Finding Users of Interest in Micro-blogging Systems

Camelia Constantin
Univ. Pierre et Marie Curie
2 Place Jussieu
F75005 Paris, France
camelia.constantin@lip6.fr

Ryadh Dahimene
CNAM
2 rue Conté
F75141 Paris, France
ryadh.dahimene@cnam.fr

Quentin Grossetti
CNAM
2 rue Conté
F75141 Paris, France
quentin.grossetti@cnam.fr

Cédric du Mouza
CNAM
2 rue Conté
F75141 Paris, France
dumouza@cnam.fr

ABSTRACT

Micro-blogging systems have become a prime source of information. However due to their unprecedented success, these systems have to face an exponentially increasing amount of user generated content. As a consequence finding users who publish quality content that matches precise interests is a real challenge for the average user. We present in this article a recommendation score which takes advantage of the social graph topology and of the existing contextual information to recommend users to follow on a given topic. Then we introduce a landmark-based algorithm which allows to scale. Our experiments confirm the relevance of this recommendation score against concurrent approaches as well as the scalability of the landmark-based algorithm.

1. INTRODUCTION

Micro-blogs have become a major trend over the Web 2.0 as well as an important communication vector. Twitter, the main micro-blogging service, has grown in a spectacular manner to reach more than 570 million users in April, 2014 in less than seven years of existence. Currently around 1 million new accounts are added to Twitter every week, while there were only 1,000 in 2008. 500 million tweets are sent every day and on average a Twitter user follows 108 accounts[1]. Facebook is another example with 1.26 billion users who publish on average 36 posts a month. A Facebook user follows on average 130 "friends" which results in 1,500 pieces of information a user is exposed on average when he logs in[1]. Other similar systems like Google+, Instagram, Youtube, Sina Weibo, Identi.ca or Plurk, to quote the largest, also exhibit dramatic growth.

This fast and unprecedented success has introduced several challenges for service providers as well as for their users. While the former have to face a tremendous flow of user generated content, the latter struggle to find relevant data that match their interests: they usually have to spend a long time to read all the content received, trying to filter out relevant information. Two (complementary) strategies have emerged to help the user to find relevant data that matches his interest in the huge flow of user generated content: posts filtering like in [16, 17, 11] and posts/account searches and/or recommendation like in [10, 6, 7]. Social network systems usually offer the ability to search for posts or accounts that match a set of keywords. This could be a "local" search to filter out the posts received, or a "global" search to query the whole set of existing posts/accounts. For the latter search, there exist two options: some pre-computed posts sets that correspond to the hot topics at query time, or customized searches where the query result is built according to the keywords specified by the user. However, the broad match semantics generally adopted by the searching tools is very limited. Even a ranking score based on the number of keywords is not sufficient to retrieve all posts of interest. Combined with the lack of semantics and the number of posts a day, the large number of searches performed every day also raises scalability issue. For instance in 2012, more searches were performed each month (24 billion) on Twitter than on Yahoo (9.4 billion) or on Bing (4.1 billion).

We have introduced in a previous work scalable filtering techniques and structures on server's side that are specifically designed for micro-blogging systems [9]. In this paper, we consider the problem of discovering quality content publishers by providing efficient, topological and contextual user recommendations on the top of a micro-blog social graph. Micro-blogging systems are characterized by the existence of a large directed social graph where each user (accounts) can freely decide to connect to any other user for receiving all his posts. In this paper we make the assumption that a link between a user u and a user v expresses an interest of u for one or several topics from the content published by v. We consequently choose to model the underlying social network graph as a *labeled social graph* (LSG), where labels correspond to the topics of interest of the users. Our objective is to propose a recommendation score that captures both the topological proximity and connectivity of a publisher along with his authority regarding a given topic and the interest of the intermediary users between the one to be recommended and the publisher.

[1]http://expandedramblings.com

The size of the underlying social graph raises challenging issues especially when we consider operations that involve a graph exploration. In order to speed up the recommendation process we propose a fast approximate computation based on landmarks, *i.e.* we select a set of nodes in the social graph, called *landmarks* which will play the role of hubs and store data about their neighborhood. This set of landmarks is selected using different strategies we compare experimentally in Section 4.

Contributions. In this paper we propose a recommendation system that produces personalized user recommendations. Our main contributions are:

1) considering the idea that measures based on the graph topology are good indicators for the user similarity, we propose a topological score which integrates semantic information on users and their relationships;

2) furthermore, we introduce a adapt landmark-based approach to improve recommendation computation time and to achieve a 2-3 order of magnitude gain compare to the exact computation;

3) an experimental validation of our approach, including a comparative study with other approaches (TwitterRank [30] and Katz [20]).

Observe that we illustrate our proposal in the context of micro-blogging systems, but our model is general and may be used for any social networks where users publish content and receive posts from the accounts they follow.

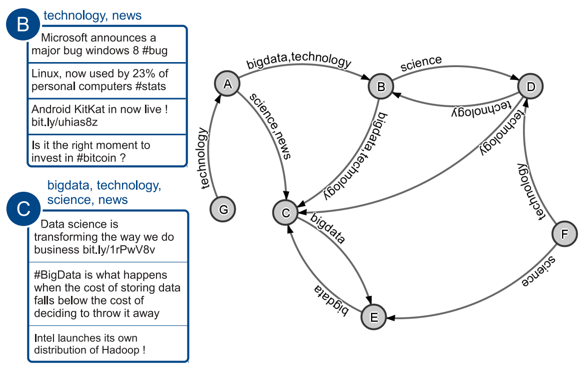
The paper is organized as follows: after the introduction in Section 1, we present our model and our recommendation scores along with their composition property in Section 2. Then we describe our fast recommendation computation based on a landmark strategy in Section 3. Our experimental validation is presented in Section 4. Section 5 presents the related work while Section 6 concludes the paper and introduces future work.

2. MODEL

We introduce in this section the underlying social graph model and our recommendation scores.

Labeled social graph.

We model the *Twitter* social network as a directed labeled graph G=$(U, E, \mathcal{T}, l_u, l_e)$ where $\mathcal{U}$ is set of vertices such that each vertex $u \in \mathcal{U}$ is a user (account). $E \subseteq \mathcal{U} \times \mathcal{U}$ is a set of edges where an edge $e = (u, v) \in E$ exists if u follows v, *i.e* u receives the publications of v. The labeling function $l_u : \mathcal{U} \rightarrow 2^{\mathcal{T}}$ maps each user u to the set of topics that characterize his posts, chosen in a topic vocabulary $\mathcal{T}$. The topics associated by the labeling function $l_e : E \rightarrow 2^{\mathcal{T}}$ to an edge $e = (u, v)$ describe the interest of the user u in the posts of v. Topic detection is a well studied problem [10] and is out of the scope of the present work. We assume here a given set of topics that can be explicitely defined by the users or inferred from theirs posts, *e.g.* as described in Section 4. An example of such graph is depicted in Figure 1 where for readability reasons each edge is labeled with a single topic. For users B and C we display their topics of interest along with an excerpt of their tweets.



Microsoft announces a major bug windows 8 #bug

Linux, now used by 23% of personal computers #stats

Android KitKat in now live ! bit.ly/uhias8z

Is it the right moment to invest in #bitcoin ?

Data science is transforming the way we do business bit.ly/1rPwV8v

#BigData is what happens when the cost of storing data falls below the cost of deciding to throw it away

Intel launches its own distribution of Hadoop !

Figure 1: A labeled social graph

Recommendation.

For a user u and a query composed of several topics $Q = \{t1, \ldots, t_n\}$, our model recommends users v based on the following criteria that consider both graph topology and content semantics: (i) *user proximity* (the recommendation score of v decreases with his distance from u); (ii) *connectivity* (v is more likely to be important for u if they are connected through many paths); (iii) *topical path relevance* of the connections between u and v with respect to Q.

