Leveraging Spatio-Temporal Clustering for Participatory Urban Infrastructure Monitoring

Matthias Budde, Julio De Melo Borges, Stefan Tomov, Till Riedel, Michael Beigl
Karlsruhe Institute of Technology (KIT), TECO / Pervasive Computing Systems, Karlsruhe, Germany
email: {budde, borges, tomov, riedel, michael}@teco.edu

ABSTRACT

Internet-enabled, location aware smart phones with sensor inputs have led to novel applications exploiting unprecedented high levels of citizen participation in dense metropolitan areas. Especially the possibility to make oneself heard on issues, such as broken traffic lights, potholes or garbage, has led to a high degree of participation in Urban Infrastructure Monitoring. However, duplicate reporting by citizens leads to bottlenecks in manual processing by municipal authorities. Spatio-temporal clustering can serve as an essential tool to group and rank similar reports. Current data mining techniques could be used by municipal departments for this task, but the mandatory parameter selection can be unintuitive, time consuming and error-prone. In this work, we therefore present a novel framework for clustering spatio-temporal data. We first apply an intuitive transformation of the data into a graph structure and subsequently use well-established parameter-free graph clustering techniques to detect and group spatio-temporally close reports. We evaluate our method on two real-world data-sets from different mobile issue tracking platforms. As one of the datasets includes labels for duplicate reports, we can show how our framework outperforms existing techniques in our exemplary use-case (duplicate detection).

Keywords

Crowdsourcing; Spatio-temporal Clustering; Duplicate Detection; Civic Issue Tracking; Issue Ranking;

1. INTRODUCTION

Participatory Sensing [2] has enabled a multitude of novel applications in recent years, ranging from collaborative noise pollution maps [16] to automatically characterizing places [3]. The crowdsourced gathering of content is becoming easier and faster as mobile and pervasive technology continues to spread [18]. Instead of merely being a data collection paradigm, Participatory Sensing presents a powerful tool for harnessing civic engagement. Many different platforms for crowdsourced, mobile *Participatory Infrastructure Monitoring* or *Civic Issue Reporting* have emerged in the last years and enable citizens to make themselves heard and let them actively shape and connect to the urban spaces they live in.

Urb-IoT 2014, October 27-28, Rome, Italy

DOI 10.4108/icst.urb-iot.2014.257282

FixMyStreet [8], *SeeClickFix* [10], *CitySourced* or *KA-Feedback* [4] are just some examples.

Citizens have a strong intrinsic motivation to enhance their living environment. As argued by FENNEL [6], crowdsourcing systems paradoxically may be negatively affected by a high degree of participation. Issue reporting will only be useful if the flow of records does not exceed the receiving entity's capacity to process or respond to reports. However, an increased data flow is often caused by submitted records being duplicates rather than complements:

- *The same* citizen may repeatedly report the same issue to emphasize the perceived urgency of an issue.
- *Different* users may see and report the same issue at different times and/or categorize the issue differently.

We confirmed that the frequent occurrence of duplicate reports is in fact an actual problem in the underlying real-word systems by getting in contact with both the operator of a large platform, *SeeClickFix*[1], as well as the government partner of a small one: *KA-Feedback*[2], in the City of Karlsruhe, Germany. *SeeClickFix* currently does not have any mechanism that allows their government partners to handle similar reports in the system, but emphasized the necessity of such a function in the future. A surprising revelation was that the current strategy for duplicate handling by case officers in Karlsruhe is keeping a mental record of the already processed reports. If this fails, they stated that the duplication is eventually detected by the maintenance crew that is sent out to fix the issue. Needless to say, this approach does not scale. Instead, employing intelligent (ideally real-time) data processing in general and spatio-temporal clustering in particular has large potential:

- *Similar report detection and aggregation*: By automatically finding and grouping reports that are likely to describe the same unique issue, data processing entities can significantly speed up issue handling and better allocate their resources.
- *Implicit prioritization*: The size of the clusters discovered by data analytics can reveal infrastructure problems which are reported by many distinct users. This information can be exploited to prioritize the processing of certain specific issues.

In this paper, we present and evaluate a novel framework for clustering spatio-temporal data, based on initial transformation of the data into a graph structure and subsequent clustering of the graph. This discards the need of setting unintuitive density parameters, as dense substructures can be directly discovered with traditional parameter free graph clustering approaches. We compare the performance of our framework to contemporary spatio-temporal clustering methods for the use case of duplicate detection.

