Data Management and Smart Cities

Martin Brugnara, Cristian Consonni, Daniele Foroni, Sivam Pasupathipillai,
Giulia Preti, Paolo Sottovia, Yannis Velegrakis
University of Trento, Italy
{martin.brugnara, cristian.consonni, daniele.foroni, s.pasupathipillai}@unitn.it
{giulia.preti, paolo.sottovia, velgias}@unitn.it

***Abstract*—Modern cities use information and communication technologies to obtain deep insights on the different aspects of the way they operate, which can allow officials to make informed decisions to improve the operational efficiency of different operations and improve the life of their citizens. Analyzing the data about the different activities poses significant challenges. It is not merely the volume that recent hardware and software advancements have helped to achieve, but also challenges regarding the variety, velocity, and veracity of the data. All this is often known as the Big Data paradigm. In this document, we analyze some of these challenges, which we believe have not yet received considerable attention, we explain their value, and we describe some of the advanced solutions we have developed.**

I. Introduction

By the end of this century, most people will be living in grand metropolitan areas. This phenomenon is putting a burden on governments and city officials in ensuring the efficient and effective operation of the city activities and guaranteeing the quality of life of the citizens. Any current and future buildup of urban development, alongside any repair and maintenance of the existing settlement and infrastructure, cannot be done in an ad-hoc way. Solid and substantial evidence should inform every design and implementation decision taken by city officials, which can be obtained by analyzing the data regarding the activities that are taking place in the city. This paradigm is known as Smart Cities.

Recent advances in information and communication technologies have played a significant role in the successful materialization of the Smart City paradigm. Smart cities have enabled vast amounts of data in almost every aspect of human activity in a city to be collected and become available for analysis. These analyses offer valuable insights into the way the city operates and can drive the decision-making process. For quite some time, the datasets that were analyzed were few and inserted in the system mainly manually by experts. The advent of the web, social media, and the access in communication technologies made available to the general public, turned the passive Internet users into active data producers, contributing with significant volumes of additional data. The penetration of sensors and other monitoring hardware in our daily lives increased the amount of the available data even further since it also turned machines into data producers. As a result, the amount of complex data produced and made available daily has become massive. Traditional data management techniques started to become limited in coping with such data, calling for a novel data management paradigm. That paradigm is known as Big Data. Big Data is data characterized by high Volume, Variety, Velocity, and Veracity. A great deal of work has been devoted to managing the high data Volume, but the other three dimensions have been largely overlooked. This overlook has not been a significant issue when dealing with business data that is often well-curated and targeting specific information needs, but data from a city data is a profoundly different case: a city is a complex living entity. Datasets from many different sources need to be collected and used to obtain a panoramic and complete view of a city. These datasets are often of diverse nature and have been designed with different requirements in mind, which makes them, naturally, highly heterogeneous. Another particular characteristic of a city is that there is a need for immediate reaction to certain situations, which means that data needs to be monitored and analyzed as it is produced. Decisions must be taken immediately, even with limited resources, and without the ability to look much in the past. Last, but not least, the collection process is not always flawless, which means that the data may be incomplete or inconsistent, leading to inaccurate analytic results. To exploit the full potential of Big City Data it is necessary to tackle all the above challenges.

However, Big Data exploitation is not merely analytics. Before any data can be leveraged by data analysis, it will have to be integrated, cleaned, restructured, and understood. This is the typical task requested to data experts or data scientists, and it has been recognized that it consumes more than 50% of the time of the overall process. Thus, it is significant to develop the necessary tools, techniques, and methodologies that facilitate data analysts in performing all their analysis tasks.

In this work, we describe a number of the above challenges and the solutions we have developed for dealing with them.

II. Data Quality

To have confidence in any data-driven decision, there should be trust in the data on which those decisions are based. For this trust to materialize, the data has to be of high quality, i.e., to be accurate, complete, consistent and up-to-date. Unfortunately, most real-world datasets have significant quality issues. Sensors operating in a natural environment are exposed to weather and other environmental conditions that affect their mechanic readings. Instruments used by public-transport vehicles are often obstructed by other objects in the surrounding resulting in less accurate information. Communication networks used in

data collection processes are unreliable, causing parts of the collected data to be lost. Different systems operate on different hardware that produces data of different granularities, formats or semantics. On top of all the above, there is always the human factor. In most activities throughout the city, there are humans involved, that by nature introduce errors in the data.

Coping with the natural presence of data quality issues in these datasets has attracted significant attention. A great deal of work has been devoted to quantifying the amount of data in a dataset with different quality issues. The identification and quantification of the quality issues can be used either for informative purposes or can form the basis for a data reparation task where these issues are either repaired or eliminated. Unfortunately, it is not always easy to repair the data or eliminate the issues. For this reason, some efforts have been devoted to developing data management and analytic techniques that are immune to them, but the cases in which this is possible are very limited.

Although quantification of data quality is useful, it is often not enough. First of all, for this quantification to be made, there is a need for knowing the original clean dataset, which often is not available or known. Second, even if the clean dataset is known, and the difference of an available (dirty) dataset to the clean one has been calculated, this difference is not very informative. Small differences in the dataset may have a very big impact on the results of an analytic task, or none at all, depending on what the analytic task is. For this reason, any data quality issues observed in a dataset need to be considered in the context of the task for which the data are used. For instance, we consider a dataset related to New York, if the goal is to use this dataset to count the number of people living in the city, even if the records contain thousands of mistakes, this hardly affects the outcome of the counting. If, on the other hand, the goal is to cluster people into groups, then the incorrect values will lead to significant variations from the expected result.

