Available online at www.sciencedirect.com

ScienceDirect

Procedia Computer Science 52 (2015) 1186 – 1191

Procedia Computer Science

The 2nd International Workshop on Machine Learning and Data Mining for Sensor Networks (MLDM-SN 2015)

A Framework for Distributed Cleaning of Data Streams

Saul Gill[a,*], Brian Lee[a]

[a]*Software Research Institute, Athlone Institute of Technology, Athlone, Ireland*

Abstract

Vast and ever increasing quantities of data are produced by sensors in the Internet of Things (IoT). The quality of this data can be very variable due to problems with sensors, incorrect calibration etc. Data quality can be greatly enhanced by cleaning the data before it reaches its end user. This paper reports on the construction of a distributed cleaning system (DCS) to clean data streams in real-time for an environmental case-study. A combination of declarative and statistical model based cleaning methods are applied and initial results are reported.

Peer-review under responsibility of the Conference Program Chairs

Keywords: Internet of Things; Data Stream Cleaning; Model Based Cleaning; Regression; Declarative Cleaning; Distributed Cleaning; Sensor Data Modelling

1. Introduction

The exponential growth of the IoT in recent years is leading to the generation of data on a very large scale including for environmental applications. Quick and universal access to this environmental data could yield benefits for health and general societal quality of life. However, there are some common difficulties that are experienced during the acquisition of environmental data from sensors. Such data can be dirty i.e. sensor readings are often out of range, null values can be produced or duplicate values can even be produced. This may be due to: damage, low battery power, incorrect calibration and other issues. This can render the data produced by sensors unusable for analysis. These data are greatly improved by cleaning the data streams.

Many different techniques have been used to clean streaming data. Bayesian Methods have been used for streaming and batch data cleaning[1]. Kalman Filters have been implemented using an extensible toolkit with graphical visualization for cleaning of streaming data[2]. Declarative languages have been used together with cleaning frameworks[3]. Model-based techniques have been used to help cleaning systems adapt to the characteristics and behaviour of particular kinds of data[4].

Although a number of authors have examined techniques for distributed data stream cleaning[5], there are gaps in the existing research that suggest areas for improvement. Firstly, more specialized techniques may yield improvements when examining environmental data because many contextual factors can affect the magnitude of the reading e.g. the location where the reading was taken can be important to understanding and analysis of the data. Proximity models, interpolation models, line dispersion models, integrated emission-meteorological models, hybrid models and Land

* Corresponding author. Tel.: +353-852392237
E-mail address: sgill@research.ait.ie

Peer-review under responsibility of the Conference Program Chairs
doi:10.1016/j.procs.2015.05.156

Use Regression Models (LUR)[6] could be used to provide a contextual basis for environmental data stream cleaning. Secondly, the growth of Fog computing[7], suggests the use of Big Data stream processing frameworks can facilitate the development of distributed stream cleaning systems. Fog Computing[7], is a distributed cloud computing architecture that pushes processing to the network edge, suggesting that benefits of cost and scale could be attained through distributed real-time cleaning of data streams. This work describes the ongoing development of such a distributed data cleaning system.

Our research aims to investigate the use of statistical models that incorporate contextual data such as land use factors - to provide high quality, high performance data cleaning for environmental data streams. The main goals of the research are:

1. To develop and evaluate data cleaning statistical models that incorporate use of land use semantics. Land use includes factors such as the building mix in the locale, distance from the nearest highway etc.
2. To develop a stream processing platforms for real time data stream cleaning. This will be done initially using the models developed in the previous step and the design will later be generalised for a variety of models and stream deployment scenarios.

Data streams, cleaned by these processes, will be the source of reliable information for users of the cloud, whether they be academic, industrial or private clients. This paper reports on our progress to-date, primarily on the second goal.

This rest of this paper is structured as follows: First an overview of the distributed cleaning system (DCS) is presented. The second section details the previous work that provided a foundation for the creation of this system, through a brief exploration of related research. The third section of the paper describes the parts of the system that have already been implemented and the evaluation of those parts. A final section details the conclusions that can be drawn from this study along with a description of further work to be carried out.

2. System Overview

This section describes the overall structure of the distributed cleaning system as it will be developed for the current project. As such it gives a context to the description of work and results to date in the following sections. Ultimately the DCS will be developed as a configurable multi-stage processing framework enabling various cleaning functions to be combined or pipelined as appropriate for individual scenarios. Typical of such functions are those in the ESP framework[3].