[1] http://www.seeclickfix.com
[2] http://www.ka-feedback.de

2. RELATED WORK

In the field of spatio-temporal analytics, clustering methods have been distinguished by whether the analyzed data consists of *events* or *trajectories and moving points* [9]. In the Participatory Infrastructure Monitoring scenarios described above, we focus on analyzing reports, i.e., events. Spatio-temporal event analysis has been employed for a wide range of applications to better understand important aspects of urban phenomena. H. SILVA ET. AL. have analyzed data from *Instagram* and *Foursquare* to analyze user's movement patterns and popular regions in a city [15]. However, to the best of our knowledge, applying spatio-temporal clustering for prioritizing the treatment of infrastructure issues by grouping similar and potentially duplicate reports is still an unexplored application.

Regarding current data mining techniques for spatio-temporal clustering, a well-established approach for spatial clustering is DBSCAN [5]. It clusters circular regions in which the density of an event is higher than outside within a certain (spatial) radius, but it only considers spatial events. Based on DBSCAN, some algorithms have been proposed in order to separately identify spatial clusters and temporal clusters, and then combining the results in order to provide spatio-temporal clusters. Some examples are ST-DBSCAN [1] and ST-GRID [17]. ST-DBSCAN is an extension of DBSCAN with an additional threshold for the temporal dimension (Eps_1 & Eps_2). ST-GRID is based on partitioning of the spatial and temporal dimensions into cells and defining dense cells as clusters. Adjacent dense cells get merged into a single cluster representation. They have in common the requirement of a density threshold $minPts$, suggested to be set as ~$ln(n)$ on the number of data points in the database. However, the main problem of these density based approaches is that they cannot cluster data sets well with large differences in densities, since the combination of $minPts$ and the spatio-temporal thresholds cannot be chosen appropriately for all clusters [12]. Recent publications have addressed these drawbacks of density based clustering [13]. These strategies however do not consider the temporal issue for the clustering.

Rather than relying on the definition of different density thresholds, we propose a novel approach based on modeling first the spatio-temporal data in form of a neighborhood connectivity graph. In a later step, dense structural subgraphs forming the spatio-temporal clusters are detected with parameter-free graph partitioning algorithms based on the structure of the modeled graph. We thus additionally overcome the known drawback of density based approaches of merging adjacent clusters which are connected by a very narrow dense link, as this is detected directly in the graph structure.

3. SPATIO-TEMPORAL ANALYTICS

In this work, we propose a generic approach that can be applied to the problem of grouping spatio-temporally close reports. We first describe preliminary considerations for the formation of our methods and subsequently present the clustering framework.

3.1 Preliminary Considerations

In order to develop our analyses, we define several terms and formal relations that we use to describe the spatio-temporal relationships. The data records we analyze are the notifications on infrastructure issues that residents of a city submit. These ***reports*** are by nature spatio-temporal observations, i.e., they have both a temporal and a spatial dimension (reporting time and location of the issue) and an issue description. Throughout this work we use the term ***user*** or *citizen* to describe a person who creates a report. The receiving entities that are responsible for processing, responding and reacting to the submitted reports are called ***case officers*** or *civil servants*.

Our main assumption is, for two reports to be similar and potentially describe the same infrastructure issue, they must be spatio-temporally neighbored. Two reports x and y ($x \neq y$) are considered to be ***spatio-temporal equivalent (ST-equivalent)*** or spatio-temporally neighbored, if for a temporal threshold (Δt) and a spatial distance (D) the following conditions are fulfilled:

$$|L(x) - L(y)| < D \quad \text{and} \quad |t(x) - t(y)| < \Delta t$$

The location of an observation x reported at time $t(x)$ is denoted as $L(x)$. Two reports x and y are therefore spatio-temporal equivalents, if for report x, submitted at time t, another report y is submitted within the timeframe Δt, and y lies within the spatial distance D from x. The spatio-temporal equivalence is by definition symmetric. If *x is a spatio-temporal equivalent of y*, reciprocally *y is a spatio-temporal equivalent of x*. In this sense, there is no notion of an original first report. Naturally, there is a temporal order in which the reports are submitted, but it is not part of our model.

The spatio-temporal equivalence relation can be modeled and visualized as a graph, in which the vertexes represent the reports and which has an edge connecting two reports if they are spatio-temporally neighbored. We call this undirected and unconnected graph representation of the reports and their spatio-temporal relation the ***spatio-temporal graph (ST-Graph)***. In this graph, the spatial-temporal neighborhood is defined by linkage relationships between the objects.