To help users understand - and not simply quantify - the quality of their datasets, we have developed a system that measures the fitness of a dataset for a specific task. In particular, it generates in a systematic way, multiple types of noise and measures the observed change in the results of the analytics task of interest. By considering all the observed changes alongside the amount and the type of noise that was generated in the data for producing these changes, it is possible to derive a single metric that indicates the effect of a kind of noise for a specific task. This novel metric is referred to as 'fitness for use'.

Fitness for use finds significant applications in smart cities since it can provide city officials and data analysts with an idea on whether a dataset that is available to them can be used for a specific purpose or not, by quantifying the trust to be put in the results of some analysis applied on that dataset.

III. Dynamic Stream Management

Many smart city processes produce continuous and infinite streams of data. Monitoring and analyzing such streams is a challenging task since there are no infinite resources to store all the data, neither it is possible to wait long enough before taking a decision. Thus, specialized querying, processing, and analysis techniques have been developed specifically for streams.

The nature of the smart city environment also imposes additional requirements: one of them is the ability to manage dynamic streams, which produce data at variable rates. In fact, in most cases, these data are generated by human activities, thus the amount of data collected varies in time. Consider for instance the sensors throughout the city that monitor and report the traffic. Clearly, there are moments throughout the day of very high traffic, and other moments, like late in the night, where the traffic is practically non-existent. This fluctuation in traffic is reflected in the amount of demand for resources. Allocating too few resources puts a strain on the system for the moments of high traffic, whereas, allocating too many resources means that they will remain underutilized during the moments of low demand. This situation is inefficient and, in the end, ineffective: underutilized computational resources could have been employed in other activities that needed them, or they could simply be turned off, reducing the overall cost. Thus, there is a need for dynamic allocation and management of the different resources. This ability becomes even more apparent nowadays that many computational resources can be deployed on-demand in the cloud.

The Moira [1] system comes to fulfill this need. Moira is a framework that allows the dynamic rescheduling of streaming applications given a specific goal. It allocates the resources dedicated to a task according to a given goal and input data rate. In this way, the overall cost for managing a stream is optimized to satisfy the chosen goal at minimum cost. Moira has been implemented for the Flink platform, an open source stream management platform for Big Data. Thus, it can be easily plugged in any of the many different applications in which Flink is used. Nevertheless, the idea is easily implementable in other systems as well.

IV. Anomaly Detection for Smart Cities

One interesting problem related to data collected from sensors in cities is how to detect unusual events from the continuous stream of information automatically. This data analysis task is known as anomaly detection. In the context of smart cities, anomalies can represent unusual weather conditions, traffic jams, large-scale social happenings, and other interesting events. The anomaly detection approach consists in learning a model of normal behavior and then detecting anomalies by comparing data with the learned model. An anomaly detection model is essentially a classifier, that is a function discriminating normal data from anomalies. There are several anomaly detection methods already proposed in the scientific literature, and many of them are already used in practice [2].

The detection of anomalies in mobile phone data is of particular interest for smart cities. Mobile phones have become ubiquitous in our society. Each mobile phone is equipped with a wide array of sensors; thus mobile phone data represents a particularly rich source of information for analyzing and

understanding smart cities. Mobile phone data is collected by telecommunication carriers in the form of Call Detail Records (CDRs). These records contain information about the mobile user activity, and they usually include location and timestamp. Mobile phone data has been used extensively in the field of urban sensing to analyze the built environment [3], social networks and behavior of people during events [4], [5].

One of the biggest challenges in analyzing smart city data streams is non-stationarity. Non-stationary data sources change their behavior over time in an unpredictable way. Data from smart cities present this characteristic since it is generated by a variety of complex phenomena. As an example, the mobile phone activity in a city depends on several factors including the period of the year, the hour of the day, the weather, holidays, or sports events.

Smart cities present several challenges to anomaly detection. The high volume and the velocity of smart cities data call for highly scalable anomaly detection techniques, which must be able to work in an online fashion. Additionally, non-stationary data sources require the anomaly detection method to continuously adapt to the data, in order to maintain acceptable performance levels. These reasons make anomaly detection for smart cities a particularly challenging use case for anomaly detection.

Only a few proposed anomaly detection methods are both scalable and adaptive [6], [7]. However, their capabilities have not been tested at very large-scale. Furthermore, most anomaly detection methods rely on the assumption that data instances can be represented as real-valued vectors. Often, this assumption does not apply. The very general case, namely the mixed-attribute relational data, has received very few attention.

We have developed a novel anomaly detection method based on an ensemble of classifiers. Ensemble methods have recently been applied to anomaly detection with good results [8], [9], [7]. Our novel method has been designed to be adaptive, and it can leverage a distributed computing environment in order to achieve horizontal scalability. Additionally, our method is designed to work with mixed-attribute relational data streams. The method is validated on real-world mobile phone data of Telecom Italia[1] to ensure that our methods are robust and effective enough to be applied to real-world scenarios.