Our recommendation score combines the topical relevance of paths with a topological measure which considers *all existing paths* between two nodes u and v of the graph. More precisely, the recommendation score $\sigma(u, v, t)$ of the user v for user u on topic t on paths $p = u \rightsquigarrow v$ is the sum of all path scores $\overline{\omega}_p(t)$ and is expressed as follows:

DEFINITION 1 (RECOMMENDATION SCORE).

$$\sigma(u, v, t) = \sum_{p \in P_{u,v}} \overline{\omega}_p(t) = \sum_{p \in P_{u,v}} \beta^{|p|} \omega_p(t) \tag{1}$$

where $P_{u,v}$ denotes the set of all paths between u and v and the score $\overline{\omega}_p(t)$ of a path of length $|p|$ consider both the topical relevance $\omega_p(t)$ and the distance score $\beta^{|p|}$ on p. The decay factor $\beta \in [0, 1]$ gives more importance to shorter paths.

The final recommendation score for the query Q is computed as a weighted linear combination (some are proposed in [2]) where user scores for each individual topic $t_i \in Q$ are weighted by the relevance of t_i for the posts of u which is computed by the topic extraction method (see Section 4).

The topological score obtained by ignoring the topical relevance of paths ($i.e$ setting $\omega_p(t)$ to 1) is higher if there exist many short paths between u and v. It is similar to the idea of the Katz scores [20] that have been successfully employed for link prediction. We denote this topological score as :

$$topo_{\beta}(u, v) = \sum_{p \in P_{u,v}} \beta^{|p|} \tag{2}$$

Topical path relevance

The topical relevance $\omega_p(t)$ a path $p = u \rightsquigarrow v$ for a topic t considers both the relevance of nodes (user authority) and of edges (topical similarity) on the path p $w.r.t$ to the topics of the query Q.

Edge relevance:
Each edge on p contributes to the score of the path with a semantical score which depends on its topics. Distant edges contribute less to the recommendation score than edges close to u. More precisely, the relevance of an *edge* at distance d from u on path p for a topic t is defined as :

$$\varepsilon_{edge}(t) = \alpha^{d} \times max_{t' \in l_e(edge)}(sim(t', t)) \quad (3)$$

where $l_e(edge)$ is the labeling function that returns the topics associated to *edge*. The decay factor $\alpha \in [0, 1]$ decreases the influence of an *edge* when its distance from u is higher. The function $sim : \mathcal{T}^2 \to \mathbb{R}$ computes the semantic similarity between two topics t and t'. We use in the present paper the Wu and Palmer similarity measure [31] on top of the WORDNET [1] database, but other semantic distance measures, such as RESNIK or DISCO [2] could also be used. The choice of the best similarity function is beyond the scope of the current paper. When an *edge* is labeled with several topics, we only keep the maximum similarity to t among all its topics to avoid high scores for edges labeled with many topics that have small similarity to t.
User authority:
We define a per node topical authority function $auth(u, t)$ of u on a topic t which depends on the number of users who follow u on t. The authority score is decomposed into two scores: (i) the *local authority score* that gives a higher score to users that are specialized on topic t than to users u who publish on a broad range of topics and (ii) the *global popularity score* that gives higher scores to users that are more followed on t. Combination of local and global scores has also been used to compute authorities for Web pages [15] or micro-blogging [13]. The authority score $auth(u, t)$ of a user u on a topic t is consequently defined as follows :

$$auth(u, t) = \underbrace{\frac{|\Gamma^u[t]|}{|\Gamma^u|}}_{local} \times \underbrace{\frac{\log(1 + \Gamma^u[t])}{\log(1 + max_u(\Gamma^u[t]))}}_{global}$$

where $|\Gamma^u[t]|$ is the number of followers of u on t, and $|\Gamma^u|$ is its total number of followers. We used the logarithm function to smooth the difference between popular accounts and accounts with very few followers.

The local authority is 1 when u is followed exclusively on t and the global popularity is 1 when u is the most followed user on t. If no other user follows u on t both scores are 0. The authority scores for a given topic t are high for users that are mainly followed on topic t and that have a significant number of followers. The combination of both local and global scores leads to similar authority scores for very specialized accounts with few followers and for very popular but generalist accounts. Observe for scores update that $|\Gamma^u|$ and $|\Gamma^u[t]|$ can be computed on local information of each user, without graph exploration. Oppositely the computation of $max_u(\Gamma^u[t])$ may be costly since it requires to query the complete graph. However, the log strongly limits the impact of a variation in the popularity of an account with millions of followers, and we can assume this value is stored (and re-computed periodically).

EXAMPLE 1 (LOCAL AND GLOBAL AUTHORITY). *Consider the example graph in Figure 1, with a sample of tweets for the users B and C along with their topics. User B is more relevant for technology than C. Indeed B and C have the same global popularity with two followers on this topic for both accounts. However the local authority of B on technology is higher that the one of C since 2 out of the three topics on which B is followed are technology, whereas for C only 2 out of the 6 topics on which it is followed are technology. For the topic bigdata, the local authority of B and C is the same (1 out of 3 for B and 2 out of 6 for C) but C is more followed on bigdata (2 users that follow him) than B (1 user). Therefore, the total authority of C on bigdata is higher.*

Finally, we consider that the path relevance of p is high when both the relevance of the nodes and the one of the edges of p are high:

$$\omega_p(t) = \sum_{edge \in p} \varepsilon_{edge}(t) \times auth(end(edge), t) \quad (4)$$

where $end(edge)$ returns the end node of *edge*. The recommendation score of v for the user u on topic t is then obtained by replacing $\omega_p(t)$ in equation (1) by its formula given by equation (4). The obtained user recommendation score thus captures the topology (proximity and connectivity) of the graph along with the followers interests (expressed as labeled edge) and the authority score regarding the topic of interest for each user on the path.

EXAMPLE 2 (TOPICAL PATH RELEVANCE). *In Fig. 1, we want to recommend to A users on topic technology (we suppose a search within a range $k = 2$). Users D and E can be reached with respectively the paths $p_1 = A \to B \to D$ and path $p_2 = A \to C \to E$, each of length 2. The relevance of the edge $A \xrightarrow{bigdata,technology} B$ is higher than the one of $C \xrightarrow{bigdata} E$, since the first one is at distance 1 from A, whereas the second is at distance 2. Moreover, the authority on technology of node B computed as $(local) \times (global) = \frac{2}{3} \times \frac{log(1+2)}{log(1+2)}$ is higher than the authority of C on technology, computed as $\frac{2}{6} \times \frac{log(1+2)}{log(1+2)}$. Overall, the semantic relevance of the edges on p_1 for technology is higher than the one of edges on p_2 and D obtains a higher recommendation score than E.*

Score composition:
The recommendation score given by the equation (1) allows us to deduce the score of paths p from the score of of its sub-paths already computed using the following property :

PROPOSITION 1 (RECOMMENDATION SCORE COMPOSITION). *Assume a path $p = p_1.p_2$, with $\overline{\omega}_{p_1}(t)$ and $\overline{\omega}_{p_2}(t)$ the path scores already computed of respectively p_1 and p_2 for a topic t. The recommendation score of p can be computed as:*

$$\overline{\omega}_p(t) = \beta^{|p_2|}.\overline{\omega}_{p_1} + \beta^{|p_1|}.\alpha^{|p_1|}.\overline{\omega}_{p_2}$$

PROOF. Using equations (3) and (4), the recommendation score $\overline{\omega}_p(t) = \beta^{|p|}$ of a path p with length $|p|$ can be deduced from the score of its prefix path p_1 with length $|p-1|$ and the score of the last edge E as: $\overline{\omega}_p(t) = \beta\overline{\omega}_{p'}(t) + \beta^{|p-1|}\alpha^{|p-1|}(\beta\alpha max_{t' \in l_e(E)}(sim(t', t))auth(end(E), t))$

The score $\beta\alpha.max_{t' \in l_e(E)}(sim(t', t))auth(end(E), t)$ represents the score of a path p_2 contains only E. By induction we prove the proposition for longer paths. □

[1]http://wordnet.princeton.edu/
[2]http://www.linguatools.de/disco/disco_en.html

Score computation convergence.