1. *Point* operates on a single value in a sensor stream with the primary goal of filtering individual values e.g. obvious outliers.
2. *Smooth* applies temporal windowing across multiple values to correct missing values or correct outliers in a single stream.
3. *Merge* is used to process readings of a number of spatially related sensors of a single type
4. *Arbitrate* is used to resolve conflicting readings between nearby spatial groups
5. *Virtualise* allows readings to be combined across sensors of different types or different spatial granules.

While ESP uses a declarative language for all functions, in general they may be implemented with different technologies, including statistical models. These five functions form a rich toolset with which to develop a cleaning framework, though not all functions will be needed for every such system. This is indeed is the case for the present work. In particular we use only the elements *point* and *smooth* as these are appropriate for the data set we are using. This consists of atmospheric gas readings that were recorded on wireless motes, mounted on buses in the Serbian city of Pancevo. The DCS is designed to process a number of concurrent streams, all sampled at the same time instants. We refer to the values read at any instant as a Tuple.

The main logic of the system is illustrated in both Figure 1 and Figure 2. Data streams are fed in to the DCS based on a Stream Processing Engine(SPE) e.g. SparkStreaming[8]. The current system version uses declarative cleaning to implement a *point* function and regression models to implement the *smooth* function. A combination of Simple Regression (SR) and Multiple Regression (MR) will be used - SR for variables such as weather, temperature, pressure, humidity etc. and MR for atmospheric gas readings (see Figure 2). For the SR models, time is the independent variable and models are built for every data segmentation. The data segmentation is configurable and refers to the number of past values included in the model. The MR models will also be built for every data segmentation to deal with concept

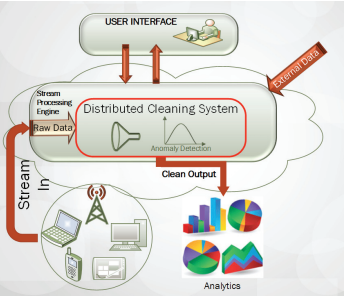

Fig. 1: Stream Cleaning Overview

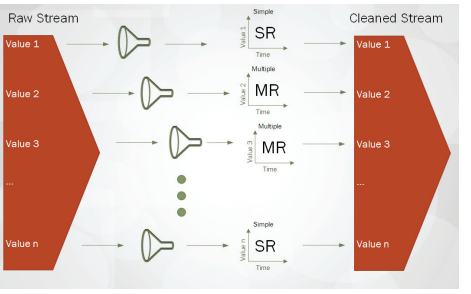

Fig. 2: Distributed Cleaning System: Internal

drift - however, this is not addressed in the current work. In the longer term, the DCS will be extended to include different model types and be fully configurable in parameter specification.

In Figure 1, the DCS processes the tuples from the data streams using 2 main operations. In the *point* operation, data values from the tuple that are obviously incorrect are removed. An example of this would be if there was an environmental temperature reading of 60 degrees Celsius, the tuple containing that value would be removed. In the second operation i.e. the *smooth* function, statistical models are applied to each of the values in each stream. Anomalies are removed from the stream and replaced by predicted values that have been produced by the models.

The DCS will be fully configurable. The type of statistical models[1] built and the parameters used in those models are options selected by the user, via a user interface. Available options for the configuration of the system will be displayed to the user in a menu format including the number of past values on which the models are to be built. The user will also have the option to include external data where needed. e.g. parameters for land use such as elevation, population density etc.

3. Related Work

Although there are many tried and tested cleaning methods that have been applied successfully for static data[9,10], a different approach is needed when cleaning streaming data. The continuous updating of the data presents a different set of challenges for data cleaning. There have been many successful efforts to build structures around streaming data that allow the streams to be accessed in a similar way to static structures such as database tables, using declarative languages[5]. Filtering, aggregation, transformation and calculations can be carried out on the incoming streams of data using these query languages.

Staged frameworks have been employed widely for the cleaning of static data[9]. In this particular staged framework, declarative queries are used to perform various filtering steps on the data to improve its quality. ESP is one such framework[3].