V. Data Profiling through Dependency Discovery

The volume and complexity of the datasets that get generated and become available for exploitation nowadays have increased dramatically. Data practitioners face increasing challenges in understanding the information that the datasets provide and turning it into the needed knowledge. One of the critical tasks in understanding the data is finding hidden dependencies. On the technical side, dependencies across different parts of the data play a significant role in query optimization, since redundant information may be ignored making the query evaluation faster. Furthermore, parts of the data may be replaced with others that are easier to manipulate, without affecting the final result. On the application side, dependencies are equally important. Since data records the reality, dependencies may reveal relationships between real life activities or events. Furthermore, dependencies may also help in ensuring the quality of the data since they can highlight constraints that may exist in the data but the mechanism to enforce them has not been put in place by the data administrator. Finding dependencies is a challenging task since many different possibilities have to be considered, and even then since correlation does not necessarily mean causality, a level of confidence has to be computed. The challenge becomes even more significant for the massive volumes of the complex city data that can be analyzed. Thus, it is fundamental to have efficient and effective techniques that can discover hidden dependencies in the data in an automatic.

Dependency discovery is not a new topic. For quite some time and many different reasons, scientists have studied the problem of finding dependencies in data sets that are not explicitly enforced by their respective schema. Functional and inclusion dependencies are the most common and most studied type of dependencies. A *functional dependency* states that if two different data elements have the same part A, then some other part B should also have the same value. An *inclusion dependency* states that the values of a part A of a set of data elements must be a subset of the values found in the part B of some other set of data elements.

As an example, consider the table below that describes information about the traffic at a specific part of the city.

TABLE I
Air Quality

Borough	Car transit	Truck transit	Pop density	Pollution score	PM10 level
Duomo	480	60	1680	1	34
Villazzano	750	76	2185	1	38
Povo	900	85	1780	2	44
Mattarello	1100	110	1600	3	54
Gardolo	1100	150	1665	3	60

For each `borough` of the city, it provides the number of cars and trucks that have gone through it (fields `car_transit` and `truck_transit`, respectively), the population density (field `pop_density`), the perceived level of air quality (field `pollution_score`) in a scale between 1 (best) and 10 (worst), and the level of pollutants in the air (field `PM10_level`). One would expect that the air quality is associated with the level of pollutants in the air. However, there is no functional dependency observed between `PM10_level` and `pollution_score`, while we have a functional dependency between `pollution_score` and `PM10_level`, which indicated that places with better perceived air quality have lower levels of pollutants. There is also a functional dependency between `car_transit` and `pollution_score` meaning that the number of cars in a borough influences the perceived air quality.

Apart from the traditional functional dependencies, there are other, stronger, forms of dependency that take into consideration the order of attributes. For example, we can qualify more precisely the relation between traffic, perceived air quality, and pollutants level in the air. From the above table, we can

[1] https://www.telecomitalia.com/tit/en.html

learn not only that there is a functional dependency between the number of cars and the perceived pollution, but that when the number of cars grows the `pollution_score` also increases. We can also learn that there is a direct proportion between the number of trucks circulating in a borough (field `truck_transit`) and the level of pollutants in the air (field `PM10_level`). We also see that the number of trucks and cars grow together, i.e., if we order the table entries based on the value of the `truck_transit`, each one of the `car_transit` and `PM10_level` columns will also end up being ordered. This form of dependency is known as an *order dependency* and is denoted by the symbol $\mapsto$.

Apart from the benefits from the real-life application perspective, order dependencies have also advantages at the technical data management level. In the data storage design phase, order dependencies can be exploited to assist schema design and index selection. If data is extracted from unstructured sources, order dependencies can be used for knowledge discovery by finding hidden data properties. In the context of data profiling, data integration, and cleaning, order dependencies can be used to describe a dataset. Order dependencies can also be used as requirements or constraints, guaranteeing higher quality data. The most important application of order dependencies is their use in query optimization. In particular, they can be used to rewrite the `ORDER BY` clauses in SQL queries in ways similar to that of functional dependencies for the `GROUP BY` statements. Consider, for instance, the following query:

```
SELECT truck_transit, car_transit, PM10_level
FROM AirQuality
ORDER BY truck_transit, car_transit, PM10_level
```

Given that the order dependencies `truck_transit` $\mapsto$ `car_transit` and `truck_transit` $\mapsto$ `PM10_level` hold, the query optimizer may infer that sorting by `truck_transit` makes the ordering on the other two columns redundant, so the `ORDER BY` clause can be simplified to `ORDER BY truck_transit`.

Dependencies are typically derived from design specifications, from the context of queries or other known dependencies using inference rules.

Dependency discovery from the data requires to check which of all the possible dependencies hold in the database instance under examination, which may become highly time-consuming. This makes the development of strategies that limit the number of combinations to be checked very important. The task becomes more challenging in the case of order dependencies, where the order of attributes matters, leading to a search space much larger than that of functional dependencies.

We have developed an efficient technique for order dependency discovery [10]. It is based on a bottom-up approach in which we start by checking short lists of columns and progressively check longer and longer lists. In this process, once an order dependency between two lists of attributes is found not to hold, we prune the search space by ignoring larger lists that include them. In this way, many of the combinations that would have normally been checked are avoided. We are also studying solutions for discovering the most significant order dependencies in big datasets that represent a challenge given their high dimensionality.

VI. Event Extraction

A great deal of the world's knowledge is currently contained in documents in an unstructured textual form. Unfortunately, this textual form is hard to understand, query, manage and in general exploit. For the contained knowledge to reach its full potential, it is of paramount importance to be able to identify, extract, and convert it into some structured form, that can then be exploited more easily. Wikipedia is one of the most substantial textual resources, encoding the collective knowledge of millions of users, which is why there has been an increasing interest in extracting its contained knowledge into some structured knowledge base.

