Prog Artif Intell (2014) 2:113–127
DOI 10.1007/s13748-013-0040-3

REGULAR PAPER

Event labeling combining ensemble detectors and background knowledge

Hadi Fanaee-T · Joao Gama

Received: 2 August 2013 / Accepted: 24 October 2013 / Published online: 26 November 2013

Abstract Event labeling is the process of marking events in unlabeled data. Traditionally, this is done by involving one or more human experts through an expensive and time-consuming task. In this article we propose an event labeling system relying on an ensemble of detectors and background knowledge. The target data are the usage log of a real bike sharing system. We first label events in the data and then evaluate the performance of the ensemble and individual detectors on the labeled data set using ROC analysis and static evaluation metrics in the absence and presence of background knowledge. Our results show that when there is no access to human experts, the proposed approach can be an effective alternative for labeling events. In addition to the main proposal, we conduct a comparative study regarding the various predictive models performance, semi-supervised and unsupervised approaches, train data scale, time series filtering methods, online and offline predictive models, and distance functions in measuring time series similarity.

Keywords Event labeling · Event detection · Ensemble learning · Background knowledge

1 Introduction

Event labeling is recognized as a basic function in surveillance and monitoring systems. Labels are essential for evaluation of the algorithms and for incorporation in real time systems. However, event labeling is an expensive and time-consuming task which requires synergy of one or more human experts. Several solutions have been developed to avoid performing human-based labeling. The first group of methods relies on the creation of artificial and simulated data [31,38,42,44] so that both normal and abnormal instances are generated via simulation. In the second group, events are injected on real background data [8,28]. However, the ideal scenario is to have access to ground truth data [35] where both normal and abnormal instances are labeled without simulation. The first and second solutions suffer from two issues. Firstly, they do not reflect the reality [41] and secondly it is extremely difficult to develop a simulator that generates data close to the ground truth [19]. Besides, availability of ground truth data is limited or has been under some criticisms (e.g. [37,50]).

Regardless of learning methodologies, evaluation of event detectors is still highly dependent on human efforts. In supervised event detection, both normal and abnormal instances are required to be labeled by human experts. In semi-supervised approaches normal instances should be labeled by humans. In unsupervised methods, detected events are required to be verified by human experts. However, in practice, labeling or verification of events by human experts can be extremely time-consuming and expensive [41]. In order to solve this problem some efforts have been made to assist users to label data more efficiently via a graphical user interface [41]. However, such methodologies still are human-dependent to a great extent.

An automatic event detection system ought to operate without intuitive dependency on human resources neither in providing labeled data nor in verification of alarms. One alternative for human knowledge can be computer-based knowledge resources. Although there is a lot of non-human knowledge sources, however, unfortunately still a large number of

H. Fanaee-T (✉) · J. Gama
Laboratory of Artificial Intelligence and Decision Support,
INESC Porto, University of Porto, Campus da FEUP,
Rua Dr. Roberto Frias 378, 4200-465 Porto, Portugal
e-mail: hadi.fanaee@fe.up.pt

J. Gama
e-mail: jgama@fep.up.pt

event detection systems rely on human-based knowledge. For instance, available knowledge inside search engines, meta-image and meta-video databases, data archives, data warehouses rarely is incorporated into event detection problems. Only a few researches addressed this issue. For instance, [3] use background knowledge from reliable sources of information (e.g. CNN, BBC) for matching and validation of detected events on Twitter streams. Collier et al. [12] use an event ontology, called BioCaster, as background knowledge for detection of infectious disease outbreaks from linguistic signals on the web. SanMiguel et al. [43] present an approach for creation of domain ontology as background knowledge for detection of events in surveillance video. Xu et al. [55] use web cast text as background knowledge source for detection of events in sport videos. However, in none of the above works, the target has not been the eliminating of human role from the detection cycle, rather background knowledge is used as a supportive role.

The essential tool for elimination of the human role from the detection cycle is the possession of a highly reliable detector. However, different event detectors are not equally capable of detecting events and subsequently different detectors perform differently on different environments [2,48,49,52]. This is due to the inconsistency of detector performance [22,52]. But how can we overcome this difficulty? When we want to make an important decision in our daily routine, we probably ask for recommendations from different peoples from *different perspectives*. The similar idea is already broached in the machine learning which is entitled *ensemble learning*. Ensemble learning is robust solution for more accurate and relatively domain-independent classification and clustering [2,16]. It also can be embedded in parallel computing paradigm to improve the efficiency [47]. However, the application of ensemble methods in event detection has received a little attention in the research literature [20,23,30,45,53] while theoretically it is believed that combining different detectors should provide a better anomalous space coverage [20,49].

There are few works in the literature [2] that adapted ensemble methods to event detection. The first work [4] applies multiple classifiers for anomaly detection from real network traffic data. The authors showed that a few judiciously selected classifiers outperform many diverse classifiers. They propose a method called standard deviation normalized entropy of accuracy as a strategy for combining the classifiers. In another work [20] authors combine four diverse anomaly detectors for automated event labeling of network traffic data and create a data with ground truth. The strength of their approach relies on the synergy between detectors with different granularity. However, it is not specified as to how data can be considered as ground truth while not validated by an external knowledge source or human expert. Besides, the role of randomness and chance is not considered in the combination of the outputs of detectors.

In this work we aim at development of an approach for labeling and detection of events in unlabeled data by exploiting a combination of both ideas of ensemble learning and background knowledge. This approach has two main applications. Firstly, it can be used for creating benchmark data sets for evaluation of event detection algorithms and secondly can be used in a real world event detection problem when data nature is unknown and there is no access to human experts for labeling of data or verification of alarms.

In parallel to our main contribution, we perform a comparative study on different important issues in event detection such as learning strategy composed of unsupervised and semi-supervised, scale analysis, multiple denoising approaches, offline and online regression models and distance function in time series similarity estimation.

The rest of the paper is organized as follows: The next section identifies the main concepts for event detection and introduces the proposed model. The Sect. 3 presents a case study using a real data set, discusses the obtained results and presents a sensitivity analysis. The last section concludes the exposition presenting the final remarks.

2 Proposed solution

2.1 Definitions

The central concept in this paper is related to *event* that there is as yet no formal agreed definition about that in the literature. It is sometimes interpreted as a sub-category of anomaly [11] or in some circumstances equivalent to anomaly [29] or change [24]. Several definitions exist in different contexts. However, more appropriate definition that can distinguish event from anomaly, outlier, change or other equivalent terms is the definition with emphasis on spatial-temporal dimension of an event [13,36]. We have to note that in this paper, even though we do not conduct a spatiotemporal analysis on data, each event of interest implies an occurrence of something in a specific place (e.g., Washington, D.C.) and a time period (e.g., 2012/05/16). In the following, we define and distinguish some of the concepts used in the paper.

Definition 1 An *Event* is something that happens in space and time and creates change in the environment under study.

Definition 2 *Event labeling* is the process of marking events in unlabeled data.

Definition 3 An *Event Detector* (or *Detector*) is a method or algorithm that discriminates events from non-event instances.

Definition 4 An *Ensemble Detector* is a group of event detection algorithms that assign a score to each instance of

data. The score usually represents the chance of that instance not be an event.

Definition 5 *Background Knowledge* is a sort of knowledge that cannot be used directly in the training phase due to privacy, computational complexity or competitive reasons but can be queried directly or indirectly. Some examples of Background knowledge sources are as follows.

- *Homogeneous sources* This category may include data archives, data warehouses, domain ontologies and other homogeneous sources. The assumption in this category is that we have computational limitations for dealing with big data sets. However it is assumed that it is possible to query the higher scale data set through an efficient DBMS gateway.
- *Heterogeneous sources* Heterogeneous sources differ in nature with train and test sets. The well-known example is the World Wide Web. There is huge heterogeneous information available on the web that cannot be integrated in the learning process because of both volume and competitive issues. But a direct query or query over API is possible over these sources. Our work is concentrated on this type of knowledge sources. We use existing knowledge inside Google™web and image search and YouTube™for verification of detected events.
- *Confidential sources* Sometimes due to the privacy or security matters is not possible to have access to whole database. However, the third party provide secure gateway to perform a limited queries over the databases.