In order to show the convergence of the iterative computation of recommendation scores $\sigma(u, v, t)$ of users v for user u on topic t (Equation (1), we express this computation in matrix form as :

$$R^{(k+1)} = (\beta A)R^{(k)} + (\beta\alpha)S.T_{\alpha\beta}^{(k)}$$

where $R^{(k)}$ is the recommendation vector for topic t computed at step k ($R_v^{(k)}$ is the recommendation score $\sigma(u, v, t)$ computed at step (k)). Matrix A is the adjacency matrix ($a_{vu} = 1$ if u follows j). Matrix $S(t)$ is the similarity-authority matrix ($s_{vu} = sim(max_{t' \in l_e(u \to v)}(t', t)) \times auth(v, t)$). Vector $T_{\alpha\beta}^{(k)}$ is the topological vector at step k that can be expressed as follows :

$$T_{\alpha\beta}^{(k+1)} = \alpha\beta.A.T_{\beta}^{(k)} + I$$

where $I[u] = 1$ and $I[u] = 0$ for all $u \neq v$. We deduce that the computation convergence is achieved under the following condition :

PROPOSITION 2 (SCORES COMPUTATION CONVERGENCE). *If $\beta < 1/\sigma_{max}(A)$, where $\sigma_{max}(A)$ is the highest eigen value for A, then the iterative scores computation of our recommendation scores converges.*

PROOF. Based on the recursive formula which defines the topical vector for a given node n, the topical scores matrix defined by the series expansion

$$T_{\alpha\beta} = \sum_{i=1}^{\infty} \alpha\beta^i.A^i = (I - \alpha\beta.A)^{-1} - I$$

converges when $I - \alpha\beta A$ is positive definite, so $\alpha.\beta < 1/\sigma_{max}(A)$. Consider a step k' when convergence is reached for $T_{\alpha\beta}$. Then for any $k > k'$, we have the recursive computation $R^{(k+1)} = (\beta.A).R^{(k)} + C$ with $C = (\alpha\beta).S.T_{\alpha\beta}^{(\infty)}$ constant. The convergence for $R^{(k+1)}$ is reached when $R^{(\infty)} = (\beta.A).R^{(\infty)} + C$ thus when $R^{(\infty)} = (I - \beta.A)^{-1} \times C$. This can be achieve if $\beta < 1/\sigma_{max}(A)$. Since $\beta > \alpha\beta$, this later condition is sufficient to assure convergence. $\square$

3. LANDMARK-BASED COMPUTATION

The recommendation score computation presented in the previous Section assumes to explore *all* paths from the node that corresponds to the account to be recommended. In order to propose real-time recommendations, we proposed a landmark-based algorithm, with a two-step approach: (i) a preprocessing that computes for a given set of nodes, the recommendations for each topic in the topic vocabulary and (ii) an approximated query-time recommendation computation for a node on a given topic using precomputed recommendations.

3.1 Landmark-based approach overview

Computing recommendation scores by graph exploration at k hops of a graph with n nodes supposes to handle candidate sets of size out_{avg}^k for the average case (out_{avg} denotes the average out degree) and of size n^k in the worst case for a complete graph. This might be prohibitive in the context of social graphs with millions of nodes and edges and we rely on a landmark-based approach to propose fast approximate recommendations.

The computation is performed in two steps: (i) in the preprocessing step we precompute for a sample of nodes in the graph, named landmarks, top-k recommendation scores (k being a parameter of the system) for every topic $t \in \mathcal{T}$ and (ii) at runtime we compute approximate recommendations by exploring the graph until a given depth (also a parameter of the system) and collect precomputed recommendations from landmarks encountered during this exploration. Figure 2 illustrates this approach, where n is the query node and λ_1, λ_2, λ_3 and λ_4 are landmarks. When performing the BFS from n represented by the blue dashed-line, only λ_1, λ_2 and λ_4 were encountered. For the recommendations finally proposed, we see that r_1 was not encountered by the BFS exploration and its score is the aggregate score for score combinations $n - \lambda_1 - r_1$ and $n - \lambda_2 - r_1$. r_2 score results from the aggregate score combinations $n - \lambda_1 - r_1$ and $n - \lambda_2 - r_1$ with the path score at distance 2 for $n - \lambda_2$ since it is encountered during the BFS. Oppositely, recommendation r_4 was retrieved thanks to the sole landmark λ_2.

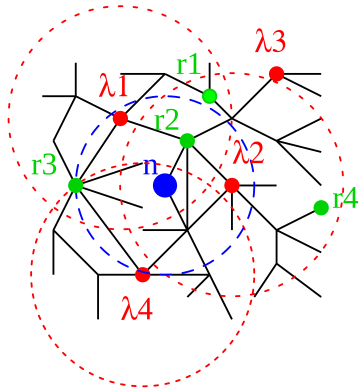


Figure 2: Example of landmark-based recommendation

3.2 Preprocessing

For the preprocessing step we consider a subset $\mathcal{L} \subset N$ of nodes, so-called the landmarks, with $|\mathcal{L}| \ll |N|$. Instead of a random sampling, several strategies may be considered to determine $\mathcal{L}$. For instance, landmark-based approaches for computing shortest paths within a large graph rely mainly on centrality properties (betweenness or closeness centrality) to determine the sampling. The publisher-follower characteristics of our graph allow other topology-based sampling, like a selection of the nodes with the most important number of followers (most popular accounts) or the ones that follow the highest number of accounts (most active readers). Additionally we can force two landmarks to respect a minimal distance between them. While the choice of the landmarks may impact the global performances of our approach we do not investigate further the sampling strategies in the current paper. Nonetheless some of these sampling techniques are experimentally compared in Section 4.

For each landmark $\lambda \in \mathcal{L}$ we perform a Breadth-First Search (BFS) but without pruning paths with cycles. We name the set of reached nodes at depth k from λ the *k-vicinity* of λ, denoted $\Upsilon_k(\lambda)$ and $\Upsilon_\infty(\lambda)$ denotes the set of reachable nodes from λ. We compute for each node

$n \in \Upsilon_\infty(\lambda)$ a recommendation vector $R_{\lambda,n}$ for each topic t with $R_{\lambda,n}[t] = \sigma(\lambda, n, t)$ (the recommendation score of the landmark λ for the account n on topic t as defined in Equation 1) and a topological score $topo_\beta(\lambda, n)$ with decay factor β (Equation 2) used to estimate the final recommendation score at runtime (see below). Then we aggregate the different $R_{\lambda,n}$ for all users $n \in \Upsilon_\infty(\lambda)$ to get a sorted list R_λ for each term t where users are sorted according to their recommendation score $\sigma(\lambda, n, t)$. Only the top-k recommendations for each topic are finally stored.

Algorithm 1 performs the recommendation computation (we remove the different initialization to simplify the presentation). The input parameters are the decay factor chosen for topical scores and the maximal depth for the BFS. This value is used for the BFS of the query nodes (see Algorithm 2), but for landmarks computation, since we compute until the convergence this value is useless and consequently can be set to a large value. For a landmark we call it with the set of topics $\mathcal{T}$. Iteration in line 2 allows to explore the k-vicinity of λ. For each iteration we add in the k-vicinity nodes that could be reached with an additional hop (l. 5). For each node v reached at this step (l. 6), we compute (or update if the node has been already encountered)) the recommendation score for each term of the topic vocabulary (l. 7-8), and the node's topological score (l. 10). The new score of v is added to the sorted topical (l.12) and topological (l. 13) lists of λ.