A fundamental first task of turning a text document into some structured knowledge is entity extraction, i.e., the recognition of references to real-world entities [11], and the extraction from the text of information about them, such as characteristic properties and relationships [12], [13]. Another equally important task is the extraction of events. Event extraction [14], [15], [16] deals with the discovery of 'unique things that happened at some point in time' [17]. Unfortunately, event extraction has not so far received the considerable attention it deserves, but some solutions already exist. One is to decide the structure or type of an event in advance and then analyze the text to find descriptions that match that predefined structure [18], [14], [19]. This, of course, means that the structure of an event is known in advance and that any description in the text complies with that structure, which is not always the case. To overcome this limitation, other approaches have looked for irregular changes in some monitored quantity, to identify that an event is occurring. For instance, search engines may be monitoring for irregular popularity (like peaks) of a set of terms and use this as an indication that an event characterized by these terms has occurred [20]. This online monitoring is effective for events that are occurring in the present, i.e., during the monitoring time, but cannot be used to discover historical events for which there was no monitoring operation in place. Wikipedia is a set of pages, with each page about some real-world entity. Although each page contains some structured information (referred to as an *infobox*), most of the information is actually contained in its textual part. The latter may contain references to other pages when the entity represented is mentioned in the text.

We have developed a novel approach to identify historical and contemporary events from Wikipedia content, based on their co-reference relationships in specific time periods. An entity is associated with another if in the text of the first there is a reference to the second. The association can, of course, be mutual, and this is known as *co-reference*. Associations have also a temporal validity. In Wikipedia, this temporal validity is often stated in the text. For instance, a part of Barack Obama's page states that '... in June 1989, Obama met Michelle Robinson ...' meaning that the association 'met' of Michelle Robinson to Barack Obama has the temporal validity of June 1989. Note that the temporal validity of an association

refers to the event in the real world, and not the time that the link was introduced in the text of the page. Our position is that an event can be considered the situation in which a set of entities in a specific period have a dense set of established associations between them. This offers us the flexibility to detect events independently of the kind of events that they are.

Identifying events in the Wikipedia content through co-reference poses many challenges. First, a set of entities with a number of temporally restricted co-references among them may not always correspond to some real-world event. For this, it is important to increase the likelihood that such a set constitutes an event. Second, there may be multiple co-references among certain pairs of entities, and it is important to decide which of them to consider. Last but not least, it is important to be able to scale to the level of co-references across the millions of pages of which the current Wikipedia consists. To cope with these challenges we have devised a technique that employs the context of the co-references as an additional criterion for improving the precision of our results and avoid combining unrelated co-references. Furthermore, the performance is significantly improved through smart indexing.

The ability to identify events in text collections through coreference can find significant application in the smart city context. By analyzing documents of public administration that have recorded various aspects of the city governance, one will be able to identify significant events that have affected the city and look easier for information on how these issues were resolved.

VII. Frequent Patterns in Multi-weighted Networks

Every real-life situation to be better understood and analyzed, it has first to be modeled. For many cases of a modern city, the model that most naturally represents the reality is that of a graph. A graph is a network of components (called nodes) that are pairwise interconnected through edges. Graphs have found applications in a great deal of application scenarios like urban planning [21], [22], zoning regulations [23], public transport design, disease outbreak control, energy management [24], and disaster management [25], [26].

One of the most typical analysis tasks that can be performed on graph data is the identification of relevant patterns: this task is known as graph pattern mining. Graph pattern mining has been extensively studied, and many successful algorithms have been applied successfully applied to fraud detection [27], biological structures identification [28], anticipation of user intention [29], graph similarity search [30], traffic control [31], and query optimization [32].

The goal of these algorithms is to identify structures that appear frequently in the graph, under the assumption that frequency signifies importance. Determining the number of appearances of a structure entails solving graph isomorphism, a well-known NP-hard problem. For this reason, metrics based on the *apriori* property have been proposed in the literature [33], [34], [35], with the most prevalent one being the MNI support [36]. According to this property, the frequency of a pattern decreases monotonically with the increase of its size, and thus the mining process can safely start from small patterns and extend to larger ones only when their frequency is above a given threshold.

A multitude of applications can require to be modeled as weighted graphs, i.e., are graphs with weights assigned to the nodes, the edges, or both. In genomic networks, for example, weights indicate the strength between genomes [37], in knowledge graphs they quantify the degree a piece of data is qualified as an answer to a user [38], and in computer networks they indicate congestions [39]. For these particular graphs, the importance of a pattern should be determined not only by the number of its appearances but also by the weights on the nodes or edges of those appearances. However, scoring functions based on weights do not satisfy in general the apriori property, because the weights of the extra nodes/edges of a larger pattern may offset its lower frequency. As a consequence, existing works on weighted pattern mining have proposed solutions that employ less efficient pruning strategies [40].

The problem becomes even more challenging in the case of multiple weights on the same nodes/edges. Multiple weights are present, for example, in scenarios where we want to offer personalized products and services rather than generic preferences [41].

