Industrial and Applications Paper

Scouter: A Stream Processing Web Analyzer to Contextualize Singularities

Badre Belabbess
Atos Innovation Lab
Bezons, France
badre.belabbess@atos.net

Musab Bairat
Atos Innovation Lab
Bezons, France
musab.bairat@atos.net

Jeremy Lhez
LIGM, CNRS, ENPC, ESIEE, UPEM
Marne-la-Vallée, France
jeremy.lhez@u-pem.fr

Zakaria Khattabi
Atos Innovation Lab
Bezons, France
zakaria.khattabi@atos.net

Yufan Zheng
Atos Innovation Lab
Bezons, France
yufan.zheng@atos.net

Olivier Curé
LIGM, CNRS, ENPC, ESIEE, UPEM
Marne-la-Vallée, France
olivier.cure@u-pem.fr

ABSTRACT

Anomaly detection is a key feature of applications processing singularities using IoT sensor measures. In a recent project, we highlighted that to guarantee high quality detections, metadata providing spatio-temporal contexts on sensor measures are needed. These metadata hence become an integral component of the anomaly detection computation and need to be processed using a streaming approach. In this paper, we introduce Scouter, a generic tool that helps in capturing, analyzing, scoring and storing the contextual information of a given application domain. The processing depends on a semantic-based approach that exploits ontologies to score the relevancy of contextual information. The paper provides details on the system's architecture, describes lessons learned and introduces practical aspects.

1 INTRODUCTION

Large amounts of data generated by IoT streaming devices are continuously accumulated and processed in Big Data platforms. The analysis of these data supports intelligent software functionalities usually based on machine learning or semantic approaches. Among these features, singularities leading to anomaly detection is predominant and is tackling domains as diverse as medicine (*e.g.*, to identify malignant tumors in MRI images), finance (*e.g.*, to discover cases of credit card transaction frauds), information technology (*e.g.*, to detect hacking situations in computer networks). In the WAVES project[1], we are interested in detecting anomalies in large potable water networks managed by an international company, *i.e.*, Suez company, referred to as the domain field expert in the paper. The automatic detection of such anomalies is an important issue at both the environmental and economic levels: the volume of lost water in the world has reached more that 32 billions m^3/year (corresponding to a loss of US$ 14 billion/year) with 90% of them being invisible due to the underground nature of the network. As a matter of fact, water leaks can be detected based on pressure and flow measures retrieved from sensors installed on strategic spots within such a network. During several experiments, conducted on the French network, consisting of 100.000 km of pipelines equipped with over 3.000 sensors distributing drinkable water to more than 12 million clients, field experts found out that to guarantee high accuracy in anomaly detection, a contextualization of the measures is needed especially when a singularity appears. For instance, abnormal high pressure or peculiar flow signatures could potentially indicate a water leak. However, in many cases of popular sport or cultural events, these singularities could easily be explained because of some arrangements made by the city hall to provide water fountains. The same singularities can also account for hot weather conditions implying garden watering in a suburb area or a wildfire forcing firemen to use important amounts of water. Hence, an efficient approach for singularity contextualization has to integrate a spatio-temporal dimension on the analyzed measures.

The tasks related to the capturing, analyzing, scoring and storing of contextual metadata are demanding since they require interactions between software components that may be hard to set up, especially in a distributed environment. These components generally handle stream, natural language and ontology processing as well as the storing of complex data structures. Designed as a generic system, Scouter[2] aims at simplifying these tasks by proposing implementations of the main functionalities and by taking care of installation and configuration aspects.

2 CONTRIBUTIONS

In this paper, we present Scouter, a novel singularity contextualization system built on top of Apache Kafka that automatically proposes relevant explanations to detected anomalies using a continuous filtering approach.

Scouter proposes a powerful web analysis tool leveraging on ontology building and semantic web technologies to automatically retrieve, score and rank relevant events extracted from the web. Scouter also exposes a powerful set of natural language processing functions such as topic extraction, topic relevancy, topic matching and sentiment analysis as well as an original method for geo-profiling to classify areas. Finally, Scouter reduces the time complexity of heavy web scrapping using big data frameworks and eliminates the dataset size limitation by implementing a continuous filtering method. In summary, the main contributions of Scouter are:

- An efficient method to fetch both streaming and static data from various sources on the web based on a hierarchy graph of concept labels, henceforth denoted an ontology.
- An original geo-profiling method that gives a detailed portrayal of the areas where the events occur.

