Ontology Based Recommender System Using Social Network Data

Mohamad Arafeh

Computer Science Department, Università degli Studi di Milano, Milan, Italy

Paolo Ceravolo

Computer Science Department, Università degli Studi di Milano, Milan, Italy

Azzam Mourad

Computer Science and Mathematics Department, Lebanese American University, Beirut, Lebanon

Ernesto Damiani

Center of Cyber-Physical System, Khalifa University, Abu Dhabi, UAE

Emanuele Bellini

Mathematics and Physics Department, University of Campania "Vanvitelli", Caserta Italy

---

Abstract

Online Social Network (OSN) is considered a key source of information for real-time decision making. However, several constraints lead to decreasing the amount of information that a researcher can have while increasing the time of social network mining procedures. In this context, this paper proposes a new framework for sampling Online Social Network (OSN). Domain knowledge is used to define tailored strategies that can decrease the budget and time required for mining while increasing the recall. An ontology supports our filtering layer in evaluating the relatedness of nodes. Our approach demonstrates that the same mechanism can be advanced to prompt recommendations to users. We broke our experiments into two sets, testing on data generated randomly and other collected from Twitter. Our test cases and experimental results emphasize the importance of the strategy definition step and the application of ontologies on the knowledge graph.

Keywords: social network, data miner, big data, data analysis, data sampling, ontology, recommender system.

1. Introduction

In recent years, significant attention has been spent on mining Online Social Network (OSN) in real-time. Decision analytics methods, from marketing to emergency management, from politics to business and management benefit of real-time or near real-time event processing. Event detection is for example crucial in traffic management [1], fire control [2], TV show hosting [3], and smart-city management systems [4]. In combination with other data sources, OSN can boost complex decision making and risk management methodologies [5]. For instance, Twitter has been effectively exploited in many real-world incidents to communicate disaster warnings and disseminate information, capture the evolving trends, control resource consumption, or discover effective mitigation strategies bottom-up [6], [7], [8].

However, OSN data must be treated with proper confidence levels. What is happening because of the Coronavirus is a lively case study. In fact, if during the epidemics risen before the advent of social media, experts had to wait for a publication in an academic journal to know the progress of the disease outbreaks. Nowadays sharing information between experts is much faster. On the other hand, the ease with which information is published and the speed with which it spreads pose new challenges when this information is incorrect or false (e.g. fake news). The so-called *infodemics* needs to be promptly dealt trying to eliminate the noise generated by the unverified news and by the alarms caused by the fear of contagion, spreading reliable information in the shortest time possible. In fact, the reliability of the social network-based event analysis depends on several factors. The actual presence of users on the ground acting as a sensor is the first one. In Florence, during a recent Arno river embankment collapse caused by a water pipe disruption, as the event happened at 6.15 AM on 25/05/2016, no relevant variation on Twitter had been detected, simply because there were

no Twitter users on the site to comment [9]. Social media are general-purpose communication platforms, for this reason filtering the activities that are related to the domain of analysis is crucial to avoid introducing selection bias.

To do so, a new class of agile and cost-effective methods and tools has been proposed to support operators in analyzing at a deeper level and closer to real-time the huge amount of data generated by OSN is paramount. Twitter, for example, gives researchers a gateway providing them with billions of information about users' links, written contents, and community circles, giving analytics a gateway for improving their algorithms primarily in Natural Language Processing [10], Link Prediction [11], Community Detection [12] and Sentiment Analysis [13]. However, such methods require a relevant amount of data to be processed that impacts on the timeliness of the result provided as well as the resources needed.

Several approaches have been proposed for mining OSNs while limiting the time and the budget required for mining [14], [15], [16], [17], [18], [19], [20], [21]. However, to the best of our knowledge, none of them is capable of using the mix of strategies we propose in this paper. In the present work, we propose a mining platform to help researchers and data collector to mine and directly analyze social networks, defining API-specific and budget-constrained strategies able to filter data collection based on concurrent sampling and ontology-enhanced filtering algorithms [22]. To test our approach we exploited it in creating a content-based recommender system. In our proposed architecture, recommendations are the results of the graph projection of social network nodes with their relationship and roles. In our approach, we are seeking machine learning to find people's entity from their shared contents, and ontologies to build a knowledge graph that maps the relations between the accounts and their environment. Additionally, we consider our platform for building ontology enhanced knowledge graph and use it for recommendation purposes.

The paper is organized as follows: In Section 2 we present the related work. In Section 3 we describe our proposed platform, including its architecture and components. In Section 4, we show the implementation of the ontologies as part

of our platform and the complete workflow of our recommender system analysis. In section 5, we present our system implementation. In Section 6, we present our experimental analysis, and finally the conclusion in Section 7.

2. Related Works

The value of the data collected from OSN lies in the potential they have to reveal hidden patterns or predict future dynamics or trends [23], [24] that are mostly impossible to do in other ways. However, the quality of the results of the analysis is intrinsically related to the method through which the datasets are created. Although extensive research has been carried out on data collection, some uncertainty remains about the existence of a standard sampling methodology to efficiently collect datasets from OSN [25].

Most of the OSN managing platforms provide APIs allowing anyone to query and amass large amounts of information in a relatively short time. Usually, the process requires the registration of an application first, then platform returns a set of tokens granting access to the streaming API. As of 2015 Twitter, which is the main source of information for researchers, decided to limit the number of queries that can be executed in a 15-minute window, which results in lowering the amount of information available for the analysis while increasing the time to mine the required resources. Another limitation includes the definition of a maximum number of request calls in a period that affects the informative potential of the generated datasets in case a huge amount of tweets is generated (and then lost) in certain cases (e.g. emergency). Finally, the impossibility to access to historical Twitter via the Twitter API, the limited number of characters of the message, and so on, force the developers to set up specific architectures and strategies for collecting tweets, while attempting to get them with a sufficient reliability [26], [27].

To avoid the aforementioned limitation, upgrade from standard to premium or enterprise API is needed and is subject to the payment of fees. However, researchers are finding new ways to address this issue, where a web scraper

mentioned in [28] works by parsing hypertext tags and retrieving plain text information embedded onto them. Since web scraper does not get information directly using API, they are not restricted by the limitation posed by OSN providers and thus can mine large amounts of data with less time and budget. But using a web scraping tool has its limitations. A web scraper cannot be used for long term monitoring since websites are in constant changes over time, thus web scrappers must be updated constantly to be aligned to the new updates. This condition is particularly evident in [29], where a web-based crawler for collecting vulnerabilities information from the dark web should be adapted each time a harvesting campaign is about to start. Additionally, each OSN requires a custom web scraper. This issue is not different from using OSN API since each platform provides its own. However, the development of a reliable scrapper requires a fair amount of work and knowledge that a researcher may not have. Finally scrapping may pose the researcher to trials, e.g. in [4], LinkedIn sued peoples that anonymously scraped their website for different reasons like a violation of computer fraud and abuse act (CFAA), trespass and breach of contract.

Another approach discussed in the literature is sampling where a small fraction of the OSN users is mined to create a sampling representative of the whole OSN. There are several sampling techniques for OSN that aim to optimize the effort in terms of time, computation load, and dataset representativeness. In [30] a sampling-based algorithm for efficiently exploring a user's social network respecting its structure and for quickly approximating quantities of interest is proposed. In [31][32] and [27] the sampling strategy is based on the concept of Channel, which consists of a set of simple and complex search queries performed on the Twitter platform by the Crawler engine. The simplest Channel to be monitored can refer to collect and analyze tweets referring to a single Twitter user, user citation, hashtag, or keyword. Complex Channels may consist of several queries designed according to the search query syntax of Twitter APIs by combining keywords, user IDs, hashtags, citations, etc., with some operators (e.g., and, or, from). Thus, a user can design its channel and run a collection

process on OSN. However, this method is limited by the fact that the user should make some assumptions in defining the query filters (e.g. hashtag) that could not be appropriate to create a comprehensive dataset to analyze a specific phenomenon.

