IEEE Access
Multidisciplinary | Rapid Review | Open Access Journal

Received December 26, 2017, accepted February 5, 2018, date of publication February 9, 2018, date of current version March 15, 2018.

Digital Object Identifier 10.1109/ACCESS.2018.2804623

Real-Time Probabilistic Data Fusion for Large-Scale IoT Applications

ADNAN AKBAR[1], GEORGE KOUSIOURIS[2], HARIS PERVAIZ[1], JUAN SANCHO[3], PAULA TA-SHMA[4], FRANCOIS CARREZ[1], AND KLAUS MOESSNER[1]

[1]Institute for Communication Systems, University of Surrey, Guildford GU2 7XH, U.K.
[2]Institute of Communication and Computer Systems, National Technical University of Athens, 15779 Athens, Greece
[3]ATOS Research and Innovation Labs, 28037 Madrid, Spain
[4]IBM Research, Haifa 3498825, Israel

Corresponding author: Adnan Akbar (adnan.akbar@surrey.ac.uk)

This work was supported by the European Union's Horizon 2020 Projects FP7 COSMOS 609043 and CPaaS.io under Grant 723076.

ABSTRACT Internet of Things (IoT) data analytics is underpinning numerous applications, however, the task is still challenging predominantly due to heterogeneous IoT data streams, unreliable networks, and ever increasing size of the data. In this context, we propose a two-layer architecture for analyzing IoT data. The first layer provides a generic interface using a service oriented gateway to ingest data from multiple interfaces and IoT systems, store it in a scalable manner and analyze it in real-time to extract high-level events; whereas second layer is responsible for probabilistic fusion of these high-level events. In the second layer, we extend state-of-the-art event processing using Bayesian networks in order to take uncertainty into account while detecting complex events. We implement our proposed solution using open source components optimized for large-scale applications. We demonstrate our solution on real-world use-case in the domain of intelligent transportation system where we analyzed traffic, weather, and social media data streams from Madrid city in order to predict probability of congestion in real-time. The performance of the system is evaluated qualitatively using a web-interface where traffic administrators can provide the feedback about the quality of predictions and quantitatively using F-measure with an accuracy of over 80%.

INDEX TERMS Complex event processing, data analysis, internet of things, real-time systems, intelligent transportation systems.

I. INTRODUCTION

Many Internet of Things (IoT) applications enable smart city initiatives all over the world by leveraging ubiquitous connectivity, big data and analytics [1]. The capabilities introduced by these new applications are tremendous such as the ability to remotely monitor, manage and control devices without human intervention, create new insights and actionable information from massive streams of data, which in return is improving standard of human living to a great extent. IoT offerings are transforming cities by improving public health management, enhancing public transportation and creating more efficient and cost effective municipal services [2].

Forward-thinking cities and smart city solution providers recognize that the full potential of IoT cannot be reached by providing disparate smart city point solutions but rather the focus should be an efficient and scalable IoT infrastructure that integrates multiple systems or data streams. Currently, most of the IoT data is under-used by limiting it for specific applications. For example, weather data sensors have been deployed for years but are seldom used for other applications. In the new world of connectivity, weather data has huge potential for a range of applications. It can be correlated with shopping centers sales to determine the effect of weather on shopping patterns or can be combined with traffic conditions to understand how weather can effect traffic. According to McKinsey, 40 percent of the total value that IoT can provide requires different IoT systems to work together [3].

There are many research challenges in this regard from data ingestion and storage at one end to scalable and efficient data analytics at the other end. Different IoT systems were built on different protocols and building a single and global ecosystem for IoT that can work together is certainly a challenging and non-trivial task. In this context, any proposed IoT data analytics solution should fulfil the following requirements;

1) A proposed solution should be generic enough to ingest data coming from different IoT systems in

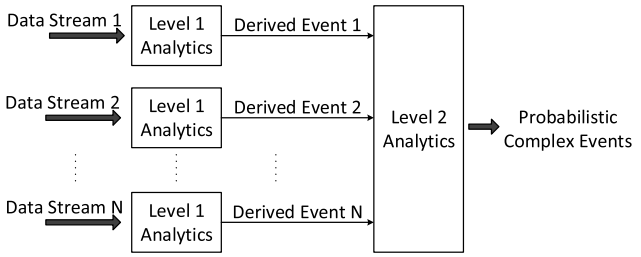

FIGURE 1. Proposed approach for hierarchical processing of data streams.

all types of format. It should have the capability to handle heterogeneous data streams with minimal modification of existing components.

2) An efficient and scalable solution is provided for storing, managing and performing complex analytics on this ever increasing data.
3) It should have the capability to perform real-time analytics for time sensitive applications.
4) Finally, it should be able to take inherent uncertainty of real-world events into account while infusing data from heterogeneous sources and able to make decisions with missing or uncertain data as well.

In this regard, we propose a two layer analytics solution as shown in Figure 1. The first layer of analytics is responsible for ingesting and analysing individual IoT data streams generated by different IoT systems in order to extract high-level knowledge which we called derived events in our system. Whereas the second layer of analytics provides the framework for probabilistic fusion of derived events by extending complex event processing (CEP) with Bayesian networks (BNs) for large-scale IoT applications. The hybrid framework based on CEP and BNs enables to combine derived events probabilistically and extract high-level knowledge in the form of probabilistic complex event.

There are several terms used in this paper which have different meanings in different contexts. In order to have a common understanding, these terms are defined here in the context it has been used in this paper.

Derived Event: Derived events can be defined as high-level events which are extracted from raw data streams; for example a hot weather derived from temperature data or bad traffic derived from traffic data are the examples of derived events.

Probabilistic Complex Events: Probabilistic complex event in our system is referred as the final output which is extracted from the combination of different derived events using Bayesian networks and CEP.

Heterogeneous data: Heterogeneous data is defined as the combination of data produced by different type of sensors or devices. A dataset consisting of traffic, weather and social media data is an example of heterogeneous data.

Large-scale: Large-scale data or Big data is referred in this paper as the data sets which are voluminous and complex enough that it cannot be analysed using conventional methods.

The key contributions of this paper are summarized below:

- A generic architecture is proposed to ingest data from heterogeneous IoT data streams,[1] store it efficiently and provide scalable methods to extract high-level events in near real-time.
- We propose a novel method for fusion of high-level events in real-time. Our proposed method extends the current CEP technology with BNs using a scalable approach based on open-source tools.
- The proposed solution is demonstrated using a real-world use-case of intelligent transportation system (ITS) where heterogeneous data streams including traffic, weather and social media data were collected. We developed a complete end-to-end solution for managing traffic efficiently in real-time and evaluated it qualitatively using web-interface and quantitatively using F-measure with an accuracy of over 80%.

The remainder of the paper is organized as follows. Section II highlights the leading research work in the domain of big data analytics and event processing. Section III describes the use-case of ITS which we used to demonstrate the application of our proposed solution. Section IV explains the first layer of analytics of our proposed architecture followed by the description of second layer analytics in Section V. Section VI demonstrates the implementation of our proposed solution for ITS use case alongside its evaluation. Finally we draw conclusion and highlight our future work in Section VII.

II. RELATED WORK

IoT data analytcis has long been divided into two categories of Batch and Event processing [4]. Batch processing is an efficient way of processing high volumes of data where data is collected over a period of time; whereas, event processing involves analyzing data on the fly without storing it and making decisions on per single entry of system. MapReduce [5] is perhaps one of the most renowned batch processing technique which provides the basis of other open source solutions such as Hadoop [6]. MapReduce is a programming model for carrying computations on large amounts of data in a distributed manner. It was originally developed by Google as a generic but proprietary framework for carrying analytics on Google's own big data. Although MapReduce provides a generic and scalable solution for big data but it is not designed for running iterative machine learning algorithms. A new mapper and reducer is initialized for every MapReduce iteration. Mappers read the same data from the disk again and again; and after processing, the results are stored in the disk in order to carry the next iteration. In such scenarios, disk access is a major

[1] In this work, the term "heterogeneous IoT data streams" is used to refer the data streams coming from multiple and diverse sources such as traffic sensors data from Madrid city council, social media data in the form of twitter and weather data.

bottleneck for iterative algorithms and hence degrading the overall performance [7].

