Querying Influence Graphs over Online Social Networks Effectively and Efficiently (Discussion Paper)

Vincenzo Moscato[1], Antonio Picariello[1], Giancarlo Sperlí[1], and Alfredo Cuzzocrea[2]

[1] DIETI Dept., University of Naples "Federico II", Naples, Italy
{vincenzo.moscato,antonio.picariello,giancarlo.sperli}@unina.it
[2] DIA Dept., University of Trieste, Trieste, Italy
alfredo.cuzzocrea@dia.units.it

Abstract. From the *Social Networks Analysis* (SNA) perspective, *Viral Marketing* has the aim to maximize the number of people that become aware of a given product/service by identifying a few number of individuals, considered more "influential" that can be promoted products or services. In this paper we propose a novel concept of *influence graph* that can be easily derived by querying social network modelled as a graph. Furthermore, spread diffusion model has been defined as a *Combinatorial Multi-Armed Bandit* (CMAB) problem for retrieving the most influential users, without any kind of preliminary knowledge.

Keywords: Social Networks Analytics · Viral Marketing · Influence Diffusion and Maximization · Online Social Networks Models

1 Introduction

The use of *Online Social Networks* or OSNs have rapidly become an essential part of every day life, and today about 2 billions of users are connected on simple smart platforms, where single people or communities of people share personal information, feelings, opinions about their life or on public facts [1, 5], and, of course, such environments represent a novel channel for assessing, choosing and buying goods and services .

Sociologists find out that individuals are frequently, explicitly or implicitly, influenced by their social contacts while deciding whether to adopt an innovation (such as a political idea, or to buy a new product): the way in which new practices spread through a population should so mainly depend on the fact that people influence each others' behavior.

It appears really important for big companies to target "opinion leaders", because if they are able to influence users of a community, it will start a large cascade of further recommendations. One of the main problems here is to maximize the number of people that become aware of a given product, finding the best set of users to start the diffusion and, consequently, to maximize the spread: this is, for example, the basic idea behind each social advertisement campaigns, and it corresponds to solve an *Influence Maximization* (IM) problem, which has the goal to find a small subset of nodes that could maximize the spread of influence over a social network graph.

An application of this strategy in social advertising is surely the so-called *Viral Marketing*. Viral Marketing allows to improve the spread awareness about a given product/service through a sort of "word-of-mouth" advertising. In a nutshell, a company selects a fixed (small) number of individuals, considered more influential, and offers or discounts them the product/services, hoping that they will be recursively recommended: as in epidemic diffusion, the viral effect starts and new products can reach a large number of people. In this way, users will influence their neighbors/friends triggering a "cascade effect". Viral marketing thus capitalizes on the advantages of social networks, and in particular their high capacity of fast and effective information diffusion. In this paper, we propose a novel approach for viral marketing based on modeling OSNs as a *graph database*. Using *regular path queries*, we can then query the database and extract relevant "social paths", by which users influence the other ones, considering a number of different social interactions. Relevant social paths are thus opportunely merged into an *influence graph* that describes the influence spread over a social network, following the *Independent Cascade* (IC) model. Finally, we exploit a particular online learning problem, namely the *Combinatorial Multi-Armed Bandit* (CMAB) framework, to determine the most influential users. The chosen approach allows us to automatically estimate the influence probabilities during the influence maximization stage, without any preliminary knowledge.

2 OSN modeling

Our idea consists in modeling a generic OSN as a *graph database* that is an undirected edge-labeled graph $G = (V, L, E)$ where:

- V is the set of graph vertices, representing the main entities of a social network;
- L is a set of labels, usually belonging to a given vocabulary and describing the different kinds of relationships that can occur among the social network entities;
- $E \subseteq V \times L \times V$ is the set of edges;
- V and E are abstract data types with a set of properties (expressed using several attributes that can be different depending on the type of nodes/edges).

To better explain the proposed approach, we consider a case study based on YELP social network. YELP is an On-line Social Network in which users can build friendship relationships and submit reviews about business objects, described by a set of attributes (i.e. business hour, accessibility and parking and so on), across all industries. Furthermore, an evaluation metrics based on a 5-point rating is assigned to both entities for estimating their relevance with respect to their reviews and actions on YELP.

Considering our model, we can represent Yelp as a graph in which the vertices set is composed by *Users* and *Business objects* while friendship and reviews represent respectively the relationships between two users or a user and business object. Furthermore, vertices and edges can be described by a set of attributes.

Using this general model we have defined a methodology to analyze users' behavior, proposing the novel concepts of "social path".