In [25], the authors explored the use of random search algorithms to sample OSN such as a Brownian walk (based on a normal distribution), a spiral-inspired walk, and a Reservoir sampling algorithm. The scope is to define a standard sampling methodology applicable where the OSN information flow is readily available. In [33], four sampling algorithms such as DLAS, EDLAS, ICLA-NS, and FLAS based on learning automata, are explored to produce a scale-down representative subgraphs from OSN. The random walk exploring strategy, adopted in [14], [15], [16], [17], provides the base method to ensure unbiased sampling. Random walk, however, requires a long mixing time, i.e. it requires a long startup period before guaranteeing good accuracy [18]. An effective way for overcoming this issue is to incorporate uniform node sampling (UNI) into random walk sampling and enable the strategy to jump to other parts of the graph. Different authors developed this random walk with a jump approach. An alternative solution to address the same issue is developing a multi-layered social network, where multiple sources can be followed to exit the blind roads or the local boundaries that a random walk can enter. Additionally, another form of traversal algorithm are mentioned in [34], [35], [36] to boost data transmission performance and to reduce energy and data consumption.

Researchers have also pointed out that sampling based on social media APIs is biased by policies that are constructed to save the vendor's resources and not for optimizing the sampling power of data [37], [38]. This means scholars using data obtained via API need to apply caution when drawing inferences from such data. In particular, it has been observed that the source of biases arise from the order connected nodes are returned [23] based on the age of the link created between two nodes and on the fixed time-frames used for selecting the nodes to be included in the sample APIs [39].

In the domain of the recommender systems, ratings and features are widely

used to infer the recommendation probabilities. Depending on the needs, researchers used either one or both of them to achieve their promised results. In [40], Nilashi et al. propose an ontology-based recommendation system combined with dimensionality reduction in order to reduce the issues of a sparse dataset. It uses users' ratings and features as an input to infer a probability of recommendation. Similarly, the use of dimensionality reduction is also discussed in [41], where the authors deploy the said methods under two Real-world experiments and compare it with collaborative filtering. In [42], the authors demonstrate an existing relationship between an item and its location by developing a location-aware recommendation system. The system takes advantage of the data localization and the ratings to produces recommendation probabilities. Alternative to the recommendation that focuses on the ratings, Yao et al. propose in [43] a recommendation system in the auto industry where the availability of item ratings are believed to be scarce and inapplicable. Hence, the authors take advantage of the customer's common features to build relations between them.

The approach followed in the present work is to let the user compose strategies using a mixture of approaches and constraints. This supports the setting of API-specific and domain-specific solutions with the ability to compare alternative strategies in order to assess them in real-world scenarios. Additionally, we ought to explore our platform to build ontologies by taking advantage of the graphical nature of the saved data. Furthermore, we propose an ontology enhanced graph analysis in the scope of recommendation systems, which to the best of our knowledge, none of the current approaches has addressed it yet.

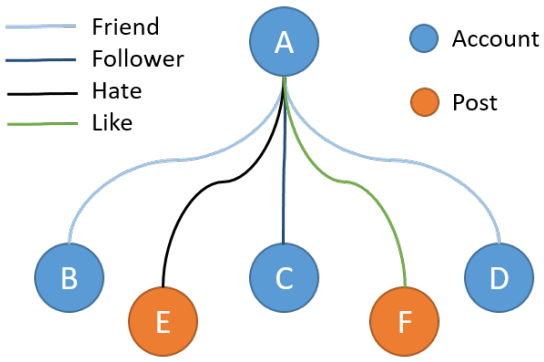

Figure 1: Relationships Between Nodes in a Graph Database

3. A Framework for Sampling Social Networks

In this section, we present our framework. Accordingly, we discuss the system hierarchy organizing our architecture with its abstraction layer, we detail the workflow guiding the mining procedures, the network space exploring algorithms and the different filtering strategies that make the system effective.

3.1. System Architecture and Components

A Social Network is an ever-expanding data source. For this reason, an effective mining procedure must rely on real-time data collection. Also, an evolving domain may require to extend the computational capacity of the system. In order to address this issue, we used different technologies that helped to achieve maximum scalability in our architecture, by interfacing separated components using abstract classes. Figure 2 presents the hierarchy of our architecture listing the abstract classes that compose it.

The system architecture directly reflects the elements of a strategy with a software component for managing each element independently. At the root level, we have a class for defining mining strategies. Each strategy is a combination of multiple settings managed by separated components.

Data Sources. A strategy has to contain a connection to an input source that the miner uses for querying data. Sources could be online like Twitter or Facebook, or locally available like a local SQL Database, or data files (CSV,

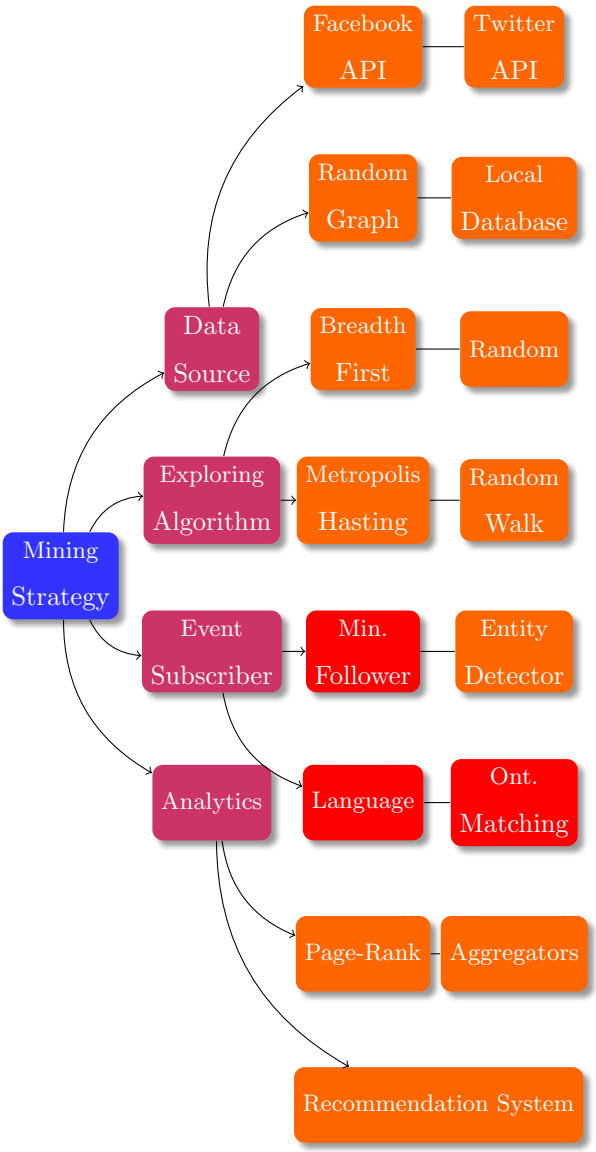

Figure 2: The Architectural Hierarchy

Excel). Abstracting such component allowed us to have a limitless source in which we were able to sustain the daily increasing number of the available data.

Network Space Exploring Algorithm. As described in Section 3, the network space exploring algorithm is used to navigate through a data source embedding the mined data as a graph. Moreover, such algorithms are also used to sample the data source decreasing the time and budget spent on mining and data analysis.

