Detecting Urban Anomalies Using Multiple Spatio-Temporal Data Sources

HUICHU ZHANG, Shanghai Jiao Tong University, China
YU ZHENG, JD Finance, China
YONG YU, Shanghai Jiao Tong University, China

Urban anomalies, such as abnormal movements of crowds and accidents, may result in loss of life or property if not handled properly. It would be of great value for governments if anomalies can be automatically alerted in their early stage. However, detecting anomalies in urban area has two main challenges. First, the criteria to determine an anomaly on different occasions (e.g. rainy days vs. sunny days, or holidays vs. workdays) and in different places (e.g. tourist attractions vs. office areas) are distinctly different, as these occasions and places have their own definitions on normal patterns. Second, urban anomalies often exhibit complex forms (e.g. road closure may cause decrease in taxi flow and increase in bike flow). We need an algorithm that not only models the anomaly degree of individual data source but also the combination of changes in multiple data sources. In this paper, we propose a two-step method to tackle those challenges. In the first step, we use a similarity-based algorithm to estimate an anomaly score for each individual data source in each region and time slot based on the values of historically similar regions. Those scores are fed into the second step, where we propose an algorithm based on one-class Support Vector Machine to capture rare patterns occurred in multiple data sources, nearby regions or time slots, and give a final, integrated anomaly score for each region. Evaluations based on both synthetic and real world datasets show the advantages of our method beyond baseline techniques such as distance-based, probability-based methods.

CCS Concepts: • **Information systems → Geographic information systems**;

Additional Key Words and Phrases: urban computing, anomaly detection, spatio-temporal data mining, multiple data sources

ACM Reference Format:
Huichu Zhang, Yu Zheng, and Yong Yu. 2018. Detecting Urban Anomalies Using Multiple Spatio-Temporal Data Sources. *Proc. ACM Interact. Mob. Wearable Ubiquitous Technol.* 2, 1, Article 54 (March 2018), 18 pages. https://doi.org/10.1145/3191786

1 INTRODUCTION

Urban anomalies, like unexpected crowd gathering, may pose tremendous risks to public safety if not timely handled. For example, on Dec. 31th, 2014, a tragic stampede took place in Shanghai, where more than 300,000 people flocked to the Bund (a famous riverfront promenade in Shanghai) for a popular light show on New Year's Eve. 36 people were killed and 49 injured. The authority later admitted that they had underestimated the crowd size and were not well prepared for the event. A similar stampede happened in the 2010 German Love Parade. For

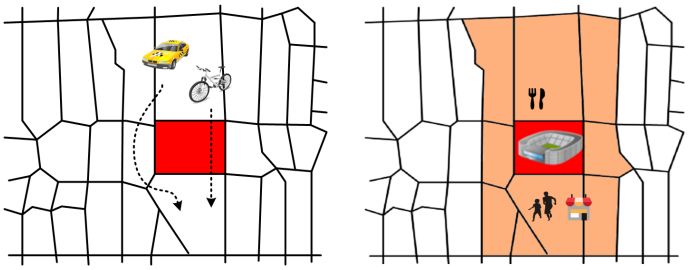
The research was done under the supervision of Yu Zheng who is the correspondence author of this paper.
Authors' addresses: Huichu Zhang, Shanghai Jiao Tong University, Apex Data & Knowledge Management Lab, 800 Dongchuan Road, Shanghai, 200240, China, zhc@apex.sjtu.edu.cn; Yu Zheng, JD Finance, Urban Computing Lab, 18 Kechuang 11 Street, BDA, 101149, China, msyuzheng@outlook.com; Yong Yu, Shanghai Jiao Tong University, Apex Data & Knowledge Management Lab, 800 Dongchuan Road, Shanghai, 200240, China, yyu@apex.sjtu.edu.cn.

2474-9567/2018/3-ART54 $15.00
https://doi.org/10.1145/3191786

government and policy makers, it would be of great value if anomalies can be timely detected or even reported at its early stage, enabling them to take immediate actions and prevent serious incidents from happening.

In this paper, we focus on detecting urban anomalies like events (e.g. concerts, unexpected gatherings) or accidents (e.g. traffic accident) which take place in a small range of regions [11, 31] rather than city-scale ones [26] like thunderstorms or holidays. Detecting *local* urban anomalies has more practical value because they are less foreseeable than city-scale ones. Also, it is a more challenging task because the criteria for anomaly will change due to external influences (detailed later in this section). With the help of massive spatio-temporal data such as traffic conditions, user check-ins generated in cities, we are able to detect these urban anomalies. When an anomaly happens, uncommon pattern will be observed from these data sources. Moreover, values of these data sources in nearby regions or previous time slots may also be influenced. The following are two examples:

Example 1.1: As illustrated in Figure 1a, a road located in the red area was temporally closed due to construction. Drivers have to bypass it through other routes, causing the traffic flow in this area decreased by 10%. On the contrary, more people chose to ride bikes rather than taxis when traversing this area, leading to a 10% increase of the bike flow.



(a) Road Closure (b) Concert

Fig. 1. Examples of urban anomalies

Example 1.2: As shown in Figure 1b, an influential pop star held a concert in the stadium at 20:00. Many fans arrived at the stadium several hours before the concert started, having dinner in nearby restaurants or wandering around the stadium. As a result, the traffic flow around the stadium during 18:00-19:30 as well as that in nearby regions all increased by 10-15%.

To accurately capture urban anomalies and timely alert for potential ones as shown in the above examples, we propose a method to detect urban anomalies based on uncommon patterns observed from multiple spatio-temporal data sources of a certain region along with values of nearby regions and time slots. Our approach has two advantages beyond traditional methods:

1) **Detecting underlying anomalies**: Some anomalies may not be detected in terms of a single dataset, but show unusual patterns if we check multiple datasets simultaneously. As illustrated in Example 1.1, if we check the increase and decrease individually, they look like normal fluctuations over the corresponding dataset, thereby cannot be detected as anomalies. However, if checking the two datasets simultaneously, we find the opposite change (i.e. increase vs. decrease) of flows is quite abnormal, as they change with the same trend in most cases.

2) **Earlier detection**: When an event took place in a certain region, it might influence the behavior of nearby regions or consecutive time slots. Sometimes the anomaly degree for the values in a single region or time slot may not be large enough to be marked as an anomaly, however, when taking the values of nearby regions and consecutive time slots into consideration, anomalies become obvious. This kind of pattern often appears in the early stage of an event, enabling us to issue alert before the anomaly reaches its peak. As shown in Example 1.2,

if we only check the traffic flow at the stadium during 18:30-19:00, 15% might not be a significant jump. But, with the help of nearby regions and consecutive time slots like traffic flow during 18:00-18:30, the fact that all those values increased by 10-15% is a relatively rare situation, and we can safely separate it from normal fluctuations and mark the stadium as anomalous during 18:30-19:00.

While timely detect urban anomalies may benefit a lot for both governments and citizens, it is a challenging problem for the following reasons:
1) **Distinctiveness and uncertainty of urban data.** Data in different regions and time or under different external influences may have different definitions of normal patterns. For example, on holiday nights, the *normal* value of taxi flow in office area should be much less than that in commercial area, while on weekday mornings, the opposite holds. Furthermore, the *normal* bike flow in rainy day should be much less than that in sunny days. We need to define different criteria for anomalies under those different conditions.
2) **Complex forms of urban anomalies.** As Figure 1a shows, anomaly lies in the combination of increase in bike flow and decrease in taxi flow, which is a rare case. This requires the algorithm be capable of evaluating the anomaly degree of different combinations of changes in multiple data sources.