Algorithm 1: LANDMARK_RECOMM(k, β)

```
Require: a maximal path length $k$, topological decay factor $\beta$.
Ensure: a recommendation set of lists $R$, a connectivity vector
    $topo_\beta$
 1: $\Upsilon_0 := \lambda, \quad i := 0, converged := false$
 2: while $i < k$ and $converged = false$ do
 3:   for all $n \in \Upsilon_i$ do
 4:     $\Upsilon_{i+1} := \Upsilon_{i+1} \cup \Gamma^+(n)$
 5:     for all $m \in \Gamma^+(n)$ do
 6:       for all $t \in \mathcal{T}$ do
 7:         $\sigma^{(i+1)}(\lambda, m, t) =$
            $\sigma^{(i+1)}(\lambda, n, t) + \beta \times \sigma^{(i)}(\lambda, n, t) + \beta^{i+1}.\alpha^{i+1}.\varepsilon_{n,m}(t))$
 8:       end for
 9:       $topo_\beta^{(i+1)}(\lambda, m) = topo_\beta^{(i+1)}(\lambda, m) + \beta \times topo_\beta^{(i)}(\lambda, n)$
10:     end for
11:     $R_{\lambda,n}^{(i+1)}[t] = R_{\lambda,n}[t] + \sigma^{(i)}(\lambda, n, t)$
12:     $topo_\beta(\lambda, n)^{(i+1)} = topo_\beta(\lambda, n) + topo_\beta^{(i)}(\lambda, n)$
13:   end for
14:   if $|topo_\beta(\lambda, n)^{(i+1)} - topo_\beta(\lambda, n)^{(i)}| / topo_\beta(\lambda, n)^{(i+1)} < \nu$
      then
15:     $converged := true$
16:   end if
17:   for all $t \in \mathcal{T}$ do
18:     if $|R_{\lambda,n}^{(i+1)}[t] - R_{\lambda,n}^{(i)}[t]| / R_{\lambda,n}^{(i)}[t] < \nu$ then
19:       $converged := true$
20:     end if
21:   end for
22:   $i := i + 1$
23: end while
24: return for all
    $t \in \mathcal{T}$ top_recommendations$(R_{\lambda,n}[t]), topo_\beta(\lambda, n))$
```

3.3 Fast Approximate recommendation

We now present our algorithm which computes fast approximate recommendation scores based on the pre-computations performed for each landmark. We assume that we want to recommend to an account u other accounts for a topic t.

We first perform from u a Breadth-First Search similar to the one used for the pre-computing (see Section 3.2), for a given maximal depth k. The objective of the BFS is triple: (i) we explore the *k-vicinity* of u to discover the closest landmarks, (ii) we compute the path scores for the topic t for paths from u to each encountered landmark and (iii) we update the recommendation scores stored by the landmarks considering the paths from u to the landmarks.

More precisely, we denote $\Lambda \subseteq \mathcal{L}$ the set of landmarks encountered by the BFS exploration and R_λ the recommendation top-k lists associated to each $\lambda \in \Lambda$. R_λ contains for each topic t a list of recommended accounts v along with their recommendation scores $R_{\lambda,v}[t] = \sigma(\lambda, v, t)$ and their topological score $topo_\beta(\lambda, v)$. The approximate recommendation of a node v for a user u is an aggregation of the scores of v computed by all the landmarks $\lambda \in \Lambda$:

DEFINITION 2 (APPROXIMATE RECOMMENDATION SCORE). *The approximate recommendation score of a node v for a node u on the topic t with respect to the set of landmarks Λ is defined as :*

$$\widetilde{\sigma}(u, \Lambda, v, t) = \sum_{\lambda \in \Lambda} \widetilde{\sigma}(u, \lambda, v, t)$$

where the score $\widetilde{\sigma}(u, \lambda, v, t)$ is the score of v that takes into consideration the set of paths $P_{u,\lambda,v}$ from u to v that pass through the landmark λ.

The score $\widetilde{\sigma}(u, \lambda, v, t)$ is computed by the composition of the scores $\sigma(u, \lambda, t)$ and $topo_{\beta\alpha}(u, \lambda)$ obtained during the exploration phase with the scores $\sigma(\lambda, v, t)$ and $topo_\beta(\lambda, v)$ that are stored in the sorted lists of λ.

PROPOSITION 3 (APPROXIMATE SCORE COMPUTATION). *The recommendation score of v for u with respect to the landmark λ can be computed as follows :*

$$\widetilde{\sigma}(u, \lambda, v, t) = \sigma(u, \lambda, t) \times topo_\beta(\lambda, v) + topo_{\beta\alpha}(u, \lambda) \times \sigma(\lambda, v, t)$$

PROOF. Any path p from $P_{u,\lambda,v}$ could be decomposed into p_1 and p_2, with $p_1 \in P_{u,\lambda}$ and $p_2 \in P_{\lambda,v}$. Obviously any path $p = p_1.p_2$ with $p_1 \in P_{u,\lambda}$ and $p_2 \in P_{\lambda,v}$ is a path from $P_{u,\lambda,v}$. Consequently, based on Proposition 1) we have:

$$\widetilde{\sigma}(u, \lambda, v, t) = \sum_{p \in P_{u,\lambda,v}} \overline{\omega_p}(t)$$

$$= \sum_{p_1 \in P_{u,\lambda}} \sum_{p_2 \in P_{\lambda,v}} \beta^{|p_2|}.\overline{\omega}_{p_1}(t) + \beta^{|p_1|}.\alpha^{|p_1|}.\overline{\omega}_{p_2}(t)$$

$$= \sum_{p_1 \in P_{u,\lambda}} \overline{\omega}_{p_1}(t). \sum_{p_2 \in P_{\lambda,v}} \beta^{|p_2|} + \sum_{p_1 \in P_{u,\lambda}} (\beta.\alpha)^{|p_1|}. \sum_{p_2 \in P_{\lambda,v}} \overline{\omega}_{p_2}(t)$$

$$= \sigma(u, \lambda, t).topo_\beta(\lambda, v) + topo_{\beta.\alpha}(u, \lambda).\sigma(\lambda, v, t)$$

□

Note that our approach estimates a lower-bound of the recommendation scores while landmark-based approaches traditionally proposed for shortest paths computation provide score upper-bounds, based on the triangular inequality. Indeed in our setting the approximate scores do not consider all the paths from u to v, but only the subset $P_{u,\lambda,v}$ that pass through λ.

Consequently we perform our approximate recommendation for a node n and a topic t thanks to Algorithm 2. We use the LANDMARK_RECOMM algorithm to compute recommendation scores from n to all nodes within a distance k along with their topological score (l. 1). Observe that in reality, unlike LANDMARK_RECOMM algorithm, we compute recommendations only for a single topic. Note also that the decay factor is here $\beta.\alpha$. For all landmarks encountered (l. 2-3) we combine its recommendation for the topic t with the recommendation score computed from n to the landmark according to Proposition 3 (l. 5).

Algorithm 2: APPROX_RECOMM(n, k, t, β, α)

```
Require: a node $n$, a max. depth $k$ for BFS, a topic $t$, the
    decay factor for path $\beta$ and for edge $\alpha$.
Ensure: an ordered list of recommendations $\widetilde{R_n}$ for $n$
1: $(R_n, topo_{\beta.\alpha}(n)) \leftarrow$ LANDMARK_RECOMM$(G, n, k, t, \beta.\alpha)$
2: for all $m \in R_n$ do
3:     if $m \in \mathcal{L}$ then
4:         for all $o \in R_m[t]$ do
5:             $\widetilde{R_n} := Merge(\widetilde{R_n}, (o, \sigma(n,m,t).topo_\beta(m,o) +$
               $topo_{\beta.\alpha}(n,m).\sigma(m,o,t)))$
6:         end for
7:     end if
8: end for
9: return $\widetilde{R_n}$
```

4. EXPERIMENTS

In this section we present the experiments we have conducted on a real *Twitter* and *DBLP* datasets to validate our structures and algorithms.

4.1 The dataset

The Twitter dataset we use in our experiments contains around 2.2 million of users (with their 2.3 billion associated tweets acquired in 2015, between February and April) linked by more than 125 million edges (following relationships). Table 1 describes the main topological properties on the generated dataset. Observe that the values of topological properties for our *Twitter* dataset are very close to the values for the real *Twitter* network observed in [22].