2.2 Event labeling model: a proposal

There are two classic event labeling models that rely on human-based knowledge. In the first model (Fig. 1a) a desired detector is applied to the data and then detected events are verified by one or more domain expert(s) [7,15,32,34]. In this model, checking of all instances is not essential; rather a limited number of candidates are finally verified by the expert. This model has two main drawbacks: on one hand there is no guarantee that the detector algorithms work well on that particular data set and could detect all potential events and on the other hand in the evaluation phase is not possible to measure the accuracy of the detector.

In the second model (Fig. 1b), all instances are checked individually by knowledge expert(s) and events are labeled manually [5]. This model has also three main drawbacks: firstly, it is an infeasible task for large databases to check instances individually. Secondly, the opinion of one expert may not be sufficient and affects the labeling quality. Finally, different experts have different perspectives and therefore it is hard to assume that they have the same agreement on the event labels [54].

A recent automatic model is proposed in [20] which does not rely on human-based knowledge. As depicted in Fig. 1c, output of ensemble detectors is combined based on the detec-

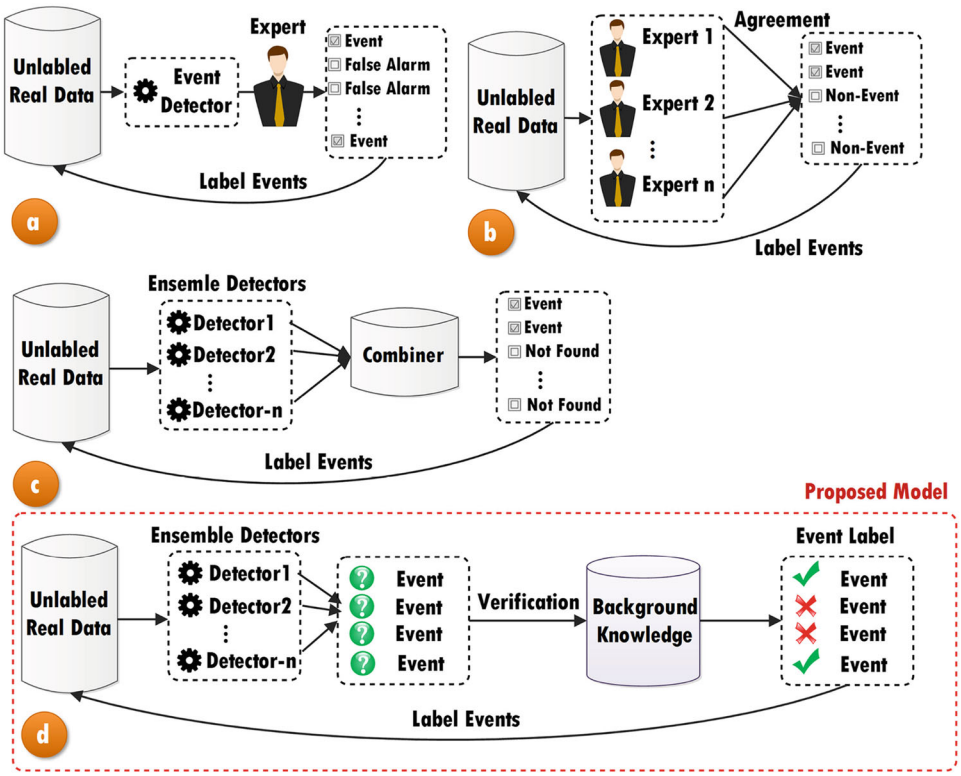

Fig. 1 Event labeling models. **a** Domain expert(s) verify the alarms raised by a single detector. **b** Domain expert(s) label the instances by manual inspection. **c** Events are labeled by applying an ensemble of detectors. **d** Proposed model: extension of previous model with this difference that alarms are verified by background knowledge before labeling

tors' output similarity. The drawback of this method however is that the result is highly dependent on the detectors' selection and no knowledge source (human nor machine) validates the outputs of detectors. Therefore false alarms might be raised, more than expected, due to not considering chance and randomness.

We extend the third model to a new model (Fig. 1d) which uses potential knowledge resources for verification of alarms. It also has no dependency on human-based knowledge and is less dependent on the methods. It is also less affected by randomness. Since it is based on ensemble detectors, it is also capable of working in parallel computing framework and thus can be computationally efficient. Based on these explanations we define our research hypotheses as follows.

Hypothesis 1 Ensemble detectors improve the detection performance comparing individual detectors.

Hypothesis 2 Background knowledge along with ensemble detectors improves the performance of event detection systems.

In the following we try to examine and validate the above hypotheses through comprehensive experimental study and evaluation tasks.

3 Experimental evaluation

There are several public data sets for outlier and anomaly detection. However it is difficult to find a real data set that for each instance, the corresponding environmental data and background knowledge are available. The most challenging part is the background knowledge which hardly can be found as an open access source. For this reason many data sets used in the outlier and anomaly detection literature are not so useful for our research goals. However, we could manage to find a data set that has a potential to be adapted for both above-mentioned issues. In the following we first describe this data set and then explore the concepts related to the method.

3.1 Data set

The data set under study is related to 2-year usage log of a bike sharing system namely Captial Bike Sharing (CBS) at Washington, D.C., USA. There are three reasons why we think this data set may fit our research goal. Firstly, it includes at least two of full life-cycle of the system and therefore seems be suitable for supervised and semi-supervised learning. Secondly, there exist some external sources that corresponding historical environmental values such as weather conditions, weekday and holidays are extractable. And finally the alarms are verifiable through open access knowledge sources (search engines, meta-image and meta-video sources).

Bike sharing systems are new generation of traditional bike rentals where whole process from membership, rental and return back has become automatic. Through these systems, user is able to easily rent a bike from a particular position and return back at another position. Currently, there are about over 500 bike-sharing programs around the world which is composed of over 500 thousands bicycles [40]. Today, there exists great interest in these systems due to their important role in traffic, environmental and health issues. Presently, the top three bike-friendly countries are Spain (132 programs), Italy (104 programs) and China (79 programs) [40]. The number of major cities that are becoming bike-friendly is growing day-by-day. It is expected that in a near future, most major cities provide this service along their other public transport services.

Apart from interesting real world applications of bike sharing systems, the characteristics of data being generated by these systems make them attractive for the research. Opposed to other transport services such as bus or subway, the duration of travel, departure and arrival position is explicitly recorded in these systems. This feature turns bike sharing system into a virtual sensor network that can be used for sensing mobility in the city. Hence, it is expected that most of important events in the city could be detected via monitoring these data. Some few researches have already addressed bike sharing data analysis [6,51] mostly via spatiotemporal analysis to aid operation-oriented decisions. However, our work differs from such works. In this paper, our main concentration is not specifically on bike sharing data, rather we use bike sharing data as a supportive source for examining our event labeling model.

In the CBS system when a rental occurs, the operation software collects basic data about the trip such as duration, start date, end date, start station, end station, bike number and member type. The historical data set of such trip transactions is available online via [9]. To avoid trend issues, we select only corresponding data to years 2011 and 2012 consisting of 3,807,587 records. Later, we aggregate the data into two scales of hourly and daily. The hourly time series includes 17,379 h and the daily time series includes 731 days. Next, we divide both daily and hourly scale time series into two sets of 2011 (train) and 2012 (test). The test set is illustrated in both scales in Fig. 2 (daily scale) and Fig. 3 (hourly scale).