To tackle the above challenges, we propose a two-step method for detecting urban anomalies, which consists of a similarity-based method to compute individual anomaly scores (*CIAS*) and an algorithm based on one-class Support Vector Machine [19] to aggregate the individual anomaly scores (*AIAS*). The contributions of our work are as follows:
1) *CIAS* leverage the values of historically similar time series and give a precise estimation of the anomaly score for values of each data source in each region or time slot under different external influences.
2) In *AIAS*, we perform a two-stage *OC-SVM* with radial basis function (*rbf*) kernel using the input from *CIAS* to select final anomalies. *AIAS* takes nearby regions and time slots into consideration. It also explicitly evaluates the anomaly degree of different combinations of changes in multiple data sources.
3) We perform extensive evaluations on both synthetic and real world data. Our model outperforms baselines in the quantitative test based on synthetic datasets where we manually inject anomalies and use them as the ground truth. We also evaluate our method on real world event detection task and study two representative cases which are detected by our method based on taxi and bike trip data of New York City.

2 OVERVIEW

In this section, we show some preliminary definitions and the goal of our work. We also provide an overview of our method.

2.1 Preliminaries

Definition 2.1 (Region). There are many definitions of location in terms of different granularity and semantic meanings. In this study, we partition a city into regions $\mathbf{r} = \{r_1, r_2, \ldots, r_{n_r}\}$ by major roads, such as highways and arterial roads, using a map segmentation method [29]. This is the practice adopted by previous reserach [31]. n_r stands for the number of regions. Consequently, each region is bound by major roads, carrying a semantic meaning of neighborhoods or communities, as illustrated in Figure 2.

Definition 2.2 (Data Sources). A data source s is a stream of values, each of which is a triplet $< r, t, v >$ representing that value v is observed in region r at time slot t. From another view, each data source is consisted of n_r time series $v^{r,s}_{0:t}$ which stands for the values of data source s observed in region r from time slot 0 to t. n_s stands for the total number of data sources in the dataset.

(a) Raw road network (b) Segmented regions

Fig. 2. Map segmentation and regions

Definition 2.3 (Goal). Given r_n regions and historical values of n_s data sources, $v_{0:t-1}^{r_i,s_j}, i \in [1, n_r], j \in [1, n_s]$ and all the newly observed values at time slot t, $v_t^{r_i,s_j}, i \in [1, n_r], j \in [1, n_s]$, we aim at detecting k anomalous regions $\mathcal{A} = \{r_{i_1}, r_{i_2}, \ldots, r_{i_k}\}$, where k depends on the proportion of regions we want to mark as anomaly every day. The degree of anomaly is evaluated based on the observed values of all data sources in each region r as well as values in nearby regions and consecutive time slots.

2.2 Framework

Figure 3 shows the procedure of our method. *CIAS* takes all observed values as input and gives an anomaly score for each individual data source s, region r and time slot t. The individual anomaly score computed represents how much the observed value deviates from the *normal* value. These scores are then fed into *AIAS* where we try to detect rare patterns occurred in the data and finally output anomalous regions.

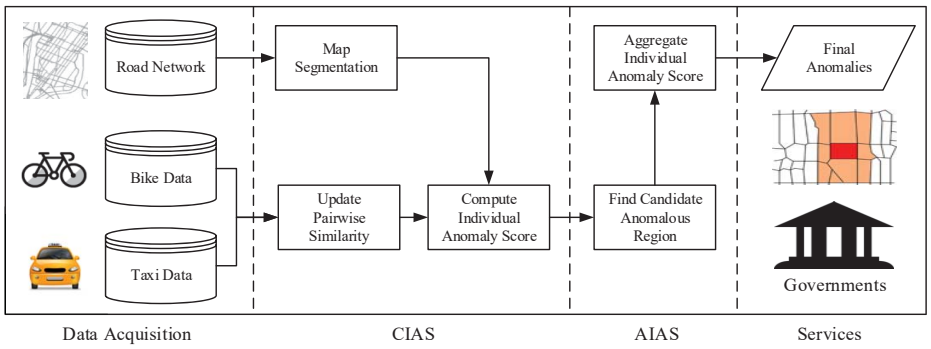


Fig. 3. Framework

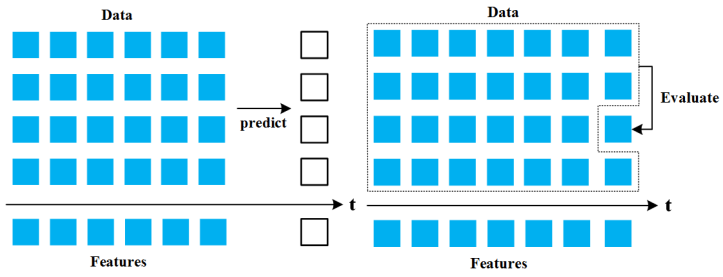
3 COMPUTE INDIVIDUAL ANOMALY SCORES

3.1 Insight

In this step, we aim at estimating the anomaly score for every $< r, t, v >$ triplet, a.k.a. computing individual anomaly scores. *CIAS* transforms the raw observed values into same-scale anomaly scores which stand for how much the newest values deviates from what they should be. These anomaly scores will be fed into the *AIAS* (Section 4) for further process.

As mentioned in Section 1, urban data are significantly influenced by complex factors, which makes it hard to define how much the latest value deviates from its *normal* range. In [31][27][22], researchers first built a prediction model based on historical values and external features, then calculated anomaly scores according to the predictor. These kinds of solutions might work but they face some problems. Because urban data are influenced by tons of factors, including time of day, function of regions, weather, temperature, holiday, government policies, it is difficult to train a precise prediction model to cover so many different environmental situations. For example, although people do not need to go to work on weekdays, Christmas and Thanksgiving Day, we still cannot treat them altogether as *holidays*, for they clearly have different impacts on the traffic flow in cities. However, due to lack of training data, most methods simply mark them as *holidays* in contrast to *workdays*, which will lead to imprecise prediction.

In fact, in the context of anomaly detection, paying huge effort on building an accurate prediction model is not necessary. As shown in Figure 4a, in prediction task, what we got is all the historical data and other environmental features. We need to use those values to predict all the values in the latest time slot. However, as Figure 4b shows, when detecting anomalies, we usually have access to all the values generated in the current time slot. This would make a big difference because we can leverage the newest values of other data sources in other regions to evaluate the anomalousness of the value in a certain region. With this observation in mind, we propose a similarity-based method where we use historically similar regions as a guide and make full use of all the newly observed data.



(a) Prediction Task (b) Anomaly Detection Task

Fig. 4. Illustration of prediction and anomaly detection

As shown in Figure 5a, we plot one time series $v^{r_1,s_1}_{t_1:t_2}$ and the weighted average of time series which has high similarity with $v^{r_1,s_1}_{t_1:t_2}$ (time series plotted are first standardized to have mean 0 and standard deviation 1). Here we use Pearson Correlation Coefficient (*PCC*) as the similarity measurement, the reason of using *PCC* will be explained later. From the figure, we can see that the weighted average is a good estimation of the value of $v^{r_1,s_1}_{t_1:t_2}$. But in Figure 5b, we observe that the new value deviates a lot from the new weighted average. This phenomena indicates a potential anomaly. Generally speaking, if a group of urban time series has high similarity, it implies that they may serve similar function in city, e.g. all of them are the taxi flow in office area. Since they serve similar function, they tend to exhibit similar behavior even when external environment changes. For example, suppose the government issued a temporary holiday decision due to bad weather, then the taxi flow in office area would all decrease. If, however, some region r in the group does not behave like others, there is probably an anomaly. Since we have access to all the latest observed data, we do not have to directly model the influence these external factors exert on the data sources. Instead, we use the latest values of historically similar time series to evaluate the anomaly degree for the current time series. This observation leads to the basic idea of *CIAS*. Suppose two time series $v^{r_1,s_1}_{0:t-1}$ and $v^{r_2,s_2}_{0:t-1}$ has high similarity, if after observing value in time slot t, the similarity drops significantly, then this is a sign for anomaly.