Event Subscribers. This component allows integrating further components beyond the one provided in the architecture. The idea came from the need for adding constantly additional features in one hand and a data transformation on another hand. Upon each step, this component broadcasts the current status to subscribers, thus allowing them to modify and transform the data. This component adopts a limitless amount of custom-defined filters and data embedders. As en example, we have introduced an ontology enhanced event subscriber as an advanced filtering technique to eliminate nodes that are unrelated to the specified case. More information is available in Section 4.

Analytics. This component contains the algorithms used to run analytics from the mined data. Currently, we support all the graph algorithms natively supported by Neo4j, including Centrality algorithms, Community detection algorithms, Pathfinding algorithms, Similarity algorithms, and Link Prediction algorithms. Additionally, we have introduced a new recommender system algorithm that assigns recommendation probability for each node in the knowledge graph through a given ontology model.

3.2. System Workflow

Figure 3 describes in detail the data processing workflow of our system. The process starts when the network space exploring algorithm navigates the social network graph choosing the first node. The results differ based on the algorithm selected while defining the strategy. The next step is scanning the selected node, thus allowing further routs of the social graph to explore. Moreover, more detail on the nodes demands additional specialized requests to fetch them.

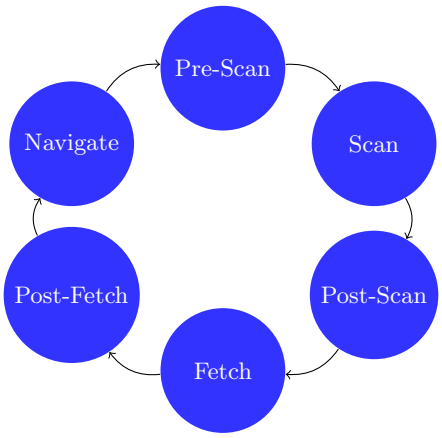

Figure 3: The System Workflow

To reduce the number of API requests, we introduced a caching system that answers the call in case it was already available in the cache. Pre-Scan, Post-Scan, and Post-Fetch are event subscribers that run before and after scanning and after fetching. Such subscribers can be used as data filters and mappers introducing new procedural information to the knowledge graph. An example of Pre-Scan could be max-level filters that prevent adding any additional node located after the specified level, and max-fetch filters that limit the number of nodes that can be scanned thus reducing further the number of API requests. Post-Fetch filters are usually used for filtering data based on nodes attributes where such information is only available after the fetch request. Other usages of post fetch are like Entity Detectors, which introduce new information to the knowledge graph based on predefined procedures subscribed to receive updates when such an event arouses.

3.3. Network Space Exploring Algorithm

An important part of our framework depends on the space exploring algorithms as they represent a crucial component when defining a strategy. With such algorithms, we can navigate through social networks and embed them as a graph in our data storing system. Graph Databases are used to store OSN

accounts and posted-contents as nodes, while the link between them is captured by edges. Figure 1 shows a graphical representation of the stored data. Edges can be labeled to define the type of relationship interconnecting two nodes, i. e. friends, follower, co-authors, etc. A post and its originator can be presented as two nodes connected by an edge labeled as posted-it. A post and a reader can be connected by like-it, hate-it. Addressing the limits posed by OSN providers, navigation algorithms combined with filters are used as data samplers working on a subset of data selected to be representative of the whole dataset. Sampling social network reduces the time and budget required to collect the minimum information needed. A similar approach has been discussed in [44], in which the authors comparatively assessed the accuracy of deterministic and probabilistic navigation algorithm. Since each case requires a different strategy, we have built our platform in such a way that allows the implementation of different navigation algorithms, allowing us to compare their performance and accuracy under different settings. The algorithms that are available in the platform can be divided into two groups: deterministic like Breadth-First, probabilistic like Forest Fire, Random Walker, and Metropolis Hasting. Additional focus has been given to probabilistic approaches, which can be supplied with hyper-parameters that are capable of changing the shape of the mined data, thus fitting more for data sampling work, while further widening the traversed space of the network. Frequent hyper-parameter used in the platform are forward weight and iterations. Forward weight controls the onward and backward jump rate of the random walker, the number range between 0 and 1, and the higher the number to deeper the level explored by the algorithm. This parameter significantly affects the accuracy of the results.

3.4. Node Filtering Strategies

Usually, any data analytics procedure includes data cleaning. Filtering is intended to prune irrelevant data, thus reducing the number of wasted requests. Filters such as minimum account followers, creation dates, and scam detectors can be exploited to identify fake accounts. In our framework, filters are part of

the event subscribers. A filter can be attached to receive continuous updates about the nodes in all of its three states, before scanning (Pre-Scan), after scanning (Post-Scan), and after fetching (Post-Fetch). Information provided to the event subscribers is relevant to the state they are subscribing to. In the Pre-Scan state, only the Id of the node and its current level are available, making it appropriate for a filter that depends on the level of the node. Post-Scan state provides more information about the shape of the network and how it will be extended. The scanned node will now reveal all of its possible children. Therefore, filters that depend on the number of node children like minimum twitter account followers are ideal for this state. The importance of the Post-Scan event resides by providing the last line of information that could eliminate a node before the actual fetch happens, thus claiming limited resources. Finally, the Post-Fetch state has the most information about the node and typically a higher impact on computational resources. Taking into consideration all the capabilities of the filter, it is possible to significantly narrow the area of our interest, therefore decreasing the time and the budget required.

4. Ontology Enhanced Event Subscribers for a Social Network-Based Recommender System

We built our ontology-based recommender system as a layer upon the mentioned framework. Using the capabilities of the event subscribers, we were able to intercept the process of saving the node and update the graph accordingly with the selected ontology.

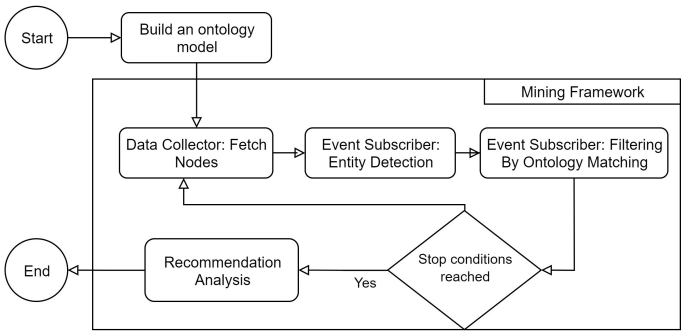

Figure 4: Recommender System Component Workflow

Figure 4 presents the complete steps of the proposed recommender system. The processes start by building an ontology model. In our approach, such a model is manually created using prior knowledge from the domain of analysis. The next steps are built independence on the mining framework, which will handle the role of collecting and fetching data from social networks. Furthermore, with the continuous update of the states of the graph provided by the framework, we examine each received node and detect its roles in the graph. Later, after we complete the required knowledge on the node, we start the filtering procedure by matching the node with the model. The process repeats until the stop conditions are reached, and later, the recommendation analysis is executed to assign a recommendation value for each node.

4.1. Ontology Model

In the adopted scenario, we are studying the users interacting with the twitter account of an academic conference for creating a recommendation system. An academic-related ontology has been manually developed and is presented in Figure 5. The conference class is one of its kind and it is manually provided upon the start of data collecting. The followers of each conference will be devised into a teacher, student, and attendee. To increase the accuracy of the recommender system, other properties that may affect the results have been also taken into consideration, e.g. the location of the conference, the location of the students and teacher, and the institution in which a student `study_in` or

a teacher `teach_at`.