As we discuss later, if we apply a regular anomaly detection algorithm on the daily or hourly time series we would not be able to detect all events. We only can detect severe events because bike rental process is probably under effect of seasonality and environmental settings such as weekday, holiday, temperature, precipitation, wind speed, humidity, etc. Therefore, event signature cannot be directly observed in these time series. In order to study such effects we need

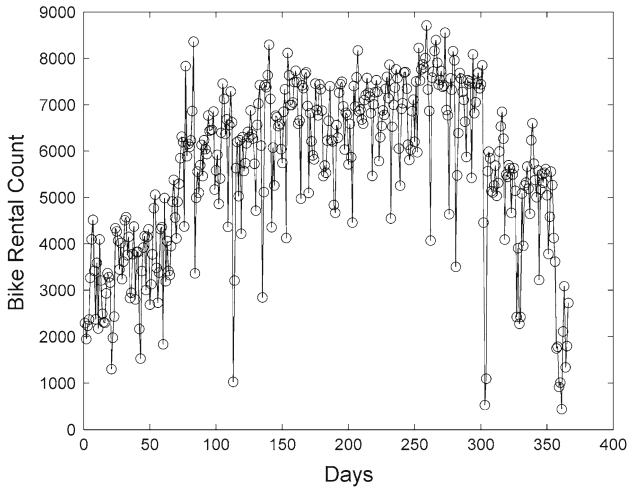

Fig. 2 The number of rented bikes in 2012 in daily scale

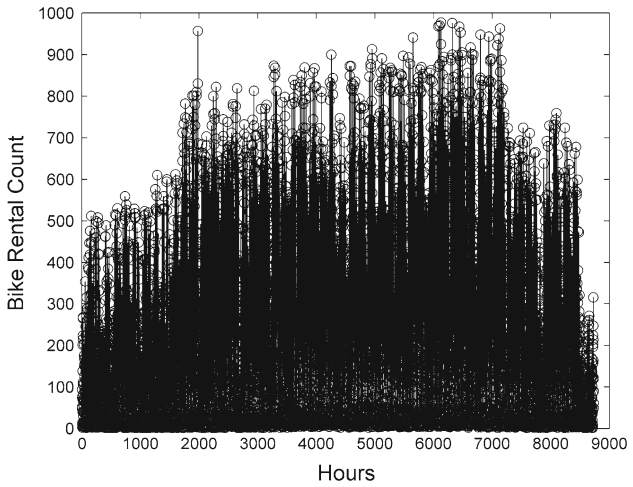

Fig. 3 The number of rented bikes in 2012 in hourly scale

to extract weather data. There exists several weather data sources, however, most of them provide only forecasting data and do not contain historical weather reports. There is another group of forecasting sources that contain historical weather reports for specific last days (e.g. 14 days). Another group also contains weather historical report but in daily scale. However, we could manage to find a source that provides the hourly historical data [21]. We therefore, extract from this source some attributes such as weather temperature, apparent temperature, wind speed, wind gust, humidity, pressure, dew point and visibility for each hour from the period 1 January 2011 to 31 December 2012 for Washington, D.C., USA. Next, we map each hour in bike rental time series with corresponding weather reports. There are some missing weather reports for some hours. Thus, we map the closest report for that hour. The maximum temporal difference is 292 min, with mean of 3 min and standard deviation of 14 min. We also extract the official holidays of Washington, D.C. from [14] and map them to the corresponding dates. Afterward, holidays are combined with weekends such that finally each day is classified as a working day or non-working day. Additionally, according to weather conditions provided in the weather data, we mark each hour by four weather grades: good, cloudy, bad and very bad.

As a result we create two sets in two scales. In the hourly scale set, each record includes hour, month, working day, season, weather grade, temperature, filling-temperature, humidity and wind speed as variables and hourly aggregated count of rented bikes as target value. In daily scale, each record consists of month, working-day, season, daily average weather-grade, daily average of temperature, daily average of filling-temperature, daily average of humidity and daily average of wind-speed as variables and daily aggregated count of rented bikes as target value.

We then perform a feature selection step on the data set to identify the most significant features. As a result, month, hour, working day and temperature are selected as most important features for hourly scale and month, working-day and temperature are selected as final features for daily scale. The final processed data set is available online via [17].

3.2 The proposed method

Our event labeling system is depicted in Fig. 4. We first apply ensemble detectors (see Sect. 3.2.1 for details of detectors) with highest possible disagreement rate. To assess the degree of disagreement of the detectors, we perform Fleiss' Kappa test (see Sect. 3.2.3). The more disagreement on alarms results in more coverage on anomalous space. In addition, more alarms increase the chance of false alarms and thus alarms are required to be validated by an external knowledge source. We run the detectors and combine all alarms to make a candidate events list. The output list is much limited comparing whole instances in data set and therefore imposes lower cost for verification. Then we combine all outputs together by adding distinctive instances together. In the next step we verify each candidate via Google web and image search and YouTube (we choose Google due to its prominent coverage). The verification phase works as follows: a spatiotemporal query is submitted to Google (Fig. 5). If an important event is detected from the result we mark the date with that event. For instance by querying 2012-10-30 Washington, D.C" we notice that the Sandy storm has happened on this date. So this date is marked as "Sandy". If we could not find any significant event from the search result we try Google images and YouTube or try another query this time including the keyword "weather" (e.g. "2012-10-30 weather Washington, D.C"). Note that due to the relatively small volume of our data set we did not perform some text processing steps. However, in a fully automatic system we could process the retrieved textual result and count the most repeated terms.

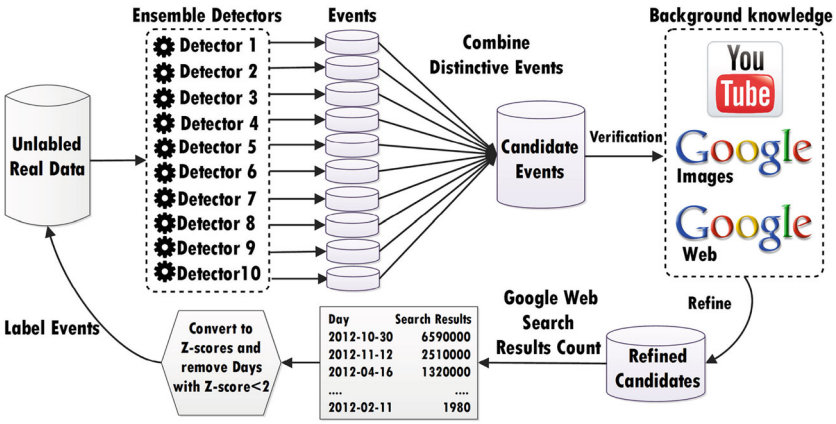

Fig. 4 Our event labeling system

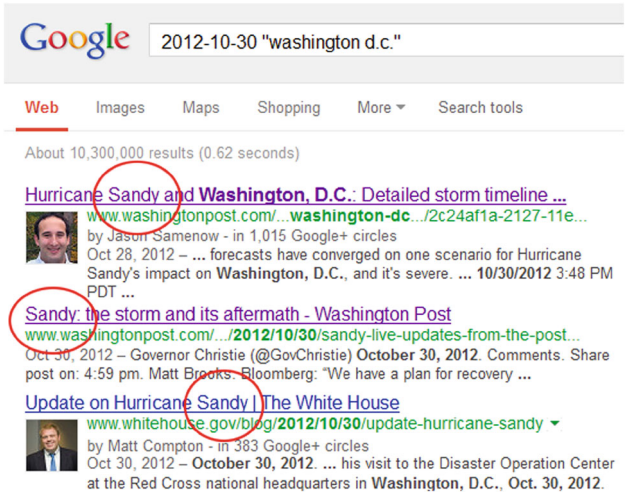

Fig. 5 A query with format of date + place is submitted to Google web/image search and YouTube for understanding occurred event

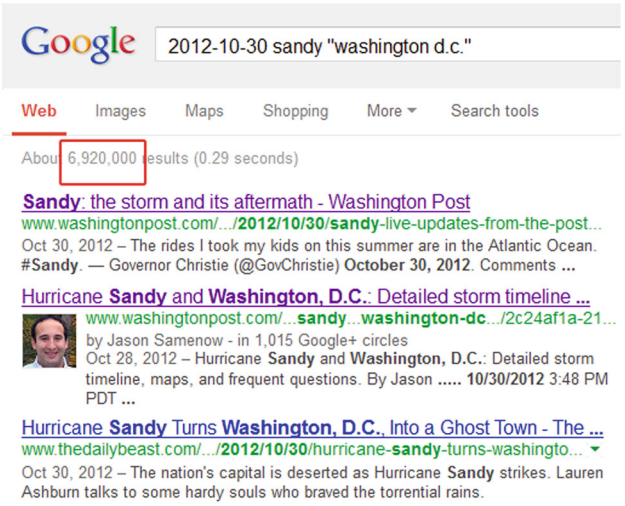

Fig. 6 After understanding event, the query with format of date + event + date is submitted to Google to measure the weight of event

In the next phase we compute the weight of the event by a method similar to [56] where search results count is used as a criterion for extraction the correlation between two terms (e.g. food + shopping vs. food + drink). The search results count itself is meaningless; however, it would be considered as an appropriate criterion for comparative purposes. For instance if query "food + shopping" result in one million and "food + drink" leads to five million pages, then it reveals that food is more correlated to drink than shopping. We adapt this idea to measure the weight of candidate events. To this end, as Fig. 6 shows, we add event title (e.g. "sandy") to the previous query and then extract the count of retrieved results. For instance, as is depicted in the figure, 6.920.000 results are returned for this query. This can be used as a criterion to measure the weight of the event. After obtaining this weight for each event candidate we transform all weights to their corresponding z-scores. Suppose the vector $x = (w_1, w_2, \ldots, w_n)$ of obtained weights from Google result count. z-score corresponding to each weight is obtained by the following equation.

$$z\text{-score} = \frac{w - \mu}{\sigma} \tag{1}$$

where w is the obtained weight, μ is the mean and σ is standard deviation of vector x.