(a) Normal (b) Anomalous

Fig. 5. Weighted average of similar regions

3.2 Methodology

For each time slot t, we compute a pairwise similarity matrix $\mathbf{S}^t \in \mathbb{R}^{r \cdot s \times r \cdot s}$ for each region r and data source s, where $\mathbf{S}^t_{r_1,s_1;r_2,s_2}$ stands for the similarity between data source s_1 in region r_1 and data source s_2 in region r_2 at time slot t. $\mathbf{S}^t_{r_1,s_1;r_2,s_2}$ is calculated by the following equation:

$$\mathbf{S}^t_{r_1,s_1;r_2,s_2} = \rho(v^{r_1,s_1}_{(t-l+1):t}, v^{r_2,s_2}_{(t-l+1):t}) \tag{1}$$

where ρ is the Pearson Coefficient Correlation (*PCC*). l is a parameter controlling how many values we are interested when computing similarity, since urban data usually exhibit strong weekly periodicity, we set l to be the length covering a whole week. We choose *PCC* as the similarity measurement for the following two reasons. First, *PCC* reveals the true similarity between time series regardless of the scale and length [12]. Second, *PCC* is known to be vulnerable to outliers. This is one of the drawback of *PCC*. However, in the context of anomaly detection, we exactly require the similarity measurement be sensitive to anomalous points.

After computing the pairwise similarity matrix at each time slot t, we define the concept of historically similar series as $< r_1, s_1 >, < r_2, s_2 >$ pair whose *PCC* is larger than a certain threshold θ. That is, $< r_1, s_1 >$ and $< r_2, s_2 >$ are regarded as historically similar at time slot t if $\mathbf{S}^{t-1}_{r_1,s_1;r_2,s_2} > \theta$. Also, we use $\mathcal{HS}^t_{r,s}$ as the set of all historically similar series $< r', s' >$ for $< r, s >$ at time slot t.

Then we calculate a Similarity Decrease Matrix $\mathbf{SC}^t \in \mathbb{R}^{r \cdot s \times r \cdot s}$. $\mathbf{SC}^t$ stands for the decrease of pairwise similarity from time slot $t-1$ to time slot t. $\mathbf{SC}^t_{r_1,s_1;r_2,s_2}$ represents the drop in similarity of data source s_1 in region r_1 and data source s_2 between time slot $t-1$ and time slot t, which is calculated by the following equation.

$$\mathbf{SC}^t_{r_1,s_1;r_2,s_2} = max(0, \mathbf{S}^{t-1}_{r_1,s_1;r_2,s_2} - \mathbf{S}^t_{r_1,s_1;r_2,s_2}) \tag{2}$$

since we are only interested in the drop in similarity, negative values which represents an increase in similarity will be omitted.

We compute the anomalous degree $ad^{r,s}_t$ for $v^{r,s}_t$ based on Equation 3

$$ad^{r,s}_t = \frac{\sum_{<r',s'> \in \mathcal{HS}^t_{r,s}} \mathbf{S}^{t-1}_{r,s;r',s'} \cdot \mathbf{SC}^t_{r,s;r',s'}}{\sum_{<r',s'> \in \mathcal{HS}^t_{r,s}} \mathbf{S}^{t-1}_{r,s;r',s'}} \tag{3}$$

Notice that $ad^{r,s}_t$ only depicts the anomalous degree for $v^{r,s}_t$, it cannot tell if $v^{r,s}_t$ is abnormally larger than its normal value or smaller than what it should be. To differentiate between those two conditions, we need to add a positive or negative sign for $ad^{r,s}_t$. If $v^{r,s}_t$ is larger than the average value of all its historically similar regions, we multiply $ad^{r,s}_t$ by $+1$ otherwise -1. However, since data of different regions or data sources have different scales, we cannot directly use the average of raw data as the estimation of *normal* value. Instead, we first normalized the

data and use the weighted average of standardized value to decide if a positive sign or negative sign should be added. Basically, we transform the raw data by removing the mean value of each time series, then scale it by dividing by their standard deviation, as Equation 4 shows.

$$v\_scaled_t^{r,s} = \frac{v_t^{r,s} - mean(v_{(t-l+1):t}^{r,s})}{std(v_{(t-l+1):t}^{r,s})} \tag{4}$$

Then we compute the sign for $ad_t^{r,s}$ by the following equation.

$$sign(ad_t^{r,s}) = \begin{cases} +1, & v\_scaled_t^{r,s} > \frac{\sum_{<r',s'>\in \mathcal{HS}_{r,s}^t} S_{r,s;r',s'}^{t-1} \cdot v\_scaled_t^{r',s'}}{\sum_{<r',s'>\in \mathcal{HS}_{r,s}^t} S_{r,s;r',s'}^{t-1}} \\ -1, & \text{otherwise} \end{cases}$$

Finally, the individual anomaly score $score\_ind_t^{r,s}$ is calculated based on the following equation.

$$score\_ind_t^{r,s} = ad_t^{r,s} \times sign(ad_t^{r,s})$$

4 AGGREGATE ANOMALY SCORES

In this section, we introduce how we aggregate the individual anomaly scores (*AIAS*). *AIAS* takes *score_ind*s computed by *CIAS* as input, and find anomalous regions based on the *score_ind*s of all data sources in that region as well as *score_ind*s in nearby regions and consecutive time slots. We adopt *OC-SVM* with *rbf* kernel as detection algorithm because *rbf* kernel introduces higher order terms which explicitly represent the interaction between different data sources and nearby regions. *AIAS* consists of two stages. In the first stage, we compute the anomaly score by *OC-SVM* for each region using *score_ind* of the data sources in that region. We select regions with top anomaly scores as candidate anomalous regions. Then in the second stage, we perform *OC-SVM* again using the *score_ind*s of each region as well as *score_ind*s in its nearby regions and time slots. Candidate regions with top anomaly scores computed in the second stage will be output as final anomalies.

4.1 One Class Support Vector Machine

First, we briefly review the concept of *OC-SVM* algorithm. Suppose we are given a dataset with n d-dimension points where most points are considered *normal*, i.e. drawn from a certain distribution, while a small part of points are *anomalous*, i.e. drawn from other distributions. *OC-SVM* tries to find those anomalous points without explicitly telling it which points are anomalous and which are normal. It is basically an unsupervised version extension of Support Vector Machine. It first maps the data into feature space corresponding to the kernel. Then instead of separating the dataset according to their label, it separates them from the origin $\vec{\mathbf{0}}$ using a hyperplane. The algorithm tries to maximize the distance from this hyperplane to the origin $\vec{\mathbf{0}}$. Additionally, similar to *SVM*, *OC-SVM* also introduces slack variables ξs. Slack variables let *OC-SVM* tolerate some data points (anomalous) falling into the other side of the hyperplane. For a new point $\mathbf{x}$, if it falls in the same side of hyperplane where most training data fall, then $\mathbf{x}$ is marked as a *normal* data points, otherwise anomalous. Moreover, the distance from new point $\mathbf{x}$ to the separating hyperplane can be used as the measurement of degree of anomalousness.

The quadratic programming minimization for *OC-SVM* is slightly different from *SVM*, as Equation 5 and 6 shows.

$$\min_{w,\xi,\rho} \quad \frac{1}{2}\|w\|^2 + \frac{1}{\nu n}\sum_i \xi_i - \rho \tag{5}$$

$$\text{subject to} \quad (w \cdot \Phi(\mathbf{x_i})) \geq \rho - \xi_i, \xi_i \geq 0 \tag{6}$$

Where parameter ν characterizes the fractions of support vectors and outliers. ξs are the nonzero slack variables. Φ is the function that maps original points into high dimensional feature space. The minimization problem also has a dual form where only the inner product of data points are interested, thus we can use all the kernel trick same as *SVM*.

4.2 Methodology

Definition 4.1 (Proportion of anomaly α). α is a manually set parameter. It stands for the expected proportion of anomalous regions for each day (consecutive 24 hours). For example, suppose the length of each time slot is 30 minutes, then there are $60/30 \cdot 24 \cdot n_r < r, t >$ pairs for every 24 hours. Given α, the number of anomalies we expected to report during every 24 hours will be $60/30 \cdot 24 \cdot n_r \cdot \alpha$.