Example. *Consider a heterogeneous citation network that includes authors, papers, venues, and terms (keywords). In this graph, edges connect papers with their authors, the works they cite, the venue where they were presented at, and the keywords appearing in the title and the keyword list. Frequent pattern mining finds patterns that contain mainly top venues and terms related to research fields with high engagement because these labels appear very often in the graph and thus are characterized by a larger support.*

On the other hand, we can weight the nodes and edges of the network according to the user preferences, which can be inferred from the papers she published, from her coauthors, or from the keywords she used and liked. Since multiple users can access the graph, each one with their own preferences, each single edge is associated with multiple weights, one for each user. Mining relevant patterns in such a multi-weighted network allows us to guide each user in the exploration of the literature related to their own research interests. For instance, a researcher working in the field of machine translation is unlikely to be interested in patterns describing the largely studied area of data mining.

For these multi-weighted graphs, there is a need for solutions able to mine patterns based on the weights of each individual user, as opposed to *one size fits all* solutions where only a single set of weights is considered. We have recently introduced [42] a novel approach to mine patterns in multi-weighted graphs that goes beyond frequencies, yet its performance does not significantly different from pattern mining in unweighted graphs. This approach uses a new family of scoring functions that respect the apriori property and thus can rely on effective pruning strategies. To avoid running the algorithm one time for each different weighting function, all the scores of each pattern are computed at the moment we

are generating and examining it, and only the patterns with a high score with respect to at least one weighting function are returned to the application.

We have developed both an exact and an approximate solution, which reduces the number of weighting functions to consider by aggregating those having a high probability of generating similar results into a single representative function. In addition, we have developed a distributed version that can scale to large graphs and runs on top of the distributed graph processing system Arabesque [43].

VIII. Regions of Correlation in Dynamic Networks

In many data analytics scenarios, the datasets that are analyzed are static. They describe a specific snapshot of a situation or the collective information about a specific period. However, a city is a live environment. It is an entity that changes continuously, and a great deal of additional information can be extracted if this dynamic nature is also taken into consideration.

Given that many city situations can be modeled as graphs, the dynamic nature is translated into an evolving graph, i.e., into a graph where nodes and edges are continuously added and removed. Of particular interest in such graphs is the analysis to discover different parts of the graph that indicate a high correlation, i.e., they evolve in a more or less similar way. This kind of behavior could signal, for instance, different parts of the city, the traffic of which affects each other. Thus, there is a need for efficient and effective solutions for the detection of dense groups of correlated edges in dynamic networks, i.e., groups of edges that are topologically close and changed in a similar way in a period of time.

Example. *Dense groups of correlated edges are useful in network fault diagnosis, where the goal is to identify the cause of a set of symptoms. Consider a road network where sensors are placed in each road segment to monitor the level of traffic during the day and the night. The stream of values recorded induces a sequence of weighted graphs where each graph represents a snapshot of the road network and its weights the traffic load in that status of the network. Suppose an accident happens in some segment of the network. The corresponding sensor suddenly detects lower driving speeds, and gradually, high levels of traffic are observed by the sensors in the adjacent roads, as more and more cars get stuck in the traffic jam caused by the accident. As a consequence, the sequences of measurements of these sensors exhibit a positive correlation. A search for dense correlated structures in the sequence of graphs will detect this group of sensors and allow the network analyst to attribute their anomalous measurements to the car crash.*

Unfortunately, not much research has addressed the task of finding subgraphs of correlated changes in a dynamic network [44], [45], and to the best of our knowledge, no work performs a complete enumeration. Thus, we have considered the problem of enumerating all the dense groups of correlated edges in dynamic networks. We have developed two measures to compute the density of a group of dynamic edges. The first measure is the minimum average node degree in the subgraph induced by the edges measured in the snapshots of the network. The second measure, instead, is the average between the average node degrees in all the snapshots. Similarly, we propose two measure to express the temporal correlation of a group of edges. The first one is the minimum Pearson correlation between the series of values associated with the edges in the group, while the second one is the average Pearson correlation. Then, given a density and a correlation threshold, our goal is to find all the maximal groups of edges with density and correlation greater than the respective threshold. Furthermore, since some types of dynamic networks may naturally contain a large number of dense groups of correlated edges, we have also studied the problem of identifying a more compact subset of results that are representative of the whole set. In particular, given a threshold on the maximum pairwise Jaccard similarity between groups of edges, we can find a set of groups with pairwise overlap less than the threshold.

IX. Data Exploration

Day after day, companies, organizations, and governments have started to realize the value of data analytics, or, as the saying goes among the practitioners, that *"Data is the oil of the 21st century"*. This appreciation has lead to a new race in collecting data. Massive amounts of data are collected daily, and huge datasets are created and stored with the intention of being analyzed at a later time to discover useful knowledge. Data comes in etherogenous formats, and thus novel solutions has been proposed to store them, ranging from columnar store [46], to graph databases [47].

However, in most cases, data analysts are not cognizant of what kind of knowledge they can extract from the dataset. They may not know what kind of analytics to run, on which part of the dataset to run them, or how to run them. So before doing any analysis, it is essential first to understand very well what kind of information the dataset contains. This search is typically known as *data exploration*, where the data analysts and a domain expert explore together the different parts of a dataset in order to understand what it contains. Given the size and complexity of the datasets that are available today, there is an urgent need for tools, techniques, and methodologies that support this effort.