[1] https://www.waves-rsp.org/

[2] https://badrebelabbess.github.io/Scouter-1/

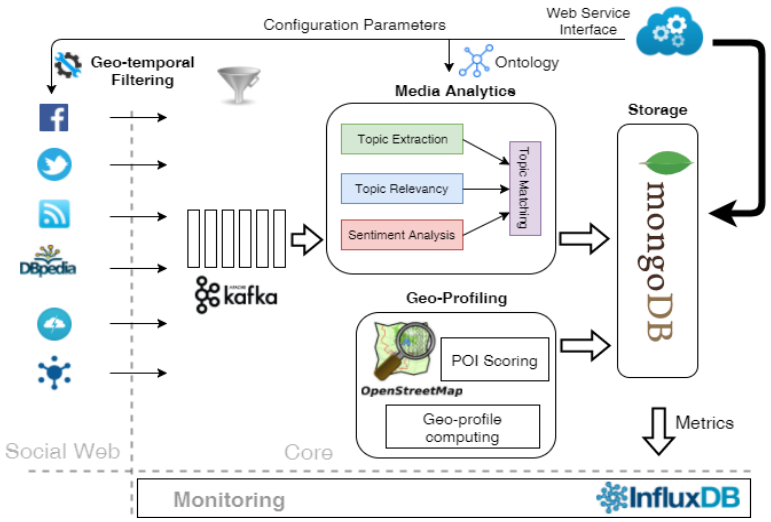

Figure 1: Scouter Architecture

- A novel approach to select relevant non duplicate events using powerful Natural Language Processing (NLP) functions in a three dimensional pipeline: topic extraction, topic relevancy and sentiment analysis.
- A continuous dynamic data filtering technique based on a combination of media analytics scoring and geo-profiling annotations.

3 ARCHITECTURE

As a component of the WAVES platform [1], Scouter was developed to be a generic system that can handle both streaming and static events and analyze them using a powerful set of NLP functions and semantic methods. Fully configurable, the primary goal of Scouter is to fetch data efficiently from different web sources, process them in a short amount of time in order to score every event regarding its capability to explain anomalies detected by the platform. As detailed by Figure 1, the main components of our system are: a set of Web data connectors, a media analytics unit, a geo-profiling unit, a storage mainframe, a messaging broker and a web service provider.

The web connectors consume data from different data sources at a certain frequency based on predefined configurations which can be specified using a web interface. These sources include social networks namely Twitter and Facebook (*e.g.*, citizens commenting on water leaks nearby) but also media sources such as RSS feeds from various newspapers listed in Table 1(*e.g.*, an article from the French newspaper Le Monde mentioning a fire), weather information from Open Weather Map (*e.g.*, climate conditions during a specific event), organized events from Open Agenda (*e.g.*, concerts, exhibitions or sporting events) and also specific information retrieved from DBpedia (*e.g.*, number of inhabitants or type of neighborhoods). All of these data sources are consumed in a powerful multi-threading mechanism using rest APIs. The concepts, subconcepts and properties used to fetch data are represented in an ontology that formalizes the different relationships in a sound manner. More details about the ontology are provided in the section 4.1.

The media analytics unit digests fetched feeds from Kafka and leverages on the Apache Spark distributed framework to analyze feeds in real-time. Feeds are recorded as events annotated with location, start/end dates and description. In order to filter the most relevant events without any duplicates in the database, a topic extraction approach and a sentiment analysis are combined to match topics. The topic extraction parses the text of feeds to discover term occurrences. Then, the scoring module takes advantage of user defined weights, *i.e.*, a real value in the [0, 1] range, associated to ontology concepts to provide an overall scoring for each text. The sentiment analysis classifies the feeds into positive or negative categories using the maximum entropy algorithm [2]. It builds a model using multinomial logistic regression to determine the right category for a given text. Simultaneously, the geo-profiling unit provides geographical characteristics for the analyzed area. It determines the type of terrain surrounding the anomaly location. Based on points of interest and terrains for a given zone, a profile is generated that describes the category of the targeted region using a configurable classification.

After the scoring step, events are recorded into a scalable and distributed document database (namely MongoDB). As a result, we obtain in real-time spatio-temporal and scored contexts that can assist the operator to explain an anomaly detected in the underground water pipeline network. Scouter also provides a metrics monitoring tool to track the performance of the system including query times, event processing times, events count and topic extraction training times. These metrics are stored in a time series database with very high read/write access (namely InfluxDB). Finally, the Web services component is used for configuring the system. It provides Rest-based interface that can be integrated with a graphical user interface to deliver configuration parameters in an user-friendly and readable way.

4 MEDIA ANALYTICS & NLP

In this section, we detail the methodology of collecting data from the various types of sources available on the web. This process is built on four major components that work together in order to:

- Extract the most relevant events based on a complex graph of concepts and properties.
- Identify the appropriate summaries from each event with the maximum expressiveness.