First stage. In the first stage, we aim at finding candidate anomalous regions. These are the regions which are relatively anomalous by only checking the data sources of its own, without considering nearby regions or other time slots. Specifically, for every region r at time slot t, we construct a data point

$$\mathbf{x\_r_t^r} =< score\_ind_t^{r,s_1}, score\_ind_t^{r,s_2}, \ldots, score\_ind_t^{r,s_{n_s}} >$$

$\mathcal{X}_r$ is a set containing all $\mathbf{x_r_t^r}$s. An *OC-SVM* with *rbf* kernel will be trained on $\mathcal{X}_r$. *Rbf* kernel maps the original training data into infinite dimension space. Higher order terms like $score\_ind_t^{r,s_1} \cdot score\_ind_t^{r,s_2}$, $score\_ind_t^{r,s_1} \cdot score\_ind_t^{r,s_2} \cdot score\_ind_t^{r,s_4}$ explicitly model the combinations of changes in multiple data sources, enabling the algorithm to identify underlying anomalies, e.g. the one specified in Figure 1a.

For each time slot T, we first construct data points $\mathbf{x\_r_T^r}$ for $r \in [1, n_r]$. Then we use the previously trained *OC-SVM* model to evaluate an anomaly score $score\_r_T^r$ for each region, i.e. the distance from $\mathbf{x\_r_T^r}$ to the separating hyperplane. $\mathbf{x_r_T^r}$s will be added to $\mathcal{X}_r$ for future training. Next, we select top-β $score\_r_t^r$ during the last 24 hours and their corresponding $< r, t >$ index as the candidate anomalous regions. Here β is a parameter larger than α. For example, in the experiments setting, we set α to 1% and β to 5%. Formally, the candidate set C is given by

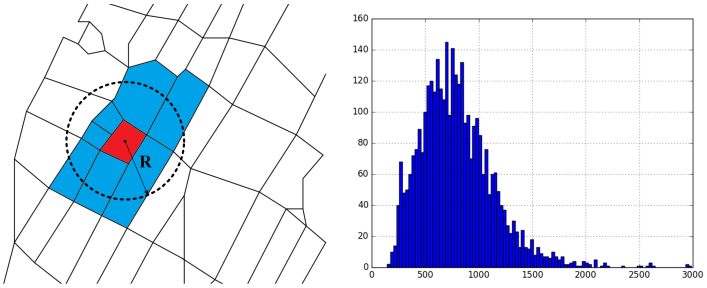
$$C = \underset{r,t}{\mathbf{argmax}}[l \cdot n_r \cdot \beta] \{score\_r_t^r | r \in [1, n_r], t \in [T - l + 1, T]\}$$

Where l is the number of time slots which can cover 24 hours, $\mathbf{argmax[k]}$ stands for the function which returns the index of largest k values.

Second stage. In this stage, we take nearby regions and consecutive time slots into consideration. The difference with the first stage lies in the construction of data points. Specifically, for every region r at time slot t, we construct a data point

$$\begin{aligned}\mathbf{x\_int_t^r} =< & score\_ind_t^{r,s_1}, score\_ind_t^{r,s_2}, \ldots, score\_ind_t^{r,s_{n_s}}, \\ & score\_ind_{t-1}^{r,s_1}, score\_ind_{t-1}^{r,s_2}, \ldots, score\_ind_{t-1}^{r,s_{n_s}}, \ldots, \\ & score\_ind_{t-t_\Delta+1}^{r,s_1}, \ldots, score\_ind_{t-t_\Delta+1}^{r,s_{n_s}}, \\ & score\_nearby_t^{r,s_1}, score\_nearby_t^{r,s_2}, \ldots, score\_nearby_t^{r,s_{n_s}} >\end{aligned}$$

Line 1 is exactly the same as $\mathbf{x\_r_t^r}$. Line 2 to Line 3 are the *score_ind*s of region r in previous $t_\Delta - 1$ time slots, where t_Δ is a manually set parameter. t_Δ represents the number of consecutive time slots we want to check. Line 4 stands for the *score_ind*s of r's nearby regions. $score\_nearby_t^{r,s}$ is the average of $score_ind_t^{r',s}$s, where r' are the neighbors of r. As illustrated in Figure 6a, two regions are regarded as neighbors if their centers are within distance R. R is set empirically depending on the dataset. Take regions in NYC as example, we plot the histogram of distances between each region and its 5 nearest regions in Figure 6b. Since most distances are around $800m$, we set R to $800m$ for NYC dataset. Similar to $\mathcal{X}_r$, $\mathcal{X}_{int}$ denotes the set containing all $\mathbf{x_int_t^r}$s.



(a) Neighbor of a region (b) Histogram of distances

Fig. 6. Settings of *score_nearby*

For each time slot T, we follow the same process as stage one and get $score_int^r_T$s through an *OC-SVM* trained on $\mathcal{X}_{int}$. Finally, we sort $score\_int^r_t$ for all $< r, t >$ in the candidate set C and select $< r, t >$ with highest scores. $< r, t >$ with $t = T$ will be identified as anomalies at time slot T. Formally,

$$\mathcal{A} = \{r| < r, T > \in \underset{r,t}{\textbf{argmax}}[l \cdot n_r \cdot \alpha] \{score\_int^r_t| < r, t > \in C\}\} \tag{7}$$

Notice that if we only perform the second stage and mark regions with highest *score_int*s as anomalies, we may select regions with low *score_ind*s but high *score_nearby*s. This conflicts with our original goal. We leverage the values of nearby regions and time slots in order to better evaluate the anomaly degree of each region. If a region is anomalous, the anomaly degree within the region itself should at least be relatively high.

Algorithm 1 summarizes *AIAS*. In practice, we retrain the *OC-SVM* in a daily basis.

Algorithm 1: Aggregating Individual Anomaly Scores

input : *score_ind*, current time slot T
output: Anomalous regions, $\mathcal{A}$

Construct $\mathbf{x\_r^r_T}$ for $r \in [1, n_r]$;
$\mathcal{X}_r \leftarrow \mathcal{X}_r \cup \{\mathbf{x\_r^r_T} | r \in [1, n_r]\}$;
$score\_r^r_T \leftarrow OCSVM_r.predict(\mathbf{x\_r^r_T})$ for $r \in [1, n_r]$;
$C \leftarrow \underset{r,t}{\textbf{argmax}}[l \cdot n_r \cdot \beta] \{score\_r^r_t | r \in [1, n_r], t \in [T - l + 1, T]\}$;
Construct $\mathbf{x\_int^r_T}$ for $r \in [1, n_r]$;
$\mathcal{X}_{int} \leftarrow \mathcal{X}_{int} \cup \{\mathbf{x\_int^r_T} | r \in [1, n_r]\}$;
$score\_int^r_T \leftarrow OCSVM_{int}.predict(\mathbf{x\_int^r_T})$ for $r \in [1, n_r]$;
$\mathcal{A} \leftarrow \{r| < r, T > \in \underset{r,t}{\textbf{argmax}}[l \cdot n_r \cdot \alpha] \{score\_int^r_t| < r, t > \in C\}\}$;
$OCSVM_r.train(\mathcal{X}_r)$;
$OCSVM_{int}.train(\mathcal{X}_{int})$;
return $\mathcal{A}$

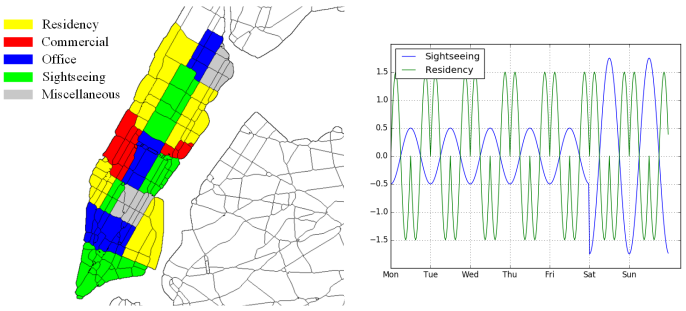
5 EXPERIMENTS

We conduct experiments on both synthetic and real world datasets to show the effectiveness of our method. Due to lack of labeled urban anomalous events, we first conduct quantitative analysis on synthetic datasets. Data

sources are generated manually and several types of anomalies are injected. Those anomalies serve as the ground truth. We also evaluate our method on real world event detection task and perform two case studies on real world data to show some of the advantages of our method. Code and data are available on Github[1].