Statistical models have been applied in different systems to clean data streams. Polynomial regression and Kalman filters were compared in one piece of research[2] which is part of the HiFi project[11]. It was found that some success was achieved when using a high degree polynomial but that the models were not as successful when used on data with high variability. Modelling was also used for data cleaning using Chebyshev regression[4] where time is used as the independent variable. However, neither of these studies use multiple regression. Using multiple regression assumes that there are factors, other than time, which can be used to examine the typical distribution of the dependent variable in the model. This has been explored to some degree in a study regarding cyber-physical systems[12]. This work uses polynomial models which include independent variables like latitude, longitude, dew point, visibility and several others to model the outcome variable of interest. Anomalous values are then identified as those values which lie outside certain bounds of the model. This research works only with polynomial models, which can be difficult to fit. No consideration is given to linear models or spline models which may show more success. Although several independent variables are considered, variables such as land use, population density, wind speed and wind direction have not been explored in these models. These independent variables have been used in other contexts like proximity models, interpolation models, line dispersion models, integrated emission-meteorological models, hybrid models and

[1] As noted regression models are the focus of our current work but the DCS will eventually be capable of handling multiple models

Land Use Regression Models (LUR)[6]. These variables could be used to provide a contextual basis for environmental data stream cleaning.

Processing data from a large network of wireless sensors would potentially cause performance bottlenecks if executed on a centralized system. Some research has been carried out to explore the possible deployment of applications like data cleaning frameworks on distributed networks[11]. This work details a bow-tie shaped network where the edges of the network consist of many sensor devices. The data from these devices would be aggregated locally in storage and further aggregation would take of smaller storage areas in to a larger storage area. The system aims to perform functions like data cleaning at the edge of the network due to chances of bottlenecking at the centre of the network. This system does not distribute the computational load evenly over the network but instead pushes the computational load to the edge. Apache Spark[8] provides the means to distribute the computational load as required. Spark can process both batch and streaming data on a large scale across a distributed network of computers. However, no research currently exists that deploys a distributed model based cleaning system using Spark or Spark Streaming. Some work has been carried out on streaming data in an application that employs wearable sensor devices and mobile devices to collect large amounts of air quality data. The application uses Apache Storm to process the data and send them back to the user in a relevant way[13]. However, this research does not specifically address the problem of data cleaning.

4. Initial Experimentation and Findings

During the implementation and testing stages of this research, the same dataset was used for building and validation. The data was taken from 39 different wireless sensor motes, mounted on top of buses, which travelled along regular routes through the city of Pancevo, Serbia from 2011 to 2013. The sensors took measurements for Carbon Dioxide (CO2), Carbon Monoxide (CO), Nitrogen Dioxide (NO2), temperature, pressure, humidity, time, latitude and longitude.

4.1. Streams-Esper

Streams Esper[14] is an integration of the Streams Framework and the Esper Complex Event Processing (CEP) engine. The graph flow style of the streams framework allows easy manipulation and processing of Streams. Esper's Event Processing Language (EPL) is a declarative language that facilitates use of statistical and aggregation functions that can act on a specified time window in one or multiple streams.

The Streams user can prototype top-level elements called *containers*. Within each *containers*, the user can define several pieces of logic which manipulate and process the incoming data stream. This logic is laid out in a pipelined manner. The stream of data is fed from one manipulation to another depending on how many steps the user implements in the pipeline. We defined a container that took a stream of data from a file. This stream of data was put through a series of manipulations. Firstly, using an EPL query, the data was converted to a format that made the values contained within the stream easier to process. A second query imposed thresholds, outside of which, data was removed from the stream. These thresholds varied depending on the kinds of values contained within the data stream e.g. temperature. The resultant filtered stream was written to an output file to make it available for use for building models in R.

4.2. Building Regression Models in R

The filtered data were then imported in to R to determine whether effective models could be built from the data. As we also wished to determine whether addition of meteorological and geographical values to the data would improve the models, R was used to access online APIs for these values[15]. Three kinds of multiple regression models were built: linear, polynomial and generalized additive models (GAM). NO2 was used as the test outcome variable for these models

For linear regression, all of the different terms available in the dataset were used as potential predictors of NO2. Best subsets selection was performed using packages available in R. The criteria used for the best possible model was the one that resulted in the lowest root mean squared error (RMSE). As a means of further validation of these models, only the data from the North of the city was used as training data, while the data from the south of the city was used as test data. The model with the best validated RMSE was chosen as the optimum model. The most effective model of each size is plotted in Figure 3. The increase in success of prediction on the test data levels off at 10 predictors. The 10 predictors in this optimal model were: timestamp, CO, atmospheric pressure, CO2, temperature, longitude, humidity and distances to 3 different locations in the city: a regional landfill, an oil refinery and a church.