A new cluster computing framework called Spark [8] was developed to overcome the limitations of MapReduce. Spark provides the ability to run computations in the memory which enables it to provide much faster computation times for complex and iterative applications as compared to systems based on traditional MapReduce. Spark is designed to be fast and general purpose for different data analysis applications including iterative algorithms such as machine learning. Recently, a lot of research effort has been put into spark streaming for real-time applications but it is still in its early stages with drawbacks such as the inability to process real-time data on per event basis [9].

In order to address the requirements of real-time data processing for IoT, solutions based on the concept of event driven architecture (EDA) [10] were proposed in recent times which has led to the development of new research area called complex event processing (CEP). CEP includes processing, analyzing and correlating event streams from different data sources using distributed message-based systems to extract high-level or actionable knowledge in near real-time [10]. The research on CEP has been multi-disciplinary including active databases [11], business process management [12] and service-based systems [13]. Common goal for all these multi-disciplinary research was to provide low delay processing of incoming primitive events in order to analyze streaming data on the run. It has often lead to the design of simplified solutions with no inherent support for handling uncertainty as shown by Cugola and Margara [14].

The dichotomy of batch processing and event processing has resulted into multiple systems analyzing the same data for various applications. In order to address this issue, Nathan Marz proposed the Lambda architecture [15] which provides a scalable and fault tolerant architecture for processing both real-time and historical data in an integrated fashion. In Lambda architecture, the output of batch and speed layer are calculated separately and then the results are served through service layer. Batch layer provides more in-depth analysis but with time delay whereas speed layer provides quick results by using only recent data and compromising on the level of analysis. Both layers work independently of each other. In contrast to it, the Hut architecture was proposed in [16] for IoT data analytics based on the combination of batch processing and event processing where both layers work together to provide the best of both worlds. In addition, the modular approach of their architecture enables both batch processing and event processing to work independently as well.

In addition to providing scalable solution for managing and analyzing real-time IoT data streams, handling inherent uncertainty in real-world IoT data streams represents another challenge. The scope of the IoT is global where the aim is to combine data generated by different devices in order to extract higher-level events. In this regard, it is important to provide solutions which can incorporate the uncertainty when fusing events from multiple sources. One of the early work for introducing uncertainty in probabilistic processing of different events was presented in [17] where Khoussainova *et al.* build their system probabilistic event extractor (PEEX) on the top of traditional relational dataBase management system (RDBMS) in order to detect probabilistic events from RFID data. They validated the system on a real-world deployment of radio-frequency identification (RFID) tags for recognizing activities. From an implementation point of view, Peex uses a RDBMS for storing all information received from data sources and confidence score about them. Rules are then translated into SQL queries before running periodically. It introduces a small delay between the real occurrence time of events and the detection time of events. Another initial approach for handling uncertainty in events processing was proposed by Wasserkrug *et al.* [18], where authors discussed two types of inherent uncertainty in CEP systems; uncertainty in data and uncertainty in relation between events. They presented theoretical framework for embedding uncertainty in events and later same authors extended their work in [19] where they proposed a mechanism for constructing probability space that captures the semantics of defined rules and used an abstraction based on Bayesian networks (BNs) to define the probabilities of possible combinations. Bayesian model was automatically created based on defined rules and explicit events. Experiments were performed on simulated data with the assumption of conditional probabilities given for BN. The authors highlighted the difficulties in obtaining conditional probabilities for real-world scenarios which limits the practical implementation of their work. Existing literature on using Bayesian network with CEP assumed that conditional probabilities are given which are vital for constructing BN. According to the best of our knowledge, there is no work in the literature focusing on the computation of conditional probabilities for real-world problems in order to construct BN with CEP. As the complexity of Bayesian inference process increases with an increase in the data size and number of data sources [20], it is essential to propose a scalable and efficient solution.

III. ILLUSTRATIVE SCENARIO

This work demonstrates the functionalities of proposed framework with the help of Intelligent transportation system (ITS) use-case. The use-case of ITS is chosen for several reasons. First, it represents a truly big data problem where different sensors at city-scale are generating large amounts of data. Second, it requires analysing this large data in real-time so that traffic administrators can manage traffic proactively. Third, ITS has an immense impact on the social and economical development of smart cities.

The advent of IoT has made many data sources available but currently, most of them are deployed as a stand-alone systems. Such individual systems are limiting the true potential of IoT. If we take the example of ITS, the use of traffic sensors is the most conventional method for traffic administrators to observe and manage traffic. Although, many modern day

TABLE 1. Input data streams and derived events.

No.	Input Data Stream	Derived Event
1	Traffic data	Traffic Congestion (C)
2	Twitter data	Large Crowd Concentration (LCC)
3	Weather data	Weather conditions (W)

cities have weather sensors installed and different research efforts as mentioned in [21] and [22] highlights the correlation of weather on the traffic patterns but it is not being yet exploited at city-level applications. Similarly, in the current era of digital technology social media is one of the quickest way to detect city events effecting traffic patterns. For example, a musical concert or a football match happening in nearby region can lead to unexpected increase in traffic and data analytics carried out on social media data streams can help us to detect it in a timely manner. Although different research efforts demonstrate the potential of using social media data such as twitter to detect city events in real-time as outlined in [23], they are seldomly preferred against the more conventional sensor based methods. In our proposed approach, we emphasis that the social media data and weather data can be exploited in order to predict traffic patterns more accurately and take pro-active measures. We propose to ingest and analyze different data streams in real-time and proposed a probabilistic event processing system for combining multiple high-level events extracted from these data streams.

For ITS use case, we ingest traffic data captured by various traffic sensors deployed by Madrid city council,[2] social media data in the form of twitter[3] and weather data[4] to derive high-level events as summarized in Table 1. In order to get fair analysis of results, three different locations were selected from city of Madrid as shown in the Figure 2. More details about the methodology and individual data streams can be found in next section.

IV. DATA COLLECTION AND ANALYTICS

The diverse requirements of IoT data processing need event processing (for real-time data) and batch processing (for historical data) to work in parallel. The combination of two technologies poses many challenges. In our earlier work [24], we highlight these challenges and presented an approach to address these challenges. In this paper, we improve our initial approach to make it generic for heterogeneous data streams, extend our experimental evaluation and implement it using open source components which are optimized for large-scale IoT data streams

Our proposed solution for data collection and analytics is shown in Figure 3. Node-RED [25] serves as the front end of our solution which is an open source visual tool for wiring IoT events coming from heterogeneous data sources. It provides a service oriented gateway which enables to ingest data

[2]http://informo.munimadrid.es/informo/tmadrid/pm.xml
[3]https://developer.twitter.com/
[4]http://api.wunderground.com/

FIGURE 2. Monitoring locations for analytics with bounding boxes in madrid city.

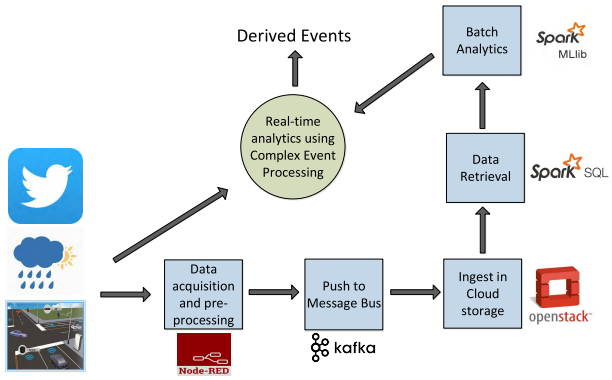

FIGURE 3. Level 1: Data collection and analytics.

coming from different data sources, performs pre-processing and push the data on a Kafka [26] topic in a defined format and schema. It acts as a generic interface for ingesting data from different IoT systems. In our system, we have used Apache Kafka as the message broker for real-time generated events because of its high throughput messaging capability and durability. One of the unique feature of Kafka which makes provides it an edge on other message brokers is its persistent nature to hold the messages for a given amount of time in the form of a log (ordered set of messages).