5.1 Experiments Based on Synthetic Datasets

5.1.1 Data Generation. We generate two data sources, taxi and bike flow, on the map of Manhattan. To mimic real world situation, we divide Manhattan into five functional regions. Notice that the partition is not exactly the same as real world, we just used Manhattan as a map. As shown in Figure 7a, regions are marked by color, each of which represents a certain function in city. For each function, we first design a base curve (standardized) for both taxi and bike flows. We try to simulate the behavior of each functional region in real world. As shown in Figure 7b, we plot the base curve for *sightseeing* and *residency* area. We let traffic flow in *sightseeing* area be high at weekends and low at weekdays. For *residency* area, we make the traffic flow in peak hours higher than other. Besides the base curve, we also design three types of anomalies and two types of external influences, as shown in Table 1. We will show that our method can robustly detect anomalies without knowing the external influences and its effects on data. The anomalies and external influences will be randomly injected to the base curve. After that, we add a Gaussian noise with mean 0 and standard deviation 0.03. Finally, we randomly draw *mean* and *std* for taxi and bike flow for each region from a Gaussian distribution and use them to transform the base curve to final values. For the experiments, we generate 6 weeks of data. We randomly inject 240 anomalies for each anomaly type in the first 4 weeks and 120 each in the last 2 weeks. Besides, we randomly inject 3 rainy days in the first 4 weeks and 3 in the last 2 weeks. Also, we randomly inject 4 holidays in the first 4 weeks and 3 in the last 2 weeks. The method will be trained upon the first 4 weeks and tested on the last 2 weeks.



(a) Partition (b) Base curve

Fig. 7. Synthetic data generation

5.1.2 Baselines and Metric. We first compare our method with three classical and widely used anomaly detection methods. Since we are dealing with urban data, we follow the traditional process [5] to divide data by region r, data source s, time slot t and *weekends* or *weekdays*. For example, data with same r,s,t and are all collected *weekdays* will be grouped together. Anomaly detection method will be conducted on each group. We compare with the following baselines.

- *EE*: Fit an elliptic envelope and use Mahalanobis distance as the anomaly score. [17]
- *Neighbor*: Use the Euclidean distance between the values of a region and the mean value of its nearby regions as the anomaly score.

[1]https://github.com/uc-zhc/DetectUrbanAnomalies

Table 1. Anomalies and external influences

Anomalies	**Effect**
ID	increase bike flow by 0.15 and decrease taxi flow by 0.15
TS	increase or decrease the taxi or bike flow by 0.15 for two consecutive time slots
R	increase or decrease the taxi or bike flow by 0.15 for region r and regions within $800m$
External Influences	**Effect**
Rain	Decrease all traffic flow by 0.5 in one day
H	Mimic holiday, decrease traffic flow in office area by 0.5 and increase traffic in sightseeing and commercial by 0.5

- *LRT*: Fit a Poisson distribution on historical data and use likelihood ratio test as the anomaly score.

Besides the classical baselines, we also show the effectiveness of both components *CIAS* and *AIAS* by substituting or simplifying the components. Also, we show the advantages of checking multiple data sources, nearby regions and time slots. Specifically, we compare our full method with the following baselines.

- $ind + int_{taxi}$: Perform the full method but with only taxi flow.
- $ind + int_{bike}$: Perform the full method but with only bike data.
- $ind + int_s$: Do *CIAS* and only perform the first stage of *AIAS*, i.e. only check *score_ind* of all data sources of a region, without considering nearby regions or time slots.
- $ind + int_{str}$: Do *CIAS* and only perform the second stage of *AIAS*, i.e. without first selecting candidate anomalous regions.
- $raw + int$: Use raw values instead of *score_ind*s computed by *CIAS* as the input of *AIAS*
- *ARIMA*+int: Use prediction method to evaluate *score_ind*s instead of *CIAS*, notice that external information like *Rain* and *H* are not available for the prediction algorithm.
- $ind + int$: Full method

We use the hit rate (recall) of anomalies injected as main metric. Also, we evaluate parameter sensitivity of our method and compare with other baseline methods using F1-Score.

5.1.3 Settings. The threshold for historically similar regions θ is set to 0.8, since a Pearson Correlation Coefficient larger than 0.8 is considered as strongly correlated according to [32]. R which defines the neighbors of each region is set to $800m$ as mentioned in section 4. Number of consecutive time slots considered t_Δ is set to 2. The expected proportion of anomalous regions α is set to 1% and β is set to 5%. Note that t_Δ and α can be adjusted to meet the need for different real world scenario. If more anomalies are expected then α can be increased. For *OC-SVM*, we set hyperparameter ν to 0.1 as it is used in the original paper [19].

5.1.4 Results. Table 2 shows the results of different baselines and our method. As we can see, the full method $ind + int$ gets the highest overall hit rate. It detects 91.06% of all types of anomalies and remain high hit rate even when there are external influences, i.e. Rain and Holiday. For the first three classical methods, the performance is extremely poor, indicating that classical anomaly detection methods are not capable of handling the complex nature of urban data. For the *ID* anomaly, $ind + int_s$ performs slightly better than the full method, because it focuses on detecting anomalous patterns of region r, regardless of the situation of nearby regions and time slots. The performance of $ind + int_{str}$ is not satisfying, which proofs that without selecting candidate anomalous regions, one may incorrectly mark regions as anomalies due to the anomaly scores of nearby regions or time slots. The low hit rate in $ind + int_{taxi}$ and $ind + int_{bike}$ shows the importance of detecting anomalies using multiple data sources. Poor performance of $raw + int$ indicates that it is necessary to first transform raw urban data into

some kind of anomaly scores. One can't directly apply classic anomaly detection method on raw urban data. Lower hit rate in *ARIMA + int* shows the advantages of similarity-based *CIAS*. We are able to recover the true anomaly degree even when external influences are unknown, while prediction model will certainly fail.

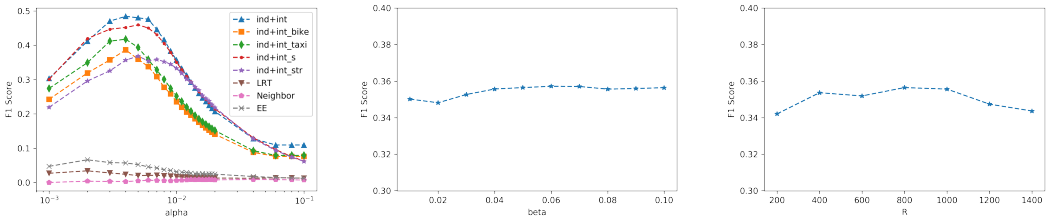
Table 2. Comparison between baselines and our method on synthetic data. For the definition of *ID,TS,R,Rain,H* please refer to Table 1

anomalies count		ID 120	R 120	TS 120	ID,TS,R 358	ID,TS,R(w/ Rain) 80	ID,TS,R(w/ H) 45
EE	hit	10	15	9	34	7	11
	hit rate	8.33%	12.50%	7.50%	9.50%	8.75%	24.44%
Neighbor	hit	3	1	2	6	3	1
	hit rate	2.50%	0.83%	1.67%	1.68%	3.75%	2.22%
LRT	hit	8	6	3	17	10	2
	hit rate	6.67%	5.00%	2.50%	4.75%	12.50%	4.44%
$ind + int_{taxi}$	hit	97	55	64	216	50	31
	hit rate	80.83%	45.83%	53.33%	60.34%	62.50%	68.89%
$ind + int_{bike}$	hit	107	57	51	214	51	24
	hit rate	89.17%	47.50%	42.50%	59.78%	63.75%	53.33%
$ind + int_{s}$	hit	117	94	90	300	64	40
	hit rate	**97.50%**	78.33%	75.00%	83.80%	80.00%	88.89%
$ind + int_{str}$	hit	103	91	93	285	63	38
	hit rate	85.83%	75.83%	77.50%	79.61%	78.75%	84.44%
raw + int	hit	6	9	7	22	3	3
	hit rate	5.00%	7.50%	5.83%	6.15%	3.75%	6.67%
ARIMA + int	hit	3	3	1	7	5	0
	hit rate	2.50%	2.50%	0.83%	1.96%	6.25%	0.00%
ind + int	hit	114	106	107	326	72	42
	hit rate	95.00%	**88.33%**	**89.17%**	**91.06%**	**90.00%**	**93.33%**