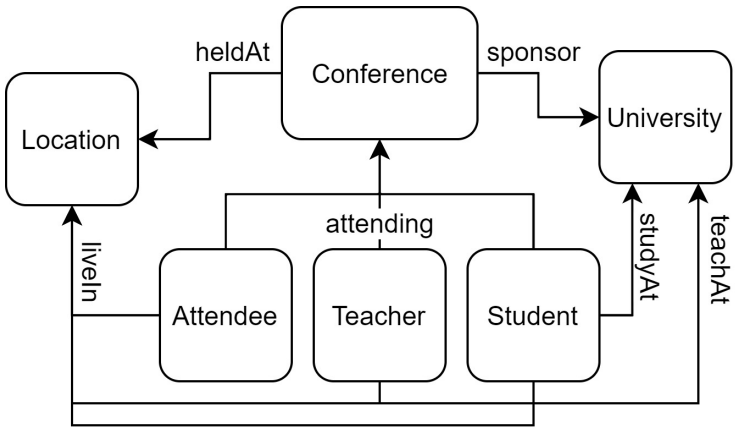

Figure 5: Academic Ontology Model

4.2. Data Collection

Data collection is handled by our mining framework using a mining strategy. For example, the level breakdown of a space exploring algorithm allows us to focus on a specific area of the network rather than the whole, thus reducing a large amount of data processing that may lead to no or few results. In our case, the focus was on the lower level. Therefore, as a network space exploring algorithm, Breadth-First could be a relevant algorithm for this task. However, other algorithms can be tuned to focus also on the lower levels. For example, with a small forward probability, RandomWalker and MetropolisHasting emphasize the backward moving rather than moving forward, causing the lower level nodes more significant.

4.3. Entity Detection Approaches

For a recommendation system to work accurately, nodes must be assigned to classes specified by the ontology model. Usually, specialized networks have well-defined entities assigned to each node. This is however not the case in generalist networks like twitter or other public social networks. We rely on labeled accounts with their description to train an entity classifier. With the

description as input and the known entity as a label, we obtained an efficient training and test sets. The entity classifier is made using a classical supervised learning algorithm. Therefore support vector machine was a solid selection. Once we built the classifier, we can determine the entity of each node using its description. Such a classifier is used as Post-Fetch event subscriber, thus receiving nodes when they are fetched and updating their entity accordingly.

4.4. Filtering & Ontology Matching

The amount of data a social network can provide is substantial but uncontrolled. As a consequence, a large portion can be cleaned and filtered out. Apart from the filters that we mentioned in the previous section (MaxLevel Filter and MinFollowers), a new filter is deployed to reduce the amount of data exposed to the recommendation system. In this step, we are seeking the elimination of all possible inaccurate or faulty recommendations. A graph-based ontology matching filter is proposed while considering the ontology model to match the labeled graph available after the discovery of the node entity. An approach similar to the DSSim-ontology [45] is exploited to extract the similarities between the node environment and our model. The goal is eliminating nodes that did not follow a shape equivalent to the provided model.

4.5. Recommendation Assignemnt Module

The recommendation assignment module of our ontology-based recommendation system approach. Figure 6 shows a sample of a graph results after the mining process. In this phase, we use the extracted entities from the generated knowledge graph to test for content similarity with the defined ontology model. Each relationship is associated with a weight relative to its equivalence in the model. For instance, a node that follows a similar structure as the model will be assigned with the same weight.

$$weight = \begin{cases} w_x & if\, relation\, x\, exists \\ 0 & o/w \end{cases} \tag{1}$$

Nodes level have been also taken into consideration. The farther the following node is, the lower is the possibility of it being recommended. Such a case is being handled by powering the weight by the level of the node.

$$weight(x_L) = weight^{\,L} \tag{2}$$

Moving to the recommendation algorithm, we adopted an improved version of the *Adamic Adar algorithm* to calculate the possibility of a node being recommend to another. Let $P(x, y)$ be the possibility of node y being recommended to attend an event x. Let $N(x)$ be the nodes adjacent to the node x, $N(y)$, the nodes adjacent to node y, and $N(z)$ the nodes that are adjacent to y and are remotely related to the node x. Finally, let U be the intersection between $N(x)$ and $N(y)$ including $N(z)$, and $N(u)$ their adjacent nodes. The recommendation evaluation can be defined as:

$$P(x, y) = \sum_{u \in N(x) \cap N(y) \cup N(z)} \frac{w_u}{log\,|N(u)|} \tag{3}$$

Based on equation (3), if $w_u = 0$, then the node log allocation index will be ignored and the results of the calculation will be lowered. Such a case happens when a relationship exists between both nodes that are not described in the defined model. For example, a follower node lives in a location that no one of the attendees lives in.

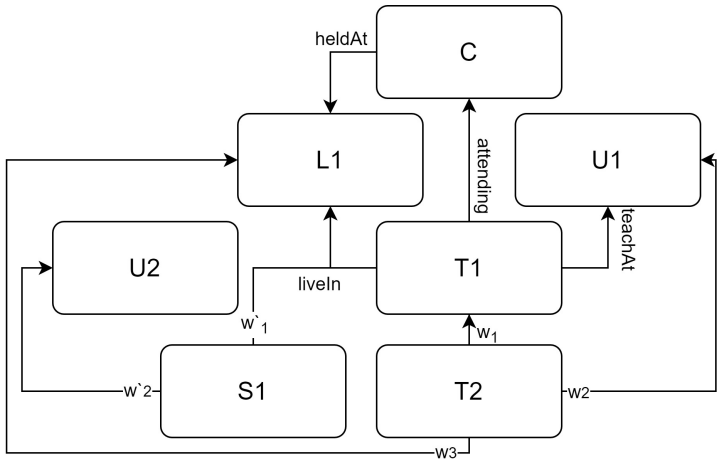

Figure 6: Example of a Labeled Graph Result

5. System Implementation

While the process of mining is always the same, mining source, graph navigation algorithm, event subscribers, and graph analysis always vary between strategies. Accordingly, in order to reduce the complexity and to increase the scalability, we tend toward abstracting all of the architecture main components, therefore, allowing them to have multiple implementations. Mined social network graphs are handled by Neo4j graph database. Using this technology, we are capable of maintaining the dynamic nature of social networks. Additionally, Neo4j is known for performant querying with further access to various graph algorithms that assist data analysis like centrality algorithms (e.g. PageRank). MongoDB is used to keep a history of the defined strategy and cache social network query results.

5.1. Mining Process

The mining process manages the interaction between the architecture components using a set of seed nodes to start the exploring process. It provides the navigator with the required data source while broadcasting regularly to a set of event subscribers. Additional information is presented in Algorithm 1. We start by creating a root for the graph to be mined. Then the main components are initialized from the strategy that includes the implementation to be adopted in the mining process, e.g. Breadth-First Navigator for exploring and Twitter as a data source.

The initialized navigator scans the root to get the seeds nodes. For each seed, it branches a navigator to explore further nodes. Navigation starts from the seed and ends when the navigator has no longer nodes to provide. The selected node that is going to be scanned will be broadcasted first to pre-scan event subscribers, thus allowing them to modify the node properties preventing it from being fetched or scanned further by marking it as unscannable. Later,

Algorithm 1: Mining Process

```
Data: strategy
let root denote the starting node of the graph;
extract navigator from strategy;
extract dataSource from strategy;
extract events from strategy;
observe(navigator, root);
analyse(root);
Function observe(navigator: Navigator, root: Graph) : void is
    scan(root);
    foreach seed ∈ root do
        navigate(navigator, seed);
    end
end
Function navigate(navigator: Navigator, node: Node) : void is
    do
        broadcast(node, "PreScan");
        if node == dead then
            broadcast(node, "Failed");
        else if node ≠ scannable then
            broadcast(node, "Failed");
        else
            scan(node);
            broadcast(node, "PostScan");
            if node == dead then
                broadcast(node, "Failed");
            else
                fetch(node);
                broadcast(node, "PostFetch");
            cache();
        node = navigator.next(node);
    while node ≠ null||stopped;
end
```

after a node is accepted and the scan is performed, the node will be sent to Post-Scan subscribers. Similarly, a node is marked as rather fetchable or not by comparing the new properties with the working strategy. Further, when the node is fetched, it will be sent to post fetch event subscribers to decide whether to keep this node if it has proven beneficial to the study or prune it otherwise. As aforementioned, the difference between pre-scan, post-scan, and post-fetch is that the later has access to an array of attributes that are not available in previous states. Finally, the fetched node is cached and used later when the same node is requested again.

