a stream of records to one or more Kafka topic, to be consumed by applications that can subscribe to one or more topic to read data streams. The stream is distribute and balance between receivers within the same group for the sake of scalability.

3 SYSTEM OVERVIEW

3.1 Problem formulation

We follow the terminology of [7, 15, 20] to formalize the problem we tackle. Given a set of K real-time streams of events $S = \{s_1, s_2, ..., s_k\}$ as input, which are associated with a set of K objects $O = \{o_1, ..., o_k\}$. Where each stream $s_i = \langle e_1, e_3 ..., e_t, ... \rangle$ is a time-ordered sequence of events, these events are connected to a single reference object $o_i \in O$, e_t refers to the current time event within the unbounded stream, we give the definition of the input event sample as follows:

Definition 3.1. Each event is defined as a tuple of attributes $e_i = (type, \tau, a_1, a_2 ....., a_n, id)$: where $type$ is the event type attribute that takes a value from a set of finite event types/symbols Σ, τ represents the timestamp of the event tuple, the $a_1, a_2, ..., a_n$ are spatial or other contextual features (e.g., speed), these features are varying from one application to another, while the id attribute connects the event tuple to the associated domain object.

A user-defined pattern $\mathcal{P}$ is given in the form of regular expression over Σ [6], and the main goal is to provide predictions

about the full matches of $\mathcal{P}$ within each event stream $s_i \in S$ in real-time.

The setting that is considered in this work is described in the following:
A large-scale patterns prediction over multiple input event streams system that consists of $K = |S| = |O|$ distributed predictor nodes $n_1, n_2 \ldots, n_k$, each of which consumes an input event stream $s_i \in S$ and provide an online predication service. Each node $i \in [K]$ consumes a single event stream s_i associated with a single object $o_i \in O$, in addition, it maintains a local prediction model f_i for the user-defined pattern $\mathcal{P}$. The online predictions about the full match of the pattern $\mathcal{P}$ in s_i is provided for each new arriving event tuple, based on the current received event $e_t \in s_i$ and the observed previous events sequence $\{e_j \in s_i \mid j < t\}$. In summary, we have multiple running instances of an online prediction algorithm on distributed nodes for multiple input event streams, each instance provides online predications about a defined pattern of events. We consider massive event streams that describe trajectories of moving objects, more specifically, event streams of moving vessels in the context of maritime surveillance.

The defined pattern $\mathcal{P}$ is monitored over each event stream s_i by a predictor nodes n_i that maintains a local prediction model f_i, where there is one node for each vessel's event stream. The prediction model f_i gives the ability to provide an online predictions about when the pattern will be completed in the form of an expected number of future events before a full match does occur.

3.2 The Proposed Approach

We design and develop a scalable and distributed patterns prediction system over massive input event streams (e.g., event streams of moving objects). We exploit the event forecasting with Pattern Markov Chains [6] as the base prediction model (i.e., f_i). We propose to enable the dynamic merge of the prediction models of the input event streams, by adapting the distributed online predication protocol [12] to synchronize the distributed models, i.e., Markov transitions probabilities of the Pattern Markov Chain (PMC) predictors.

We propose a *synchronizationoperation* for the distributed Pattern Markov Chain (PMC) models based on the maximum-likelihood estimation [8] for the transition probabilities matrix of the underlaying Markov Chain described by:

$$\hat{p}_{i,j} = \frac{\sum_{k \in K} n_{k,i,j}}{\sum_{k \in K} \sum_{l \in L} n_{k,i,l}} \quad (1)$$

Our approach relies on enabling the collaborative learning between the prediction models of the input event streams. By doing so, we assume that the underlying event streams belong to the same distribution and share the same behavior (e.g., mobility patterns). We claim that assumption is reasonable in many application domains, for instance, in the context of maritime surveillance, vessels travel through defined routes by International Maritime Organization (IMO). Additionally, vessels have similar mobility patterns in specific areas such as moving with low speed and multiple turns near the ports [14, 18]. That allows our system to dynamically construct a coherent global prediction model for all input event streams based on merging its local models.

Our proposed approach is imposing a speed up in the learning of the prediction models with less training data, in addition, we expect to gain an improvement of the predictive performance compared to the no-distributed version of event forecasting with Pattern Markov Chains system.

3.3 Distributed architecture

In Figure 1 the architecture and components of our system are given.

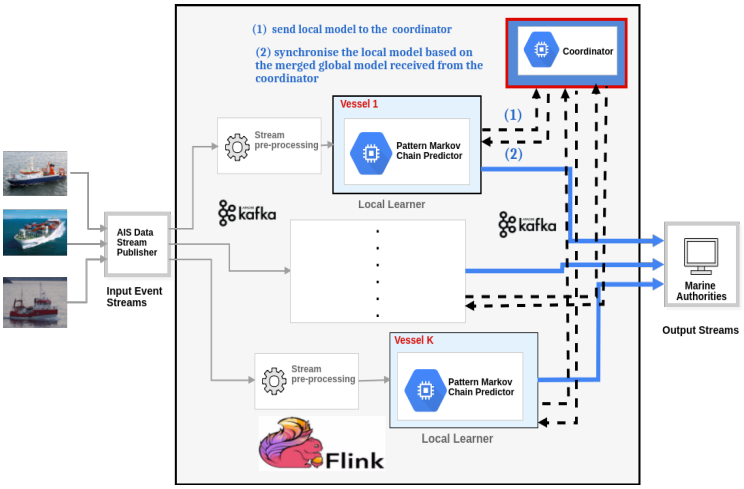

Figure 1: System Architecture.