5.1.5 Parameter Analysis. In this section, we evaluate the performance of our method under different parameter settings, i.e. α, β and R. The result is shown in Figure 8. α controls the expected proportion of anomalous regions each day. Larger α will result in more anomalies as well as false alarms, smaller α will result in less anomalies and lower hit rate. It is similar to the threshold parameter in other anomaly detection algorithms. As shown in Figure 8a, we test different α values, i.e. $\{0.001, 0.002, \ldots, 0.019, 0.02, 0.04, \ldots, 0.08, 0.1\}$, no matter how we choose α, our method *ind + int* consistently outperforms all other baselines with respect to F1-Score. β controls the proportion of candidate anomalous regions. We fix α to 0.01 and test with β ranging from 0.01 to 0.10. As shown in Figure 8b, β does not largely impact the performance, this is mainly because candidate anomalous regions with high anomaly scores are very likely to be real anomalies. But we do see that best performance is achieved when β is 5 or 6 times α. R defines the neighbor of a region. As shown in Figure 8c, we achieve highest F1-Score when R is $800m$, which corresponds with the underlying ground truth (see Table 1). However, other choices of R also show comparable performance with the best choice.

5.2 Experiments Based on Real World Datasets

5.2.1 Datasets. We evaluate our method on two real world datasets generated in New York City:
1) *Taxi data*: This dataset is generated by over 14,000 taxicabs in NYC. Data includes pick-up and drop-off locations

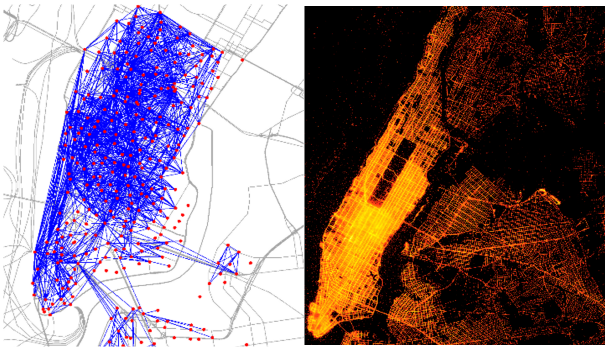


(a) α (b) β (c) R

Fig. 8. Parameter Analysis

and times, the duration and distance of each trip, taxi ID and the number of passengers for each trip.
2) *Bike renting data*: The data is generated by the bike sharing system in NYC, which has 340 bike stations and about 7,000 bikes. Each record in the data includes the time, bike ID, station ID, and an indication of check-out or return. The location of each station is also disclosed to the public.

Figure 9 presents the geographical distributions of the taxi and bike on a digital map. As shown in Figure 9a, each red point stands for a bike station and a blue edge denotes the aggregation of bike commutes between two stations. Figure 9b is a heat map of the drop-off and pickup points of all the taxi trips. The lighter the denser.



(a) Bike data (b) Taxi data

Fig. 9. Visualization of data

5.2.2 Settings. We partition each day into 48 30-minutes time slots and count the volume of taxi and bike flow for each time slot. For other parameters, we follow exactly the same setting in the synthetic experiments.

5.2.3 Results. Quantitatively evaluate urban anomaly detection algorithm on real world settings is an open challenge, mainly because it is impossible to achieve a full set of ground truth, leading to inaccurate false alarm rate. In this paper, we evaluate our method by the same procedure done in a previous work [31], where they correlated the anomalies detected with the events reported by *nycinsiderguide*[2] during Nov. 1st 2014 to Nov. 30th 2014. Time and location of the 20 reported events are listed in Table 3.

[2]https://www.nycinsiderguide.com/

Table 4 shows the results of our method and different baselines. Following the same procedure in [31], a detected anomaly is regarded as a correct recall if the anomaly has an overlap with a reported event in both spatial and temporal spaces. Among all methods, our method *ind + int* detects 12 out of 20 events, outperforms all other baselines. Lower hit rate of $ind + int_{taxi}$ and $ind + int_{bike}$ demonstrates the importance of using multiple data sources. Notice that in this experiment, all we get is the taxi and bike flow data, the result shows that our method is able to detect real world events without knowing any information about external influences (e.g. working day, holiday, weather).

Table 3. Events reported by *nycinsiderguide*

	Event Name	Address	Start Time	End Time
1	Bowlloween 2014 New York Halloween	624 W 42nd St	10/31 9PM	11/1 2AM
2	Largest Halloween Singles Party in NYC	247 W 37th St	10/31 7AM	11/1 3AM
3	Kokun Cashmere Sample and Stock Sale	237 W 37th St	11/5 10:30AM	11/1 3AM
4	Big Apple File Festival	54 Varick St	11/5 6PM	11/9 11PM
5	InterHarmony Concert: The Soul of Elegiaque	881 7th Ave	11/6 8PM	11/6 10PM
6	Hiras Master Tailors New York Trunk Show	301 Park Ave	11/6 9AM	11/9 1PM
7	First Fridays!	881 7th Ave	11/7 6PM	11/7 10PM
8	Thomas/Ortiz Dance Show	248 W 60th St	11/7 7PM	11/8 9PM
9	Rebecca Taylor Sample Sale	260 5th Ave	11/11 10AM	11/15 8PM
10	The News NYC Sample Sale	495 Broadway	11/13 9AM	11/15 6AM
11	Giorgio Armani Sample Sale	317 W 33rd St	11/15 9:30AM	11/19 6:30PM
12	Get Buzzed 4 Good Charity Event NYC	200 5th Ave	11/15 1PM	11/15 4PM
13	Ment' or Young Chef Competition	462 Broadway	11/15 2PM	11/15 6PM
14	Gotham Comedy Club	208 W 23rd St	11/17 6PM	11/17 9PM
15	Kal Rieman NYC Sample Sale	265 W 37th St	11/18 11AM	11/20 8PM
16	Inhabit Cashmere Sample Sale	250 W 39th St	11/18 10AM	11/20 6PM
17	Shoshanna NYC Sample Sale	231 W 39th St	11/19 10AM	11/20 6:30PM
18	ICB/J. Press NYC Sample Sale	530 7th Ave	11/19 12AM	11/21 12AM
19	Thanksgiving in New York City 2014	1675 Broadway	11/27 6AM	11/27 10PM
20	Thanksgiving Day Dinner	108 W 40th St	11/27 12PM	11/27 9PM

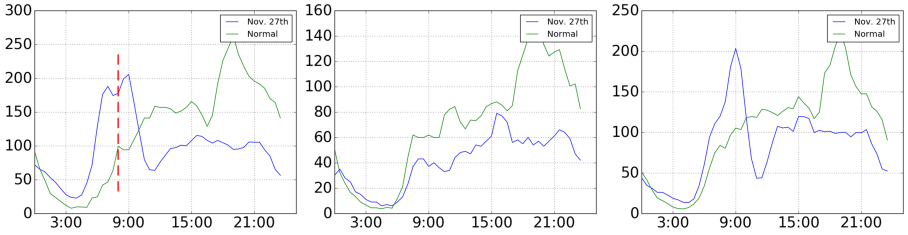
Table 4. Results on real world event detection

	EE	*Neighbor*	*LRT*	*ind+int*	*ind+int*	*raw+int*	*ind+int*
hit	5	2	5	10	9	4	**12**
hit rate	25%	10%	25%	50%	45%	20%	**60%**

5.2.4 Case Studies. Among the all anomalies detected, we selected two representative cases to illustrate the advantages of our method.