Then we remove from candidate list those events whose z-score is lower than 2. In other words, we keep only those events that there is a low probability that be produced by chance. After this filtering step, the final list contains the event labels.

3.2.1 Event detectors

Although at first glance, data looks like a time series. However, based on our prior knowledge we can argue that it is rare that someone rides a bicycle in some circumstances such as midnight, heavy rains and very cold or very hot weather. Conversely, it is very likely that people rent more bikes in the peak working hours or in good weather conditions in weekends. In short, it seems that rental count would have a close relationship to environmental settings. To validate this

hypothesis, we design our detectors such a way that could support both of these perspectives. In other words, in some detectors we assume that data is a time series with auto-correlated instances (unsupervised detectors) and in other detectors we assume that instances are temporally independent and are correlated to some environmental settings (semi-supervised detectors). However, we give more weight to the latter detectors since they are more reasonable based on our prior knowledge. Please note that the term *semi-supervised* should not be confused with its equivalent in classification or anomaly detection where classes or anomalies labels are specified in the train data. We here deal with the count time series, therefore, when we use the term *semi-supervised* we refer to a scenario in which we have access to each instances corresponding environmental settings in the train set and not the class labels.

Ten detectors are designed in this study such that each has its own distinctive ability. Different techniques are involved such as regression trees, control chart, hierarchical clustering [25, p. 520] with two different distance functions of Euclidean and Dynamic Time Warping (DTW) [46]. We also employed Principal Component analysis (PCA) [1] and Multi-channel Singular Spectral Analysis (MSSA) [39] for denoising purpose in some detectors. Schematic representation of the detectors is presented in Fig. 7. For semi-supervised detectors, we make a predictive model from the train set based on environmental and periodicity setting (To ease the further explanations, from now on, when we refer to environmental setting we mean both environmental and periodicity settings) and then make a forecast on the test set and then compare the predicted bike rental count with the actual bike rental count in test set and then monitor the residuals to detect events. In some detectors we also apply a filter for denoising data. For unsupervised detectors, we monitor test time series irrespective of the environmental settings.

As already mentioned, the data set is made in two scales: hourly and daily. If we perform analysis only on daily scale

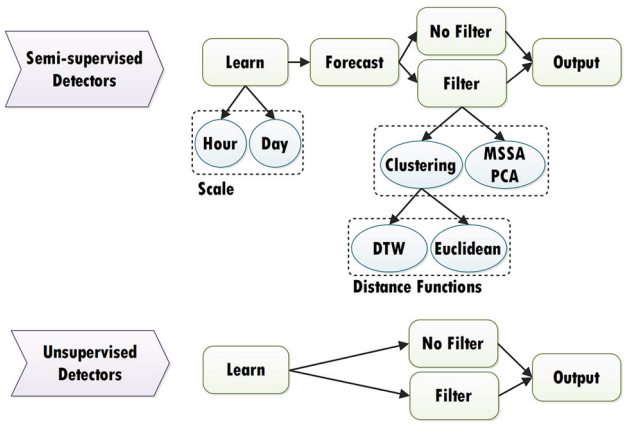

Fig. 7 A general architecture of the ensemble detectors. See Table 1 for more details

we would not be able to detect those events that affect the city only in specific hours during the day. Such events are also interesting and need to be detected. For instance, suppose that in 12/05/15 a severe event is happened during 8 a.m. and in the rest of the hours, we witness a calm day. The daily scale analysis probably would not be able to detect such kind of events, because some events manifest themselves in hourly scale.

In order to provide a unit output, alarms in hourly scale are upgraded to the daily scale (e.g. detector 10). For instance in the above example, *12/5/15, 8 a.m.* is transformed to its corresponding higher scale *12/05/15*. In this case all the detectors despite of different scale inputs generate the same output and their outputs can be combined. Each method finally returns the corresponding p values of each day. This p value indicates the probability of that day not be an event. So if we determine a threshold like 0.05 then each instant with p value lower or equal to 0.05 should be reported as an event.

In the following, each individual detector is described in detail. Note that the selected methods for detectors are optional and can be replaced with any other desired methods. However, we take into account two factors in our ensemble detectors architecture (Fig. 7). Firstly, well-known but different techniques be involved to make the maximum diversity and secondly, be more appropriate for automatic settings.

- Detector 1: The predictive model predicts the expected value on the hourly test set according to the corresponding environmental setting. Then the residuals of hourly expected value and hourly actual values on test set will be transformed to z-scores. Next, we compute the daily mean of z-scores for each day and again transform the obtained daily means to z-scores and consequently to p-values.
- Detector 2: The predictive model predicts the hourly expected value on the hourly test set according to the corresponding environmental setting. Then we compute the mean of hourly residuals for each day. Afterward, the computed daily means are transformed to z-scores and consequently to p-values for each day.
- Detector 3: The predictive model makes a forecast for the daily test set according to the corresponding environmental setting. Then the daily residuals are computed as a difference between daily predicted counts and daily actual counts. Then the residuals are transformed to z-scores and consequently p-values.
- Detector 4: This method does not need the train data. It operates directly on the daily test set. The count corresponding to each day is transformed to first z-scores and consequently p-values.
- Detector 5 and 7: This method operates as follow. First, The predictive model makes a forecast for the hourly test set according to the corresponding environmental setting and then we compute the residuals as difference between hourly predicted count and hourly actual count. Next,

matrix of $Days \times Hours$ is built such that each cell represents the residuals corresponding that day and hour. In the next step, MSSA (Method 5) or PCA (method 7) is applied on this matrix. The result is a reconstructed matrix. Later, the residual corresponding each day of original and reconstructed matrix is transformed to z-scores and consequently p-values.

- Detector 6: This method does not need train data. The hourly test data is converted to matrix of $Days \times Hours$. Afterward, MSSA is applied on this matrix and then residuals corresponding to each day of original and reconstructed matrices are transformed to z-scores and consequently p-values.
- Detector 8 and 9: The predictive model predicts the expected value on the hourly test set according to the corresponding environmental setting. Then the residuals of hourly expected value and hourly actual values on test set are clustered using agglomeration hierarchical clustering algorithm one time with Euclidean distance and one time with DTW distance. Outliers are then are chosen using a manual inspection and are reported as events. Note that this kind of approach is not appropriate for automatic detection and is only provided here for comparison to the other approaches.
- Detector 10: The predictive model predicts the expected value on the hourly test set according to the corresponding environmental setting. Then the residuals of hourly expected value and hourly actual count on test set are transformed to z-scores. Reported z-scores are still in hourly scale so we select the maximum obtained z-scores for each day. This z-scores corresponding each day are then transformed to p-values and is reported.

3.2.2 Detectors settings

Table 1 illustrates the settings used for each detector. All detectors except detector 4 and 6 are semi-supervised. For semi-supervised methods we apply REPTree regression tree as our predictive model (see Sect. 3.2.4 for justification). As it can be seen in Table 1, even though some detectors receive hourly train set as input, they score events in the daily scale. Three of detectors (5, 6 and 7) also use a filtering strategy such as MSSA and PCA for data denoising. Two of detectors (8 and 9) that are based on agglomerative clustering can only detect events and are not able to score each instant. In detector 8, Euclidean and in detector 9, DTW distance is used. Note that these clustering-based detectors generally are not appropriate choice for automatic settings since need some prior knowledge for parameter setup. However, we include them in ensemble to have more diverse detectors.