5.2. RandomWalk

Algorithm 2: RandomWalk Navigator

```
Function next(node: Node) : Node is
    let R be a random generated number;
    extract strategy.weight into W;
    extract strategy.maxDepth into M;
    initialise f := forward if node == root;
    initialise f := backgward if node.level == M;
    initialise f := backgward if node is Leaf;
    initialise f := forward if R < W else backward;
    if f == forward then
        return a child selected randomly from node;
    else
        if parent(parent(node)) ≠ null then
            initialise ancestor := parent(parent(node));
        else
            initialise ancestor := parent(node);
        return a child selected randomly from ancestor;
end
```

As an example of the navigator procedure invoked by the Mining Process, we illustrate the RandomWalk algorithm. This algorithm serves under the network space exploring algorithm components and one of the three implementations besides breadth-first and metropolis hasting. The pseudo-code in Algorithm 2 shows the process in detail. It first initializes a random number and compares it with the weight defined in the strategy to decide to go further deep in the graph or returning to a higher level. The bigger the weight is, the deeper the walker will go.

6. Experimental Results and Analysis

In this section, we present the experimental environment in which we used to perform our tests. Also, we analyze in the detail the results of the comparison between different space exploring algorithms and the best use case of each. Finally, the enhanced recommendation algorithm is compared with its original equivalence while showing the difference between the results of both algorithms.

6.1. Experimental Setup

For the experimental setup, we ran our framework on a virtual machine setup with 2 cores and 4 processors each, thus resulting in a machine with 8 threads with a frequency of 3.6 GHz. Rams is tuned to use only 8GB. To hasten our experimental analysis, we point our miners to local data sources, which helped us comparing different strategies under different conditions in a minimum amount of time. We evaluate the accuracy of different strategies on two different data sets. In the first one, we built and tested our strategies using data provided from random generators. In the second one, experiments were performed on mined twitter accounts. For the recommendation system, we used a graph simulator to generate a knowledge graph that went through all the steps of the framework to finally be analyzed. The throughput of the analysis has been shown and compared to its original algorithm.

6.2. Recommendation System Analysis

Figures 7 and 8 show the difference between the analysis results of the Ontology Enhanced Adamic Adar algorithm and the original one over a set of multiple nodes. Since we are giving a different weight for each level, Figure 7 includes nodes that are the first-degree followers of the attendee, while Figure 8 includes nodes that are second-degree followers of the attendee. One of the most noticeable differences is the height of the line that is scaled down in the case of the enhanced algorithms. The reason is that the weight assigned to each relationship is being a constant to 0.5, in which the lines are shifted down since the results scale proportionally with the weight.

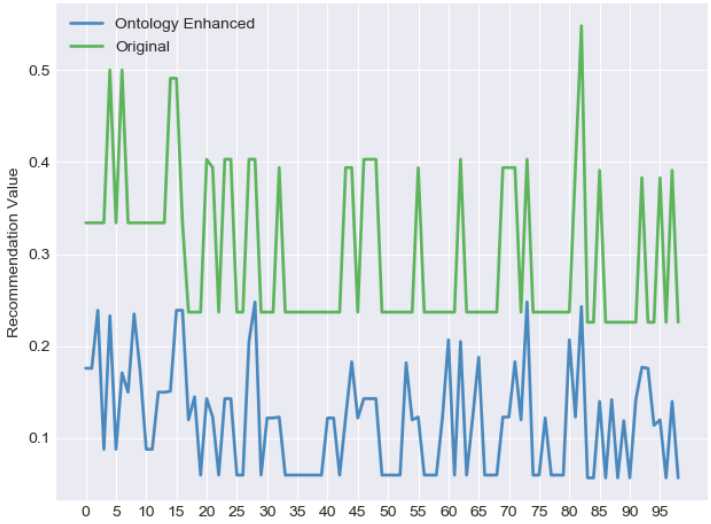

Figure 7: Level 1 Comparison

Moreover, in Figure 8, the changes in weight cause a scale down by two with respect to the results of the recommendation system in Figure 7. Other noticeable changes are related to points in the graphs that show a variation between the original and the enhanced algorithms. For instance, at point 25, points ranged from 30 to 45 have quite different results. The reason is related to the remote nodes that are adjacent to the current node and the node of interests. For example, a second-degree follower is the one that lived in the same location as someone attending the event. Such differences are due to assigning weights

based on the similarities between the nodes in the knowledge graph and its equivalence in the ontology model.

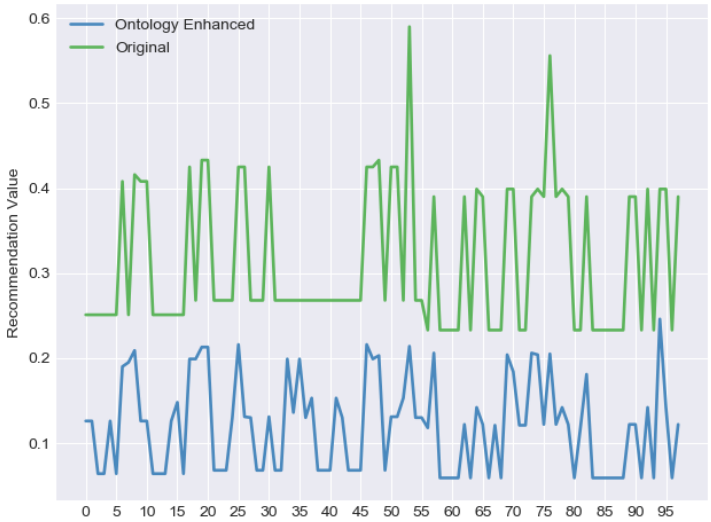

Figure 8: Level 2 Comparison

6.3. Recommendation System Experiments

In this experiment, we employed data produced by MoviesLens, which consists of 100k ratings from different users [46]. It also contains additional demographic information about each user including gender, age, and occupations. We have built the new ontology illustrated in Figure 9 for MoviesLens dataset that benefits from these features. For each movie, we have calculated the ratio of watching per occupation, age, and gender. For ages, we split them into four groups: Child, Teen, Grown, and Elder. We then devise a graph using the aforementioned ontology and apply our proposed algorithm to calculate the likelihood of each user for watching a specific movie. The results are normalized between 1 and 5 and act as a possible rating of a user for a movie.