On a set of reports R, we define the ***spatio-temporal neighborhood*** $N_{st}(x) \subseteq R$ of a report x as the subset of reports from R that are spatio-temporal equivalents of x (not including x itself):

$$N_{st}(x) := \{y \in R \,|\, x, y \text{ are ST-equivalent}\}$$

From these two definitions it is clear that if for a report x to be *similar* to another report and even represent a possible duplicate, its spatio-temporal neighborhood set is not empty and there must exist at least one edge connecting x to another report in the ST-Graph:

$$|N_{st}(x)| > 0 \Leftrightarrow Degree(x) > 0 \text{ in ST-Graph}$$

The spatio-temporal equivalence relation is not transitive, i.e. if x and y are ST-equivalent and y and z are ST-equivalent, that does not necessarily mean that x and z are as well. In order to reflect such an indirect connection between records, we introduce the notion of ***spatio-temporal connectivity***: A report x is spatio-temporally connected (*ST-connected*) to another report y, if there is a chain of reports $r_0, ..., r_n$, such that $r_0 = x$ and $r_n = y$ and all r_i, r_{i+1} are pairwise spatio-temporal equivalent:

$$\begin{aligned} x, y \text{ ST-connected} :\Leftrightarrow \exists\, r_0, ..., r_n \,|\, & r_i, r_{i+1} \text{ ST-equivalent}, \\ & r_0 = x, r_n = y, \\ & (\text{with } i \in \{0, ..., n-1\},\ n \in \mathbb{N}_{>0}) \end{aligned}$$

In the ST-Graph, if a path exists between two vertexes, these two are ST-connected.

Finally, we define a ***Spatio-Temporal Cluster*** C_{ST} as a non-empty subset of the set of reports R, where all reports in C_{ST} are spatio-temporally connected, fulfilling following condition (Maximality):

$$\forall x, y \in R : x \in C_{ST} \wedge x \text{ is ST-connected to } y \rightarrow y \in C_{ST}$$

The spatio-temporal clusters are primarily the connected components in the ST-Graph. Using these preliminary considerations, the problem of finding and aggregating similar reports can now be reduced to the problem of constructing the ST-Graph with adequate parameters and performing graph clustering on its top.

3.2 Spatio-Temporal Clustering Framework

Using the spatio-temporal relationships introduced above, we developed a two-step framework (see Figure 1): In a first step (*Data Modeling*), we detect spatio-temporal equivalent reports and form the ST-Graph. In a second step (*Graph Clustering*), we apply graph clustering algorithms to detect and extract densely connected subgroups based on the structural similarity of the nodes in the ST-Graph. Spatio-temporal clusters are then extracted from the connected components of the graph and individual graph objects are defined as outliers. Some reports that were clustered in the first step may be removed from their cluster in this step.

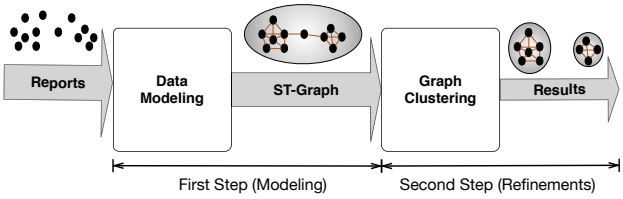

Figure 1: Two-step framework for clustering spatio-temporal data. In a first step, spatio-temporal proximity is modeled with the use of links between nodes. In a second refinement step, densely connected spatio-temporal clusters are partitioned.

3.2.1 Step 1: Data Modeling

The step of modeling the data into a graph structure is done by generating the *ST-Graph* with all reports as nodes linked to their respective spatio-temporal equivalent reports. In the framework depicted in Figure 1, the STGraph is the resulting unconnected and undirected graph after the data modeling. We introduce an optional parameter set *Constraints* as input to the algorithm which allows specifying additional (non-spatio-temporal) conditions that should be included. This can contain arbitrary other semantic constraints or relationships between reports, such as *having the same report category* or *same user IDs of the submitting citizens*.

Input: A set of reports R, temporal threshold Δt, spatial range D, and optionally an additional set of conditions $Constraints$
Output: Spatio-Temporal Clusters in $STgraph$

$STGraph \leftarrow$ new `STGraph`(R)
$Clusters \leftarrow new$ `List()`
`/* First Step: data modeling */`
foreach $i, j \in R : i \neq j$ **do**
  if $\bigwedge_{CT \in Constraints} CT(i,j)$ **then**
    if $|L(x) - L(y)| < D \wedge |t(x) - t(y)| < \Delta t$ **then**
      $STGraph$.`addEdge`(i,j)
    end
  end
end
`/* Second Step: graph clustering */`
$CCS \leftarrow ConnectedComponents(STGraph)$
foreach $CC \in CCS$ **do**
  $GPS \leftarrow GraphClustering(CC)$
  foreach $GP \in GPS$ **do**
    if `size`$(GP) > 1$ **then**
      $Clusters$.`add`(GP)
    end
  end
end
return $Clusters$

Algorithm 1: Clustering framework based on the generation of the *ST-Graph* and its subsequent partition.