3.2.3 Fleiss' Kappa test

The benefit of combining different detectors relies on diversity among detectors ensembles [20]. Hence, an ideal ensemble detectors is required to include a sort of diverse and different detectors. In order to ensure about the detectors right choice we apply an agreement test on detectors outputs to measure the disagreement rate of the detectors (the more disagreement, the better).

Since we want to evaluate the overall agreement rate between all detectors and not individual agreements between pairs of detectors, we cannot use the common Cohen's kappa [10]. Instead we use Fleiss's kappa [18] which is a statistics that measures the level of agreement between multiple raters when assigning categorical ratings to a number of items or classifying items. It is considered as an extension of Cohen's kappa statistic that works for multiple raters. If a fixed number of raters assign numerical ratings to a number of items then the kappa reveals the consistency of ratings. Fleiss's kappa is always between 0 and 1. Table 2 shows how K values can be interpreted [33].

So far, Fleiss's kappa has been used in psychology and bio-informatics for measurement of agreement of different human agents on a subject. Here we use it for measuring the rate of agreement between multiple event detectors. Opposed

Table 1 Event detectors settings

Detector	Type	Predictive model	Train	Filter	Test	Comment
1	Semi-supervised	REPTree	2011, Hour	No	2012, Day	–
2	Semi-supervised	REPTree	2011, Hour	No	2012, Day	–
3	Semi-supervised	REPTree	2011, Day	No	2012, Day	–
4	Unsupervised	–	–	No	2012, Day	–
5	Semi-supervised	REPTree	2011, Hour	MSSA	2012, Day	3 RCs
6	Unsupervised	–	–	MSSA	2012, Day	3 RCs
7	Semi-supervised	REPTree	2011, Hour	PCA	2012, Day	3 PCs
8	Semi-supervised	REPTree	2011, Hour	No	2012, Day	Agglomerative clustering ($K = 5$), distance = euclidean
9	Semi-supervised	REPTree	2011, Hour	No	2012, Day	Agglomerative clustering ($K = 5$), distance = DTW
10	Semi-supervised	REPTree	2011, Hour	No	2012, Day	–

Table 2 Fleiss'es Kappa interpretation

K range	Interpretation
$K < 0$	Poor agreement
$0 \leq k \leq 0.2$	Slight agreement
$0.2 \leq k \leq 0.4$	Fair agreement
$0.4 \leq k \leq 0.6$	Moderate agreement
$0.6 \leq k \leq 0.8$	Substantial agreement
$0.8 \leq k \leq 1$	Perfect agreement

to the psychology and bio-informatics that a k closer to 1 is more desired, we want a closer value to zero. Because we look for a group of non-similar detectors that could detect more range of events. If all detectors for instance agree and have k equal to 1 then it means that use of multiple detectors is meaningless and one detector is enough. In contrast, if k could approach zero it means that idea of using ensemble detector is helpful and result in better coverage of discovery of unknown events.

After the experiment we obtain Fleiss's kappa equal to 0.0034. That means that the ensemble detectors exhibit a very slight agreement. In other words, if we define H0 hypothesis as *the observed agreement is accidental* we cannot reject the hypothesis due to the observed low agreement. This is the desired result since it reveals that selected detectors exhibit high degree of diversity.

3.2.4 Predictive model selection

One of the main components of the introduced semi-supervised detectors is the predictive model. We apply some different regression and classifier algorithms in Weka [26] on the train data to measure the accuracy of the built model. Table 3 illustrates the comparison of the models in terms of correlation coefficient, relative absolute error (RAE), root relative squared error (RRSE), train time in seconds and test time in seconds. Train time is the time that takes to model be built on the train set and test time is a time that takes model be evaluated through tenfolds cross validation. As it can be seen, among all, REPTree [26] has a better performance in terms of the trade-off between accuracy, train and test time. It is from the regression tree family and thus presents interpretable model. IBk, and decision table both provide relatively the same accuracy but with higher test time. Therefore, we select REPTree as the predictive model in the detectors. RRSE, RAE and correlation coefficient in Table 3 also can be calculated using the following equations [26, p. 180].

$$\text{RAE} = \frac{|p_1 - a_1| + \cdots + |p_n - a_n|}{|\bar{a} - a_1| + \cdots + |\bar{a} - a_n|} \tag{2}$$

$$\text{RRSE} = \frac{(p_1 - a_1)^2 + \cdots + (p_n - a_n)^2}{(\bar{a} - a_1)^2 + \cdots + (\bar{a} - a_n)^2} \tag{3}$$

$$\text{Correlation coefficient} = \frac{S_{PA}}{\sqrt{S_P S_A}} \tag{4}$$

where:

$$S_{PA} = \frac{\sum (p_i - \bar{p})(a_i - \bar{a})}{n - 1}, \tag{5}$$

$$S_P = \frac{\sum (p_i - \bar{p})^2}{n - 1}, \tag{6}$$

$$S_A = \frac{\sum (a_i - \bar{a})^2}{n - 1} \tag{7}$$

In the above equations, a denotes actual target values, p denotes predicted target values, $\bar{a}$ represents the average of actual target value, $\bar{p}$ denotes the average of predicted target values and n denotes the sample size.

Table 3 Predictive models tested

Model	Correlation (%)	RAE (%)	RRSE (%)	Train time (s)	Test time (s)
REPTree	91.57	30.56	40.24	0.04	2.20
Linear regression	81.12	54.67	58.47	0.69	7.08
RBF network	22.77	96.65	97.36	0.25	2.08
IBk	91.99	29.98	39.27	0.01	7.03
LWL	65.54	76.47	76.25	0.01	60.00
Additive regression	71.57	67.54	69.83	0.10	2.01
Random subspace	84.74	59.47	62.78	0.20	3.10
Regression by disc	90.91	34.02	41.65	0.09	2.00
Conjunctive rule	35.59	92.70	93.43	0.05	1.70
Decision table	91.69	30.11	39.91	5.95	70.70
Decision stump	35.63	92.59	93.42	0.02	2.20
FIMTDD	68.29	71.89	71.49	–	–

3.3 Results

3.3.1 Event labels

Table 4 demonstrates detected events by our system that have passed verification phase by background knowledge. Bold items are those events that meet the condition of z-score ≥ 2. The primary candidates list before verification phase contains 69 events and as it can be seen this number is decreased to 30 events. To validate this result we ask a domain specialist to rate the impact of each detected event corresponding date from 0 to 5. The third column in Table 4 indicates the impact rates specified by specialist for that date. To evaluate the effectiveness of condition z-score ≥ 2, we define a cut line on impact rates given by specialist and then see how detected events match with the events over the cut line. Since the given impact rates are between 1 and 5, we define four cute point of 2, 3, 4 and 5 and compute the true and false alarm rate one time for all items of Table 4 and one time for bold items. The result is shown in Fig. 8. As it can be seen, applying condition of z-score ≥ 2 can decrease the false alarm rate up to an average of 20 %.

Table 4 Detected Events after verification phase by background knowledge

Date	Event	Impact
29-10-2012	**Sandy**	5
30-10-2012	**Sandy**	5
19-10-2012	**Storm**	5
04-07-2012	**Washington DC fireworks**	5
23-11-2012	**Black Friday**	4
24-12-2012	**Christmas day**	4
08-10-2012	**Columbus memorial celebration**	4
27-05-2012	**Memorial day**	4
22-11-2012	**Annual Thanksgiving day**	3
12-11-2012	**Veterans day**	2
16-04-2012	**Tax day**	1
23-03-2012	National cherry blossom festival	5
18-09-2012	Heavy rain	5
18-07-2012	Severe thunderstorm	5
01-06-2012	Tornado	4
04-12-2012	Warm weather floods	4
13-05-2012	Bike DC	3
11-02-2012	Cupid Undie run 2012	3
23-01-2012	March for life	3
29-09-2012	Green festival Washington DC	3
25-11-2012	The coldest morning of the season	2
07-10-2012	Unseasonably cool weather	2
07-04-2012	D.C. United vs. Seattle Sounders FC	2
26-05-2012	D.C. United vs. NE revolution	2
21-05-2012	Occasional showers and storms	2
15-09-2012	United vs. NE revolution	2
11-10-2012	D.C. Baseball v.s Tigers	2
12-10-2012	Hockey Capitals vs. NJ devils	2
29-01-2012	Occupy DC	1
19-05-2012	Survive DC 2012	1