One typical situation where data exploration plays an important role is in the case of Open Data. Modern governments are enacting policies making their data publicly available online in an open format. These efforts are motivated by the need for transparency; furthermore, they enable the creation of new services for the citizens. However, since the different datasets have no global schema and refer to many different aspects, analysts need to perform several investigative queries first.

A variety of tools and techniques has been proposed with many different purposes for helping data analysts in understanding the datasets they have at hand. Many tools provide interactive visualizations, as they show facets of the data and allow the user to dig deeper into the dimension she is more interested in [48], [49]. Others summarize the data

with clustering or sampling techniques, possibly providing guarantees on the quality of the latter [50]. Others again, help in finding relationships and similarities among datasets in the same data-lake [51]. Finally, a few techniques describe the result of a query, usually by highlighting the main difference with the other data in the dataset [52], or some specific structures in the data [53].

We have studied the problem of generating informative descriptions of a dataset. The notion of description may have different meanings. To understand it better, we consider the following simple dataset.

TABLE II
EXAMPLE DATASET

ID	Name	City	Employer	Job
0	Alice	**Rome**	Amazon	Engineer
1	Bob	**Rome**	Facebook	Analyst
2	Clara	**Rome**	Amazon	Analyst
3	Dean	Venice	Facebook	**Developer**
4	Evelyn	Trento	Google	**Developer**

1. A description that can be provided by the person releasing the data may look like 'Italians' residence and job'. 2. A database expert may instead prefer to state the following: 'a table with a serial id and 4 text columns'. 3. An analyst may notice the high frequency of *Rome* and conclude that 'the dataset is about Rome'.

All these descriptions provide some valuable information about the dataset; however, they are limited in scope and applicability. The first description is based on prior knowledge, and as such cannot be automatically generated from the data. Furthermore, even when a description is available, it describes the source of the data, its purpose, and what data should contain, but it provides no clue on what the actual tuples are or which are the possible relations between them. The second description captures only the structure of the relation; it belongs to a family of descriptions based on listing properties like unique column combinations or functional dependencies. While these aspects are indeed informative, they do not capture the meaning of the data nor its intrinsic pattern. Moreover, these description are of less value for non-technical users. The third description carries information about the data, however, it does not describe all the tuples, and falls short in capturing the general meaning of the whole dataset.

We have focused on generating descriptions that can be understood both by expert and non-expert users, that can capture the meaning of the data and the essence of the dataset. Our notion of a description is based on the nominal behavior of a person that wants to describe something complex and unknown. A person would start decomposing the problem in pieces and then would try to map such pieces to something known; if the pieces are still too complex, she would recursively repeat the process. We describe a dataset by slicing it in manageable pieces characterized by some common *property*. By *property* we mean that they all share a common set of values. The final dataset description then consists in the set of the individual descriptions that have been generated. To minimize redundancy, our approach aims at minimizing the descriptions that cover each value in the dataset.

X. PERSONALIZED RELEVANCE RANKING ON GRAPHS

Apart from data analysis, there is also often the need to find the data that is related to a task at hand, or a specific user. Traditional query techniques assume that the user knows very well what she is looking for. In the case of the vast amounts of data, this is not the case anymore. There is a need for techniques that can automatically or at least semi-automatically identify the elements in a dataset that best fit the user needs. This can be used for personalization or simple information retrieval.

Within the context of the ENGINEROOM[2] project, we have analyzed Wikipedia and the links that exist between its pages. We have developed a novel algorithm for finding the most relevant nodes in the Wikipedia link network related to a topic. The technique, called LOOPRANK, takes advantage of the loops that exist between the links and produces a ranking of the different articles related to one chosen by a user. This technique can provide results that are more accurate than those produced by the well-known Pagerank.

XI. CONCLUSIONS

Activities in modern cities are excellent sources of information. With the modern advances of information and communication technologies, it is possible to collect massive amounts of data about the way the different activities take place in a modern city, and then analyze this to produce valuable insights that can drive the creation of new services. The amount, velocity and complexity of the data that can be collected, however, is pushing existing technologies to their limits and calls for novel data management paradigms. Thus, smart cities have become platforms that can offer challenging opportunities in data management with significant impact on the quality of life of modern citizens. In this article, we have analyzed several such applications and discussed which new requirements they pose.

XII. ACKNOWLEDGMENT

This project is supported by the IEEE Smart Cities Initiative, Student Grant Program.

REFERENCES

[1] D. Foroni, C. Axenie, S. Bortoli, M. A. H. Hassan, R. Acker, R. Tudoran, G. Brasche, and Y. Velegrakis, "Moira: A goal-oriented incremental machine learning approach to dynamic resource cost estimation in distributed stream processing systems," in *Proceedings of the International Workshop on Real-Time Business Intelligence and Analytics, BIRTE 2018, Rio de Janeiro, Brazil, August 27, 2018*, pp. 2:1–2:10, 2018.

[2] C. C. Aggarwal, "Outlier analysis," in *Data mining*, pp. 237–263, Springer, 2015.

[3] M. De Nadai, J. Staiano, R. Larcher, N. Sebe, D. Quercia, and B. Lepri, "The Death and Life of Great Italian Cities: A Mobile Phone Data Perspective," in *Proceedings of the 25th international conference on world wide web*, pp. 413–423, International World Wide Web Conferences Steering Committee, 2016.

[2]https://ngi.delabapps.eu/

[4] F. Calabrese, L. Ferrari, and V. D. Blondel, "Urban sensing using mobile phone network data: a survey of research," *Acm computing surveys (csur)*, vol. 47, no. 2, p. 25, 2015.