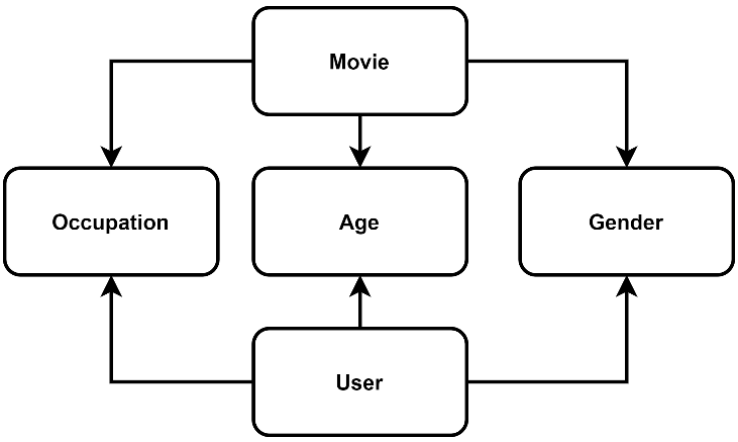

Figure 9: MovieLens Representative Ontology

For evaluating the results, we apply the Precision and Recall techniques on each user over his/her watched movies to explore the capabilities of our solution in accurately recognizing the possibilities for an item to be highly rated by the users. In this context:

$$Correctly\ Identified\ =\ High\ Rated\ Items \bigcap Rated\ Identified\ Items \quad (4)$$

$$Precision\ =\ \frac{Correctly\ Identified}{High\ Rated\ Identified\ Items} \quad (5)$$

$$Recall = \frac{Correctly\ Identified}{High\ Rated\ Items} \quad (6)$$

$$F1\ =\ \frac{2 \times P\ r\ e\ c\ i\ s\ i\ o\ n\ \times\ Recall}{Precision\ +\ Recall} \quad (7)$$

An item is considered Highly Rated if the rating is more than 3. The precision values reflect the percentage of the correctly identified items, while the recall values allow us to observe the percentage of identified items from the original set. Additionally, the Mean Absolute Error (MAE) is used to assess the accuracy of the results. These metrics provide insights about the expected

error margins.

$$MAE = \frac{\sum |R - N|}{T} \tag{8}$$

$$R \; = \; rating \; of \; user \; i \; to \; movie \; j \tag{9}$$

$$N \; = \; normalised \; calculated \; rating \; of \; user \; i \; to \; movie \; j \tag{10}$$

$$T \; = \; total \; number \; of \; ratings \tag{11}$$

Figure 10 presents the results of our experiments. In each iteration, we evaluate a single user based on the list of ratings. The results explore a high precision value for the majority of the users followed by a low recall, which indicates that the number of highly identified items is low but they are identified with high precision. Moreover, MAE results are considered fairly high compared to other algorithms that rely on ratings [47]. It is worth mentioning that, compared to the literature, the original ratings of the users are not needed in our proposed approach, which is a major advantage when there is limited or no knowledge about the current and new users preferences and historical data.

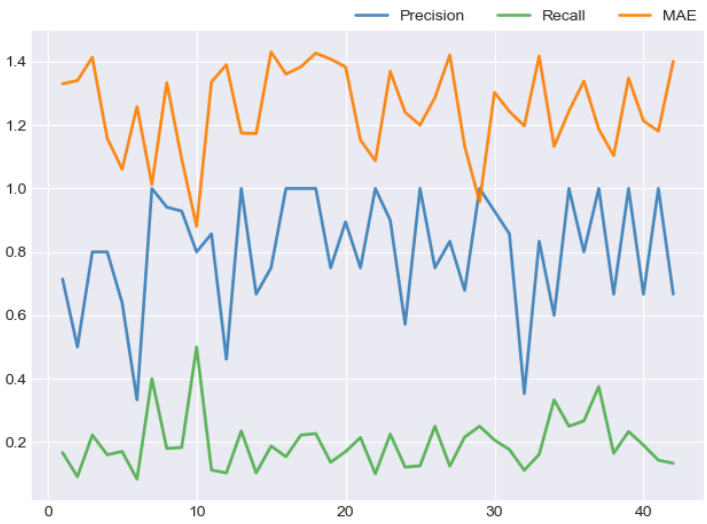

Figure 10: Recommendations Precision, Recall, and MAE

6.4. Network Space Algorithm Comparison

We propose a case where we need to mine the first level followers of a set of twitter accounts that respects predefined attribute conditions described in the strategy. Table 1 contains an overview of the data generated using four seed nodes to serve the experiments, focusing on the percentage of the seeds followers of Italian origins. Our experimental results are based on our miners fetching followers of this particular country.

Seed	1	2	3	4
Italy	62%	27%	50%	25%

Table 1: Set 1 Data Overview

The mining strategies tested are illustrated in Table 1. In all of them, we filtered the data and excluded any account that is not located in Italy. The first mining strategy uses Breadth First as exploring algorithms, fetching only 10% of the maximum account followers. The rest of the strategies are a combination of

using Random Walk and Metropolis Hasting as navigation algorithms. Different percentage of the maximum account number to be fetched has been used to test the algorithm accuracy. Lower fetching percentage results in a high sampling ratio since a lower number of accounts will be fetched and included in the test results. For Random Walk and Metropolis Hasting, we have 500 Iterations and 0.2 forward weight. Additionally, for metropolis hasting, we used Normal Distribution to generate the next mining position.

	S1	S2	S3	S4	S5
Exploring Algorithm	BF	RW	MH	RW	MH
Account Fetched	10%	10%	10%	5%	5%
Location Filter	Italy	Italy	Italy	Italy	Italy
Iterations	-	500	500	500	500
Forward Weight	-	0.2	0.2	0.2	0.2
Distribution	-	-	Normal	-	Normal

Table 2: Set 1 Mining Strategies

Experimental results in Figure 11 show the density of the Italian accounts for each seed in each strategy. We observe that the first strategy displays the worst case compared to the original data. Using breadth-first as navigation algorithm yields 53% for seed 1 compared to 62%, 78% for seed 3 compared to 50%, and 78% for seed 4 compared to 25%. This scenario happens when using breadth-first supplied with an attribute filter and a fetch filters for sampling while an important portion of the data of interest is located in the least of the data set. Other strategies show acceptable results since they reflect proportionally the main data even when using a higher sampling ratio.

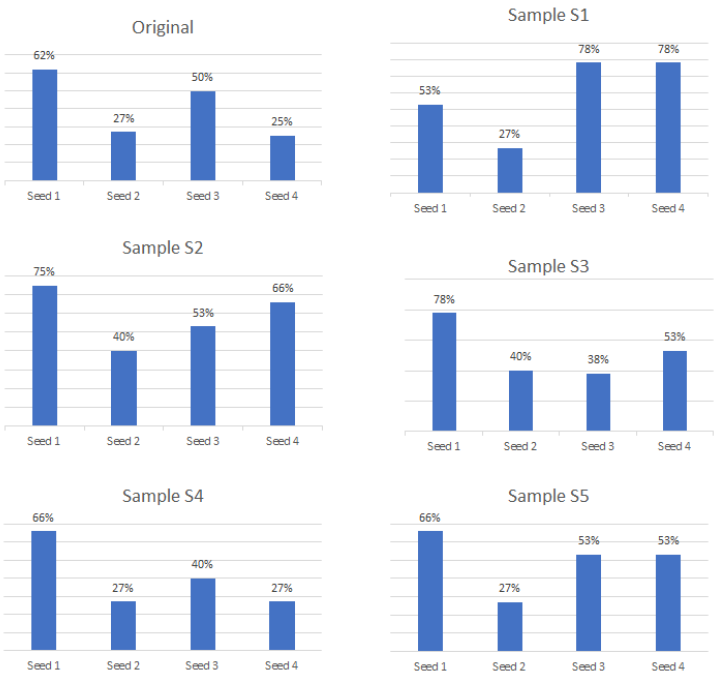

Figure 11: Density of the Italian accounts

In the second experiment, we used data provided from Kaggle [6], called Twitter Friends and hashtags. It is a collection of Twitter users that includes users information, friends, last seen, language, trending topics the user tweeted about, and others. We are interested in forming communities based on the frequent tags used by a user when posting on twitters. Our results show the density of the communities detected from the data. Since we do not have a global overview of the data, we include a strategy that iterates over a large portion of the dataset. Additionally, for the comparison, we have included other strategies that can be considered samples regarding the original one.