Bold items are verified detected events (Events with z-score ≥ 2) and non-bold items are those events that their z-score $\prec 2$. The numbers in third column is the impact rate (from 0 to 5) given by a human domain specialist for that date indicating the impact of event

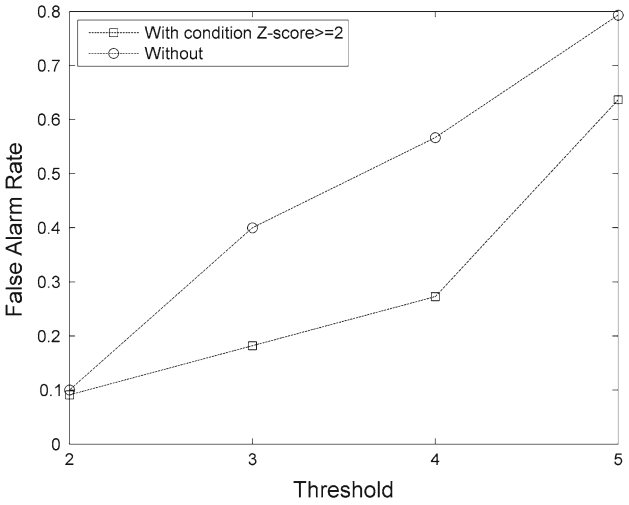

Fig. 8 Effect of condition z-score ≥ 2 on false alarm rate

Having look to the Table 4 we notice that five of alarms with impact rates of 5 and 4 that are specified with human domain specialist are not appeared in the verified alarms list. This can be due to two reasons. On one hand, the specialist admitted that our verification method based on investigation on Google web and image search and YouTube is more reliable that his evaluation. This is to some extent logical because existing knowledge in these sources represents the collaborative knowledge and intelligence of thousands of people with different insights and perspectives. This definitely might outperform a single knowledge source that has a limited insight to the subject. For this reason one reason for this observed gap can be because of the effectiveness of our approach comparing the specialist knowledge. In other hand, it can be due to some existing problems in our methodology details. For instance, probably due to the naming complexity of the events, our submitted query has not been well enough for measuring the weight of event. Four of these events are related to weather related events. It indicates that the query of "time + place + weather" might not be a good idea and we should find for more appropriate alternative query. The reason can be this fact that the weather events on these days are entitled with different terms and the weight corresponding to these events is distributed in different terms. For instance, some sources might call "Tornado" with different terms such

as high speed wind, storm or severe weather, etc. However, as it can be seen, only one non-weather event is missed in our final event labels. It reveals that the condition of z-score ≥ 2 reasonably is able to filter non-significant alarms and consequently avoid false alarms more effectively.

3.3.2 Event labeling in the absence of background knowledge

Here we study an event labeling model with relies only on ensemble detectors and have no access to any external knowledge (Fig. 1c). To compare this model with our proposed model (Fig. 1d), we need to compare the outputs of both models. If we assume model d and its result as reference we can formulate the problem as an information retrieval problem. We run model c and then measure the similarity of retrieved events with the reference detected events. In other words, we want to know how we can reproduce the same result as model d by using model c. We consider our model as reference model because its output is already checked with a human domain expert.

To this end we use a voting strategy to combine ensemble detectors alarms. We first define a confidence threshold equal to p value $= 0.05$ and compute the total of detectors votes corresponding each day. For detector 8 and 9 that are based on clustering and do not return p value we assume that the detectors are confident enough (p value ≤ 0.05) and thus include them in the voting process.

We count the votes of detectors for each instant and then compute the similarities of detected list with Table 4. The results is presented in Table 5. N in this table denotes the detector votes. $N > 1$ for instance indicates that at least two detectors agree that a hypothetical instant should be recognized as event. The terms $N > 2$, $N > 3$ and $N > 4$ means that at least three, four or five detectors are respectively suspicious to a particular instant. As it can be seen, event signals corresponding to $N > 1$ are 72 % similar to events marked with bold in Table 4. It means that if we have vote of at least two detectors for an instant, the alarms would be over 70 % similar to final output of our proposed model. However, by increasing the required vote, F-measure decreases to 58 % for $N > 2$ and 0.23 for $N > 3$ and $N > 4$.

Table 5 Ensemble detectors retrieval performance in the absence of background knowledge

Votes	Precision	Recall	F-measure
$N > 1$	0.81	0.64	0.72
$N > 2$	0.63	0.53	0.58
$N > 3$	0.27	0.20	0.23
$N > 4$	0.27	0.20	0.23

Reference events: bold items in Table 4

Table 6 Individuall detectors retrieval performance in the absence of background knowledge

Detector	Precision	Recall	F-measure
1	0.60	0.50	0.55
2	0.25	0.22	0.23
3	0.25	0.28	0.26
4	0.40	0.11	0.17
5	0.31	0.28	0.29
6	0.43	0.17	0.24
7	0.17	0.11	0.13
8	1.00	0.22	0.36
9	0.75	0.17	0.28
10	0.33	0.33	0.33

Reference events: bold items in Table 4

We repeat the same procedure for individual detectors to compare their individual performance. Comparing the ensemble detectors F-measures in Table 5 with individual detectors in Table 6 reveals the effectiveness of ensemble detectors comparing the individual detectors. As it can be seen the maximum F-measure obtained for individual detector is related to detector 1 which is equal to 0.55. This is about 20 % lower than ensemble detectors with $N > 1$ condition.

3.3.3 Evaluation of individual detectors with ROC analysis

ROC curves are robust tools for evaluation of classifiers and event and anomaly detection algorithms. Vertical axis of a ROC curve corresponds to true positive rate and horizontal axis is related to false positive rate.

As we already mentioned our detectors (except detector 8 and 9) return a p value for each instant indicating the chance of that instant is not an event. In most event detection systems, usually a pre-defined threshold (0.05) is set by user as confidence level. Then, if p value returned by detector is lower than the threshold, an alarm is raised.

In order to plot ROC curve of each detector, we first define validated bold items in Table 4 as the target set. Then we vary threshold from 0 to 1 with step of 0.001 and then compute the true positive and false positive rate of detected set against target set. Next, in order to compare the accuracy of individual detectors we compute area under the obtained ROC curves. The result is presented in Table 7. As it can be seen, detector 10 provides the best accuracy and detectors 7, 1, 2 and 5 are ranked in the subsequent places, which all provides over 70 % accuracy. The values in this table do not indicate the strength of the detectors in general; rather reveals their specific performance on this particular condition. In other words, if we would not consider ensemble detectors and we would not have access to background knowledge, detector 10 could reproduce the same result as our system with accuracy

Table 7 Area under ROC curves for each individual detectors

Detector	AUC
1	0.74
2	0.70
3	0.60
4	0.47
5	0.70
6	0.39
7	0.75
8	N/A
9	N/A
10	0.76

of 75 %. There is no guarantee that this detector performs the same on other circumstances.

3.4 Sensitivity analysis

Here we present our comparative study results. Note that the following results are domain specific and may not be generalized to other different problems and settings. Due to the difficulty and limitations on obtaining data with required characteristics we were not able to perform experiments on different domains and applications. Hence, the following specific findings emphasis on bike sharing data and the similar regression problems and domains and are required to be validated for other domains.

3.4.1 Learning: semi-supervised vs. unsupervised

Table 7 shows the area under ROC curve for the individual detectors. As it can be seen, semi-supervised detectors outperform unsupervised detectors. Area under curve for unsupervised detectors (detectors 4 and 6) is below 0.6. This shows that bike sharing data is highly non-sequential and each instant is temporally independent of other instances. Instead, instances are dependent on environmental settings. Unsupervised detectors are able to detect only severe events while semi-supervised detectors are able to detect more meaningful events.

3.4.2 Scale analysis: hour vs. day

We designed the detectors such a way to be able to compare the performance of analysis on daily scale vs. hourly scale. Among semi-supervised detectors, detector 3 is the only approach that operates on daily scale. It means that makes a model from train set in daily aggregated counts (Fig. 2) and then make a forecast on test set in daily scale. Other approaches make a model from train set in the hourly scale. Area under curve of detector 3 comparing to the other methods is presented in Table 7. As it can be seen, the low AUC value for this detector reveals this fact that for detecting events in day scale it is not always a better idea to make a predictive model on the same scale, rather sometimes is better to make a model on smaller scales. The detectors that operate on hourly scale show at least 10 % more accuracy comparing detector 3. This provides evidence that for event detection in a desired scale, training on smaller scale also would worth to be considered.