Property	Twitter	DBLP
Total number of nodes	2,182,867	525,567
Total number of edges	125,451,980	20,526,843
Avg. out-degree	57.8	47.3
Avg. in-degree	69.4	53.6
max in-degree	348,595	9,897
max out-degree	185,401	5,052

Table 1: Datasets topological properties

In order to generate the labels on the edges, we first used OpenCalais[3] document categorization to tag a sub-set of the nodes in our graph with a corresponding topic based on the tweets he published. This strategy allowed to tag 2 percents of our nodes using a list of 18 standard topics for Web sites/documents proposed by OpenCalais. To complete the categorization, we used a trained Support Vector Multi-Label Model using Mulan[4], with a precision of 0.90, to tag all our nodes. The result was a list containing each user in our graph along with his publisher profile (topics on which he publishes). We determine the follower profile as the tf of the topics of the publishers we follow. Finally we select as labels for an edge, the topics in the intersection between a follower and a publisher profile with the scores beyond a threshold in the follower profile. The result was a fully labeled social graph of 2.2M nodes and 125M edges. We refer to this dataset as *Twitter*. Our generation for edge labels results in the distribution depicted Figure 3 for the 18 topics. This biased distribution is similar to the one observed for Web sites in Yahoo! Directory [21].

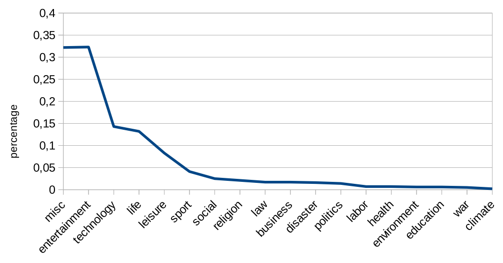


Figure 3: Distribution of edges per topic

For the DBLP dataset, we merged different versions of the ArnetMiner DBLP datasets[5]. The resulting dataset contains 2,291,100 papers, 1,274,860 authors and 4,191,643 citations. From this dataset we build a graph of author citations by creating a directed edge between author A and author B whenever it exists a citation of a paper from A to a paper from B. It results in a final dataset which contains 525,567 authors and 20,526,843 citations between authors. Observe that we kept only authors who are cited. The last step to be able to test our model and TwitterRank on this dataset was to label the authors, and the relationships between them. We used the Singapore Classification[6] to label some of the major conferences and assume a paper published in a conference is about the main topic of this conference. We then spread the labels on a conference graph with the intuition that the more authors have published in two conferences, the more their topics are close. The matrix of correlation between the topics was constructing by applying a factorial analysis. Our resulting dataset is summarized in Table 1.

4.2 Implementation

We implemented our solution in Java (JVM version 1.7). The experiments were run on a 64-bit server with a 10 cores Intel Xeon processor at 2.20GHz and 128GB of memory. The server is running Linux with a 3.11.10 kernel. We implemented our indexes as inverted lists. Given the JVM overhead, we made the design choice to rely on basic data structures (32 bits integers, arrays) in order to achieve better scaling results in term of size and execution times. We applied the same design choices for TwitterRank and Katz. For efficiency reasons, we pre-computed the Wu and Palmer semantic similarity scores using Wordnet 3.0 in a similarity

[3]http://www.opencalais.com/

[4]http://mulan.sourceforge.net/

[5]http://aminer.org/billboard/citation

[6]http://www.ntu.edu.sg/home/assourav/crank.htm

matrix, we keep in memory. We stored the triangular similarity matrix as an array with an access pattern in order optimize the storage costs to be able to handle larger numbers of topics. While we consider here only the 18 common topics for Web documents, which results in a 2.5 KB file, observe that for 10,000 topics the similarity matrix will require around 750MB so can still easily fit in memory. A similar approach was chosen for the similarity matrix for DBLP. We store the landmark recommendations as inverted lists: for each landmark, we have a set of accounts recommended along with their recommendation score for each topic from $\mathcal{T}$. Landmarks are chosen according to one of the selection strategies shown in Table 3.

4.3 Quality of the recommendation

We adopt the methodology introduced in [8] to estimate the quality of the recommendation algorithms:

i) we build a *trial* of T accounts by removing one edge from the graph according to a given edge-selection strategy;

ii) for each account from the trial, we randomly select 1000 accounts in the graph;

iii) we compute recommendation for the 1001 accounts and we form a ranked-list L;

iv) when the account from the trial belongs to the top-N from L we have a hit, otherwise a miss;

v) we iterate on the following account from the trial.

The overall recall and precision are defined similarly to [8] with $\#hits/T$ and $\#hits/N.T$ respectively. For our experiments we set the trial size $T = 100$ and we average values over 10 trials. We compare the quality of our recommendations with two competitors: the standard Katz score [20], which considers only the topology, and TwitterRank [30] which captures both the link structure and the topical similarity between users.

Unless explicitely specified we choose as β for both Katz and Tr 0.0005, reported in [20], as a good β setting for web graphs, 0.85 for our α and 0.85 for β in TwitterRank similarly to [30]. We report results obtained at convergence.

Figure 4 illustrates the accuracy of the three recommendation strategies for randomly selected edges when building the trials for the Twitter dataset. We see that TwitterRank is outperformed by other algorithms. While only for 4% of the recommendations the account corresponding to the removed edge is found in the top-1 for TwitterRank, Katz provides as first recommendation the correct account in 29% of the tests and Tr in 34%. So Tr provides a 8.5 and 1.2 gain with TwitterRank and Katz respectively for the top-1. For the top-10 the improvement remains significant: 3.8 and 1.3 with TwitterRank and Katz respectively. Figure 5 confirms that Tr outperforms other approaches: for a similar recall value greater than 0.4, the precision of Tr is at least twice the one of Katz and one order of magnitude higher than the one of TwitterRank. For instance for a 0.5 recall value, Tr presents a precision of 0.18 while precision reaches 0.06 and 0.01 for Katz and TwitterRank respectively for the same recall value. We observe in Figures 6 and 7 that the DBLP dataset exhibits similar results. The recall reaches however faster 0.8 for Tr due to the self-citations phenomenon: authors from a given paper often cite one or several of their previous papers on the topic. These papers may share some citations with the paper corresponding to the edge removed for the selected author. This also explains the faster recall increase for Katz strategy. TwitterRank whose recommendations are essentially based on the popularity (in-degree) of an account reached in the graph does not capture this phenomenon and provides slightly worst results than with the Twitter dataset.

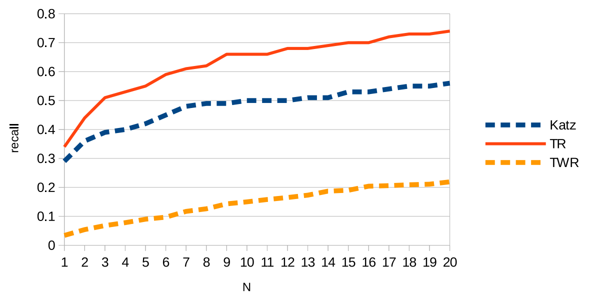


Figure 4: Recall at N (Twitter)

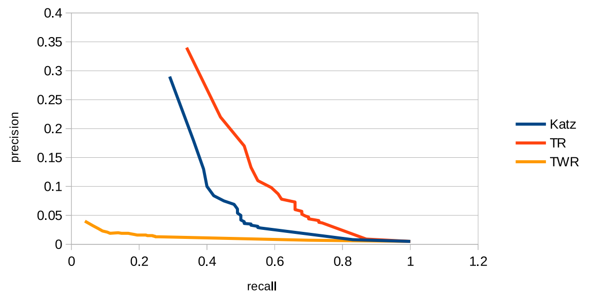


Figure 5: Precision vs recall (Twitter)

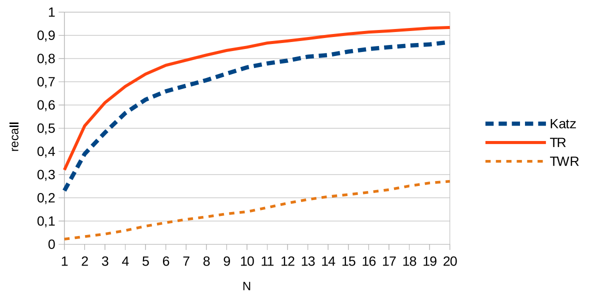


Figure 6: Recall at N (DBLP)