Table 3 shows the different mining strategies used by miners. For the first strategy, we intend to have a general idea to use it as a comparison measurement. Therefore, breadth-first is used as a navigator since we are not focusing on sampling. We have not specified any restriction on the maximum accounts that can be fetched in each node. The limitation of a maximum 8th depth level can

be considered very high since the data increases exponentially when depth level increases. For other strategies, we used the three remaining algorithms. Only 50 accounts can be fetched under each node. Breadth-first is limited for the first 3 levels, while Random Walk and Metropolis Hasting level are not set, including additional settings for "forward weight" = 0.8 allowing the navigators to expand their exploration territory deeply in the lower levels, therefore allowing them to have a more wide view on the data.

	S1	S2	S3	S4
Exploring Algorithm	BF	BF	RW	MH
Account Fetched	-	50	50	50
Max Depth	8	3	-	-
Iterations	-	-	500	500
Forward Weight	-	-	0.8	0.8
Distribution	-	-	-	Normal

Table 3: Set 2 Mining Strategies.

The amount of communities detected by miners is available in Table 4. The first strategy contains the highest possible knowledge about the dataset, therefore can be used to measure the accuracy of other sampling strategies. In total, 29 communities are identified in the first strategy, followed by 19 while employing Random Walk navigator. Metropolis Hasting (S4) ought the worst accuracy compared to the rest of the strategies.

Finally, Table 5 measures the number of API requests made, and the total execution time in seconds for each strategy. The numbers are based on simulated twitter APIs taking into consideration the limitation posed at the time of discussion. The results reveal that we can achieve through a well-defined

Strategy	Communities Detected
S1	29
S2	14
S3	19
S4	10

Table 4: Community Detected From Set 2

strategy high accuracy compared to the original with a significant reduction in time and API requests.

	S1	S2	S3	S4
API Req.	13650	379	289	177
Exec. Time (s)	56456	1879	1400	970

Table 5: Set 2 Execution Time.

7. Conclusions

OSNs can be considered as the main source of information for any Big Data analysis study. Our aim in this paper was to develop a scalable platform that keeps pace with the continuous development of OSNs and to bypass their restrictions that limit the effectiveness of the mined data. In this regard, we have introduced domain-specific sampling strategies that serve as input for platform miners. Moreover, we have demonstrated the capabilities of our platform by employing ontologies to reinforce our graphical representation with stronger relations and used them as part of the proposed recommendation system. In our

experiments, we have explored the importance of the strategy definition step as well as its impact on the quality of the results. Additionally, we have illustrated the implication of the ontologies on the graph and have used it on a real word dataset for a recommendation system.

References

[1] A.Nsouli, A.Mourad, D.Azar, Towards proactive social learning approach for traffic event detection based on arabic tweets, in: Proceedings of the IEEE International Wireless Communications and Mobile Computing Conference(IWCMC 2018), IEEE, 2018.

[2] F. Abel, C. Hauff, G. J. Houben, R. Stronkman, K. Tao, Twitcident: Fighting fire with information from social web streams, Proceedings of the 21st Annual Conference on World Wide Web Companion`doi:10.1145/2187980.2188035`.

[3] A. Crisci, V. Grasso, P. Nesi, G. Pantaleo, I. Paoli, I. Zaza, Predicting TV programme Audience by Using Twitter Based Metrics, Multimedia Tools and Applications, Springer.2017, 2017. `doi:10.1007/s11042-017-4880-x`.

[4] M. D. Lytras, V. Raghavan, E. Damiani, Big data and data analytics research, International Journal on Semantic Web and Information Systems 13 (1) (2017) 110. `doi:10.4018/ijswis.2017010101`.
URL `http://dx.doi.org/10.4018/IJSWIS.2017010101`

[5] E. Bellini, P. Ceravolo, P. Nesi, Quantify resilience enhancement of uts through exploiting connected community and internet of everything emerging technologies, ACM Transactions on Internet Technology(TOIT) 18 (1) (2017) 1–34.

[6] M. B. Habib, P. Apers, M. van Keulen, Neogeography: The challenge of channelling large and ill-behaved data streams, in: In the proceeding of

the 27th IEEE International Conference on Data Engineering Workshops, 2011, pp. 284–287.

[7] T. Hossmann, F. Legendre, P. Carta, P. Gunningberg, C. Rohner, Twitter in disaster mode, in: In the Proceedings of the 3rd Extreme Conference on Communication The Amazon Expedition - ExtremeCom 11, ACM Press, 2011. doi:10.1145/2414393.2414394.
URL http://dx.doi.org/10.1145/2414393.2414394

[8] B. A, J. And, A. MacEachren, A. Robinson, A. Jaiswal, S. Pezanowski, A. Savelyev, J. Blanford, P. Mitra, Geo-twitter analytics: Applications in crisis management.

[9] P. C. D. N. P. P. G. P. I. P. M. Bellini, E.and Bellini, An IoE and Big Multimedia Data approach for Urban Transport System management in Smart Resilient City, Informatics and Systems, Elsevier(under press, Sustainable Computing, Sustainable Computing, 2020.

[10] S. Verma, S. Vieweg, W. J.Corvey, L. Palen, J. H.Martin, M. Palmer, A. Schram, K. M. Anderson, Natural language processing to the rescue ? extracting "situational awareness" tweets during mass emergency, in: Proceedings of the Fifth International Conference on Weblogs and Social Media, 2011.

[11] P. K. A. P. D. A. Pandey, B.and Bhanodia, comprehensive survey of edge prediction in social networks: Techniques, parameters and challenges, Expert Systems with ApplicationsJan 23.

[12] F. S. Y. B. K. Dakiche, N.and Tayeb, Tracking community evolution in social networks: A survey, Information Processing & ManagementMay 1.

[13] L. Zhang, S. Wang, B. Liu, Deep learning for sentiment analysis: A survey, Wiley Interdisciplinary Reviews: Data Mining and Knowledge Discoverydoi:10.1002/widm.1253.

[14] X. Xu , C. Lee , D. Eun , A general framework of hybrid graph sampling for complex network analysis, in: Proceedings of the IEEE INFOCOM, 2014, pp. 2795–2803. doi:10.1109/INFOCOM.2014.6848229.

[15] M. B. C. M. A. Gjoka, M.and Kurant, Practical recommendations on crawling online social networks, IEEE J Sel Areas Commun 29 (9) (2011) 1872–1892.

[16] B. Ribeiro, P. Wang, F. Murai, D. Towsleyl, Sampling directed graphs with random walks, Proceedings of the IEEE INFOCOM (2012) 1692–1700doi:10.1109/INFCOM.2012.6195540.

[17] C. H. Lee, X. Xu, D. Eun, Beyond random walk and metropolis - hastings samplers: Why you should not backtrack for unbiased graph sampling, Sigmetrics Performance Evaluation Review - SIGMETRICS.

[18] A. Mohaisen, A. Yun, Y. Kim, Measuring the mixing time of social graphs, in: Proceedings of the 10th annual conference on Internet measurement - IMC 10, ACM Press, 2010. doi:10.1145/1879141.1879191.
URL http://dx.doi.org/10.1145/1879141.1879191

[19] K. Avrachenkov, B. Ribeiro, D. Towsley, Improving Random Walk Estimation Accuracy with Uniform Restarts, Springer Berlin Heidelberg, 2010, p. 98109. doi:10.1007/978-3-642-18009-5_10.
URL http://dx.doi.org/10.1007/978-3-642-18009-5_10

[20] B. Ribeiro, P. Wang, F. Murai, D. Towsleyl, Sampling directed graphs with random walks, Proceedings of the IEEE INFOCOM (2012) 1692–1700doi:10.1109/INFCOM.2012.6195540.