3.2.2 Step 2: Graph Clustering

The aim of the second step is to detect densely connected subgroups and reach a *partitioning of the clusters* in the ST-Graph, so that the resulting clusters are more *meaningful*. Densely connected subgraphs stand for clusters with high intra-cluster similarity. We assume that clusters containing similar (i.e. potentially duplicate) reports have highly dense connected vertexes and therefore dense subgraphs that are bridged by single connecting vertexes should be partitioned: Connections within graph clusters should be dense, and connections between different graph clusters should be sparse.

A number of new algorithms for graph clustering based on different principles have been proposed in recent years. It is not the focus of this work to exhaustively discuss which algorithm would fit best in different cases. We chose a parameter-free modularity based algorithm which finds densely connected subgraphs in a graph by calculating the leading non-negative eigenvector of the modularity matrix of the graph [11]. Figure 2 shows the result of applying our clustering framework to the SeeClickFix Chicago dataset.

4. DATA COLLECTION

We collected data from two separate real-world issue tracking platforms that are actively used. The first dataset originates from *KA-Feedback* (KAF), which represents an urban issue tracking platform in its beginnings (2,821 reports by February 2014). The second dataset contains open data that was extracted from the *SeeClickFix*[3] (SCF) issue tracking platform that has deployments in several cities across the U.S. The analyzed dataset is from the Chicago deployment of *SeeClickFix* and features a much denser and larger amount of reports (34,690 entries between February 2013 and 2014). We published this dataset along with an exhaustive description of the data and its attributes as well as the clustering on our website[4] as benchmark for future publications and other researchers.

The *SeeClickFix* Chicago dataset has the unique feature that a large portion of the reports (32%) have been previously marked as *duplicates*. We exploited the manual labels as evaluation metric and for the comparison of spatio-temporal clustering: clusters are sets of objects in such a way that objects in the same group are in some sense *similar*. The highest degree of similarity is equality (*duplicates*). We thus expect the spatio-temporal clusters to precisely cover a high degree of duplicate reports.

5. EVALUATION

We evaluated the performance of our framework and two contemporary density based spatio-temporal clustering algorithms (ST-GRID and ST-DBSCAN) regarding duplicate detection. As ground truth we used the labeled SCF dataset. Clustered reports which are labeled as duplicates are considered to be true positives, clustered non-duplicates are regarded as false positives. Note that the given dataset does not contain any information about the relationship between the reports: While we know if two reports are duplicates, we do not know if they are duplicates of each other. Thus, labeled duplicates in one cluster could actually be duplicates of different original reports. Nonetheless, we do can assess the performance of the clustering for separating real duplicates from non-duplicates by using the F1-Measure as evaluation metric (see Table 2).

The first evaluation regards the parametrization of the clustering framework, as it has great effect on the clustering qualities. In this specific evaluation case, we have tested the impact of the thresholds on both precision and recall on the first step of the algorithm

[3] `seeclickfix.com`, open data access under *Creative Commons BY-NC-SA 3.0* license.
[4] `http://www.teco.edu/~borges/urbiot14/`