Macy's Thanksgiving Day Parade. The first example is the famous macy's thanksgiving day parade. The parade started at 9:00a.m., Nov. 27th, 2014. As Figure 10a shows, our algorithm raised alert for anomaly at 8:00a.m. The following are two main advantages of our algorithm implied by this example:

1) **The algorithm is capable of finding anomalies even when external environment changes.** On Thanksgiving Day, traffic flow will differ greatly from regular Thursdays. The algorithm is totally unaware of the Thanksgiving Day, all it had is the raw data. However, with the similarity-based *CIAS*, our method is capable of handling these kind of external influences and evaluate anomalies based on the *normal* values on Thanksgiving Day. Figure 10a shows the normal values and values on Thanksgiving Day for the detected region r, while Figure 10b shows the same information for regions which are historically similar to it. As we can see, data for region r clearly differs from its historically similar regions. It supposed to decrease like its historically similar regions on Thanksgiving Day, however, due to the Macy's Thanksgiving Day Parade, values in region r increased. Our method captures this difference and successfully detect the anomaly.
2) **Checking nearby regions and consecutive time slots helps us detect anomalies before it reaches its peak.** We detect the anomaly one hour earlier than the parade started, before the anomaly reaches its peak during 8:30a.m. to 9:00a.m., as shown in Figure 10a. We plot the traffic flow on the region near region r in Figure 10c. Although it is not the starting point of the parade, it still shows a deviation from its *normal* value. The anomaly scores in those nearby regions, along with the scores at previous time slot, i.e. 7:00a.m.-7:30a.m., helps us confidently mark the region as an anomalous region during 7:30a.m.-8:00a.m. If we only check the values of a single region and time slot, we may not be able to detect the anomaly as earlier as we can now.



(a) r (b) Similar region (c) Nearby region

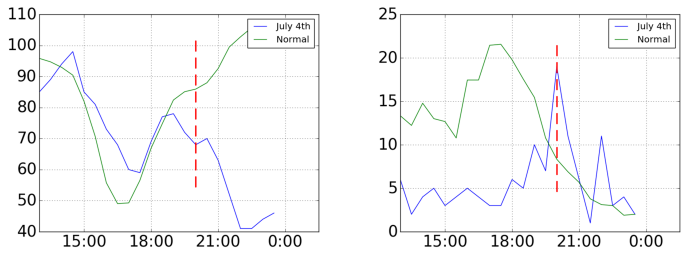
Fig. 10. Taxi flow for the parade

July 4^{th} Firework Show. The second example is the July 4^{th} Firework Show which took place along East River at 9:00p.m., on Independence Day. This case also shows two strong points of our method, which are as follows:
1) The first point is similar to the first case. However, besides the difference between the Independence Day and regular Fridays, it was also raining on that day which would clearly affect the taxi and bike flow. Our algorithm is still capable of detecting anomalies even when there are complex external influences.
2) **Discovering rare patterns in multiple data sources helps us better detect anomalies.** As Figure 11 shows, the taxi flow of region r decreased, while the bike flow, in contrary, increased. This kind of pattern of taxi and bike flow rarely happened, thus the algorithm confidently report region r as an anomaly. The underlying reason for this kind of pattern is that on that day, some roads of region r were closed due to firework show.

6 RELATED WORK

6.1 General Anomaly Detection

Anomaly detection has been studied extensively in the past decades [2][8]. Most anomaly detection method [24][14][20][16][25][28], focus on building a unified model to detect anomalies. In their problem settings, e.g. network anomaly detection, anomalous image classifying, the definition of *normal* data points is static.



(a) Taxi flow (b) Bike flow

Fig. 11. Taxi and bike flow for firework show

In our work, we focus on urban anomaly detection where data are influenced by many factors, e.g. the location and time collected, weather, etc. The underlying distribution which generates *normal* data may vary greatly. Thus the above mentioned methods are not suitable for urban anomaly detection.

6.2 Urban Anomaly Detection

Some previous works focus on urban anomaly detection with spatio-temporal data, e.g. detecting anomalous trajectories [10][30][1], identifying traffic anomalies based on trajectories [15], diagnosing traffic anomalies [3], and gleaning problematic design in urban planning [13]. However, they try to detect anomalies using only one data source and cannot be directly applied to multiple data sources. In [31][27][22], they detect spatio-temporal anomalies based on multiple data sources. The idea of their method is to first build a prediction or probabilistic model, then detect anomalies based on that model. The effectiveness of these methods largely depend on the accuracy of the prediction model. For example, in [27][22][23], they built probabilistic graphical model to predict the underlying distribution of a new value. However, these methods are not able to handle complex external influences in city. In [31], they trained prediction model on *weekdays* and *holidays* respectively, but they did not model the effect of weather and other conditions.

In this work, we use multiple data sources simultaneously to identify urban anomalies. Also, we consider combinations of changes in multiple data sources and check nearby regions and consecutive time slots to help us detect anomalies better and earlier. Moreover, with the help of similarity based *CIAS*, our model can accurately detect anomalies without any additional environmental information, such as weather, holiday, etc.

6.3 Similarity-Based Anomaly Detection Method

Similarity-based anomaly detection approaches [9][6] have generated much interest due to their relative simplicity and robustness as compared to model-based, cluster-based, and density-based approaches, especially in the filed of time series anomaly detection [4][21]. These methods involves choosing a proper similarity measurement and identify anomalies based on the similarity or dissimilarities between data samples.

In our work, we don't directly use the similarity measurement as a criteria for identifying anomalies. Instead, we use the change (sudden drop) in Pearson Correlation Coefficient with other historically similar regions as the measurement for anomaly score. The anomaly score computed by our approach is still accurate even when external environment changes. One previous work [11] also considered change in similarity measurement to evaluate the anomaly degree of vehicle traffic data. However, the similarity measurement in their method is the $L - \infty$ distance, which cannot accurately represents the true similarity between two time series.

7 CONCLUSION AND FUTURE WORK

In this paper, we propose a two-step method to detect urban anomalies using multiple spatio-temporal data sources. We propose the similarity-based *CIAS* to give an anomaly score for each data source of each region at each time slot. Instead of building a prediction model, we use historically similar regions as guides. This enables us to evaluate how much the value deviates from what it should be, without knowing any information other than the data itself, e.g. POI of region, holiday, weather, etc. These anomaly scores are fed into step two, *AIAS*. In *AIAS*, we leverage the power of *OC-SVM* and *rbf* kernel, and detect anomalies based on rare patterns of the individual anomaly scores of a region as well as its nearby regions and previous time slots.

We conduct quantitative experiments on synthetic datasets. Data sources are generated by simulating real world behavior and several types of anomalies and external influences are injected. These injected anomalies serve as the ground truth. The result shows the advantages beyond baseline techniques. It also shows the effectiveness of both components of our algorithm. We also demonstrate the main advantages of our method through event detection task and two case studies on real world datasets, i.e. being able to detect anomalies even with unknown complex external influences, being able to detect anomalies before it reaches its peak and being able to detect underlying anomalies.

There are several improvements and potential extensions that could be addressed in future work. For *CIAS* part, in this paper, we only consider sudden drop in Pearson Coefficient Correlation with historically similar regions as the indicator of anomaly. However, if values in two regions are consistently different, i.e. consistently negative Pearson Coefficient Correlation, but after new observations *PCC* rises significantly, this may also be a sign of anomaly. For *AIAS* part, it would be interesting to see if recent advances in Generative Adversarial Nets [7] performs better than traditional methods like One-Class SVM. For example, Schlegl et. al proposed *AnoGAN* and demonstrated the ability to detect more known anomalies and even novel ones in an unsupervised manner[18]. With the help of generator-discriminator framework, *GAN* based methods might capture more complex form of anomalies. Also notice that, in this paper we evaluated our method on two real world datasets, i.e. taxi and bike flow, due to limited data availability. In fact, our method can be naturally applied to any real-valued time series data, such as volume of cellular signals in base stations or number of yelp check-in records. We would release code and documents for organizations with richer data to detect anomalies based on their own data.