4 IMPLEMENTATION DETAILS

This sections describes the system's implementation over Apache Flink and Apache Kafka.

5 EMPIRICAL EVALUATION

In this section, we present experimental results on real-world event streams in the context of maritime and aviation domains provided by the datAcron project.

6 CONCLUSION

ACKNOWLEDGMENTS

This work was supported by the EU H2020 datAcron project (grant agreement No 6875).

REFERENCES

[1] Apache Kafka. https://kafka.apache.org/. (????).
[2] Apache Kinesis. https://aws.amazon.com/de/kinesis/. (????).
[3] Apache Spark Streaming. http://spark.apache.org/streaming/. (????).
[4] Apache Storm. http://storm.apache.org/. (????).
[5] 2015. Apache Flink. https://flink.apache.org/. (2015).
[6] Elias Alevizos, Alexander Artikis, and George Paliouras. 2017. Event Forecasting with Pattern Markov Chains. In *Proceedings of the 11th ACM International Conference on Distributed and Event-based Systems*. ACM, 146–157.
[7] Elias Alevizos, Anastasios Skarlatidis, Alexander Artikis, and Georgios Paliouras. 2015. Complex event recognition under uncertainty: A short survey. *Event Processing, Forecasting and Decision-Making in the Big Data Era (EPForDM)* (2015), 97–103.

[8] Theodore W Anderson and Leo A Goodman. 1957. Statistical inference about Markov chains. *The Annals of Mathematical Statistics* (1957), 89–110.

[9] Paris Carbone, Asterios Katsifodimos, Stephan Ewen, Volker Markl, Seif Haridi, and Kostas Tzoumas. 2015. Apache flink: Stream and batch processing in a single engine. *Bulletin of the IEEE Computer Society Technical Committee on Data Engineering* 36, 4 (2015).

[10] Ofer Dekel, Ran Gilad-Bachrach, Ohad Shamir, and Lin Xiao. 2012. Optimal distributed online prediction using mini-batches. *Journal of Machine Learning Research* 13, Jan (2012), 165–202.

[11] Ioannis Flouris, Vasiliki Manikaki, Nikos Giatrakos, Antonios Deligiannakis, Minos Garofalakis, Michael Mock, Sebastian Bothe, Inna Skarbovsky, Fabiana Fournier, Marko Stajcer, and others. 2016. FERARI: A Prototype for Complex Event Processing over Streaming Multi-cloud Platforms. In *Proceedings of the 2016 International Conference on Management of Data*. ACM, 2093–2096.

[12] Michael Kamp, Mario Boley, Daniel Keren, Assaf Schuster, and Izchak Sharfman. 2014. Communication-efficient distributed online prediction by dynamic model synchronization. In *Joint European Conference on Machine Learning and Knowledge Discovery in Databases*. Springer, 623–639.

[13] Michael Kamp, Sebastian Bothe, Mario Boley, and Michael Mock. 2016. Communication-Efficient Distributed Online Learning with Kernels. In *Joint European Conference on Machine Learning and Knowledge Discovery in Databases*. Springer, 805–819.

[14] Bo Liu, Erico N de Souza, Stan Matwin, and Marcin Sydow. 2014. Knowledge-based clustering of ship trajectories using density-based approach. In *Big Data (Big Data), 2014 IEEE International Conference on*. IEEE, 603–608.

[15] David Luckham. 2008. The power of events: An introduction to complex event processing in distributed enterprise systems. In *International Workshop on Rules and Rule Markup Languages for the Semantic Web*. Springer, 3–3.

[16] Michael Mathioudakis and Nick Koudas. 2010. Twittermonitor: trend detection over the twitter stream. In *Proceedings of the 2010 ACM SIGMOD International Conference on Management of data*. ACM, 1155–1158.

[17] Daniele Miorandi, Sabrina Sicari, Francesco De Pellegrini, and Imrich Chlamtac. 2012. Internet of things: Vision, applications and research challenges. *Ad Hoc Networks* 10, 7 (2012), 1497–1516.

[18] Giuliana Pallotta, Michele Vespe, and Karna Bryan. 2013. Vessel pattern knowledge discovery from AIS data: A framework for anomaly detection and route prediction. *Entropy* 15, 6 (2013), 2218–2245.

[19] Kostas Patroumpas, Alexander Artikis, Nikos Katzouris, Marios Vodas, Yannis Theodoridis, and Nikos Pelekis. 2015. Event Recognition for Maritime Surveillance.. In *EDBT*. 629–640.

[20] Cheng Zhou, Boris Cule, and Bart Goethals. 2015. A pattern based predictor for event streams. *Expert Systems with Applications* 42, 23 (2015), 9294–9306.