All the data is stored in the form of objects in OpenStack Swift cloud storage [27]. The OpenStack Object Store project, known as Swift, offers cloud storage software so that data can be stored and retrieved efficiently with a simple API. It is a scalable storage which provides durability, concurrency and availability through the entire data set. It enables to CReate, Update and Delete (CRUD) objects using a simple REST

API, and supports scalable and low-cost deployment using clusters of commodity machines. The open-source status of OpenStack was another important factor for choosing it as object storage framework.

We configured Spark SQL to retrieve data efficiently from OpenStack swift and analyze using Spark MLlib to extract insight from it. Spark MLlib is a machine learning library for Apache Spark. Apache Spark is a general purpose analytics engine that can process large amounts of data from various data sources. It performs especially well for multi-pass applications which include many machine learning algorithms. Spark SQL is Apache Spark's module for working with structured data. Real-time analytics are carried out using CEP in our architecture. CEP systems require rules based on different threshold values which have to be set by domain experts. In our proposed solution, historical data analysis using Apache Spark calculates these value automatically and keep updating it according to current context. More details about it can be found in [24]. Now we briefly describe different data streams which we ingested and analyzed using our architecture.

A. TRAFFIC DATA

We briefly describe about using our architecture for ingesting Madrid traffic data in this section. City of Madrid Council have deployed over 3000 traffic sensors in fixed locations in the city monitoring various traffic parameters including average traffic speed, density and occupancy. Traffic administrators utilize the data from these sensors to understand the traffic conditions and take reactive measures in order to ensure smooth traffic flow. Aggregated data is published as an IoT service using a RESTful API[5] and data is updated every 5 minutes. We ingest this data in our system using Node-Red flows, add meta data such as time and location, push data to kafka and store in the form of objects in openstack swift cloud storage. The published data on kafka has the following schema, where *intensidad* denotes traffic intensity, *velocidad* denotes traffic speed, *ts* denotes the timestamp in epoch format and *tf* denotes the time of day.

```
{"namespace": "traffic_data",
  "type": "record",
  "name": "MadridTrafficFlow",
  "fields": [
    {"name": "codigo", "type": "string"},
    {"name": "ocupacion", "type": "int"},
    {"name": "carga", "type": "int"},
    {"name": "nivelServicio", "type": "int"},
    {"name": "velocidad", "type": ["null","int"]},
    {"name": "intensidad", "type": ["null","int"]},
    {"name": "error", "type": "string"},
    {"name": "subarea",  "type": ["null","int"]},
    {"name": "ts", "type": "long"},
    {"name": "tf", "type": "string"}
  ]
}
```

In order to detect complex events using CEP, we require threshold values for different traffic features. These thresh-

[5]http://informo.munimadrid.es/informo/tmadrid/pm.xml

```
+-----------+---------+--------+----------+
| time_stamp|intensity|velocity|congestion|
+-----------+---------+--------+----------+
| 1.458234E9|     4212|      26|         1|
|1.4576274E9|     3784|      31|         1|
|1.4584788E9|     3162|      91|         0|
| 1.458315E9|     6118|      85|         0|
|1.4581926E9|     5952|      89|         0|
| 1.457793E9|     3880|      91|         0|
|1.4581206E9|     4766|      87|         0|
|1.4579568E9|     7100|      81|         0|
|1.4582844E9|     4940|      87|         0|
|1.4584896E9|     4210|      89|         0|
+-----------+---------+--------+----------+
only showing top 10 rows
```

FIGURE 4. Traffic events table layout in cloud storage.

old values are calculated automatically by carrying analytics on historical data. The data is collected for more than two months and is analyzed using Spark SQL and Spark MLlib to learn traffic patterns and model the expected traffic behavior for different contexts such as morning, evening, weekdays and weekends. Then this information is used to automatically generate threshold values for CEP rules. As the traffic becomes bad or approaches to congestion, CEP generates complex event with a warning message. An example rule for CEP is shown below where threshold values are generated automatically by exploiting pattern learning algorithms [24] using Spark MLlib. Complex events detected are stored in the cloud storage with current intensity and velocity readings as shown in the Figure 4 where '1' indicates a congestion event.

```
InitiatedAt(TrafficCongestion(locID), t)
HappensAt(aggr(locID, avg_int, avg_spd), t)
0 < avg_int < threshold(intensity) and
0 < avg_spd < threshold(speed)
```

B. TWITTER DATA

Social networks have been identified as a rich source of information due to their extended uptake, through which significant inference may be achieved with relation to circumstances affecting the societal status. In this context, we have used Twitter data as another source of information that can aid in the prediction of congestion. Exploring social media analytics is beyond the scope of current work, which is primarily the construction of an integrated system with heterogeneous data streams and probabilistic inference framework. In contrast to existing sophisticated methods for analyzing social media data, we have followed a rather simple yet effective approach. The intuition behind our approach is that the number of tweets coming from a specific region is the indicative of number of people gathered in a region. For example, if we are extracting tweets from a region including a football stadium, the number of tweets will be far higher on a match day as compared to other days indicating a large crowd concentration. Similarly, there will be more tweets coming from a region including city center or shopping malls during Christmas holidays as compared to a normal weekday indicating a crowd concentration.

One reason for selecting twitter data is its easily available open API [28] through which it can be ingested into our system and dynamically alert about surges in population concentration in a given area, when compared to the normal tweeting activity of the past. Increased Twitter activity in an area can be considered as an indirect indication about abnormal activity, especially when this relates to highways and not residential areas that could include an increased number of false positives (e.g. due to popular Twitter trends in the specific timeslot).

In order to focus on a specific area (a bounding box around the location), filtering of the acquired messages from Twitter needs to be performed. This is done upon registration, in which we define the geographical bounding box from where we need the relevant tweets to be forwarded. This is taken under consideration by the respective Twitter API and only the tweets that have enabled geolocation and fall within this box are forwarded (in our case the bounding boxes for every location is highlighted in Figure 2). It is necessary to stress that not all of the tweets are forwarded, but only a percentage of them, according to the Twitter API documentation [28]. Following tweets acquisition, ingestion takes place as follows:

1) Initial filtering is performed in order to reduce each tweet size, by discarding fields of no interest and thus reducing needed storage space
2) Enrichment with sentiment analysis information based on the tweet content is performed. This information is not currently used but it was considered a promising feature for future work.
3) Definition of an Apache AVRO [29] schema necessary for describing how the data fields information will be stored in the Cloud storage and which fields will be maintained, which also affects the Node-RED manipulation
4) Adaptation of the incoming tweets to the Avro format and JSON structure of the output to the Message Bus (Apache Kafka), from which it is collected by Openstack Swift

The published data on Kafka has the following schema;

```
{"namespace": "cosmos",
  "type": "record",
  "name": "TwitterData",
  "fields": [
    {"name": "ts", "type": "long"},
    {"name": "text", "type": "string"},
    {"name": "lon", "type": "double"},
    {"name": "lat", "type": "double"},
    {"name": "twitter_id", "type": "string"},
    {"name": "sentiment", "type": "double"},
  ]
}
```

Once the ingestion flow has been established and sufficient data have been collected, they can be included in the run-time operation of the LCC event identification. Historical data is analyzed using Spark SQL and Spark MLlib to learn about expected number of tweets for given location and time and

```
+----------+--------------+---------+
| time_stamp|twitter_counts|LCC_level|
+----------+--------------+---------+
| 1.459134E9|             9|        0|
|1.4592978E9|            27|        0|
| 1.462329E9|            11|        0|
|1.4589702E9|            12|        0|
|1.4621652E9|            17|        0|
|1.4626566E9|            87|        0|
|1.4588064E9|            25|        0|
|1.4624928E9|            37|        0|
|1.4586426E9|            60|        0|
|1.4620014E9|            43|        0|
+----------+--------------+---------+
only showing top 10 rows
```

FIGURE 5. LCC events table layout in object storage.