- Avoid duplicate events stored in the database by comparing summaries and assessing the similiarity between them using sentiment analysis.

4.1 Ontology

Every scrapping tool relies on a configuration file that lists the properties of words, concepts or events that it tries to fetch [3]. In Scouter's case, the fetching capabilities rely heavily on a pre-built ontology that lists the main concepts the user is looking for but also organizes the relations in two dimensions:

Vertical Hierarchy: A given concept (*e.g.*, Fire) can have multiple sub-concepts (*e.g.*, Blaze, Wildfire) or aliases and mispellings (*e.g.*, fir, wild-fire, blayz).

Horizontal Dependency: A concept can have multiple properties that describe a specific state in a certain time period. For example, water can be potable, but can also leak or have a specific color.

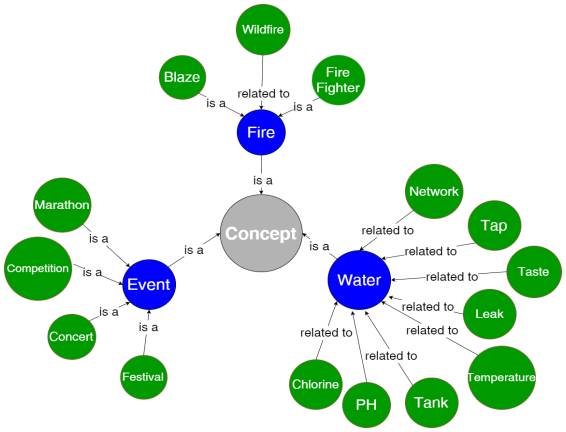

Figure 2: Ontology Overview

By combining the concepts and the properties through predicates, we can build a complex graph such as the one in Figure 2 used for the water leak use case. We can easily argue that this type of structure holds more expressiveness than a classic list of keywords exposed in a configuration file.

4.2 Topic Extraction

After fetching the proper events from the various sources based on the ontology of concepts and subconcepts, the next step is to extract meaningful topics from the events following the pipeline in in Figure 3.

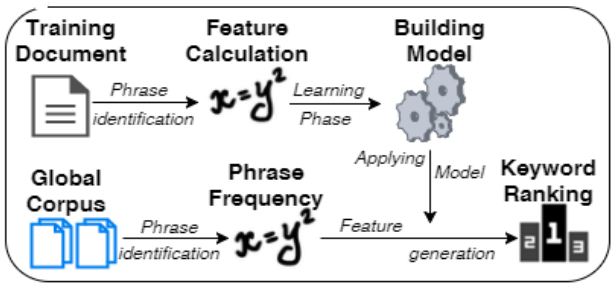

Figure 3: Topic Extraction Pipeline

The main preprocessing here concerns the cleaning of the input text, the identification of potential candidates, and finally the stemming and case-folding of the phrases. Input files are filtered to regularize the text and determine initial phrase boundaries, then the splitting into tokens alongside several modifications are made (apostrophes are removed, hyphenated words are split in two, etc). Next, we consider all the subsequences in order to determine the ones that are suitable candidate phrases. To increase the accuracy, we use a list of french stop-word list containing more than 500 words in different syntactic classes (conjunctions, articles, particles, etc). Then we case-fold all words and stem them using the iterated Lovins method [4] to discard any suffix, and repeating the process until there is no further change. Stemming and case-folding allow us to treat different variations on a phrase as the same thing.

The main processing involves two calculated features for each candidate phrase : the phrase frequency in the input text compared to its rarity in general use and the first occurrence, which is the distance into the input text of the phrase first appearance. These two features are converted to nominal data for the machine-learning process and a discretization table for each feature is derived from the training data. Finally, we generate a model that gives the scores for every candidates and ranks them using Naive Bayes techniques [5].

4.3 Topic Relevancy

Several research works tackle the issue of automatic summarization [6]. The tools proposed generally mix several classes of features such as summary likelihood, use of topic signatures or syntactic analysis [7]. In our case, we chose a direct approach based on distributional similarity that compares input and summary content. In fact, we consider that a good summary should be characterized by low divergence between probability distributions of words in the input and summary, and by high similarity with the input. For this purpose, we used two common measures: the Kullback Leibler divergence and the Jensen Shannon divergence.