[5] M. Balduini, E. Della Valle, M. Azzi, R. Larcher, F. Antonelli, and P. Ciuccarelli, "Citysensing: Fusing city data for visual storytelling," *IEEE MultiMedia*, vol. 22, no. 3, pp. 44–53, 2015.

[6] K. Yamanishi, J.-I. Takeuchi, G. Williams, and P. Milne, "On-line unsupervised outlier detection using finite mixtures with discounting learning algorithms," in *Proceedings of the sixth ACM SIGKDD international conference on Knowledge discovery and data mining*, pp. 320–324, ACM, 2000.

[7] S. Sathe and C. C. Aggarwal, "Subspace outlier detection in linear time with randomized hashing," in *Data Mining (ICDM), 2016 IEEE 16th International Conference on*, pp. 459–468, IEEE, 2016.

[8] C. C. Aggarwal, "Outlier ensembles: position paper," *ACM SIGKDD Explorations Newsletter*, vol. 14, no. 2, pp. 49–58, 2013.

[9] E. Schubert, A. Zimek, and H.-P. Kriegel, "Fast and scalable outlier detection with approximate nearest neighbor ensembles," in *DASFAA (2)*, pp. 19–36, 2015.

[10] C. Consonni, P. Sottovia, A. Montresor, and Y. Velegrakis, "Discovering order dependencies through order compatibility," in *International Conference on Extending Database Technology*, vol. 2019, 2019.

[11] A. K. Elmagarmid, P. G. Ipeirotis, and V. S. Verykios, "Duplicate Record Detection: A Survey," *IEEE Trans. Knowl. Data Eng.*, 2007.

[12] I. Witten and D. Milne, "An Effective, Low-cost Measure of Semantic Relatedness Obtained from Wikipedia Links," in *AAAI '08*.

[13] S. Siersdorfer, P. Kemkes, H. Ackermann, and S. Zerr, "Who With Whom And How?: Extracting Large Social Networks Using Search Engines," in *CIKM '15*.

[14] A. Spitz and M. Gertz, "Terms over LOAD: Leveraging Named Entities for Cross-Document Extraction and Summarization of Events," in *SIGIR '16*.

[15] A. D. Sarma, A. Jain, and C. Yu, "Dynamic Relationship and Event Discovery," in *WSDM '11*.

[16] D. Hienert and F. Luciano, "Extraction of Historical Events from Wikipedia," in *ESWC '12*.

[17] J. Allan, J. G. Carbonell, G. Doddington, J. Yamron, and Y. Yang, "Topic Detection and Tracking Pilot Study Final Report," *Broadcast News Transcription and Understanding Workshop*, 1998.

[18] A. Abujabal and K. Berberich, "Important Events in the Past, Present, and Future," in *WWW '15*.

[19] E. Kuzey and G. Weikum, "Extraction of Temporal Facts and Events from Wikipedia," in *TempWeb '12*.

[20] C. Lin, C. Lin, J. Li, D. Wang, Y. Chen, and T. Li, "Generating Event Storylines from Microblogs," in *CIKM '12*.

[21] S. Phithakkitnukoon, T. Horanont, G. Di Lorenzo, R. Shibasaki, and C. Ratti, "Activity-aware map: Identifying human daily activity pattern using mobile phone data," in *International Workshop on Human Behavior Understanding*, pp. 14–25, 2010.

[22] Y. Yuan and M. Raubal, "Extracting dynamic urban mobility patterns from mobile phone data," in *International Conference on Geographic Information Science*, pp. 354–367, 2012.

[23] J. L. Toole, M. Ulm, M. C. González, and D. Bauer, "Inferring land use from mobile phone activity," in *Proceedings of the ACM SIGKDD international workshop on urban computing*, pp. 1–8, ACM, 2012.

[24] P. Palensky and D. Dietrich, "Demand side management: Demand response, intelligent energy systems, and smart loads," *IEEE transactions on industrial informatics*, vol. 7, no. 3, pp. 381–388, 2011.

[25] J. Candia, M. C. González, P. Wang, T. Schoenharl, G. Madey, and A.-L. Barabási, "Uncovering individual and collective human dynamics from mobile phone records," *Journal of physics A: mathematical and theoretical*, vol. 41, no. 22, p. 224015, 2008.

[26] V. Traag, A. Browet, F. Calabrese, and F. Morlot, "Social event detection in massive mobile phone data using probabilistic location inference," 2011.

[27] C. C. Noble and D. J. Cook, "Graph-based anomaly detection," in *SIGKDD*, pp. 631–636, 2003.

[28] J. Huan, D. Bandyopadhyay, W. Wang, J. Snoeyink, J. Prins, and A. Tropsha, "Comparing graph representations of protein structure for mining family-specific residue-based packing motifs," *J. Comput Biol.*, vol. 12, no. 6, pp. 657–671, 2005.

[29] J. Pei, J. Han, B. Mortazavi-Asl, and H. Zhu, "Mining access patterns efficiently from web logs," in *PAKDD*, pp. 396–407, 2000.

[30] H. Jiang, H. Wang, P. S. Yu, and S. Zhou, "Gstring: A novel approach for efficient search in graph databases," in *ICDE*, pp. 566–575, 2007.