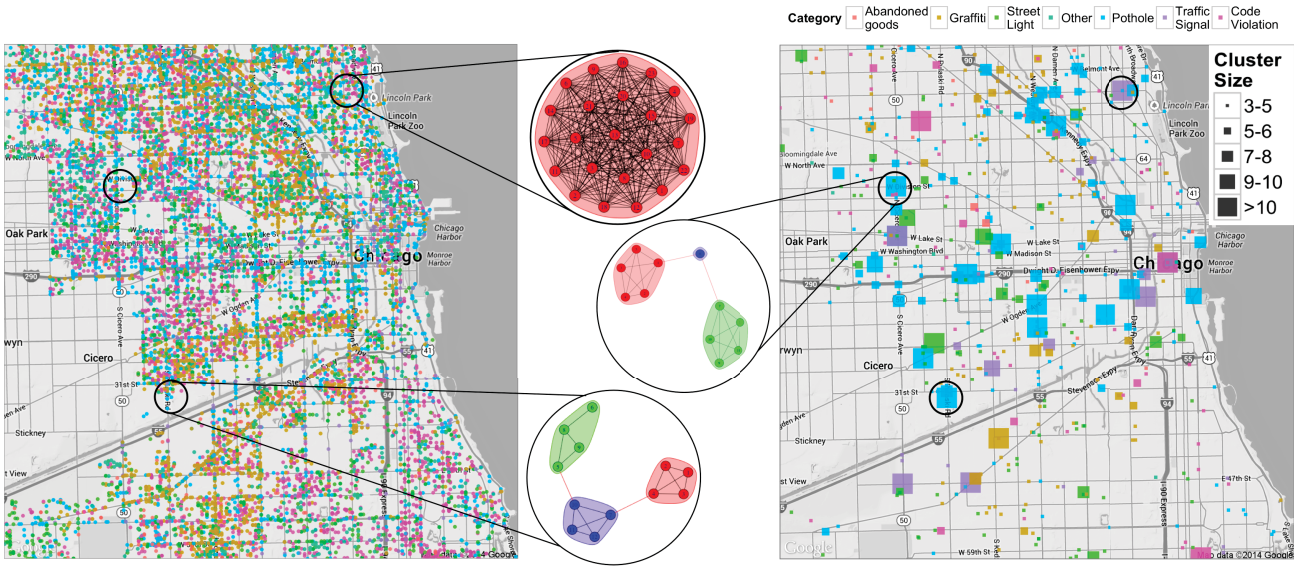

Figure 2: Spatial plot of the *SeeClickFix* Chicago dataset (left) and results after applying the proposed clustering framework (right). Depicted are three graph representations of the clusters after the graph clustering step. The refinement step resulted in one cluster being split into three distinct clusters (bottom) and another one being split into two clusters by removing a report from it (middle).

regarding its ability to cluster duplicate reports: If the thresholds are too restrictive, the precision is high but the recall low, if too loose, vice versa. We therefore used the F1-Measure, combining both precision and recall, to decide which parameters have the best impact on performance (see Figure 3). In the heatmap, the overall best parameters are marked yellow ($D = 40m$ and $\Delta t = 28$ days, F1-Measure $= 0.651$). All our experiments in this section were run using these parameters, constraining the clustering only among reports of the same category (cf. line 4 of Algorithm 1).

In the following, we present the clustering results of running our spatio-temporal clustering on the *SeeClickFix* Chicago dataset (see Table 1). Our unsupervised approach groups 14,625 reports of the SCF Dataset, classified as similar and thus as possible duplicates by the algorithm, into 5,101 clusters. The KAF Dataset, containing 2,821 reports, revealed 285 clusters for 706 clustered reports. The graph partitioning step has had just a small effect on the KAF dataset compared to its effect on the SCF. Only 12 new clusters are generated after the partition. This is mainly due to the spatio-temporal density of the data. The KAF dataset is relatively sparse, both spatially and temporally containing only a few small clusters which are already densely connected, not being further partitioned in the second step of the framework. The clustering algorithms ST-GRID [17] and ST-DBSCAN [1] were run on the same datasets using the same spatio-temporal parameters and its performance was compared to our approach. In addition to the spatial and temporal thresholds, both algorithms require a density threshold $minPts$.

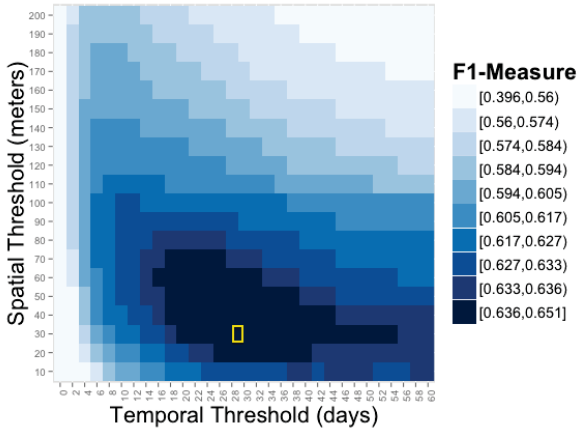

Figure 3: F1-Measure variation of our spatio-temporal clustering framework depending on spatio-temporal parameters.