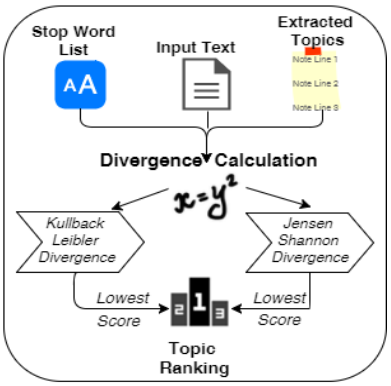

Figure 4: Topic Relevancy Pipeline

First, words in both input and summary are stemmed and separated before any computation. Then we compute the two measures:

Kullback Leibler (KL) divergence: It corresponds to the average number of bits wasted by coding samples belonging to P using another distribution Q, an approximate of P. It is given by:

$$D_{\mathrm{KL}}(P\|Q) = \sum_i P(i) \log \frac{P(i)}{Q(i)}.$$

In our case, the two distributions of word probabilities are estimated from the input and summary, respectively. Because KL divergence is not symmetric,both input summary and summary input divergences are introduced as metrics. In addition, we perform a simple smoothing using an approximating function that captures important patterns while leaving out noise and other fine-scale structures.

Jensen Shannon (JS) divergence: This one leverages on the fact that the distance between two distributions cannot be very different from the average of distances of their mean distribution. It is given by the following formula:

$$\mathrm{JSD}(P \parallel Q) = \frac{1}{2}D(P \parallel M) + \frac{1}{2}D(Q \parallel M) \quad (1)$$

$$where\ M = \frac{1}{2}(P + Q) \quad (2)$$

In contrast to KL divergence, the JS distance is symmetric and always defined. We compute both smoothed and unsmoothed versions of the divergence as summary scores. The final step is to use the output of these two functions to rank the extracted topics and keep only the ones with the best summarization score (*i.e.*, lowest divergences).

4.4 Sentiment Analysis

During the last decade, sentiment analysis has known an exponential development due to the growing usage of social networks and the popularity of websites where people can state their opinion on different products and rate them. Many solutions have been proposed and packaged in several technologies [8], we propose in this section a simple approach based on some tools provided by the Stanford CoreNLP toolkit [9].

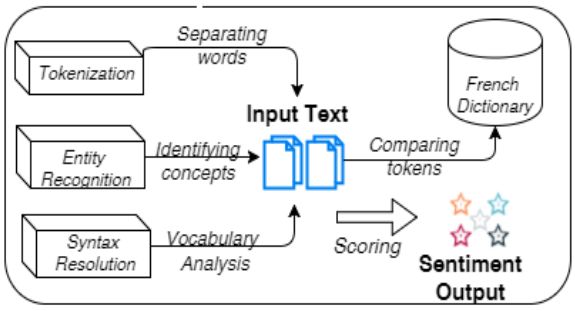

Figure 5: Sentiment Analysis Pipeline

Before applying the model, we need to launch several preprocessing steps that will improve the accuracy of the output score. Here, three steps are involved:

Tokenization: It separates the text into a sequence of tokens and saves the character offsets of each token in the input text. Then we split a sequence of tokens into meaningful sentences.

Entity Recognition: It checks if the tokens are consistent and conform to a predefined standard before trying to determine the likely gender information to names based on a dictionary. Then, the recognition algorithm annotates recognized tokens as persons, locations, organizations, numbers, dates, times or durations.

Syntactic Resolution: The system used a full syntactic analysis, including both constituent and dependency representation, based on a probabilistic parser.

Since our use case is to analyze media and social networks within the French territory, we used a French dictionary embedded in a wrapper to analyze the words.

After the preprocessing phase, we enter the main computation step where we apply the model. Among several models, we chose the compositional one over trees using deep learning. It relies on nodes of a binarized tree of each sentence, including, in particular, the root node of each sentence, that are given a sentiment score. In order to capture the sentiment of an input text, a Recursive Neural Tensor Network model (RNTN) is built based on the characteristics of the input phrases. These phrases are represented using word vectors and a parse tree, then we compute vectors for higher nodes in the tree using the same tensor-based composition function. This approach is inspired from the recursive deep models for semantic compositionality developed by Stanford[10].

4.5 Topic Matching

As stated at the beginning of this section, the goal of the different steps in the media analytics part is to extract the most relevant unique events by annotating them with a meaningful summary. Hence, we try to avoid duplicate events that refer to the same happening or occurrence. In order to complete this task with high accuracy, we are mixing several approaches relying on NLP processing from ontology building to sentiment analysis. The process follows the following steps: For each event fetched from the different sources, the topic extraction phase will propose a list of potential summaries based on a Bayesian approach. Then these summaries will be ranked using the lowest divergences (*i.e.*, KL divergence and JS divergence) in order to assess their accuracy. Among the highest ranked ones, we will check if they have the same sentiment (*i.e.*, positive, neutral or negative). If one of the selected topics during this process have the same sentiment, we assume then that they are referring to the same event in the same way. Therefore,we conclude that these events are duplicates and we only keep the content of one event. Also, we annotate the event with a reference from the other deleted event to show to the final user that this specific event is present in different sources. The overall pipeline of our media analytics module is detailed in figure 6.