A *social path* is a particular edge-labeled path, i.e. a sequence $p = (v_1, l_1, v_2, \dots, l_k, v_{k+1})$ with $(v_i, l_i, v_{i+1}) \in E$ for each $i \in \{1, , k\}$.

Given the graph in Figure 1, examples of social paths connecting two users belonging to YELP are: $p_1 = (vinni, friendship, giank)$ and $p_2 = (vinni, review, gamberorosso, review, giank)$.

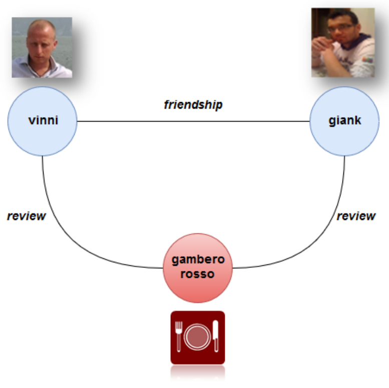

Fig. 1: An example of OSN graph using YELP data.

From a social paths, we can derive *relevant social paths*, i.e. special social paths that may be used in a variety of Social Networks Analytics applications. The elements belonging to a relevant social path must match a set of conditions Θ, defined on the attributes of nodes and edges belonging to the path.

For example, in YELP, the paths p_1 and p_2 can be relevant for an influence analysis problem and represent different ways by which a user can "influence" another one.

We explicitly note that a relevant social path can be expressed by regular path queries on a graph database [4]. A *regular path query* (RPQ) over the set of edge labels L is expressed as a regular expression over L. [3] In other terms, RPQs have the form $Q(x, y) : (x, R, y)$, R being a regular expression over the vocabulary of edge labels.

To extract the set of nodes' pairs connected by relevant social paths, we can first exploit regular path queries and then filtering the obtained results on the base of Θ conditions.

As an example, the query $Q(u_i, u_j) : (u_i, review, review, u_j)$ returns the set of users pairs that have performed a review on the same business object, and in particular, it can be represented using the graph in Figure 2.

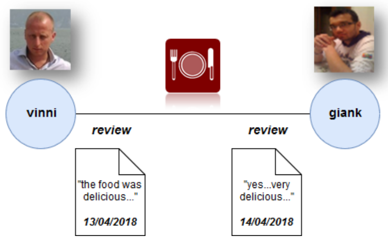

Fig. 2: An example of graph generated using RPQs on YELP.

Exploiting as further condition $\Theta = (e_2.t - e_1.t) \leq \delta t \wedge (e_1.m = e_2.m)$ (m and t respectively being the mood and timestamp of a given review), we can select only the set of users pairs in which the user u_j has performed a review immediately after u_i with the same mood.

3 Influence analysis: building an Influence Graph from an OSN

We can express more complex social paths using *conjunctive* RPQs [4]. As an example, the query $Q(u_i, u_j) : (u_i, review, review, u_j) \wedge (u_i, friendship, u_j)$ returns the set of users pairs that have performed a review on the same business object and are friends.

We define *influential paths* all the social paths that are relevant for an influence analysis problem. The execution of specific RPQS describing influential paths on the database graph (representing a particular social network) allows to determine a new homogeneous graph (vertices are only users), that we call "influence graph", we used to effectively model the influence spread over an OSN.

More precisely, an *Influence Graph* is a labeled graph $IG = (\hat{V}, \widehat{E}, \tau)$ where:

[3]The answer $Q(D)$ to a RPQ Q over a database D is the set of pairs of nodes connected in D by a social path traversing a sequence of edges forming a word in the regular language $L(Q)$ defined by Q.

– $\hat{V}$ is the set of nodes such that each $v \in \widehat{V}$ corresponds to a user $u \in U$;
– $\hat{E} \subseteq \hat{V} \times \hat{V}$ is the set of edge (with no self-loops);
– $\tau : \hat{V} \times \hat{V} \rightarrow [0,1]$ is a function that assigns to each edge $e = (v_i, v_j)$ a label, representing the probability τ that user u_i can influence user u_j.

To build an influence graph, we used a *sentiment analysis* technique to associate a predefined mood to each YELP review whilst we initially decided to exploit only the influence graph structure/topology, without considering the influence probabilities. In particular, we adopted a *Combinatorial Multi-Armed Bandit* (CMAB) approach [6] that allows us to automatically estimate the influence probabilities during the influence maximization stage. More in detail, the CMAB approach consists of multiple rounds, in each of them we leverage an IM method. Considering that we start without the knowledge of influence probabilities, the obtained spread will be low compared to the optimal solution and we refer to such situation as a *regret*. At each step, we then attempt to reduce the regret and, at the same time, to improve influence probabilities estimation using an *exploration-exploitation* trade-off.