Configuration	Dataset	SCF	KAF
	Parameters $D, \Delta t$	$40m$,$28d$	$90m$,$7d$
Before run	Total Reports	34,690	2,821
	Duplicates	11,121	*n/a*
After Step 1	Clustered reports	14,625	708
	Clusters	4,851	273
	Cluster size (μ, σ)	3.01, 2.20	2.59, 1.59
After Step 2	Clusters	5,101	285
	Cluster size (μ, σ)	2.86, 1.71	2.48, 1.05

(a)

Configuration	Dataset	SCF	KAF
	Parameters $D, \Delta t$, MinPts	$40m$,$28d$,2	$90m$,$7d$,2
ST-GRID	Clustered reports	11,909	589
	Clusters	3,963	234
	Cluster size (μ, σ)	3.00, 2.35	2.51, 1.52
ST-DBSCAN	Clustered reports	17,590	981
	Clusters	5,612	375
	Cluster size (μ, σ)	3.13, 2.39	2.61, 1.47

(b)

Table 1: Results after running (a) our spatio-temporal clustering respectively (b) ST-GRID and ST-DBSCAN on the *SeeClickFix* Chicago (SCF) and *KA-Feedback* (KAF) datasets.

	Spatial DBSCAN	ST-GRID	ST-DBSCAN	Clustering framework
F1-Measure	**0.553**	**0.599**	**0.615**	**0.651**
Precision	0.40	0.58	0.50	0.57
Recall	0.88	0.62	0.79	0.75
Error Rate	0.46	0.27	0.32	0.25
Purity	0.80	0.80	0.82	0.78
Compression	0.49	0.77	0.65	0.71

Table 2: Comparison of clustering performance for the task of duplicate detection using different clustering methods.

The proposed heuristic for this parameter [1] is to set it to $\lfloor ln(n) \rfloor$, n being the amount of reports in the dataset (see Table 1). To be able to better compare the results to our approach, which does not require a density threshold, we also tested it with $minPts = 2$ for duplicate detection. ST-DBSCAN delivered a F1-Measure of *0.615* and *0.187* and ST-GRID yields a F1-Measure of *0.599* and *0.039* for $minPts = 2$ and $minPts = \lfloor ln(n) \rfloor$ respectively. Our approach thus outperforms both ST-GRID and ST-DBSCAN with respect to the given evaluation metrics.

We also tested another density based approach, which is solely based on spatial attributes: Spatial DBSCAN. It delivered a much lower F1-Measure than the spatio-temporal competitors. Applying other classical algorithms, such as *k-means* for this task would be difficult, as the number of clusters must be known a-priori. Therefore, approaches that use the spatial and temporal aspects of the data (and optionally also semantic information) are more suitable.

Runtime: The main problem of ST-GRID (and grid based methods in general) is its discretization: it is sensitive to grid position and resolution. For example, for a given density threshold τ, a cluster of more than τ points can be missed if they lie in different cells. This explains why ST-GRID finds fewer clusters compared to other methods (cf. Table 1). Density based methods like ST-DBSCAN or our clustering framework find clusters irrespective of orientation or size. Its only drawback compared to ST-GRID is its runtime: it runs in $O(n^2)$ (or in $O(n \cdot log(n))$ using appropriate spatial data structures) as it needs to calculate the pairwise distance between the data points while ST-GRID only needs one scan of the whole data set, which is of linear time complexity (see Figure 4).

We were able to significantly improve the runtime of our algorithm through parallelization and Big Data technology. We partitioned the temporal dimension into several equal-sized (Δt) bins, generating the ST-Graph in parallel and reducing the spatio-temporal neighborhood search only among neighbored bins. Each bin was assigned to a computational unit running an In-Memory Database (*BigMemory* from *Terracotta*[5]). The optimized framework efficiently scales to large datasets: We ran our algorithm on a big heterogeneous set of $558,993$ reports from the SeeClickFix and the 311 city service of Chicago, which took 12.58 minutes on an Intel Core i7-2600 machine at 3.40GHz (8 cores) with 8 GB RAM .

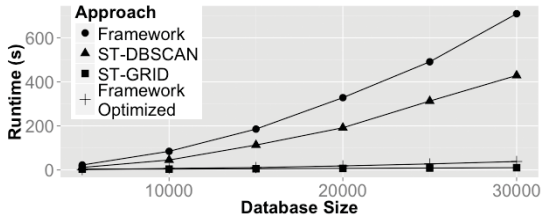

Figure 4: Runtime of the approaches vs. database size.

[5]http://www.terracotta.org

6. DISCUSSION

Quantitatively, the performance of our presented methods points to a significant potential speed up of manual processing in the underlying application case. We are convinced that, if case officers receive an aggregated view of the detected clusters, they could sort out similar reports and actual duplicates much more quickly and link back to the original report. However, whether a semi-automatic system using the proposed framework outperforms a manual system when also looking at human factors has yet to be investigated.