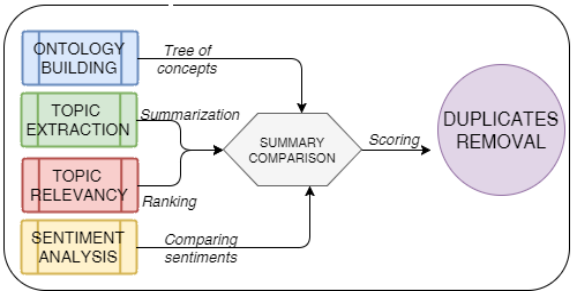

Figure 6: Topic Matching Pipeline

Even though this module provides powerful functions to filter relevant unique even, a spacial dimension is required to fully contextualize the detected anomaly. This part will be explained in the following section.

5 GEO-PROFILING

5.1 Methodology

The goal of Scouter is to provide relevant information to contextualize and potentially explain detected anomalies. The media analytics module helped in filtering the relevant events and removing duplicates, however geographical information about the area where the anomaly is spotted could further fine-tune and improve the accuracy of the context. This task is handled by two complementary modules: Geo-profiling module and reasoning module. It can be performed before the reasoning, to orientate the research of events, or after, to change the ranking of the potential sources.It is composed of two methods, that are combined and enriched with a third consumption-based method for better results.

Method 1: It extracts points of interest (POI) present in a given sector, from online geographic data sources, to determine the proportion of different types of surfaces composing it. The domain field expert selected the types of surfaces relevant to our use cases, giving us five profiling parameters: residential, natural, agricultural, industrial and touristic. Then, we created a rating file, assigning notes to each POI, in order to compute a score for each type of surface, and calculate its proportion (*i.e.*, a real value in the [0,1] range).

Method 2: It is also based on geographic data, but uses features modeled as polygons instead of POI. The inclusion tests are more complete, since some polygons may be included completely or partially inside the consumption sector. Also, the computation is not performed using the rating system, but the areas of the polygons, which are less arbitrary. Otherwise, both methods produce the same result: proportions (also as real value in the [0,1] range) for each type of surface.

Method 3: For certain types of consumption sectors, our two methods can slightly differ. To decide which method should be used in each case, we added a third method that computes what we denote as the consumption ratio. For each sector, we compute the daily flow, and make an average over a long period of time to avoid anomalies; then we divide this flow by the pipeline length on the sector to obtain the ratio. A low ratio corresponds to a sector with few consumers, such as countryside zones, a high ratio is the opposite. The program selects the best profiling using those criterion.

In case of a mixed result, we compute the average of the methods, to reflect the fact that there may be both open zones and populated locations. In Figure 7, we sum up our profiling strategies by highlighting their collaborations and the input data for each method.

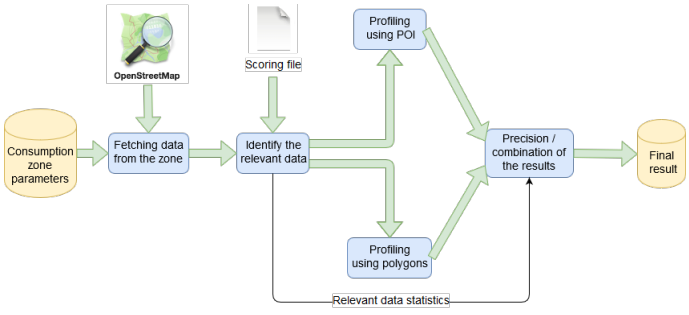

Figure 7: Profiling Overview

5.2 Tools & Resources

The geo-profiling module relies essentially on open data: the identification of the anomalies sources can only be performed with external sources, and so is their probabilistic classification.

The origins of the problems detected are generally external events that affect the quality parameters on the network. Several types of events can be responsible such as natural disasters, sport events or cultural exhibition. Some are not predictable and will be noticed by mining the proper sources, but others can be detected easily: sport events are announced a few weeks in advance, a cultural festival may be scheduled several months before. Some events can also be taken in account, even though they are not directly causing anomalies: meteorological data will help in deciding the relevancy of other events (*i.e.*, an open air festival will be more relevant as a cause of anomaly in the case of a warm weather). Static data may also provide accurate information such as Open Street Map. It has been selected because of its relative completeness compared to other online data like GeoNames. In addition to those geographic information, the Linked Open Data also provides a great amount of details on locations surrounding our water network. Some are useful to enhance the profiling or the classification of sources (*e.g.*, population density, touristic attractions or GDP), others are potential original events (*e.g.*, religious celebrations or regular typical events).