CEP is used to calculate the running sum of number of tweets. Depending on the number of tweets, it generates different levels of Large Crowd Concentration (LCC) event. After detecting LCC event, it is stored in the cloud storage along with twitter counts and time stamp. We have used three levels of LCC where '0' indicates no LCC or normal conditions, '1' indicates first level of LCC and '2' indicates second level of LCC event indicating a Large crowd in the region as shown in Figure 5. We validated our approach by performing several experiments around Santiago Bernabeu stadium in Madrid in order to detect the LCC for football matches and validate our proposed solution. Further details about the validation can be found in [30].

C. WEATHER DATA

There are number of open source weather services available for retrieving real-time weather data. For our work, we used open API provided by Weather Underground[6] to access data from the nearest weather station to our selected locations. Combined with a Node-RED flow, we parsed the received data in a JSON file and filter out the needed parameters according to our defined schema. We assigned weather data three different levels (0,1, 2) depending on the conditions where '0' indicates a clear,sunny or cloudy weather, '1' indicates light rain or light shower and '2' represents heavy rain or stormy weather. An instance of weather table stored in cloud storage after processing raw data is shown in Figure 6.

All the Node-RED flows used in this work are made available at [31].

V. LEVEL 2: PROBABILISTIC EVENT PROCESSING

After extracting derived events from individual data streams, they are combine to extract high-level knowledge using second layer of analytics. In state-of-the-art CEP systems, events are correlated using absolute rules where complex event detected is either true or false. Even if a single condition or rule is violated, complex event will not be generated. It is a major drawback given the random and probabilistic nature of real-world events. In contrast to conventional CEP systems, we propose a probabilistic CEP approach using BNs

[6]http://api.wunderground.com/

time_stamp	TemperatureC	Humidity	weather_level
1.4575644E9	7	33	0
1.4575644E9	7.0	46	0
1.4575662E9	6.0	49	0
1.457568E9	7	34	0
1.457568E9	6.0	49	0
1.4575698E9	6.0	49	0
1.4575716E9	6	40	0
1.4575716E9	5.0	57	0
1.4575734E9	5.0	57	0
1.4575752E9	5	47	0

only showing top 10 rows

FIGURE 6. Weather events table layout in object storage.

where complex events are detected in terms of probabilities. And even if any condition is not fulfilled, it will generate the complex event with reduced probability. In our system, BNs were trained using large historical data and integrated with CEP rules.

A. FUNDAMENTALS OF BAYESIAN NETWORK

A Bayesian Network (BN) is a graphical structure that allows to represent and reason the inherent uncertainty in real world problems. The nodes in a BN represent a set of random variables, $X = X_1, ..X_i, \ldots X_n$, from a specific domain. A set of directed arcs connect pairs of nodes, $Xi \rightarrow Xj$, representing the direct dependencies between variables. The strength of the relationship between different random variables is quantified by conditional probability distributions associated with each node. The only constraint on the arcs allowed in a BN is that there must not be any directed cycles i.e. you cannot return to a node simply by following directed arcs. Such networks are called directed acyclic graphs (DAGS). The relationship between different nodes is specified using a conditional probability distribution for every node. In case of discrete variables, it takes the form of a conditional probability table (CPT). BNs model the quantitative strength of the connections between variables which allows it to update probabilistic beliefs about them automatically as new data arrive.

Markovian assumption is an important property when modeling with Bayesian networks. In order to understand Markovian assumption, consider the following notation;

- $Parents(X)$ are the parents of X in DAG G, that is, the set of variables Y with an edge from Y to X.
- $Descendants(X)$ are the descendants of X in DAG G, that is, the set of variables Y with a directed path from X to Y.
- $Non - Descendants(X)$ are all variables in DAG G other than X, Parents (X), and Descendants (X).

For the given notation, Markov property can be represented as:

$$I(X, Parents(X), non - descendants(X)) \tag{1}$$

for all variables X in DAG G. That is, every variable is conditionally independent of its non-descendants given its parents.

The probabilistic semantics of BNs can be interpreted using joint probability distribution. For a BN containing the n nodes, $X1 \rightarrow Xn$, taken in that order, a particular value in the joint distribution is represented by $P(X1 = x1, X2 = x2, \ldots, Xn = xn)$, or more compactly, $P(x1, x2, \ldots, xn)$. The chain rule of probability theory allows us to factorize joint probabilities so:

$$\begin{aligned} &P(x_1, x_2, \ldots, x_n) \\ &\quad = P(x_1) \times P(x_2|x_1) \ldots, \times P(x_n|x_1, \ldots, x_{n-1}) \\ &\quad = \prod_i P(xi|x1, \ldots, x_{i-1}) \end{aligned} \tag{2}$$

Applying Markovian assumption on BN structure implies that the value of a particular node is conditional only on the values of its parent nodes, this reduces to;

$$P(x1, x2, \ldots, xn) = \prod_i P(x_i|Parents(X_i)) \tag{3}$$

IMPORTANCE OF BAYESIAN NETWORK

Graphical models like BNs have several advantages when they are used in conjunction with statistical methods for data analysis which are summarized below;

- Bayesian model encodes dependencies among all variables which enables it to handle situations where some data entries are missing. The dependencies between variables can be exploited to infer missing information.
- The model has both a causal and probabilistic semantics which makes it as an ideal representation for combining prior knowledge (which often comes in causal form) and data. Anyone having performed a real-world analysis knows the importance of prior or domain knowledge, especially when data is scarce or expensive.
- A BN can be used to learn causal relationships, and hence can be used to gain understanding about a problem domain and to predict the consequences of different combination of events.
- Finally, statistical methods in conjunction with BNs provides a rather generic and efficient solution avoiding the over-fitting of data. For BNs, there is no need to hold some data for testing, instead all available data can be used for training.

B. BAYESIAN NETWORKS FOR INTELLIGENT TRANSPORTATION SYSTEM

In particular, to construct a Bayesian network for a given set of variables, we first identify cause and effect variables and then imply arcs from cause variables to their immediate effects. Traffic Congestion (C), Weather (W), Large Crowd Concentration Event (E), Time (T) and Day (D) are the set of variables in our system and we are interested to find the causal effect of these variables on a traffic Congestion (C). Assuming conditional independence between weather, LCC,

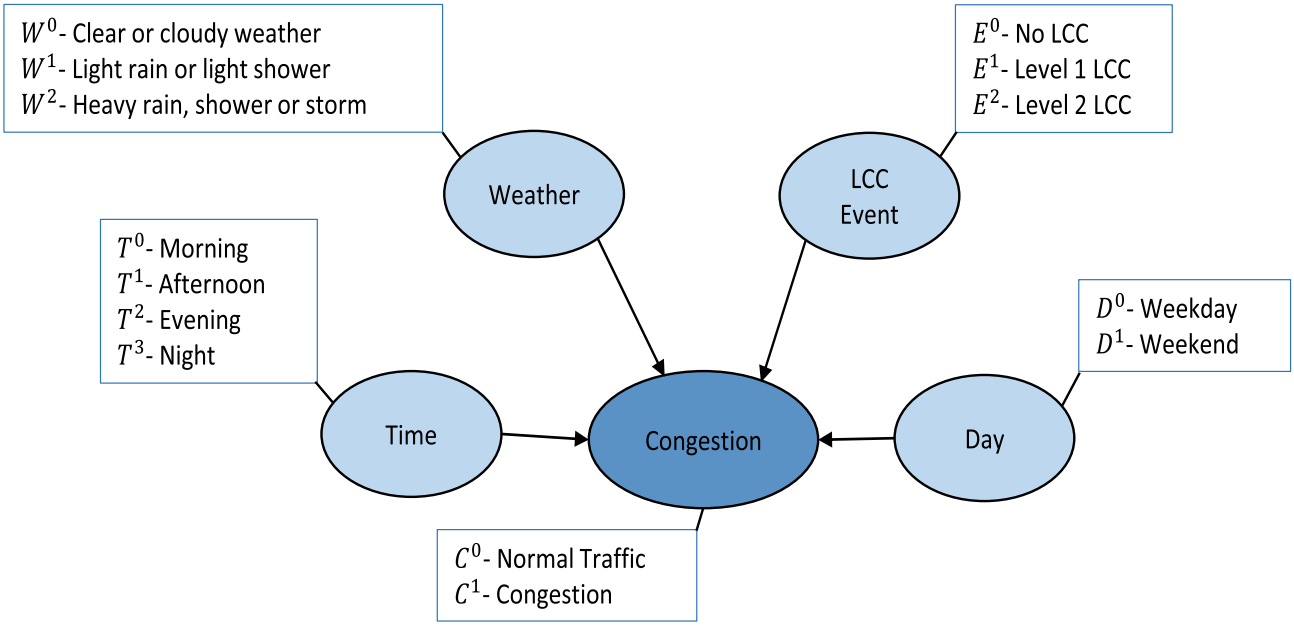

FIGURE 7. Proposed bayesian network.

time and and day, we come up with a simplified Bayesian structure as shown in the Figure 7.