However there exists a large discrepancy for accuracy when considering two dimensions of analysis: the edge removal strategy and the popularity of the topic used for the recommendation. When considering the top-10, Figure 8 shows that for Twitter we have a very low accuracy, *i.e.* a recall of 0.15, 0.03 and 0.18 for respectively Katz, TwitterRank and Tr, when trying to retrieve an account which belongs to the top-10% less followed accounts (TW min). Conversely very popular accounts, top-10% most followed

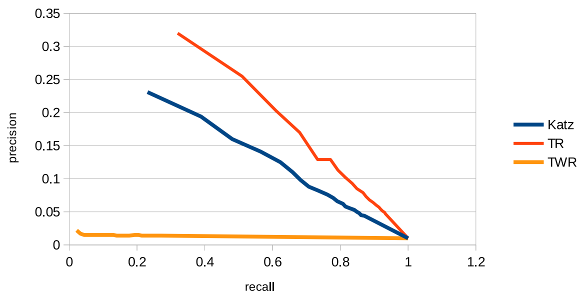


Figure 7: Precision vs recall (DBLP)

accounts, are most of time retrieved in the top-10 recommendations with a recall between 0.9 and 0.95 for all strategies. This can be explained by their path-based approach which aggregate scores on incoming paths, so high scores for accounts with numerous incoming paths. Observe that for popular accounts, TwitterRank provides the best results. Indeed most of large accounts are labeled with several topics. While TwitterRank score relies on the account popularity and on the presence or not of a label for an account, whatever the number of labels it has, Tʀ score consider for its authority score the number of incoming edges labeled with a given topic. But the more labels an account has, the lower authority score for a given topic it may have. Oppositely an account with an low in-degree rarely has several labels. Our approach that also consider semantic similarity between topics on edges is then particularly efficient. With the DBLP dataset, authors who belong to the 10% less cited are more likely retrieved than with the Twitter dataset for Katz and Tʀ due to the higher density of the graph. However TwitterRank based on the popularity fails to retrieve these authors. Even for the 10% most popular authors, TwitterRank does not achieve the good results got for the Twitter datasets, due to a different distribution of the in-degree. While the 10% most followed in Twitter include few extremely popular accounts and some moderately popular, the 10% most followed in DBLP consist in a more uniform dataset regarding the in-degree.

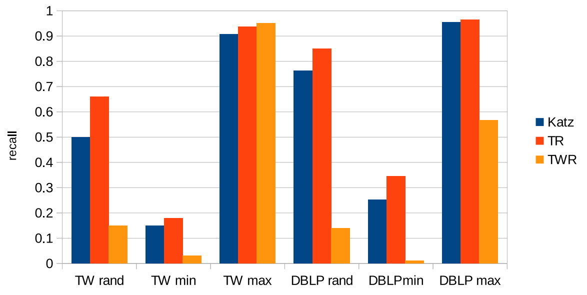


Figure 8: Recall w.r.t. popularity

Since the distribution of edge topics is very biased we also study the impact of the popularity of the topic on the recommendations. Results are depicted in Figure 9 for topics `social`, `leisure` and `technology`. Two main conclusions are underlined with this experiment. First, the less popular an account is, the better accuracy for our recommendations we get. So for an infrequent topic like `social` we get a recall-at-10 for Tʀ, Katz and TwitterRank of respectively 0.959, 0.751 and 0.253. Oppositely for the popular topic `technology` we get respectively 0.462, 0.424 and 0.09. Indeed for a popular topic many accounts may be found in a close and connected neighborhood of the account we want to recommend, possibly with a higher score than for the account formerly linked by the removed edge. Second we observe that Tʀ which considers the semantic similarity between topics always outperforms other strategies.

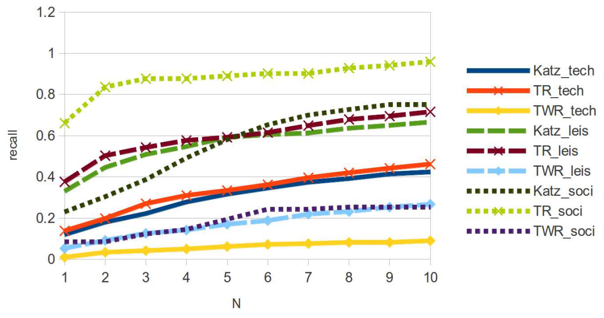
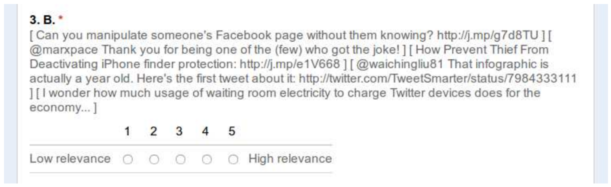


Figure 9: Recall w.r.t. topic popularity

Observe however that this experiment does not highlight the quality of the recommendations performed by each algorithm but only the ability to retrieve a removed link between two followers (link prediction). To estimate the quality of the recommendations we rely on a user validation.

User Validation Task.

In order to evaluate the relevance of the generated recommendations, we conducted a user validation task on 54 IT users (undergraduate, postgraduate and PhD students, and academics) from which 46% are regular *Twitter* users. We set up an on-line blind test where we ask users to rate the relevance of a set of recommendations for a given topic on a scale from 1 (low relevance) to 5 (high relevance). A recommendation set consists in the top-3 recommendations given by Katz, Tʀ and TwitterRank, so 9 recommended accounts for topics `Technology`, `Social` and `Leisure`. On the interface the recommendation list is shuffled, and for each recommendation we display a sample of 5 randomly chosen tweets from the corresponding account. Figure 10 displays an excerpt of the interface of our rating system.


3. B. *

[Can you manipulate someone's Facebook page without them knowing? http://j.mp/g7d8TU] [@marxpace Thank you for being one of the (few) who got the joke!] [How Prevent Thief From Deactivating iPhone finder protection: http://j.mp/e1V668] [@waichingliu81 That infographic is actually a year old. Here's the first tweet about it: http://twitter.com/TweetSmarter/status/7984333111] [I wonder how much usage of waiting room electricity to charge Twitter devices does for the economy...]

	1	2	3	4	5	
Low relevance	○	○	○	○	○	High relevance

Figure 10: Interface of the user rating system for the topic `Technology`

Figure 11 presents the results of our user validation. Note that we observe that the user during the validation usually mark with the average 2 or 3 value all accounts when he was doubtful about the relevance or not of an account what happened usually when tweets were neutral, unclear, or when they required some knowledges about a given topic (*e.g.* few European people know who is Tom Brady so can not assert

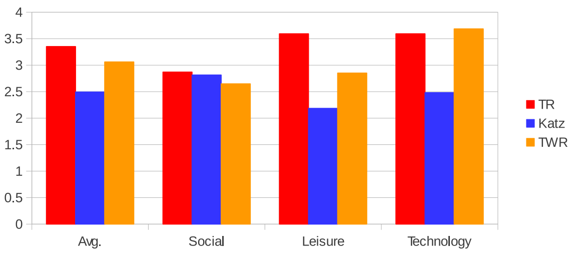


Figure 11: Relevance scores (user validation Twitter)

this tweet is about the `leisure` topic). So scores greater than value 3 are significant since they means users really observe the relevance of an account.

From this experiment we conclude that on average TR and TwitterRank provide more relevant recommendations according to the topic searched. However according to the popularity of the topics we have very different results. The `social` topic gave mor homogeneous results with a score between 2.7 for TwitterRank, 2.8 for Katz and 2.9 for TR. The reason for this result is that posts published by these accounts are generally difficult to classify since they mix social and health, or social and politics for instance where topics like `leisure` or `technology` are less ambiguous. For these topics, we see that TR and TwitterRank outperform Katz, which was expected since these two approaches consider the content published for their recommendation scores, unlike Katz. While TwitterRank generally recommends accounts with a large number of followers, TR can also recommend smaller account but more-specialized, which results in a better relevance score for topic with a medium popularity like `leisure`, when TwitterRank is slightly better for the most popular topic `technology`.