In particular, the regret ρ after n rounds can be described by the Equation 1: it practically represents the difference between spread σ related to the optimal seed set S^* (knowing the right probabilities) and that returned by the CMAB strategy applied to current seed set S^i at iteration i.

$$\rho = n \cdot \sigma(S^*) - \sum_{i=1}^{n} \bar{\sigma}(S^i) \tag{1}$$

Indeed, CMAB is based on the following *exploration-exploitation* trade-off:

– exploration allows us to improve our knowledge, incrementally learning the influence probabilities over the network;
– exploitation allows us to achieve the largest spread by exploiting the current knowledge.

3.1 Modeling Influence maximization as CMAB problem

In our model, we consider m arms each one with a random variable $X_j^i \in [0,1]$ having μ_j as mean, which represents the obtained reward when the arm j is triggered at the round i. Furthermore, in the CMAB model it is possible to play a superarm A in each round: in this way all arms in A are triggered. We indicate with p_j^A the probability of arm j to be triggered when superarm A is played.

At each round, we then adopt an approximation oracle that leverages the current means' distribution $\hat{\boldsymbol{\mu}} = (\hat{\mu}_1, \hat{\mu}_2, ..., \hat{\mu}_m)$ to find the best (super)arm to play. After that, the superarm is played and consequently several arms are triggered. In the end, we observe the rewards obtained from each arm to update their mean estimation $\hat{\mu}_j$. To this goal, we use the number of times $n_{j,i}$ that an arm j has been triggered until the round i and the process continues until $i = n$.

In other terms, the CMAB relies on the *Independent Cascade* (IC) diffusion model that mimics the spread of an epidemic disease, evolving along discrete steps.

Given an influence graph $IG = (\hat{V}, \widehat{E}, \tau)$, where at each edge $(u, v) \in \widehat{E}$ is assigned a influence probability $\tau_{u,v} \in [0, 1]$, the process starts with an initial set of active nodes S_0 and each node $u \in S_0$ has a single chance to activate each outgoing inactive node v: it happens with probability $\tau_{u,v}$. The node v can become active in the next step and attempt to influence each one of its inactive out-neighbors. In this way the cascade continues until the process converges; the final spread $\sigma(S_n)$ represents the number of nodes that are activated starting from S_0 after n iterations. Thus, IM tries to maximize the final spread using reduced seed sets.

3.2 The IM algorithm

The procedure receives as input an influence graph IG without the knowledge of influence probabilities τ. In the initial step, it is used the vector $\hat{\boldsymbol{\mu}}_0$ that assigns 0 or a low influence probability to each edge (u, v).

At each round, the algorithm attempts to reduce the regret by implementing a particular exploration-exploitation trade-off technique (*coin flip*):

- the exploitation is performed using as approximation oracle the very simple and efficient greedy strategy proposed by the authors in [2,3] that allows to select a seed set S (with $|S| = k$) representing the played superarm A (it requires to solve an IM problem on the graph IG with influence probability estimating through the current means' vector $\hat{\boldsymbol{\mu}}$);
- the exploration is obtained generating a seed set S in a random way.

When a superarm is played, the diffusion process starts and leads to active several inactive nodes by triggering other arms. At the end of the process, it is evaluated the reward considering the number of nodes that are activated. This information is the used to update the means' estimation $\hat{\boldsymbol{\mu}}$ by the following Equation:

$$\hat{\mu}_j = \frac{\sum_{i=1}^{z} n_{j,i}}{z} \tag{2}$$

$n_{j,i}$ being the number of activated nodes playing the arm j at step i and $z \in [1, n]$. Thus, we improve the knowledge to minimize the regret in the next step.

4 System Overview and Implementation Details

Here, we present an overview of the system for viral marketing purposes, whose main components together with the related workflow are shown in Figure 3.

The system relies on a multilayer architecture typical of Big Data infrastructures based on the Apache Hadoop[4] technological stack and supports four main tasks: i) *data ingestion*, ii) *data storing*, iii) *batch computation*, iv) *data visualization*.