1) PROBABILISTIC INFERENCE

Once a Bayesian network is constructed, we need to estimate prior and conditional probabilities from the historical data. Our final goal is to get the probability of congestion given observations of the other variables. It is not stored in the model and needs to be estimated at run-time. The procedure of estimating the probability of interest for given model is known as probabilistic inference. As BN for X variables determines a joint probability distribution for all X variables, we can in principle use this model to calculate any probability of interest using Bayes' theorem.

2) DATA FUSION AND ANALYTICS

Different data sources have different sampling time and every data reading has a different time stamp. Traffic data is updated every 5 minutes, weather data is refreshed every 30 minutes and twitter data generates several tweets every minute. In order to combine all data sources, we need to bring all data on a common time scale. We performed interpolation to fill missing values and aggregated available data for every 30 minutes and combine data from tables shown in Figure 4, 5 and 6. The instance of resulting table with all data is shown in the Figure 9. Once, all the data with individual derived events are combined in a single table, the calculation of conditional probabilities is a simple task using Spark SQL. As an example, probability of congestion when weather is good (represented by 0 in data), and no LCC event (represented by 0 in data) and afternoon time on a weekday can be calculated as below;

TABLE 2. Conditional probability table for location 1 for morning time.

No.	E, W, D	C^0	C^1
1	E^0, W^0, D^0	0.82	0.18
2	E^0, W^0, D^1	0.98	0.02
3	E^0, W^1, D^0	0.88	0.12
4	E^0, W^1, D^1	0.96	0.04
5	E^0, W^2, D^0	0.78	0.22
6	E^0, W^2, D^1	0.97	0.03
7	E^1, W^0, D^0	0.81	0.19
8	E^1, W^0, D^1	0.86	0.14
9	E^1, W^1, D^0	0.79	0.21
10	E^1, W^1, D^1	0.94	0.06
11	E^1, W^2, D^0	0.74	0.26
12	E^1, W^2, D^1	0.96	0.04
13	E^2, W^0, D^0	0.80	0.20
14	E^2, W^0, D^1	0.93	0.07
15	E^2, W^1, D^0	0.77	0.23
16	E^2, W^1, D^1	0.92	0.08
17	E^2, W^2, D^0	0.73	0.27
18	E^2, W^2, D^1	0.93	0.07

*E = Event, W = Weather, D = Day, C = Congestion

```
sqlContext.sql(" SELECT * FROM total_table
WHERE congestion = '0' AND LCC_level = '0'
AND day = '0' AND tf >= '08:00:00' AND
tf <= '12:00:00'  ").count()/ total_count
```

where total_table is the combined table of all data (an instance is shown in Figure 9)

Spark and Spark SQL provides an efficient and scalable approach for calculating conditional probabilities from this large amount of data from multiple tables in a timely manner. The overall data flow is shown in Fig. 8. The number of

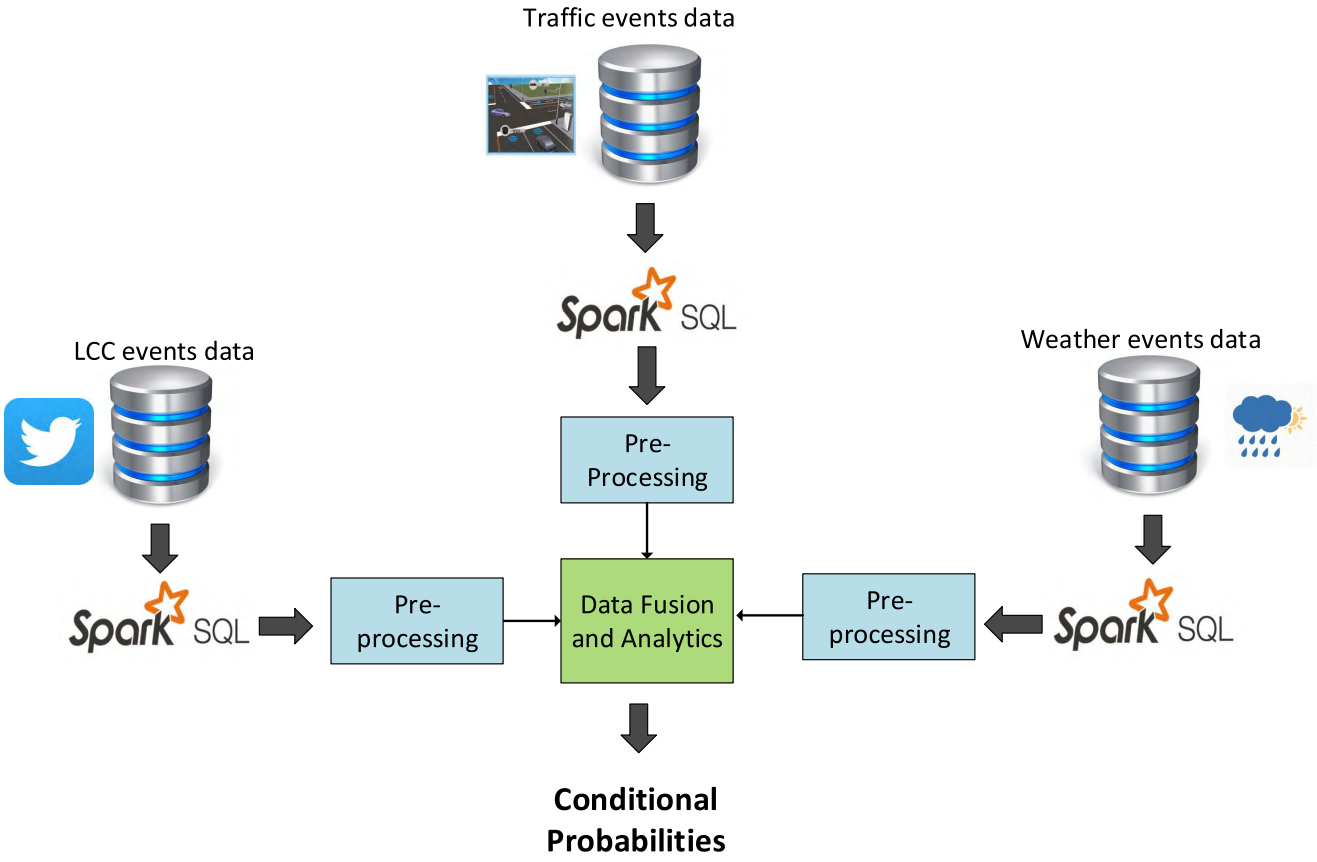

FIGURE 8. Calculation of conditional probabilities.

time_stamp	intensity	velocity	congestion	twitter_counts	LCC_level	weather_level	weekday	day
1.4586426E9	5080	86	0	60	0	0.0	2	0
1.4620014E9	4660	88	0	43	0	0.0	6	1
1.4621652E9	1418	91	0	17	0	0.0	1	0
1.4628204E9	2116	90	0	93	1	0.0	1	0
1.4620428E9	2138	90	0	88	0	0.0	6	1
1.4622066E9	3876	89	0	83	0	0.0	1	0
1.4623704E9	7076	65	0	74	0	0.0	3	0
1.4625342E9	6716	56	0	67	0	0.0	5	0
1.462698E9	4488	81	0	65	0	2.0	7	1
1.4599638E9	4132	88	0	73	0	0.0	3	0

only showing top 10 rows

FIGURE 9. An instant of all data combined in the form of table with unified time stamps.

possible combinations for conditional probability depends on the number of variables and their possible output states in the system. For our scenario, it equates to 4 x 3 x 3 x 2 x 2 = 144 possible combinations. An instance of conditional probability table for location 1 for morning time is shown in the Table 2.