We also conduct a user validation for the DBLP dataset. We build a list with the top-3 recommendations returned by each method for researchers from our lab. Observe they belong to different areas (IR, DB, OR, network, software engineering, etc). To illustrate how the different methods may help to discover relevant authors we limit to 100 the number of citations of the authors returned by each algorithm (so we avoid to propose very popular and obvious authors). We propose to each researcher the randomized list with the 9 authors retrieved based on his DBLP entry. He marks each proposal between 1 and 5 according to the relevance of the proposal (*i.e.* the proposed author could have been cited regarding the past publications done by the researcher). We collect 47 answers and results are presented in Table 2. The first row shows that both Katz and TR outperform TwitterRank on this dataset. The second row shows that around a third of the recommendations proposed by Katz and TR are considered particularly relevant by our panel (4 or 5-mark)) while only 8% are so-considered with TwitterRank. A first rationale is that papers we cite, including papers from co-authors, often cite the same relevant articles within a topically-closed research community. The importance of the semantics on edge is less than with Twitter since researchers, whatever the number of articles they published, cite/are cited by mainly researchers from their community. This explains the close score for Katz and TR. The important role of the popularity in TwitterRank explains it poor results in this context since it proposed popular authors even when there exists a small number of paths between the query author and them. Last row confirms the quality of our recommendations since TR presents for 50% of the tests the best top-3 recommendations, when Katz and TwitterRank achieve respectively 38% and 12%.

	Katz	TR	TWR
average mark	2.38	2.47	1.51
# 4 and 5-mark	46	47	11
best answer (%)	0.38	0.50	0.12

Table 2: User validation (DBLP)

4.4 Approximate computations

We perform a set of experiments to illustrate the benefits of the landmark-based approximate computations. Since results may be highly related to the choice for the landmarks selection, as underlined in [25], we decided to implement and compare recommendations based on 11 different landmark selection strategies presented in Table 3.

Algorithm	Description
RANDOM	Draw landmarks with a uniform distribution
FOLLOW	Draw landmarks with a probability depending on their # of followers
PUBLISH	Draw landmarks with a probability depending on their # of publishers
IN-DEG	Landmarks are nodes with highest in-degree
BTW-FOL	Draw landmarks among nodes with # of followers in [min_follow,max_follow]
OUT-DEG	Landmarks are nodes with highest out-degree
BTW-PUB	Draw landmarks among nodes with # of publishers in [min_publis,max_publish]
CENTRAL	Select landmarks that are reachable at a given distance from most of chosen seed nodes
OUT-CEN	Select the landmarks based on the number of different output seeds that they cover
COMBINE	Weighted combination between the CENTRAL and OUT-CEN
COMBINE2	Weighted combination between the BTW-FOL and BTW-PUB

Table 3: Landmarks selection algorithms proposed

size and building time of the landmark index.

A first experiment highlights the important time discrepancy for the landmarks selection (see Table 4). Obviously random selections of the landmarks like RANDOM, BTW-FOL and BTW-PUB are the fastest strategies (around $2ms$ per landmark), while strategies based on the centrality property are 5 orders of magnitude slower (around $17h$) due to $O(N^2.\log N + NE)$ centrality complexity (with Johnson's algorithm). Table 4 also illustrates that the recommendation computation for a given landmark is almost independent of the landmarks selection strategy (between 12 and 15 mns), which means that convergence is achieved in a similar number of steps after exploring a similar number of paths.

Comparison of the landmark selection strategies for recommendations.

We evaluate our approximate approach presented in Section 3. We perform a BFS at depth 2 from a query node and combine scores with the ones of the landmarks encountered. (see Algorithm 2). Then we compare the recommendations

Strategy	landmarks	
	select. (ms)	comput. (s)
Random	2.4	756.7
Follow	3,712.8	877.3
Publish	3,614.7	868.6
In-Deg	459.6	854.3
Btw-Fol	2.4	735.1
Out-Deg	1,815.7	918.6
Btw-Pub	1.7	822.7
Central	61,060.2	807.8
Out-Cen	66,862,3	816.5
Combine	130,461.8	818.2
Combine2	2.45	805.6

Table 4: Determining landmark w.r.t. strategies

retrieved with the ones provided by the exact computation with convergence. Average results for 100 landmarks are reported in Table 5.

Strategy	#lnd	time in s (gain)	L10	L100	L1000
Random	2.9	0.93 (338)	0.130	0.124	0.125
Follow	17.5	0.83 (379)	0.377	0.140	0.096
Publish	11.7	0.58 (539)	0.349	0.136	0.100
In-Deg	58.9	0.84 (373)	0.523	0.149	0.066
Btw-Fol	3.5	0.55 (577)	0.061	0.059	0.058
Out-Deg	6.2	0.81 (388)	0.518	0.147	0.064
Btw-Pub	2.9	0.54 (585)	0.129	0.127	0.123
Central	5.3	0.76 (414)	134	123	125
Out-Cen	4.4	0.74 (425)	0.172	0.131	0.121
Combine	4.2	0.71 (443)	0.180	0.125	0.118
Combine2	3.7	0.54 (584)	0.129	0.126	0.124

Table 5: Comparison of the landmark selection strategies

First we observe that the number of landmarks encountered during the BFS at distance 2 differs from one strategy to another and ranges from 2.9 on average for the Random strategy to 58.9 for In-Deg. Centrality approaches lead to less landmarks encountered since they select landmarks among nodes which connect connected subgraphs and a two-hops BFS is more unlikely to visit several connected subgraphs. We notice that the processing time is not connected to the number of landmarks found, what seems counterintuitive since more landmarks means more computations (score combinations) to perform. The rationale is that we perform pruning when we encounter a landmark during the BFS, to avoid considering twice paths from the BFS which pass through a landmark. Since the recommendation computation is dominated by the the BFS exploration and computation, this pruning largely reduces the whole processing time. Notice that our approximate computation allows to get a 2-3 order of magnitude gain compared to the exact computation. Finally observe that a strategy that allows to find more landmarks is more tolerant to a landmark departure or one with outdated recommendation values.

Last 3 columns of Table 5 show the average Kendall Tau distance between the approximate computation and the exact computation obtained at convergence when a landmark stores respectively the top-10, top-100 or top-1000 recommendations for all topics. Keeping 1000 recommendations at landmarks nodes allows to reach a Kendall Tau distance between 0.06 and 0.13 for the top-100 recommended accounts. Keeping a top-10 at landmarks leads to a higher Kendall Tau distance since a landmark may update at most 10 scores from the top-10 built at distance 2 from the BFS. A landmark with a large out-degree (so as observed above such landmark has also often a high in-degree) is able to contact much more accounts and its recommendation lists may largely differ from the one of another such landmark keeping only a top-10 at each landmark. Remark that even when storing the top-1000 for each topic, the landmarks recommendations can easily fit in memory since they require 1.4MB storage each.

5. RELATED WORK

Motivated by the increasing interest of the researchers community, several works were presented to characterize social network systems like [18, 1, 23, 4, 3, 27]. In [18] the authors present a topological study on a very large dataset of real *Twitter* users. They show that ranking users according to their PageRank or to their in-degree provides similar results. They also propose an approach based on the retweet behavior to measure the influence of a user. In addition to retweet behavior, [4] compares two other methods to compute user influence on micro-blogging networks which rely on in-degree and @mentions. They observe that considering only the in-degree is a good indicator of popularity but it fails to capture the user influence. They also show that a user increases his popularity when he publishes on a small number (or even a single) of topics. [23] introduces a topic authority score which allow to determine the top-authors for each topic in the whole social graph (topic authorities). Observe this score is not personalized and does not consider any topological proximity to determine local authorities. We rely on these works to understand the users behavior and propose an authority score that captures most of social network systems characteristics they present.