During the data ingestion, and using the functionalities provided by an ETL module, information about users' social interactions are:

[4]http://hadoop.apache.org/

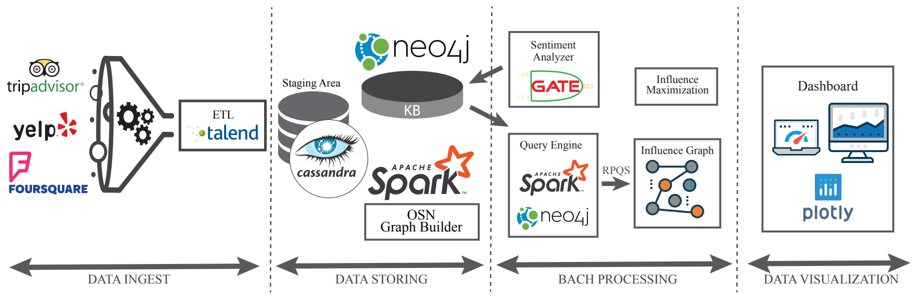

Fig. 3: System Architecture.

- periodically extracted from several OSNs (e.g., YELP, TripAdvisor, etc.) using the related API;
- cleaned and transformed in according to a particular log format containing information on user, timestamp and the performed action with several attributes (as an example ⟨ vinni, 13-05-2018, Gambero Rosso, "the food was delicious", ...⟩);
- loaded into a particular Staging Area.

Concerning implementation details, the ETL module was realized exploiting facilities provided by the Talend[5] tool. In turn, the Staging Area repository was implemented through the Cassandra[6] columnar database.

The data storing task has the goal of generating the OSN graph database by means of OSN Graph Builder module, using data from staging area. For each OSN, the final graph is then stored into system Knowledge Base (KB). The OSN Graph Builder module was realized on the top of Spark[7] engine (choosing Python as programming language), while we exploited Neo4j[8] for the KB.

During the batch computation task several activities are performed:

- textual reviews are processed to infer the related sentiment using Sentiment Analyzer module that is saved as edge property within the KB;
- on the base of a set of RPQs, the KB is queried by a Query Engine module to extract the influence graph;
- an influence maximization procedure (based on the CMAB theory) is applied on the influence graph to determine the most influential users (through Influence Maximization module).

Regarding implementation details, the Sentiment Analyzer module was implemented on the top of Spark engine leveraging several GATE[9] NLP libraries.

[5]https://www.talend.com/l
[6]http://cassandra.apache.org/
[7]https://spark.apache.org/
[8]https://neo4j.com/
[9]https://gate.ac.uk/

GATE detects *sentiment* related to each sentence in the text in terms of *score* (a numeric value for the sentiment), *sarcasm* (a boolean value to indicate whether the sentence is sarcastic or not) and *polarity* (a boolean value that can be **positive** or **negative**). RPQs are expressed in the Cypher Graph Query Language[10] and the Query Engine module was realized using Spark. The IG is stored in a textual raw format directly on HDFS. Eventually, the Influence Maximization module implements the CMAB IM algorithm in Python leveraging Spark graphs' management libraries.

Data visualization task allows to present results of several elaborations (e.g., influence graph, influential users, etc.) thanks to the functionalities provided by Data visualization module implemented using the Jupiter[11] tool.

5 Conclusion

In this paper we presented a novel approach for viral marketing based on modeling OSNs as particular graph databases. In particular, we derive the influence graph by analyzing influence paths and using CMAB technique.

Future works will be devoted to provide more evaluation of the proposed approach and to compare the proposed methodology with different and more recent influence maximization approaches.

References

1. Jordi Albors, Juan C Ramos, and Jose L Hervas. New learning network paradigms: Communities of objectives, crowdsourcing, wikis and open source. *International Journal of Information Management*, 28(3):194–202, 2008.
2. Flora Amato, Antonio Bosco, Vincenzo Moscato, Antonio Picariello, and Giancarlo Sperl. A novel influence diffusion model based on user generated content in online social networks. In *Proceedings of the 6th International Conference on Data Science, Technology and Applications*. SCITEPRESS - Science and Technology Publications, 2017.
3. Flora Amato, Vincenzo Moscato, Antonio Picariello, and Giancarlo Sperlí. Diffusion algorithms in multimedia social networks: a preliminary model. In *Proceedings of the 2017 IEEE/ACM International Conference on Advances in Social Networks Analysis and Mining 2017*, pages 844–851. ACM, 2017.
4. Angela Bonifati, Wim Martens, and Thomas Timm. An analytical study of large sparql query logs. *Proc. VLDB Endow.*, 11(2):149–161, October 2017.
5. Glen Urban. *Don't just relate-advocate!: a blueprint for profit in the era of customer power.* Pearson Education, 2005.
6. Sharan Vaswani, Laks Lakshmanan, Mark Schmidt, et al. Influence maximization with bandits. *arXiv preprint arXiv:1503.00024*, 2015.

[10]https://neo4j.com/cypher-graph-query-language/

[11]http://jupyter.org/