VI. RESULTS AND EVALUATION

Figure 10 shows the Bayesian network simulation for different combinations of input events at location 1. The simulations were done using SamIam (Sensitivity Analysis Modeling Inference And More), which is a comprehensive tool for modeling and reasoning with BNs developed at the University of California [32]. Conditional probabilities calculated in section V were used as an input to BN. Different BNs were created for every location. Figure 10(a) shows the probability of congestion when no prior information about any event is given. As soon as the current day and time events are given, probability of congestion is updated using probabilistic inference. Congestion probability increases from

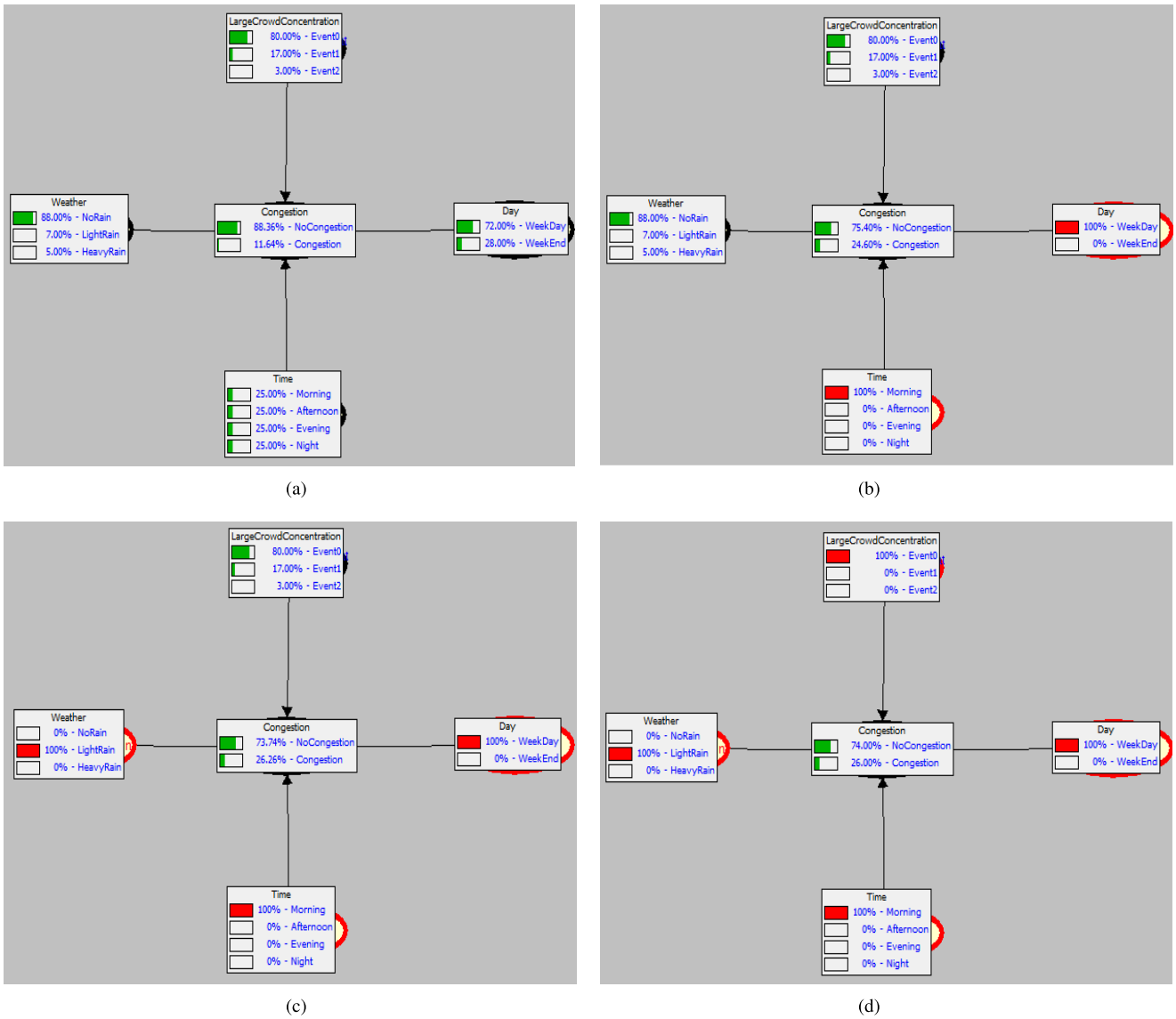

FIGURE 10. Different scenarios for location 1 simulated for bayesian network. (a) Scenario 1: when no event is given. (b) Scenario 2: with current time and day given. (c) Scenario 3: light rain event detected from weather data stream. (d) Scenario 4: no LCC detected from twitter data stream.

11.64% to 24.60% as it is morning time on a weekday. Probability of congestion further increases when *LightRain* event is detected from weather data stream as shown in Figure 10(c). Finally, as the system detects the final input event *LCC* from twitter data, it updates the probability of congestion for the given context. Bayesian network implementation enables the CEP to take decisions and predict the output with incomplete data as evident from the different scenarios demonstrated in Figure 10. After simulating different scenarios, we embed the Bayesian probabilities in CEP rules and deployed it for our scenario. As the Bayesian implementation is done outside the CEP, our proposed solution is independent of the choice of particular CEP.

The proposed solution is evaluated qualitatively with the help of Madrid city council team responsible for managing traffic. We developed a web-interface using the worldmap node of Node-RED, in order to display different events and provide a visual output of our system as shown in the Figure 11. City of Madrid have cameras installed at selective locations for monitoring traffic and we have chosen the same locations in order to verify the prediction results. The output panel in Figure 11 shows the current state of traffic, LCC event and weather and the probability of congestion for one of the selected location. Traffic administrators can use these predictions and manage the traffic in pro-active manner. As it can be seen from the figure, that it displays the current traffic state alongside with the probability of congestion for given context. The images from the camera helped traffic administrators to provide feedback about the quality of predictions using our rating system as shown in the figure.

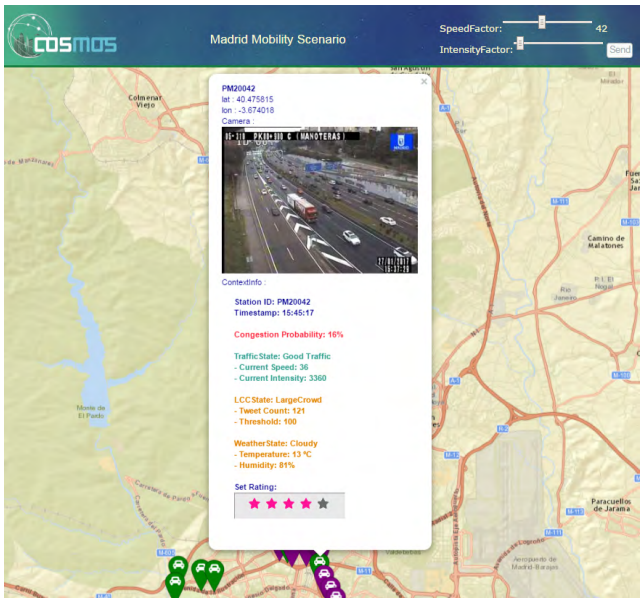

FIGURE 11. Web-interface showing the output of the system.