6 EVALUATION

In this section, we discuss the performance metrics used in our system, we evaluate the media analytics part on several dimensions such as the quality of events collected, then we focus on the geo-profiling module.

6.1 Media Analytics

In this setion, we refer to an experimentation collecting for 9 hours events (feeds) from all the mentioned sources namely social networks (Facebook and Twitter), newspapers (RSS feeds), Open Agenda and DBPedia for organized events, Open Weather Map for climate conditions. The target geographical area of interest is a group of cities in the suburb of Paris, France, denoted as Versailles and having a coordinates bounding box. The parameters used for each of the data sources are explained in Table 1.

Source	Fetch Frequency	Pages of Interest	Concepts Scores
Facebook	12 hours	Mon Versailles Versailles Officiel Public Events	meter:1 damage:10 concert:10 spray:1 fire:10 water:10 blaze:1 wildfire:10 flow:5 tank:1 chlore:5 pressure:5
Twitter	Streaming	@Versailles @monversailles @prefet78 #sdis78	
Open Agenda	24 hours		
Open Weather Map	4 hours		
DBpedia	24 hours		
RSS News Papers	12 hours	Le Parisien 78 Actu versailles.fr Sdis78 yvelines.gouv.fr	

Table 1: Data Sources and Concepts Scores

For instance, Twitter was run using a streaming API over all the tweets located in the bounding box, a special attention was given to extracted events from some feeds (*e.g.*, @Versailles or @monversailles) since they are official accounts of the city. The keywords used to query these tweets are a set of 12 concepts (each one having sub-concepts in the ontology) with an affected weight that scores the relevancy of the keyword.

For this evaluation, the system is evaluated in terms of the following measures:

Number of collected and stored events: It represents the number of received feeds from all the data sources during the run time. Figure 8 shows the number of events collected during the run and the number of events stored in the database that have a score higher than 0. We can clearly see that many of the collected events (around 28% for this experiment) are not relevant, therefore they will be useless for the operator as a potential explanation for the anomalies.

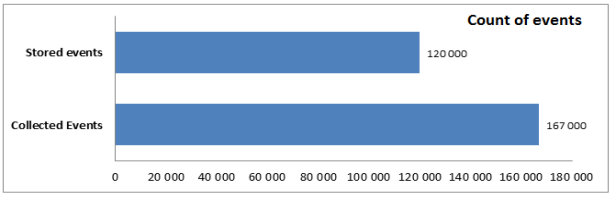

Figure 8: Collected & Stored Events for 9 Hours

Kafka Queue Messages Per Second: It represents the number of messages (events) written to Kafka broker, it can indicate the load on the messaging queue. When Scouter is running, all processors start ingesting data, then each of them will sleep until the next round after certain frequency. This explains the peak at the starting time of the figure 9, while after that, only Twitter stream feeds are being written to Kafka queue.

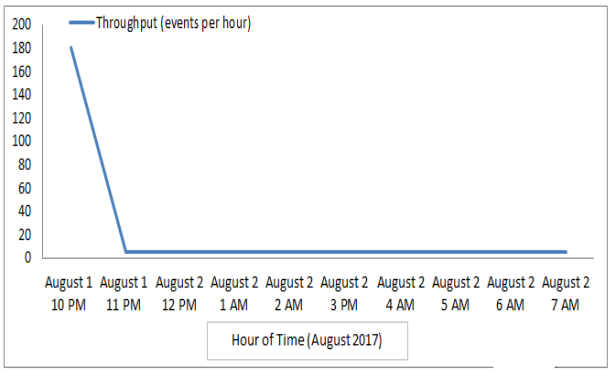

Figure 9: Kafka Throughput

Topic extraction training and Average Processing time: They represent the time spent on building the training model for detecting topics and processing the incoming events.

Table 2 shows the average time needed to score all collected events, it is calculated by dividing the sum of scoring time for each of the events by the collected events count. It also shows the training time for topic extraction algorithm to build the model that is used to detect topics. We can see that Scouter performs very well with relatively large number of events coming to the system without any failure or delay.

Measure	Time in Millisecond
Average Processing Time	7.43
Topic Extraction Training Time	474

Table 2: Scouter Processing Time

6.2 Quality of Events Collected

To evaluate the quality of the events collected, we considered the water leaks use case, where the events were fetched from the mentioned data sources over 9 hours, using the ontology in Figure 2 and the assigned score listed in Table 1. The domain expert provided the time stamp and location of all the anomalies reported on 2016 which came to 15 in total. From the database, we fetched all stored events close to the time stamp and location of each anomaly and presented them to five domain experts. For each event, they were asked if they believe that this event can give a proper and relevant explanation for the anomaly reported. A constraint was imposed, the answer had to be binary, *i.e.*, Yes or No, in order to simplify the interpretation of the results.