6.1 Deriving Issue Priority from Cluster Size

We found the great majority of the clusters to be dyads after the first step of the framework on the SCF dataset ($\mu = 3.107$, $Q_2 = 2.0$, $Q_3 = 3.0$). 21.1% of the clusters are of size 4 or more, reaching a maximum of size 34. The size of the clusters can deliver valuable information on the urgency of an infrastructure issue, which can help case officers not only to speed-up manual issue handling by processing multiple issues at once but also to potentially prioritize certain issues and better allocate their resources. As an example, we found a relatively big cluster in the KAF dataset containing descriptions of broken glass fragments on a cycleway. Such big clusters are prone to describe issues which affect a high number of citizens and can be discovered using spatio-temporal clustering.

6.2 Category-Sensitive Parametrization

Regarding the appropriate parametrization heuristic for our clustering framework, we evaluated harnessing additional meta information of the reports. One possibility we explored is to set the temporal thresholds according to the mean time it takes to the municipal government to fix an issue (*mean-time-to-fix*, *MTF*). In total 16.34% of the reports in the *SeeClickFix* Chicago dataset are marked as fixed, wherein the overall MTF is 13.47 days ($\sigma = 30.56$). The heatmap in Figure 3 shows that the best temporal parameters lie between the mean-time-to-fix plus one standard deviation. This can thus be used as an heuristic for setting the temporal parameter. Another possibility is setting the threshold differently (or even dynamically) for different categories of reports, as we observed that the MTF strongly differs across categories (see Figure 5). For example, broken traffic lights tend to get fixed within a day or two, while potholes or abandoned goods are generally tended to later. The means are depicted as white dots on the boxes. To enable better visualization the plot has been cut off at 30 days.

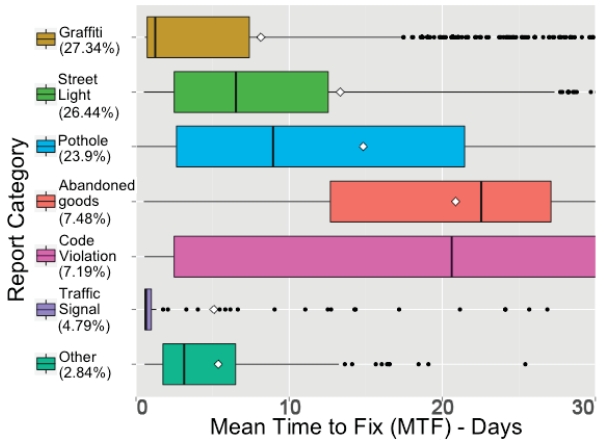

Figure 5: Mean-Time-To-Fix (MTF, cut off at 30 days) for different issue categories, sorted by frequency of occurrence.

The category *Code Violations* has a mean and a third quantile range from 78.09 and 190.0 days respectively. These categories have been artificially created through text-mining based on the description of the report. For the duplicate evaluation, applying dynamic parameters has led to a best F1-Measure of 0.638 tested with varying spatial parameters. Dynamic category-sensitive parametrization has thus not delivered any measurable advantages over the global best parameters, which yields a F1-Measure of 0.651.

7. CONCLUSION AND FUTURE WORK

In this work, we have presented a spatio-temportal data analysis technique and evaluated it on the data of two separate real-world issue tracking platforms, one in its beginnings (2,821 reports) and one mature (close to 35,000 reports). Our presented method, an unsupervised clustering framework, requires only a temporal and a spatial threshold as parameters and provides very promising results for particular applications. We have evaluated and compared our approach to the state-of-the-art spatio-temporal clustering algorithms ST-GRID and ST-DBSCAN, showing that our approach can outperform both algorithms, delivering results which give excellent agreement with expected outcomes. Furthermore, our approach requires less parameters, being more intuitive for non-experts (e.g. civil servants) to operate. Additional refinements and experiments on big datasets have shown the scalability of the approach.

This work includes an innovative use-case application of data mining techniques for urban infrastructure monitoring. We see immediate need for such technologies in detecting and grouping similar and possible duplicate reports. For this, we have made the evaluated dataset publicly available and published exhaustive information about our evaluation on our website, in order to enable other researchers to compare their findings and use our results as a benchmark for future work.

While the framework presented in this paper can already greatly facilitate manual processing in civic issue tracking, we intend to investigate how the application could benefit from further refinements or augmentation, such as leveraging natural language processing (NLP) to cluster the reports based on text similarity.