The output of the system is the probability of congestion at any time with respect to current context (location, time, day, weather and LCC events). In order to further evaluate our system, we set different threshold values on the probability of congestion and match if it correctly predicts the congestion in near future.

The performance of the system is evaluated using F-measure. F-measure represents an accurate and widely used metric for comparing the performance of classifiers and is defined as;

$$F - measure = \frac{2P.R}{P+R} \tag{4}$$

where P represents Precision and R represents Recall [33]. Precision of the algorithm is the ratio of the number of correct events to the total number of events detected; whereas recall is the number of events detected by the algorithm to the total number of events that should have been detected ideally. They can be calculated as follows,

$$Precision = \frac{TP}{TP+FP}, \quad Recall = \frac{TP}{TP+FN} \tag{5}$$

where TP is true positive, FP is false positive and FN is false negative.

Results are shown below in Table 3. It can be seen that setting lower threshold values results in better performance. As the threshold value is increased, the performance deteriorates due to low values of recall. The reason for low values of recall is that with high threshold, it will not predict congestion until the probability reaches the minimum threshold and result into missing actual congestion events.

In general, setting low threshold values on probabilities to generate congestion events results in low precision whereas high threshold values results in low recall. Finding right balance between precision and recall is important, otherwise either it will generate many false alarms or will miss actual congestion events.

TABLE 3. Evaluation for congestion prediction at location 1.

Threshold	TP	FP	FN	Precision	Recall	F-measure
40%	84	27	4	0.75	0.95	0.83
50%	81	21	13	0.79	0.86	0.82
60%	72	14	29	0.84	0.71	0.77
70%	28	4	83	0.87	0.25	0.38

DISCUSSION

The proposed architecture provides a generic solution for different IoT applications. Different components involved can easily be configured according to the requirements of a specific application. For example, the proposed solution can contribute towards the efficient management of supply chain for large retail stores. There are many factors which effect the sales of their products including weather (hot weather increases sales of certain products like cold beverages), time of the year (holiday and festive period tend to have higher sales) or an event happening in the region such as musical festival or concert. By combining all of the available data using our proposed solution, store managers can predict the demand for their product in real-time with respect to current circumstances.

In our research, all the locations are chosen from Madrid ring road (M30) as Madrid city council have installed sensors on selective locations. Data is collected for over two months (approximately over 1 million data points) and congestion instants were limited during this time period on M30. It introduces some bias as the effect of twitter events and weather data might be more prominent inside city and busy locations such as around city center or main bus stations which we intend to explore in our future work.

VII. CONCLUSION AND FUTURE WORK

In this paper, we have proposed and implemented a solution for ingesting, analyzing and correlating heterogeneous data streams in order to provide an efficient, scalable and reliable solution for IoT applications. We have proposed a two layer architecture for analyzing IoT data. The first layer provides a generic interface for ingesting and analyzing data from different IoT systems in a scalable manner. It provides an architecture based on the combination of batch and event processing methods for extracting high-level events from individual IoT data streams. Whereas the second layer extends state-of-the-art event processing mechanisms to take inherent uncertainty of events into account and provide a probabilistic solution for correlating high-level events based on bayesian network and CEP.

The feasibility of the proposed architecture was demonstrated with the help of a real-world use case from ITS where external data feeds of social media and weather were explored along with conventional traffic sensors in order to improve existing systems and generate early warnings of traffic congestion enabling system administrators to manage traffic in

a pro-active manner. All the implementation was done using open source components which are optimized for large-scale applications. In future, we aim to evaluate our architecture for other IoT applications where external data sources can be explored to find a meaningful correlation with existing data sources.

ACKNOWLEDGMENT

The authors would like to acknowledge the support of the University of Surrey 5GIC members (http://www.surrey.ac.uk/5gic) for this work.

REFERENCES

[1] E. Ahmed *et al.*, "The role of big data analytics in Internet of Things," *Comput. Netw.*, vol. 129, pp. 459–471, Dec. 2017. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S1389128617302591

[2] S. Mumtaz, A. Alsohaily, Z. Pang, A. Rayes, K. F. Tsang, and J. Rodriguez, "Massive Internet of Things for industrial applications: Addressing wireless IIoT connectivity challenges and ecosystem fragmentation," *IEEE Ind. Electron. Mag.*, vol. 11, no. 1, pp. 28–33, Mar. 2017.

[3] J. Manyika *et al.*, "The Internet of Things: Mapping the value beyond the hype. Mckinsey Global Institute," Mckinsey Global Inst., San Francisco, CA, USA, Tech. Rep., Jun. 2015.

[4] C. L. P. Chen and C.-Y. Zhang, "Data-intensive applications, challenges, techniques and technologies: A survey on big data," *Inf. Sci.*, vol. 275, pp. 314–347, Aug. 2014. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S0020025514000346

[5] J. Dean and S. Ghemawat, "MapReduce: Simplified data processing on large clusters," *Commun. ACM*, vol. 51, no. 1, pp. 107–113, 2008. [Online]. Available: http://doi.acm.org/10.1145/1327452.1327492

[6] C. Lam, *Hadoop in Action*, 1st ed. Greenwich, CT, USA: Manning Publications, 2010.

[7] I. A. T. Hashem, N. B. Anuar, A. Gani, I. Yaqoob, F. Xia, and S. U. Khan, "Mapreduce: Review and open challenges," *Scientometrics*, vol. 109, no. 1, pp. 389–422, Oct. 2016. [Online]. Available: https://doi.org/10.1007/s11192-016-1945-y

[8] M. Zaharia, M. Chowdhury, M. J. Franklin, S. Shenker, and I. Stoica, "Spark: Cluster computing with working sets," *HotCloud*, vol. 10, pp. 1–7, Jun. 2010.

[9] S. Chintapalli *et al.*, "Benchmarking streaming computation engines: Storm, flink and spark streaming," in *Proc. IEEE Int. Parallel Distrib. Process. Symp. Workshops (IPDPSW)*, May 2016, pp. 1789–1792.

[10] D. Luckham, *The Power of Events*, vol. 204. Reading, MA, USA: Addison-Wesley, 2002.

[11] N. H. Gehani, H. V. Jagadish, and O. Shmueli, "Composite event specification in active databases: Model & implementation," in *Proc. VLDB*, vol. 92. 1992, pp. 327–338.

[12] G. Cugola, E. D. Nitto, and A. Fuggetta, "The JEDI event-based infrastructure and its application to the development of the OPSS WFMS," *IEEE Trans. Softw. Eng.*, vol. 27, no. 9, pp. 827–850, Sep. 2001.

[13] K. Gomadam, A. Ranabahu, L. Ramaswamy, A. P. Sheth, and K. Verma, "A semantic framework for identifying events in a service oriented architecture," in *Proc. IEEE Int. Conf. Web Services (ICWS)*, Jul. 2007, pp. 545–552.

[14] G. Cugola and A. Margara, "Processing flows of information: From data stream to complex event processing," *ACM Comput. Surv.*, vol. 44, no. 3, Jun. 2012, Art. no. 15. [Online]. Available: http://doi.acm.org/10.1145/2187671.2187677

[15] N. Marz and J. Warren, *Big Data: Principles and Best Practices of Scalable Realtime Data Systems*, 1st ed. Greenwich, CT, USA: Manning Publications, 2015.

[16] P. Ta-Shma, A. Akbar, G. Gerson-Golan, G. Hadash, F. Carrez, and K. Moessner, "An ingestion and analytics architecture for iot applied to smart city use cases," *IEEE Internet Things J.*, to be published.

[17] N. Khoussainova, M. Balazinska, and D. Suciu, "Peex: Extracting probabilistic events from RFID data," Dept. Comput., Sci. Eng., Univ. Washington, Seattle, WA, USA, Tech. Rep. 2007-11-02, 2007.