[31] W. Jiang, J. Vaidya, Z. Balaporia, C. Clifton, and B. Banich, "Knowledge discovery from transportation network data," in *ICDE*, pp. 1061–1072, 2005.

[32] X. Yan, P. S. Yu, and J. Han, "Graph indexing: A frequent structure-based approach," in *SIGMOD*, pp. 335–346, 2004.

[33] N. Vanetik, S. E. Shimony, and E. Gudes, "Support measures for graph data," *Data Min. Knowl. Discov.*, vol. 13, no. 2, pp. 243–260, 2006.

[34] M. Fiedler and C. Borgelt, "Subgraph support in a single large graph," in *ICDM Workshops*, pp. 399–404, 2007.

[35] B. Bringmann and S. Nijssen, "What is frequent in a single graph?," in *PAKDD*, pp. 858–863, 2008.

[36] M. Elseidy, E. Abdelhamid, S. Skiadopoulos, and P. Kalnis, "Grami: Frequent subgraph and pattern mining in a single large graph," *PVLDB*, vol. 7, no. 7, pp. 517–528, 2014.

[37] J. C. Costello, M. M. Dalkilic, S. M. Beason, J. R. Gehlhausen, R. Patwardhan, S. Middha, B. D. Eads, and J. R. Andrews, "Gene networks in drosophila melanogaster: integrating experimental data to predict gene function," *Genome biology*, vol. 10, no. 9, p. R97, 2009.

[38] H. Wang and C. C. Aggarwal, "A survey of algorithms for keyword search on graph data," in *Managing and Mining Graph Data*, pp. 249–273, 2010.

[39] P. Bogdanov, M. Mongiovì, and A. K. Singh, "Mining heavy subgraphs in time-evolving networks," in *Data Mining (ICDM), 2011 IEEE 11th International Conference on*, pp. 81–90, IEEE, 2011.

[40] J. Yang, W. Su, S. Li, and M. M. Dalkilic, "Wigm: Discovery of subgraph patterns in a large weighted graph," in *SDM*, pp. 1083–1094, 2012.

[41] M. J. Shaw, C. Subramaniam, G. W. Tan, and M. E. Welge, "Knowledge management and data mining for marketing," *Decision support systems*, vol. 31, pp. 127–137, 2001.

[42] G. Preti, M. Lissandrini, D. Mottin, and Y. Velegrakis, "Beyond frequencies: Graph pattern mining in multi-weighted graphs," 2018.

[43] C. H. Teixeira, A. J. Fonseca, M. Serafini, G. Siganos, M. J. Zaki, and A. Aboulnaga, "Arabesque: a system for distributed graph mining," in *Proceedings of the 25th Symposium on Operating Systems Principles*, pp. 425–440, ACM, 2015.

[44] J. Chan, J. Bailey, and C. Leckie, "Discovering correlated spatio-temporal changes in evolving graphs," *Knowledge and Information Systems*, vol. 16, no. 1, pp. 53–96, 2008.

[45] J. Chan, J. Bailey, C. Leckie, and M. Houle, "ciforager: Incrementally discovering regions of correlated change in evolving graphs," *ACM Transactions on Knowledge Discovery from Data (TKDD)*, vol. 6, no. 3, p. 11, 2012.

[46] O. Erling, "Virtuoso, a hybrid rdbms/graph column store.," *IEEE Data Eng. Bull.*, vol. 35, no. 1, pp. 3–8, 2012.

[47] M. Lissandrini, M. Brugnara, and Y. Velegrakis, "Beyond Macrobenchmarks: Microbenchmark-based Graph Database Evaluation," in *VLDB'19, Proceedings of 45th International Conference on Very Large Data Bases, August 26-30, 2019, Los Angeles, California, USA*, 2019.

[48] D. A. Keim, "Information visualization and visual data mining," *IEEE Transactions on Visualization & Computer Graphics*, no. 1, pp. 1–8, 2002.

[49] S. Sarawagi, R. Agrawal, and N. Megiddo, "Discovery-driven exploration of olap data cubes," in *International Conference on Extending Database Technology*, pp. 168–182, Springer, 1998.

[50] R. Agrawal, J. Gehrke, D. Gunopulos, and P. Raghavan, "Automatic subspace clustering of high dimensional data for data mining applications," in *SIGMOD 1998, Proceedings ACM SIGMOD International Conference on Management of Data, June 2-4, 1998, Seattle, Washington, USA.* (L. M. Haas and A. Tiwary, eds.), pp. 94–105, ACM Press, 1998.

[51] D. Deng, R. C. Fernandez, Z. Abedjan, S. Wang, M. Stonebraker, A. K. Elmagarmid, I. F. Ilyas, S. Madden, M. Ouzzani, and N. Tang, "The data civilizer system," in *CIDR 2017, 8th Biennial Conference on Innovative Data Systems Research, Chaminade, CA, USA, January 8-11, 2017, Online Proceedings*, 2017.

[52] E. Wu and S. Madden, "Scorpion: Explaining away outliers in aggregate queries," *PVLDB*, vol. 6, no. 8, pp. 553–564, 2013.

[53] S. Aridhi, M. Brugnara, A. Montresor, and Y. Velegrakis, "Distributed k-core decomposition and maintenance in large dynamic graphs," in *Proceedings of the 10th ACM International Conference on Distributed and Event-based Systems, DEBS '16, Irvine, CA, USA, June 20 - 24, 2016*, pp. 161–168, 2016.