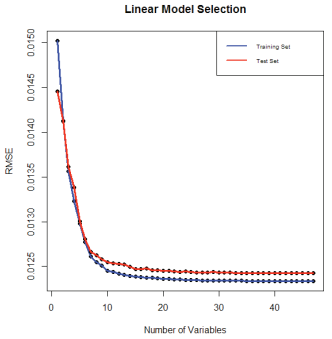

Fig. 3: Linear Selection Curve

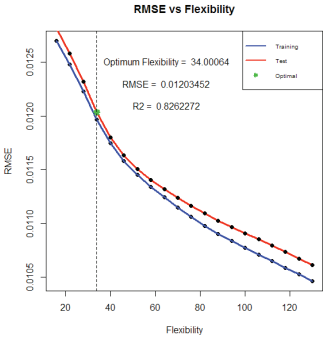

Fig. 4: GAM Selection Curve

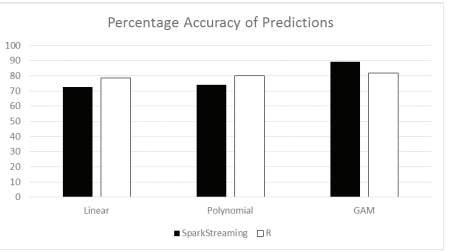

Fig. 5: Prediction Accuracy

Possible inclusion of a polynomial model was then investigated. Partial Residual plots were created in R in order to visually inspect the presence of any polynomial relationship between the NO2 outcome variable and all the predictors. Those variables which were deemed to have a non-linear relationship with the outcome variable were added as polynomial terms in to the dataset. Best subsets selection was again performed on the dataset including the polynomial terms. The models were validated on test data from the south of the sampled area and the model with the lowest RMSE was chosen as the optimum model.

When creating the GAM model, all those predictors that were found to have non-linear relationships with the outcome variable had smoothing splines applied to them. The linear predictors were also added in to the model, resulting in a model that had both linear and smoothing spline terms. The model that performed best on validation with the test data from the south of the study area was chosen as the optimum model. Figure 4 shows the model selection process. Models were created with increasing flexibility. Flexibility refers to the degrees of freedom used by the model. R automatically selects the model that will extrapolate best beyond the data used in model selection. Of the three kinds of models, the GAM model performed best overall having 87% prediction accuracy. All of these models were saved as R objects. Saving the models in this manner in the local directory allows the models to be imported at runtime in to any R script.

4.3. Spark Streaming and JVMR

After performing initial experimentation in Streams-Esper, we decided to move the project to a distributed framework. Spark Streaming was used as the SPE for testing of the models in a stream cleaning context. The JVMR project[16] allows the use of R code and packages on Spark Streaming values. JVMR was incorporated in to our Spark Streaming project. Apache Kafka[17] was used to feed data in to Spark Streaming.

The R code performed several different actions on the incoming Kafka stream. Firstly, one of the model objects was chosen (either Linear, Polynomial or GAM) and imported in to Spark Streaming. This model was then used to predict on the incoming stream of data.

R code was then used to determine whether the actual value lay within three standard deviations (or the threshold set by the user) of the models regression line. If the actual observed value was outside this threshold, the actual value was deemed an anomaly or outlier. Outliers were then replaced with predicted values.

In R values for the 3 standard deviation outlier thresholds were calculated (this threshold is configurable) on the entire test dataset. In Spark Streaming however, as the data is gradually moving through the system, standard deviation was calculated on the last 30 test values (this number is configurable). Figure 6 shows how the models performed in R on static data when compared to streaming data in SparkStreaming. In all cases, there is no very significant difference between the performance of the models in R and in Spark Streaming.

5. Conclusion and Future Work

The overall goal of this project is to develop a streaming data cleaning system that is capable of handling large amounts of data. Three parts of this goal have been explored in this paper. Firstly, a filtering *point* mechanism

for streaming data has been implemented using the declarative language provided by Streams-Esper. Secondly, the filtered data has been used imported in to R and used to investigate the building and validation of statistical prediction models using 3 different kinds of regression. Within R, online APIs were also used to add more relevant data to the existing dataset. These models were incorporated in to the DCS as smooth functions as shown in Figure 2. Finally, the models built in R were used in the Spark Streaming SPE for both the detection and replacement of anomalies or outliers which are found in the test data.

Multiple other steps must be taken in order to follow this work to its completion. As it is at the moment, the different parts of the system are separate. Streams-Esper and the model building in R lie outside the SPE. All of the functionality of the system must be integrated into the SPE. To achieve this, several steps must occur.