[18] S. Wasserkrug, A. Gal, and O. Etzion. (2012). "A model for reasoning with uncertain rules in event composition systems." [Online]. Available: https://arxiv.org/abs/1207.1427

[19] S. Wasserkrug, A. Gal, O. Etzion, and Y. Turchin, "Efficient processing of uncertain events in rule-based systems," *IEEE Trans. Knowl. Data Eng.*, vol. 24, no. 1, pp. 45–58, Jan. 2012.

[20] G. F. Cooper, "The computational complexity of probabilistic inference using Bayesian belief networks," *Artif. Intell.*, vol. 42, nos. 2–3, pp. 393–405, 1990. [Online]. Available: http://www.sciencedirect.com/science/article/pii/000437029090060D

[21] D. Akin, V. P. Sisiopiku, and A. Skabardonis, "Impacts of weather on traffic flow characteristics of urban freeways in Istanbul," *Procedia-Social Behav. Sci.*, vol. 16, pp. 89–99, Jan. 2011.

[22] J. S. Oh, Y. U. Shim, and Y. H. Cho, "Effect of weather conditions to traffic flow on freeway," *KSCE J. Civil Eng.*, vol. 6, no. 4, pp. 413–420, 2002. [Online]. Available: http://dx.doi.org/10.1007/BF02841995

[23] P. Anantharam, P. Barnaghi, K. Thirunarayan, and A. Sheth, "Extracting city traffic events from social streams," *ACM Trans. Intell. Syst. Technol.*, vol. 6, no. 4, Jul. 2015, Art. no. 43. [Online]. Available: http://doi.acm.org/10.1145/2717317

[24] A. Akbar, F. Carrez, K. Moessner, J. Sancho, and J. Rico, "Context-aware stream processing for distributed IoT applications," in *Proc. IEEE 2nd World Forum Internet Things (WF-IoT)*, Dec. 2015, pp. 663–668.

[25] Node-RED. (2016). *Node-RED: A Visual Tool for Wiring the Internet of Things*. Accessed: May 6, 2016. [Online]. Available: http://nodered.org//

[26] C.-E. Lu, "Apache Kafka: A high-throughput distributed messaging system," in *Proc. JCConf*, Taipei, Taiwan, Nov. 2014, accessed: Aug. 10, 2017. [Online]. Available: http://kafka.apache.org

[27] *OpenStack is Open Source Software for Creating Private and Public Clouds*. Accessed: Sep. 15, 2017. [Online]. Available: https://www.openstack.org/

[28] Twitter. (2017). *Twitter Developer Platform*. Accessed: Mar. 22, 2017. [Online]. Available: https://dev.twitter.com/docs

[29] AVRO. (2017). *Apache AVRO Specification*. Accessed: Dec. 22, 2017. [Online]. Available: https://avro.apache.org/docs/1.8.1/spec.html

[30] G. Kousiouris *et al.*, "An integrated information lifecycle management framework for exploiting social network data to identify dynamic large crowd concentration events in smart cities applications," *Future Generat. Comput. Syst.*, vol. 78, pp. 516–530, Jan. 2018.

[31] *COSMOS Middleware Flows Repository*. Accessed: May 22, 2017. [Online]. Available: https://github.com/COSMOSFP7/COSMOS-Platform-side

[32] SamIam. (2017). *Sensitivity Analysis, Modeling, Inference and More*. Accessed: Mar. 22, 2017. [Online]. Available: http://reasoning.cs.ucla.edu/samiam/

[33] J. Davis and M. Goadrich, "The relationship between Precision-Recall and ROC curves," in *Proc. 23rd Int. Conf. Mach. Learn. (ICML)*, New York, NY, USA, 2006, pp. 233–240. [Online]. Available: http://doi.acm.org/10.1145/1143844.1143874

ADNAN AKBAR received the B.Sc. degree from the University of Engineering and Technology, Pakistan, in 2009, and the M.Sc. degree (Hons.) from the University of Surrey, U.K., in 2011. He is currently pursuing the Ph.D. degree with the 5G Innovation Centre, University of Surrey, U.K. Before joining the Ph.D. program, he was a Research Assistant with the Institute of Space Technology, Pakistan, for two years. His current research interests lie in machine learning, complex event processing, big data, and Internet of Things

GEORGE KOUSIOURIS received the Diploma degree in electrical and computer engineering from the University of Patras, Greece, in 2005, and the Ph.D. degree in grid and cloud computing from the Department of Electrical and Computer Engineering, National Technical University of Athens (NTUA), in 2012. He is currently a Post-Doctoral Researcher with the Institute of Communication and Computer Systems, NTUA. His interests are mainly computational intelligence, cloud computing, service level agreements, optimization, analytics, artificial intelligence, and web services.

HARIS PERVAIZ received the Ph.D. degree from the School of Computing and Communication, Lancaster University, U.K., in 2016, the M.Sc. degree in information security from the Royal Holloway University of London in 2005, and the M.Phil. degree in electrical and electronic engineering from the Queen Mary University of London, in 2011. From 2016 to 2017, he was an EPSRC Doctoral Prize Fellow with the School of Computing and Communication, Lancaster University. He is currently a Research Fellow with the 5G Innovation Centre, University of Surrey, U.K. His main research interests are green communications, control data separation architecture, 5G and beyond systems, Internet of Things, UAV, and millimeter wave communication.

JUAN SANCHO received the degree in telecommunications engineering from the Universidad Politécnica de Valencia, Spain. He developed his Final Project Degree in the field of health monitoring using Wireless Sensor Networks with the Wireless Centre, Copenhagen University of Engineering, Denmark. In 2011, he became a Network and Systems Engineer, TST Systems, participating in several European FP7, ENIAC, and National projects (BUTLER, TOISE, SICRA, TSMART). Since 2014, he has been a Research and Innovation Engineer with ATOS Research and Innovation Labs, participating in COSMOS, P-SOCRATES, ROAD2CPS, and inteGRIDy projects. His research interests cover IoT platforms, M2M communications, WSN protocols, and low-cost, low-power consumption embedded systems.

PAULA TA-SHMA received the M.Sc. and Ph.D. degrees in computer science from the Hebrew University of Jerusalem. She is currently a Research Staff Member with the IBM Cloud Security and Analytics Group. She is currently involved in cloud storage infrastructure for the Internet of Things, and leads several related research efforts. She led IBM efforts in the EU funded COSMOS project and various other research projects such as continuous data protection. Her work has been presented at multiple industry conferences, including the Apache Spark Summit, the OpenStack Summit, IBM Insight, and IBM InterConnect, and academic conferences such as FAST.

FRANCOIS CARREZ received the Ph.D. degree in theoretical computer science from the University of Nancy, France, in 1991. He has been working 20 years for the Alcatel Research Centre, Paris, in areas such as security, distributed artificial intelligence, ad hoc networking, and semantics. Since he joined the University of Surrey in 2007, he has been in particular leading the ICT-FP7 Internet of Things initiative Coordination Action, which is at the origin of the IoT International Forum. He has been also a WP Leader on architecture for IoT-A, the flagship European project on architecture for the Internet of Things, which eventually released the comprehensive architecture reference model for the IoT. He is currently involved in COSMOS (SmartCity call) and FIESTA-IoT, applying the ARM methodology to those two projects.

KLAUS MOESSNER is currently a Professor with the Institute for Communication Systems, University of Surrey. His research interests include cognitive networks, IoT deployments, and sensor data based knowledge generation, and reconfiguration and resource management. He was involved in many projects in the cognitive communications, service provision, and IoT areas. He was responsible for the work on cognitive decision making mechanisms in the CR project ORACLE, and led the work on radio awareness in the ICT FP7 project QoSMOS and is currently leading the H2020 Speed5G project. In the past, he was the Founding Chair of the IEEE DYSPAN Working Group on sensing interfaces for future and cognitive communication systems. He was involved in the definition and evaluation of cooperation management between autonomous entities, in the UniverSelf project, and was a Technical Manager of the iCore project and has the same role in the H2020 project CPaaS.io, he was a Project Leader of IoT.est, SocIoTal. He currently leads the iKaaS and Speed-5G projects.