Table 3: Domain Experts Evaluation

Evaluator	Events														
	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
1	×	✓	×	✓	✓	✓	×	✓	×	×	✓	×	×	×	×
2	×	✓	×	✓	✓	×	×	✓	×	✓	✓	×	×	×	×
3	×	✓	×	✓	✓	×	✓	×	×	✓	×	✓	✓	×	×
4	×	✓	×	✓	✓	×	×	✓	×	✓	×	×	✓	×	×
5	×	×	×	✓	×	×	×	✓	×	×	✓	×	×	×	×

To evaluate the reliability of the annotation, we used Fleiss kappa measure [11]. It is a statistical measure that aims to assess the reliability of agreement between a certain number of annotators when assigning labels to categorical subjects. It can be interpreted as expressing the extent to which the observed amount of agreement among raters exceeds what would be expected if all raters made their ratings completely randomly. It follows the equation below with the calculated results for our 5 evaluators scenario:

$$kappa = \frac{\bar{P} - \bar{P_e}}{1 - \bar{P_e}} = \frac{0.84 - 0.5256888889}{1 - 0.5256888889} = 0.6626686657$$

$$where\ \bar{P_e} = \sum_{j=1}^{k} P_j^2 = \sum_{j=1}^{k} (\frac{1}{Nn} \sum_{i=1}^{N} n_{ij}))^2$$

$$And\ P_i = \frac{1}{n(n-1)} \sum_{j=1}^{k} n_{ij}(n_{ij} - 1)$$

P is the mean of Pi, the extent to which annotators agree for the event. Pe is the summation of squared Pj, where Pj is the proportion of all assignments which were to the j-th category. In these equations, N is the number of Events, n is the number of annotators and k is the number of categories we have (Yes and No in our use case).

Based on the table for interpreting kappa values[12], we can say that the annotators have substantial agreement on the events annotated as being a relevant explanation for a water leak anomaly. Therefore, we can say that Scouter system was effective in selecting the most relevant events and provided a substantial help to contextualize anomalies related to water leaks.

However, we are still aiming at improving the evaluation of stored events based on the following 2 key points :

- Only 5 experts participated in this evaluation which is not sufficient regarding the scale of the use case.Therefore, we are preparing a new evaluation with dozens of field experts from various companies to assess more accurately Scouter.
- Data are collected over a short period related to 2017. We would like to extend the evaluation to the anomalies reported in 2015 and 2017 in order to test the full capabilities of Scouter.

6.3 Geo-profiling

The profiling module can be executed "offline" in our system. This means that it does not need to be integrated within the stream processing. In fact, the data required as input are mostly static: geographic data are rarely updated, and the network configuration does not change much over time. We performed a series of tests on several data sets given by the domain expert on both national and international levels.

Area	# Sensors	Available OSM data (Mo)	Profiling		
			Consumption ratio (ms)	POI (ms)	Region (ms)
P. Laval	2	5.4	270	25	605
V. Nouvelle	16	53.8	620	282	630
Hubies D.	1	5.8	190	13	30
Brezin	1	3.1	150	8	72
Guyancourt	2	4.2	161	13	50
Louveciennes	19	123.2	850	1118	1290
Hubies H.	13	37.15	740	163	180
Haut-Clagny	4	8.6	260	21	32
Garches	3	7.0	220	205	46
Gobert	3	15.4	201	36	105
Satory	5	32.5	292	103	215

Table 4: Performance table of the different methods of the algorithm

The performances of the reasoning module mainly depend on the size of the data extracted from Open Street Map: larger consumption sectors will have more data taking more time in the end. The profiling with polygons is the longest since it needs the extraction of both POI and polygons to be processed. On average, the computation of the consumption ratio is the fastest, since it doesn't need any extraction from Open Street Map.

Table 4 details the performance of our different methods for the use case of the region of Versailles (an area of 350.000 inhabitants in the suburb of Paris), which is composed of 11 consumption sectors. For each of these sectors, we indicate the number of flow sensors present on each sector, and the size of the data to extract from Open Street Map (both POI and polygons), then we indicate the computing time for each method. As we can see, the regions with the more data and the more sensors have the longest processing time: it corresponds to the biggest zones.