[21] M. Arafeh, P. Ceravolo, A. Mourad, E. Damiani, Sampling online social networks with tailored mining strategies, in: In the proceeding of the Sixth International Conference on Social Networks Analysis, Management and Security (SNAMS), IEEE, 2019. doi:10.1109/snams.2019.8931829.
URL http://dx.doi.org/10.1109/SNAMS.2019.8931829

[22] B. Halawi, A. Mourad, H. Otrok, E. Damiani, Few are as good as many: An ontology-based tweet spam detection approach, IEEE Access 6 (2018) 63890–63904.

[23] S. Sikdar, S. Adali, M. Amin, T. Abdelzaher, K. Chan, J. . Cho, B. Kang, J. O'Donovan, Finding true and credible information on twitter, in: Proceedings of the 17th International Conference on Information Fusion (FUSION), 2014, pp. 1–8.

[24] L. Madlberger, A. Almansour, Predictions based on twitter a critical view on the research process, in: Proceedings of the International Conference on Data and Software Engineering (ICODSE), 2014, pp. 1–6.

[25] C. A. Pia-Garca, C. Gershenson, J. M. Siqueiros-Garca, Towards a standard sampling methodology on online social networks: collecting global trends on twitter, Applied Network Science 1 (1). doi:10.1007/s41109-016-0004-1.
URL http://dx.doi.org/10.1007/s41109-016-0004-1

[26] M. Oussalah, F. Bhat, K. Challis, T. Schnier, A software architecture for twitter collection, search and geolocation services, Knowledge-Based Systems 37 (2013) 105120. doi:10.1016/j.knosys.2012.07.017.
URL http://dx.doi.org/10.1016/j.knosys.2012.07.017

[27] A. Crisci, V. Grasso, P. Nesi, G. Pantaleo, I. Paoli, I. Zaza, Predicting tv programme audience by using twitter based metrics, Multimedia Tools and Applications 77 (10) (2017) 1220312232. doi:10.1007/s11042-017-4880-x.
URL http://dx.doi.org/10.1007/s11042-017-4880-x

[28] A. Hernandez, G. Sanchez Perez, V. S. V. P. M. H. Toscano Medina, K.and Martinez Hernandez, A Web Scraping Methodology for Bypassing Twitter API Restrictions, 2018.

[29] S. Shiaeles, N. Kolokotronis, E. Bellini, Iot vulnerability data crawling and analysis, in: Proceedings of the IEEE World Congress on Services (SERVICES), Vol. 2642-939X, 2019, pp. 78–83.

[30] M. Papagelis, G. Das, N. Koudas, Sampling online social networks, IEEE Transactions on Knowledge and Data Engineering 25 (3) (2013) 662676. doi:10.1109/tkde.2011.254.
URL http://dx.doi.org/10.1109/TKDE.2011.254

[31] D. Cenni, P. Nesi, G. Pantaleo, I. Zaza, Twitter vigilance: A multi-user platform for cross-domain twitter data analytics, nlp and sentiment analysis, in: Proceedings of the IEEE SmartWorld, Ubiquitous Intelligence & Computing, Advanced & Trusted Computed, Scalable Computing & Communications, Cloud & Big Data Computing, Internet of People and Smart City Innovation (SmartWorld/SCALCOM/UIC/ATC/CBDCom/IOP/SCI), IEEE, 2017. doi:10.1109/uic-atc.2017.8397589.
URL http://dx.doi.org/10.1109/UIC-ATC.2017.8397589

[32] B. P. C. D. N. P. P. G. P. I. P. M. Bellini, E., An ioe and big multimedia data approach for urban transport system management in smart resilient city.

[33] A. Rezvanian, B. Moradabadi, M. Ghavipour, M. M. Daliri Khomami, M. R. Meybodi, Learning Automata Approach for Social Networks, Springer International Publishing, 2019. doi:10.1007/978-3-030-10767-3.
URL http://dx.doi.org/10.1007/978-3-030-10767-3

[34] J. Luo, J. Wu, Y. Wu, Advanced data delivery strategy based on multiperceived community with iot in social complex networks, Complexity 2020 (2020) 115. doi:10.1155/2020/3576542.
URL http://dx.doi.org/10.1155/2020/3576542

[35] J. Wu, Z. Chen, M. Zhao, Community recombination and duplication node traverse algorithm in opportunistic social networks, Peer-to-Peer Networking and Applications 13 (3) (2020) 940947. doi:10.1007/s12083-019-00833-0.
URL http://dx.doi.org/10.1007/s12083-019-00833-0

[36] J. Wu, Z. Chen, M. Zhao, An efficient data packet iteration and transmission algorithm in opportunistic social networks, Journal of Ambient Intelligence and Humanized Computingdoi:10.1007/s12652-019-01480-2.
URL http://dx.doi.org/10.1007/s12652-019-01480-2

[37] R. Tromble, A. Storz, D. Stockmann, We don 't know what we don' t know: When and how the use of twitter 's public apis biases scientific inference, SSRN Electronic Journal 10 (2017) 2139.

[38] F. Morstatter, J. Pfeffer, H. Liu, When is it biased ?, Assessing the Representativeness of Twitter 's Streaming API. arXiv. 10 (2014) 1145.

[39] K. M. F. Pfeffer, J.and Mayer, Tampering with twitter 's sample api, EPJ Data Science 7 (1) (2018) 50.

[40] M. Nilashi, O. Ibrahim, K. Bagherifard, A recommender system based on collaborative filtering using ontology and dimensionality reduction techniques, Expert Systems with Applications 92 (2018) 507520. doi:10.1016/j.eswa.2017.09.058.
URL http://dx.doi.org/10.1016/j.eswa.2017.09.058

[41] B. M. Sarwar, G. Karypis, J. A. Konstan, J. Riedl, Application of dimensionality reduction in recommender system - a case study, 2000.

[42] J. J. Levandoski, M. Sarwat, A. Eldawy, M. F. Mokbel, Lars: A location-aware recommender system, in: Proceedings of the IEEE 28th International Conference on Data Engineering, 2012, pp. 450–461.

[43] L. Yao, Z. Xu, X. Zhou, B. Lev, Synergies Between Association Rules and Collaborative Filtering in Recommender System: An Application to Auto Industry, 2019, pp. 65–80. doi:10.1007/978-3-319-95651-0_5.

[44] F. B. E. D. E. Ceravolo, P.and Ciclosi, Assessing strategies for sampling dynamic social networks, in: M. Lytras (Ed.), Visvizi A, Research & Innovation Forum 2019. RIIFORUM 2019. Springer Proceedings in Complexity.Springer, Cham, 2019.

[45] M. Nagy, D. M. Vargas-Vera, E. Motta, Dssim-ontology mapping with uncertainty.

[46] F. M. Harper, J. A. Konstan, The movielens datasets, ACM Transactions on Interactive Intelligent Systems 5 (4) (2016) 119. doi:10.1145/2827872.
URL http://dx.doi.org/10.1145/2827872

[47] M. Nilashi, O. Ibrahim, K. Bagherifard, A recommender system based on collaborative filtering using ontology and dimensionality reduction techniques, Expert Systems with Applications 92 (2018) 507520. doi:10.1016/j.eswa.2017.09.058.
URL http://dx.doi.org/10.1016/j.eswa.2017.09.058