Firstly, the filtering functions which we performed using Streams-Esper must also be moved to Spark Streaming. Additionally, we have not yet explored the use of simple regression with time as the only predictor of the outcome variable. ESPs framework contains stages that involve the correlation of data from nearby devices as well as data from the same temporal window. Further investigation and implementation of these options will be carried out. The project will facilitate the addition of meteorological and geographical data at runtime in Spark Streaming to aid in the building of contextual models. Finally, a user interface will be developed to allow the user ease of access to the different functionality present in the system.

Acknowledgements

Many thanks to all the staff and students of the Software Research Institute in Athlone Institute of Technology for their support.

References

1. Zhao, Z., Ng, W.. A model-based approach for RFID data stream cleansing. Proceedings of the 21st ACM international conference on Information and knowledge management. The Hong Kong University of Science and Technology, Hong Kong, Hong Kong SRC - GoogleScholar FG - 0; 2012, p. 862–871.
2. Tan, Y.L., Sehgal, V., Shahri, H.H.. *Sensoclean: Handling Noisy and Incomplete Data in Sensor Networks using Modeling*. Ph.D. thesis; 2005. URL: `http://www.sccs.swarthmore.edu/users/03/yeelin/docs/finalreport.pdf`.
3. Jeffery, S.R., Garofalakis, M., Franklin, M.J.. Adaptive cleaning for RFID data streams 2006;:163–174URL: `http://dl.acm.org/citation.cfm?id=1182635.1164143`.
4. Jeung, H., Sarni, S., Paparrizos, I., Sathe, S., Aberer, K.. An End-to-End System for Cleaning Sensor Data : Model-Based Approaches. *Journal of Information Systems* 2010;.
5. Cugola, G., Margara, A.. Processing flows of information. *ACM Computing Surveys* 2012;**44**(3):1–62. URL: `http://dl.acm.org/citation.cfm?id=2187671.2187677`. doi:10.1145/2187671.2187677.
6. Jerrett, M., Arain, A., Kanaroglou, P., Beckerman, B., Potoglou, D., Sahsuvaroglu, T., et al. A review and evaluation of intraurban air pollution exposure models. *Journal of exposure analysis and environmental epidemiology* 2005;**15**(2):185–204. URL: `http://www.ncbi.nlm.nih.gov/pubmed/15292906`. doi:10.1038/sj.jea.7500388.
7. Bonomi, F., Milito, R., Zhu, J., Addepalli, S.. Fog computing and its role in the internet of things. In: *Proceedings of the first edition of the MCC workshop on Mobile cloud computing - MCC '12*. New York, New York, USA: ACM Press. ISBN 9781450315197; 2012, p. 13. URL: `http://dl.acm.org/citation.cfm?id=2342509.2342513`. doi:10.1145/2342509.2342513.
8. Zaharia, M., Chowdhury, M., Franklin, M.J., Shenker, S., Stoica, I.. Spark : Cluster Computing with Working Sets. In: *HotCloud'10 Proceedings of the 2nd USENIX conference on Hot topics in cloud computing*. 2010, p. 10.
9. Rahm, E., Do, H.H.. Data Cleaning: Problems and Current Approaches. *IEEE Data Engineering Bulletin* 2000;**23**:2000.
10. Galhardas, H., Florescu, D., Shasha, D., Simon, E.. Declaratively cleaning your data using AJAX. In: *In Journees Bases de Donnees*. 2000, .
11. Franklin, M.J., Jeffery, S.R., Krishnamurthy, S., Reiss, F., Rizvi, S., Wu, E., et al. Design Considerations for High Fan-In Systems: The HiFi Approach. In: *CIDR*. 2005, p. 290–304.
12. Javed, N., Wolf, T.. Automated Sensor Verification Using Outlier Detection in the Internet of Things. In: *ICDCS Workshops'12*. 2012, p. 291–296.
13. Rizea, D.O., Olteanu, A.C., Tudose, D.S.. Air quality data collection and processing platform. In: *RoEduNet Conference 13th Edition: Networking in Education and Research Joint Event RENAM 8th Conference, 2014*. IEEE; 2014, p. 1–4.
14. Scharrenbach, T.. Streams-Esper. 2013. URL: `https://bitbucket.org/scharrenbach/streams-esper/wiki/Home`.
15. World Weather Online, . No Title. 2013. URL: `http://www.worldweatheronline.com`.
16. Dahl, D.B.. *jvmr: Integration of R, Java, and Scala*; 2013. URL: `http://cran.r-project.org/package=jvmr`.
17. Garg, N.. Apache Kafka 2013;URL: `http://dl.acm.org/citation.cfm?id=2588385`.