3.4.3 PCA vs. MSSA

The idea of MSSA is to adapt PCA for time series. PCA looks to the instances independently while MSSA takes into account the auto-correlation between temporal instances together. Capturing this auto-correlation is not considered in PCA. We compared the performance of PCA (detector 5) vs. MSSA (detector 7). We considered only three Principal components for both and did not check other settings. They might have different performance in other settings. However in the same condition as Table 7 shows, PCA outperforms MSSA. The reason is almost clear. The bike sharing data is highly non-sequential and it is a poor correlation between consecutive instances. If there was a strong auto-correlation then MSSA would perform better. The point is that PCA and MSSA is required to be chosen depending on the data nature. PCA is recommended for independent instances and MSSA for auto-correlated instances.

3.4.4 Predictive model: online vs. offline

In Table 3 we compared the performance of 12 different algorithms. The last algorithm in this table is related to FIMT-DD algorithm [27], which is an online regression tree model. This is a streaming algorithm that scans the training data only once, using little computational resources (memory and CPU). As it can be seen, this model includes less accuracy comparing to the REPTree, however if the computation complexity is the problem, can present a relatively reasonable performance. The predicted counts correlation to the original counts is 68.29 % vs. REPTree that presents 91.57 % correlation. This difference seems reasonable for large data sets or streaming settings where REPTree fails.

3.4.5 Distance: euclidean vs. DTW

A comparison of the performance obtained using Euclidean (detector 8) and DTW (detector 9) distances is presented in Table 7. Although there is any significant difference, DTW exhibits worst results. This reveals that DTW necessarily do not always outperform Euclidean distance in time series similarity measurement. This is another evidence that envi-

ronmental attributes play an important role in bike sharing process.

4 Conclusion

We proposed a novel event labeling model based on ensemble learning and background knowledge. We provided some evidences about the effectiveness of the proposed model through a set of tasks on a real-world data set.

Our research findings can be summarized as follows: (1) When there is no access to human experts, background knowledge (if available) can an appropriate alternative; (2) scale of the train and test data sets necessarily should not be the same. We demonstrated that in some particular settings, this assumption could be violated; (3) regression tree model REPTree is promising on like bike sharing data set. We believe that this model probably work well on data sets with same nature dealing with count time series under effect of environmental and periodicity settings (4) MSSA and DTW are reconsigned as robust tools in time series analysis. However as we demonstrated they can act inverse when data is under seasonal effects. (5) On the absence of background knowledge, ensembles detectors can present a result 70 % similar to the condition where we have access to background knowledge for verification; (6) we offered evidence that ensemble detectors with at least two votes provide 20 % better result than the best individual detector; (7) we showed that bike rental data is highly correlated with environmental and periodicity settings such as temperature and hour of the day, month, work of the day (weekend, weekday, and holiday). A regression tree model can make a prediction based on these environmental attributes very close to actual counts. This shows that bike rental count time series should not be analyzed without taking into account the environmental attributes; (8) web and existing knowledge in that can be potential source for aiding event detection systems.

Event detection on bike sharing data also has two potential applications. First, it can be incorporated in a decision support system for a better planning and management of system and secondly, can be employed in a recommender system for alarming or suggestion purposes. For instance, suggesting people not going out due to severe weather conditions or encourage them to go out for participating in an ongoing event in the town.

Further research will include the following directions: (1) Verification of the proposed model performance on other data sets and with other knowledge sources apart from online sources; (2) testing different ensemble designs; (3) studying different combination techniques in ensemble detectors; (4) spatiotemporal analysis on the data to discover localized events; (5) real time detection; (6) development of some text processing methods for automated capturing of knowledge from the Web.

Acknowledgments This work is funded by the European Regional Development Fund through the COMPETE Program, by the Portuguese Funds through the FCT (Portuguese Foundation for Science and Technology) within project FCOMP — 01-0124-FEDER-022701. J. Gama also acknowledges the support of the European Commission through the project MAESTRA (Grant Number ICT-2013-612944). The authors also thank Chris Holben — the Bike sharing Project Manager and Kim Lucas—the Bicycle Program Specialist from District Department of Transportation, Washington, D.C, U. S. A for their help and feedback, as well as Capital Bike Sharing for providing data.

References

1. Abdi, H., Williams, L.J.: Principal component analysis. Wiley Interdiscip. Rev. Comput. Stat. **2**(4), 433–459 (2010)
2. Aggarwal, C.C.: Outlier ensembles: position paper. SIGKDD Explor. Newsl. **14**(2), 49–58 (2013)
3. Anantharam, P., Thirunarayan, K., Sheth, A.: Topical anomaly detection from twitter stream. In Proceedings of the 3rd Annual ACM Web Science Conference, WebSci '12, pp. 11–14, New York, ACM (2012)
4. Ashfaq, A., Javed, M., Khayam, S., Radha, H.: An information-theoretic combining method for multi-classifier anomaly detection systems. In: Communications (ICC), 2010 IEEE international conference on, pp. 1–5 (2010)
5. Barford, P., Kline, J., Plonka, D., Ron, A.: A signal analysis of network traffic anomalies. In: Proceedings of the 2nd ACM SIGCOMM workshop on internet measurment, IMW '02, pp. 71–82, New York, ACM (2002)
6. Borgnat, P., Abry, P., Flandrin, P., Robardet, C., Rouquier, J.-B., Fleury, E.: Shared bicycles in a city: a signal processing and data analysis perspective. Adv. Complex Syst. **14**(03), 415–438 (2011)
7. Brauckhoff, D., Dimitropoulos, X., Wagner, A., Salamatian, K.: Anomaly extraction in backbone networks using association rules. In: Proceedings of the 9th ACM SIGCOMM conference on Internet measurement conference, IMC '09, pp. 28–34, New York, ACM (2009)
8. Buckeridge, D.L., Burkom, H., Campbell, M., Hogan, W.R., Moore, A.W.: Algorithms for rapid outbreak detection: a research synthesis. J. Biomed. Inform. **38**(2), 99–113 (2005)
9. Capital Bike Share System: Capital bike sharing trip history data. http://www.capitalbikeshare.com/trip-history-data (2013)
10. Carletta, J.: Assessing agreement on classification tasks: the kappa statistic. Comput. Linguist. **22**(2), 249–254 (1996)
11. Chandola, V., Banerjee, A., Kumar, V.: Anomaly detection: a survey. ACM Comput. Surv. **41**(3), 15:1–15:58 (2009)
12. Collier, N., Doan, S., Kawazoe, A., Goodwin, R.M., Conway, M., Tateno, Y., Ngo, Q.-H., Dien, D., Kawtrakul, A., Takeuchi, K., et al.: Biocaster: detecting public health rumors with a web-based text mining system. Bioinformatics **24**(24), 2940–2941 (2008)
13. Dembski, W.A.: The Design Inference: Eliminating Chance Through Small Probabilities. Cambridge University Press (1998)
14. Department of Human Resources: District of Columbia. Washington D.C. holiday schedule. http://dchr.dc.gov/page/holiday-schedule (2013)
15. Dewaele, G., Fukuda, K., Borgnat, P., Abry, P., Cho K.: Extracting hidden anomalies using sketch and non gaussian multiresolution statistical detection procedures. In: Proceedings of the 2007 workshop on Large scale attack defense, LSAD '07, pp. 145–152, New York, ACM (2007)