To the best of our knowledge, there are currently no clustering or duplicate detection technologies being employed in large scale participatory sensing applications. Therefore we would like to test these methods in actual operation of such large applications in the near future. Although we applied our framework offline, it is in principle well suited to work online and in real-time. There are already several graph clustering techniques for online clustering [7]. The insertion or deletion of an object in a cluster affects the current clustering only in the spatio-temporal neighborhood of this object. Nevertheless, the main challenge is to model and evaluate an accurate data transformation for the framework in order to enable this online preprocessing in the first step of our framework and embrace a set of other attributes besides spatio-temporal ones [14].

Acknowledgments

This work was partially funded by the German Federal Ministry of Education and Research (BMBF) as part of the *ESTAData* project (grant no. 01IS12051). The authors thank PATRICIA IGLESIAS SÁNCHEZ of KIT's *Institute for Program Logic and Data Structures (IPD)* for stimulating discussions and input, as well as all reviewers. Many thanks also go to the *Research Center for Information Technology* (FZI) and to the Media Office of the City of Karlsruhe for the kind permission to analyze the *KA-Feedback* dataset, as well as to `seeclickfix.com`, for providing open access to their datasets under *CC BY-NC-SA 3.0* license.

References

[1] D. Birant and A. Kut. ST-DBSCAN: An algorithm for clustering spatial–temporal data. *Data & Knowledge Engineering*, 60(1):208–221, 2007.

[2] J. Burke, D. Estrin, M. Hansen, A. Parker, N. Ramanathan, S. Reddy, and M. B. Srivastava. Participatory Sensing. In *Workshop on World-Sensor-Web (WSW'06)*, 2006.

[3] Y. Chon, N. D. Lane, F. Li, H. Cha, and F. Zhao. Automatically characterizing places with opportunistic crowdsensing using smartphones. In *UbiComp'12*, 2012.

[4] J. De Melo Borges, V. Zacharias, and N. Plessing. PartSense: A Participatory Sensing Platform and its Instantiation KA-Feedback. In *Geoinformatik 2012*, 2012.

[5] M. Ester, H.-P. Kriegel, J. Sander, and X. Xu. A density-based algorithm for discovering clusters in large spatial databases with noise. In *KDD*, volume 96, pages 226–231, 1996.

[6] L. Fennell. Crowdsourcing land use. *Brooklyn Law Review*, 78, 2013.

[7] T. Hartmann, A. Kappes, and D. Wagner. Clustering evolving networks. *CoRR*, abs/1401.3516, 2014.

[8] S. F. King and P. Brown. Fix my Street or Else: Using the Internet to Voice Local Public Service Concerns. In *1st Conference on Theory and Practice of Electronic Governance*, 2007.

[9] S. Kisilevich, F. Mansmann, M. Nanni, and S. Rinzivillo. Spatio-temporal clustering. In *Data Mining and Knowledge Discovery Handbook*. Springer, 2010.

[10] I. Mergel. Distributed Democracy: SeeClickFix.com for Crowdsourced Issue Reporting. *Social Science Research Network*, 2012.

[11] M. E. Newman. Finding community structure in networks using the eigenvectors of matrices. *Physical review E*, 74(3):036104, 2006.

[12] M. Parimala, D. Lopez, and N. Senthilkumar. A survey on density based clustering algorithms for mining large spatial databases. *Journal of Adv. Science and Technol.*, 31(1), 2011.

[13] A. Ram, S. Jalal, A. S. Jalal, and M. Kumar. A density based algorithm for discovering density varied clusters in large spatial databases. *Int. Journal of Computer Appl.*, 3(6), 2010.

[14] P. I. Sánchez, E. Müller, F. Laforet, F. Keller, and K. Böhm. Statistical selection of congruent subspaces for mining attributed graphs. In *Int. Conference on Data Mining*, 2013.

[15] T. H. Silva, P. O. Vaz de Melo, J. M. Almeida, J. Salles, and A. A. Loureiro. A comparison of foursquare and instagram to the study of city dynamics and urban social behavior. In *2nd ACM SIGKDD Int. Workshop on Urban Computing*, 2013.

[16] M. Stevens and E. D'Hondt. Crowdsourcing of pollution data using smartphones. In *Workshop on Ubiquitous Crowdsourcing at Ubicomp'10*, 2010.

[17] M. Wang, A. Wang, and A. Li. Mining spatial-temporal clusters from geo-databases. In *Advanced Data Mining and Applications*, pages 263–270. Springer, 2006.

[18] Y. Zheng, L. Capra, O. Wolfson, and H. Yang. Urban computing: concepts, methodologies, and applications. *Transactions on Intelligent Systems and Technology (ACM TIST)*, 2014.