7 CONCLUSION

In this paper, we presented Scouter, a system that demonstrated its usefulness in the WAVES project for improving the contextualization of identified anomalies. The primary goal was to offer a generic system that can adapt to any Big Data platform whatever the engine used and without losing the capability to process various types of data sources. To achieve such target, we chose an approach based on data connectors relying heavily on messaging queue system and we offer multiple NLP functionalities such as topic extraction and sentiment analysis.

Because of its easy-to-use Docker package, Scouter is already used in other prototypes at Atos and we are aiming to extend it with novel features such as ontology enrichment based on a dictionary of concepts and the identification of duplicate events coming from different data sources. Finally, we plan to improve the implementation by supporting various ontology formats (e.g. ttl, N3, RDF/XML, etc.) and adding new data sources to fit most use cases (e.g. traffic information, etc.).

Lessons Learned: During this work, the research team learned several lessons along multiple distinct dimensions. Instead of trying to adapt our implementation to heterogeneous technologies in the Big Data world, the best approach was to create a simple but powerful bridge that would make the integration seamless. The right choice was to go with a messaging queue system such as Apache Kafka that is widely used and known for its robustness. Moreover, we found out that few data sources with high quality content outperformed multiple data sources with medium to low quality content. Therefore, we decided to keep only 4 different categories and for each one, we selected the most suited source.

Since we are aiming at designing a tool adjustable to multiple use cases, going for an ontology-based approach seemed to be a win solution. Hence, we could give the possibility to every domain expert to use his own ontology with the specific concepts, properties and keywords that suit her needs. However, the key component for a successful implementation is to find the right models and the proper scores. Even tough a lot of libraries were available for NLP functions, many of them lacked robustness and flexibility. Instead of investing much time on finding the fittest technologies, we are quite positive that we can highly improve the results by investing more time on ontology modeling and scoring.

Finally, we found out that the best way to remove complexity was to package the code into a user friendly web application. Therefore, instead of asking users to plug Scouter to their framework, they would just have to enter the location of the analysis, the specific data sources alongside with the proper domain ontology. As for the deployment part, using containerization technologies such as Docker tend to remove the burden of deployment and can be launched within a minute.

8 ACKNOWLEDGMENTS

This work has been supported by the WAVES project which is partially supported by the French FUI (Fonds Unique Interministériel) call #17. The WAVES consortium is composed of industrial partners Atos, Suez and Data publica, and academic partners ISEP and UPEM.

REFERENCES

[1] H. Khrouf, B. Belabbess, L. Bihanic, G. Képéklian, and O. Curé, "WAVES: Big Data platform for real-time RDF stream processing," in *3^{rd} Stream Reasoning workshop co-located with 15^{th} International Semantic Web Conference, Kobe, Japan.*, pp. 37–48, 2016.

[2] S. D. P. Adam Berger and V. D. Pietra, "A maximum entropy approach to natural language processing," *Computational Linguistics-MIT*, pp. 22–1, 1996.

[3] S. de S Sirisuriya, "A comparative study on web scraping," *International Research Conference*, nov 2015.

[4] J. Lovins, "Development of a stemming algorithm.," *Mechanical Translation and Computational Linguistics*, 1968.

[5] P. Domingos and M. Pazzani, "On the optimality of the simple bayesian classifier under zero-one loss," *Machine Learning*, 1997.
[6] S. Ellouze, M. Jaoua, and H. Belguith, "Machine learning approach to evaluate multilingual summaries," in *Proceedings of the MultiLing 2017 Workshop on Summarization and Summary Evaluation Across Source Types and Genres*, April 2017.
[7] C.-Y. Lin, "Rouge: A package for automatic evaluation of summaries," in *Proc. ACL workshop on Text Summarization Branches Out*, 2004.
[8] D. J. O. H. A. Collomb, C. Costea and L. Brunie, "A study and comparison of sentiment analysis methods for reputation evaluation," tech. rep., Mar. 2014.
[9] C. D. Manning, M. Surdeanu, J. Bauer, J. R. Finkel, S. Bethard, and D. McClosky, "The stanford corenlp natural language processing toolkit.," in *ACL (System Demonstrations)*, The Association for Computer Linguistics, 2014.
[10] R. Socher, A. Perelygin, J. Wu, J. Chuang, C. D. Manning, A. Ng, and C. Potts, "Recursive deep models for semantic compositionality over a sentiment treebank," in *Proceedings of the 2013 Conference on Empirical Methods in Natural Language Processing*, Association for Computational Linguistics, October 2013.
[11] J. Fleiss *et al.*, "Measuring nominal scale agreement among many raters," *Psychological Bulletin*, vol. 76, no. 5, pp. 378–382, 1971.
[12] J. R. Landis and G. G. Koch, "The measurement of observer agreement for categorical data," *Biometrics*, vol. 33, no. 1, pp. 159–174, 1977.