16. Dietterich, T.G.: Ensemble methods in machine learning. In: Proceedings of the first international workshop on multiple classifier systems, MCS '00, pp. 1–15. Springer, London (2000)
17. Fanaee-T, H., Gama, J.: Bike sharing data set. http://fanaee.com/research/datasets/bike/ (2013)
18. Fleiss, J.L.: Measuring nominal scale agreement among many raters. Psychol. Bull. **76**(5), 378–382 (1971)
19. Floyd, S., Paxson, V.: Difficulties in simulating the internet. IEEE/ACM Trans. Netw. **9**(4), 392–403 (2001)
20. Fontugne, R., Borgnat, P., Abry, P., Fukuda, K.: Mawilab: combining diverse anomaly detectors for automated anomaly labeling and performance benchmarking. In: Proceedings of the 6th International COnference, Co-NEXT '10, pp. 8:1–8:12. New York, ACM (2010)
21. Freemeteo: Washington D.C. weather history. http://www.freemeteo.com (2013)
22. Ghosh, A.K., Schwartzbard, A., Schatz, M.: Learning program behavior profiles for intrusion detection. In: Proceedings of the 1st conference on Workshop on Intrusion Detection and Network Monitoring. USENIX Association, vol. 1, pp. 6–6. ID'99, Berkeley (1999).
23. Giacinto, G., Perdisci, R., Del Rio, M., Roli, F.: Intrusion detection in computer networks by a modular ensemble of one-class classifiers. Inf. Fusion **9**(1), 69–82 (2008)
24. Guralnik, V., Srivastava, J.: Event detection from time series data. In: Proceedings of the fifth ACM SIGKDD international conference on Knowledge discovery and data mining. pp. 33–42. ACM (1999)
25. Hastie, T., Tibshirani, R., Friedman, J.J.H.: The elements of statistical learning. vol. 1, Springer, New York (2001)
26. Witten, I.H., Eibe Frank, M.A.H.: Data mining: practical machine learning tools and technique. In: Kaufmann, M. 3rd edn (2011)
27. Ikonomovska, E., Gama, J., Dzeroski, S.: Learning model trees from evolving data streams. Data Min. Knowl. Discov. **23**(1), 128–168 (2011)
28. Jackson, M.L., Baer, A., Painter, I., Duchin, J.: A simulation study comparing aberration detection algorithms for syndromic surveillance. BMC Med. Inform. Decis. Mak. **7**(1), 6 (2007)
29. Kerman, M., Jiang, W., Blumberg, A., Buttrey, S.: Event detection challenges, methods, and applications in natural and artificial systems. In: Proceedings of 14th international command and control research and technology symposium, ICCRTS, Lockheed Martin MS2, pp. 1–19 (2009)
30. Kuncheva, L.I.: Combining pattern classifiers: methods and algorithms. Wiley-Interscience (2004)
31. Lakhina, A., Crovella, M., Diot, C.: Diagnosing network-wide traffic anomalies. SIGCOMM Comput. Commun. Rev. **34**(4), 219–230 (2004)
32. Lakhina, A., Crovella, M., Diot, C.: Mining anomalies using traffic feature distributions. In: Proceedings of the 2005 conference on Applications, technologies, architectures, and protocols for computer communications, SIGCOMM '05, pp. 217–228. New York, ACM (2005)
33. Landis, J.R., Koch, G.G.: The measurement of observer agreement for categorical data. Biometrics **33**(1), 159–174 (1977)
34. Li, X., Bian, F., Crovella, M., Diot, C., Govindan, R., Iannaccone, G., Lakhina, A.: Detection and identification of network anomalies using sketch subspaces. In: Proceedings of the 6th ACM SIGCOMM conference on Internet measurement, IMC '06, pp. 147–152. New York, ACM (2006)
35. Lippmann, R., Haines, J.W., Fried, D.J., Korba, J., Das, K.: The: darpa off-line intrusion detection evaluation. Comput. Netw. **34**(4), 579–595 (1999)
36. Marins, A., Casanova, M.A., Furtado, A., Breitman, K.: Modeling provenance for semantic desktop applications. In: SEMISH-Anais do Seminario Integrado de Software e Hardware XXXIV, pp. 2101–2112 (2007)
37. McHugh, J.: Testing intrusion detection systems: a critique of the 1998 and 1999 darpa intrusion detection system evaluations as performed by lincoln laboratory. ACM Trans. Inform. Syst. Secur. **3**(4), 262–294 (2000)
38. Nychis, G., Sekar, V., Andersen, D.G., Kim, H., Zhang, H.: An empirical evaluation of entropy-based traffic anomaly detection. In: Proceedings of the 8th ACM SIGCOMM conference on Internet measurement, IMC '08, pp. 151–156. New York, ACM (2008)
39. Patterson, K., Hassani, H., Heravi, S., Zhigljavsky, A.: Multivariate singular spectrum analysis for forecasting revisions to real-time data. J. Appl. Stat. **38**(10), 2183–2211 (2011)
40. Policy Institute: Bike-sharing programs hit the streets in over 500 cities worldwide. http://www.earth-policy.org/plan_b_updates/2013/update112 (2013)
41. Ringberg, H., Soule, A., Rexford, J.: Webclass: adding rigor to manual labeling of traffic anomalies. SIGCOMM Comput. Commun. Rev. **38**(1), 35–38 (2008)
42. Rubinstein, B.I., Nelson, B., Huang, L., Joseph, A.D., Lau, S.-H., Rao, S., Taft, N., Tygar, J.D.: Antidote: understanding and defending against poisoning of anomaly detectors. In: Proceedings of the 9th ACM SIGCOMM conference on Internet measurement conference, IMC '09, pp. 1–14. New York, ACM (2009)
43. SanMiguel, J.C., Martinez, J.M., Garcia, A.: An ontology for event detection and its application in surveillance video. In: Proceedings of the 2009 sixth IEEE international conference on advanced video and signal based surveillance, AVSS '09, IEEE Computer Society, pp. 220–225. Washington, DC (2009)
44. Scherrer, A., Larrieu, N., Owezarski, P., Borgnat, P., Abry, P.: Non-gaussian and long memory statistical characterizations for internet traffic with anomalies. IEEE Trans. Dependable Secur. Comput. **4**(1), 56–70 (2007)
45. Scott, S.L.: A bayesian paradigm for designing intrusion detection systems. Comput. Stat. Data Anal. **45**(1), 69–83 (2004)
46. Senin, P.: Dynamic time warping algorithm review. In: University of Hawaii at Manoa, Technical report series (2008)
47. Shanbhag, S., Wolf, T.: Accurate anomaly detection through parallelism. Netw. Mag. Glob. Internetwkg. **23**(1), 22–28 (2009)
48. Tan, K., Maxion, R.: The effects of algorithmic diversity on anomaly detector performance. Dependable systems and networks, 2005. DSN 2005. In: Proceedings of international conference on, pp. 216–225 (2005)
49. Tan, K.M.C., Maxion, R.A.: Performance evaluation of anomaly-based detection mechanisms. In: Technical report series CS-TR-870, University of Newcastle upon Tyne, Newcastle upon Tyne, NE1 7RU, UK (2004)
50. Tavallaee, M., Bagheri, E., Lu, W., Ghorbani, A.A.: A detailed analysis of the kdd cup 99 data set. In: Proceedings of the second IEEE international conference on Computational intelligence for security and defense applications, CISDA'09, Piscataway, NJ, pp. 53–58. IEEE Press (2009)
51. Vogel, P., Greiser, T., Mattfeld, D.C.: Understanding bike-sharing systems using data mining: exploring activity patterns. Procedia Soc. Behav. Sci. **20**, 514–523 (2011)
52. Warrender, C., Forrest, S., Pearlmutter, B.: Detecting intrusions using system calls: alternative data models. In security and privacy, 1999. In: Proceedings of the 1999 IEEE symposium on, pp. 133–145 (1999)
53. Wong, C., Bielski, S., Studer, A., Wang, C.: Empirical analysis of rate limiting mechanisms. In: Proceedings of the 8th international conference on Recent Advances in Intrusion Detection, RAID' 05, pp. 22–42. Springer, Heidelberg (2006)
54. Wong, W.-K., Moore, A., Cooper, G., Wagner, M.: What's strange about recent events (wsare): an algorithm for the early detec-

tion of disease outbreaks. J. Mach. Learn. Res. **6**, 1961–1998 (2005)
55. Xu, C., Zhang, Y.-F., Zhu, G., Rui, Y., Lu, H., Huang, Q.: Using webcast text for semantic event detection in broadcast sports video. Multimedia IEEE Trans. **10**(7), 1342–1355 (2008)
56. Zheng, V.W., Zheng, Y., Xie, X., Yang, Q.: Collaborative location and activity recommendations with gps history data. In: Proceedings of the 19th international conference on World wide web, WWW'10, pp. 1029–1038. New York, ACM (2010)