REFERENCES

[1] Prithu Banerjee, Pranali Yawalkar, and Sayan Ranu. 2016. MANTRA: A Scalable Approach to Mining Temporally Anomalous Sub-trajectories. In *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*. ACM, 1415–1424.

[2] Varun Chandola, Arindam Banerjee, and Vipin Kumar. 2009. Anomaly detection: A survey. *ACM computing surveys (CSUR)* 41, 3 (2009), 15.

[3] Sanjay Chawla, Yu Zheng, and Jiafeng Hu. 2012. Inferring the root cause in road traffic anomalies. In *Data Mining (ICDM), 2012 IEEE 12th International Conference on*. IEEE, 141–150.

[4] Deepthi Cheboli. 2010. Anomaly detection of time series. (2010).

[5] Harish Doraiswamy, Nivan Ferreira, Theodoros Damoulas, Juliana Freire, and Claudio T Silva. 2014. Using topological analysis to support event-guided exploration in urban data. *IEEE transactions on visualization and computer graphics* 20, 12 (2014), 2634–2643.

[6] Zhouyu Fu, Weiming Hu, and Tieniu Tan. 2005. Similarity based vehicle trajectory clustering and anomaly detection. In *Image Processing, 2005. ICIP 2005. IEEE International Conference on*, Vol. 2. IEEE, II–602.

[7] Ian Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio. 2014. Generative adversarial nets. In *Advances in neural information processing systems*. 2672–2680.

[8] Manish Gupta, Jing Gao, Charu C Aggarwal, and Jiawei Han. 2014. Outlier detection for temporal data: A survey. *IEEE Transactions on Knowledge and Data Engineering* 26, 9 (2014), 2250–2267.

[9] Ko-Jen Hsiao, Kevin S Xu, Jeff Calder, and Alfred O Hero. 2016. Multicriteria Similarity-Based Anomaly Detection Using Pareto Depth Analysis. *IEEE transactions on neural networks and learning systems* 27, 6 (2016), 1307–1321.

[10] Jae-Gil Lee, Jiawei Han, and Xiaolei Li. 2008. Trajectory outlier detection: A partition-and-detect framework. In *Data Engineering, 2008. ICDE 2008. IEEE 24th International Conference on*. IEEE, 140–149.

[11] Xiaolei Li, Zhenhui Li, Jiawei Han, and Jae-Gil Lee. 2009. Temporal outlier detection in vehicle traffic data. In *Data Engineering, 2009. ICDE'09. IEEE 25th International Conference on*. IEEE, 1319–1322.
[12] Yuhong Li, Man Lung Yiu, Zhiguo Gong, et al. 2013. Discovering longest-lasting correlation in sequence databases. *Proceedings of the VLDB Endowment* 6, 14 (2013), 1666–1677.
[13] Wei Liu, Yu Zheng, Sanjay Chawla, Jing Yuan, and Xie Xing. 2011. Discovering spatio-temporal causal interactions in traffic data streams. In *Proceedings of the 17th ACM SIGKDD international conference on Knowledge discovery and data mining*. ACM, 1010–1018.
[14] Igor Melnyk, Arindam Banerjee, Bryan Matthews, and Nikunj Oza. 2016. Semi-Markov switching vector autoregressive model-based anomaly detection in aviation systems. *arXiv preprint arXiv:1602.06550* (2016).
[15] Linsey Xiaolin Pang, Sanjay Chawla, Wei Liu, and Yu Zheng. 2011. On mining anomalous patterns in road traffic streams. In *International Conference on Advanced Data Mining and Applications*. Springer, 237–251.
[16] Ioannis Ch Paschalidis and Georgios Smaragdakis. 2009. Spatio-temporal network anomaly detection by assessing deviations of empirical measures. *IEEE/ACM Transactions on Networking (TON)* 17, 3 (2009), 685–697.
[17] Peter J Rousseeuw and Katrien Van Driessen. 1999. A fast algorithm for the minimum covariance determinant estimator. *Technometrics* 41, 3 (1999), 212–223.
[18] Thomas Schlegl, Philipp Seeböck, Sebastian M Waldstein, Ursula Schmidt-Erfurth, and Georg Langs. 2017. Unsupervised Anomaly Detection with Generative Adversarial Networks to Guide Marker Discovery. In *International Conference on Information Processing in Medical Imaging*. Springer, 146–157.
[19] Bernhard Schölkopf, Robert C Williamson, Alexander J Smola, John Shawe-Taylor, John C Platt, et al. 1999. Support vector method for novelty detection.. In *NIPS*, Vol. 12. 582–588.
[20] A Shevyrnogov, G Vysotskaya, and E Shevyrnogov. 2004. Spatial and temporal anomalies of sea surface temperature in global scale (by space-based data). *Advances in Space Research* 33, 7 (2004), 1179–1183.
[21] Martin Steiger, Jürgen Bernard, Sebastian Mittelstädt, Hendrik Lücke-Tieke, Daniel Keim, Thorsten May, and Jörn Kohlhammer. 2014. Visual Analysis of Time-Series Similarities for Anomaly Detection in Sensor Networks. In *Computer Graphics Forum*, Vol. 33. Wiley Online Library, 401–410.
[22] X Rosalind Wang, Joseph T Lizier, Oliver Obst, Mikhail Prokopenko, and Peter Wang. 2008. Spatiotemporal anomaly detection in gas monitoring sensor networks. In *Wireless Sensor Networks*. Springer, 90–105.
[23] Apichon Witayangkurn, Teerayut Horanont, Yoshihide Sekimoto, and Ryosuke Shibasaki. 2013. Anomalous event detection on large-scale gps data from mobile phones using hidden markov model and cloud platform. In *Proceedings of the 2013 ACM conference on Pervasive and ubiquitous computing adjunct publication*. ACM, 1219–1228.
[24] Liang Xiong, Barnabás Póczos, and Jeff G Schneider. 2011. Group anomaly detection using flexible genre models. In *Advances in neural information processing systems*. 1071–1079.
[25] Liang Xiong, Barnabás Póczos, Jeff G Schneider, Andrew J Connolly, and Jake VanderPlas. 2011. Hierarchical Probabilistic Models for Group Anomaly Detection.. In *AISTATS*. 789–797.
[26] Su Yang and Wenbin Zhou. 2011. Anomaly detection on collective moving patterns: Manifold learning based analysis of traffic streams. In *Privacy, Security, Risk and Trust (PASSAT) and 2011 IEEE Third Inernational Conference on Social Computing (SocialCom), 2011 IEEE Third International Conference on*. IEEE, 704–707.
[27] Jie Yin, Derek Hao Hu, and Qiang Yang. 2009. Spatio-Temporal Event Detection Using Dynamic Conditional Random Fields.. In *IJCAI*, Vol. 9. 1321–1327.
[28] William Chad Young, Joshua E Blumenstock, Emily B Fox, and Tyler H McCormick. 2014. Detecting and classifying anomalous behavior in spatiotemporal network data. In *Proceedings of the 2014 KDD workshop on learning about emergencies from social information (KDD-LESI 2014)*. 29–33.
[29] Nicholas Jing Yuan, Yu Zheng, and Xing Xie. 2012. Segmentation of urban areas using road networks. *MSR-TR-2012–65, Tech. Rep.* (2012).
[30] Daqing Zhang, Nan Li, Zhi-Hua Zhou, Chao Chen, Lin Sun, and Shijian Li. 2011. iBAT: detecting anomalous taxi trajectories from GPS traces. In *Proceedings of the 13th international conference on Ubiquitous computing*. ACM, 99–108.
[31] Yu Zheng, Huichu Zhang, and Yong Yu. 2015. Detecting collective anomalies from multiple spatio-temporal datasets across different domains. In *Proceedings of the 23rd SIGSPATIAL International Conference on Advances in Geographic Information Systems*. ACM, 2.
[32] Kelly H Zou, Kemal Tuncali, and Stuart G Silverman. 2003. Correlation and simple linear regression. *Radiology* 227, 3 (2003), 617–628.

Received August 2017; revised November 2017; accepted January 2018