Based on these characteristics, recommendation systems for social networks were recently proposed like [20, 6, 30, 14, 10, 24, 13, 5, 32, 27]. [20] presents a comparison of different topological-based recommendation methods adapted in the context of link-prediction. In [3], authors propose to combine two ranking scores estimated with a fair bets approach on the user invitation graph and the profile browsing graph. [27] presents a user recommendation system which exploits a node similarity estimated as a combination of a local and a global score. Local score is based on the number of neighbors of the two nodes while global involved the shortest path between them. However the recommendation scores for these approaches are only based on the topology and unlike our proposal do not consider neither content nor authority of the users.

Another approach consists in considering collaborative filtering like [6, 14]. Chen et al. [6] introduce a collaborative tweet ranking based on preference information of the users, authority of the publisher and the quality of the content. They produce recommendations at a tweet level. [14] evaluate a set of of profiling strategies to provide user recommendations based on content, *e.g.* the tweets of user or the tweets of his followers, or collaborative filtering. [24, 10] also provide tweet recommendations. [24] analyze the tweets content as well as the content of the user's direct neighbors while [10] uses the user past interaction to compute rankings in real-time. All these works consider content but unlike our proposal they do not consider longer paths than direct follower/followee links.

Few papers combine content and topology of the social graph like we do. [30] presents an extension of the PageRank algorithm named TwitterRank which captures both the link structure and the topical similarity between users. However this similarity is based on the "topics" provided by LDA and their distance-based similarity computation between users does not capture the semantics similarity between topics. We also propose a authority score for an account which estimates the local and global influence of this account for a given topic. Our scoring function also provides higher weight for short paths to favor "local" recommendations. In [13], authors describe the production recommender system implemented in *Twitter*. It relies on a adaptation of the SALSA algorithm [19] which provides user recommendation in a centralized environment based on a bipartite graph with on the one hand the user's circle of trust, computed with random walks from the user considering content properties, and on the other hand the accounts followed by the users from the circle of trust, the "authorities". Our approach is different since it captures the users interest through the *LSG*, which allows us to compute scores handling semantics, authority and topology.

To scale and accelerate our recommendations, we pre-compute scores for a subset of nodes named landmarks. Landmark-based approach is a well-known *divide-and-conquer* strategy for the shortest paths problem that have been shown to scale up to very large graphs [28, 29, 12, 25, 26]. The idea is to select a set L of nodes as landmarks which store the distance to other nodes. The distance $d(u, v)$ between two nodes u and v is then estimated by computing the minimum $d(u, l)+d(l, v)$, where $l \in L$ [28]. [26] chose to pre-compute the time-consuming shortest-path operations for each node in structures called "sketches" and use them to provide shortest-path estimations at query-time which enables scaling for large web graphs. [12] have extended the sketch-based algorithm proposed by [26] used to answer distance queries to also retrieve the shortest path themselves which allows to improve the overall accuracy. [29] propose to use shortest-path trees to achieve an efficient and accurate estimation which support updates. They also introduce a landmark selection strategy that attempt to maximize the shortest-paths coverage. In [25], authors also investigated the impact of landmark selection on the accuracy of distance estimations. They proved that optimizing the landmark coverage is a NP-hard problem and showed experimentally that clever landmark selection strategies yield to better result. Those works have inspired our landmark-based approach for scaling as well as some of our landmark selection strategies in the context of user recommendation.

6. CONCLUSION

We present the Tr recommendation score which integrates topology and semantic information regarding the interests of the users. To face prohibitive computations with very large graphs, we propose a landmark-based approach which requires a pre-computation step for a small set of identified nodes and achieves a 2-3 order of magnitude gain compare to the exact computation. The experiments and user validation show that Tr outperforms other algorithms.

As future work we intend to study updating strategies. Indeed in a social network, many following links have a short lifespan, for instance when users want to cover a particular event. This graph dynamicity may impact the scores stored by the landmarks and we need a score refreshing strategy.

We have made the choice to handle the recommendation task in a fully centralized manner. This decision was motivated by the fact that for graph data manipulation the cost of distribution can quickly overcome the gain the distributed environment introduce. For example, the production system which handle the recommendation task for the whole *Twitter Who-to-Follow* service is hosted on a single server (see [13] for a detailed discussion about the distribution of the recommendation task). However, with the continuous increase in size of the social graphs, distribution strategies must be considered in the future. Regarding our approach, distribution implies to split the graph by taking into account connectivity, but also to perform clever landmark selections and distributions that allows a server/worker/node to evaluate the recommendation scores "locally" minimizing network transfer costs.

7. REFERENCES

[1] P. Achananuparp, E.-P. Lim, J. Jiang, and T.-A. Hoang. Who is Retweeting the Tweeters? Modeling, Originating, and Promoting Behaviors in the Twitter Network. *TMIS*, 3(3):13, 2012.

[2] J. A. Aslam and M. Montague. Models for metasearch. In *SIGIR*, pages 276–284, 2001.

[3] S. Budalakoti and R. Bekkerman. Bimodal Invitation-Navigation Fair Bets Model for Authority Identification in a Social Network. In *WWW*, pages 709–718, 2012.

[4] M. Cha, H. Haddadi, F. Benevenuto, and P. K. Gummadi. Measuring User Influence in Twitter: The Million Follower Fallacy. In *ICWSM*, 2010.

[5] V. Chaoji, S. Ranu, R. Rastogi, and R. Bhatt. Recommendations to Boost Content Spread in Social Networks. In *WWW*, pages 529–538, 2012.

[6] K. Chen, T. Chen, G. Zheng, O. Jin, E. Yao, and Y. Yu. Collaborative personalized tweet recommendation. In *SIGIR*, pages 661–670, 2012.

[7] A. Chin, B. Xu, and H. Wang. Who Should I Add as a 'Friend'?: a Study of Friend Recommendations Using Proximity and Homophily. In *MSM*, page 7, 2013.

[8] P. Cremonesi, Y. Koren, and R. Turrin. Performance of Recommender Algorithms on Top-n Recommendation Tasks. In *RECSYS*, pages 39–46, 2010.

[9] R. Dahimene, C. du Mouza, and M. Scholl. Efficient Filtering in Micro-blogging Systems: We Won't Get Flooded Again. In *SSDBM*, pages 168–176, 2012.

[10] E. Diaz-Aviles, L. Drumond, Z. Gantner, L. Schmidt-Thieme, and W. Nejdl. What is happening right now ... that interests me?: online topic discovery and recommendation in twitter. In *CIKM*, pages 1592–1596, 2012.

[11] S. G. Esparza, M. P. O'Mahony, and B. Smyth. Towards the Profiling of Twitter Users for Topic-Based Filtering. In *SGAI*, pages 273–286, 2012.

[12] A. Gubichev, S. J. Bedathur, S. Seufert, and G. Weikum. Fast and accurate estimation of shortest paths in large graphs. In *CIKM*, pages 499–508, 2010.

[13] P. Gupta, A. Goel, J. Lin, A. Sharma, D. Wang, and R. Zadeh. WTF: the Who to Follow Service at Twitter. In *WWW*, pages 505–514, 2013.

[14] J. Hannon, M. Bennett, and B. Smyth. Recommending twitter users to follow using content and collaborative filtering approaches. In *RECSYS*, pages 199–206, 2010.

[15] G. Jeh and J. Widom. Scaling Personalized Web Search. In *WWW*, pages 271–279, 2003.

[16] P. Kapanipathi, F. Orlandi, A. P. Sheth, and A. Passant. Personalized Filtering of the Twitter Stream. In *SPIM*, pages 6–13, 2011.

[17] K. Koroleva and A. B. Röhler. Reducing Information Overload: Design and Evaluation of Filtering & Ranking Algorithms for Social Networking Sites. In *ECIS*, page 12, 2012.

[18] H. Kwak, C. Lee, H. Park, and S. B. Moon. What Is Twitter, a Social Network or a News Media? In *WWW*, pages 591–600, 2010.

[19] R. Lempel and S. Moran. Salsa: The stochastic approach for link-structure analysis. *ACM Transactions on Information Systems*, 19(2):131–160, 2001.

[20] D. Liben-Nowell and J. Kleinberg. The Link Prediction Problem for Social Networks. In *CIKM*, pages 556–559, 2003.

[21] T.-Y. Liu, Y. Yang, H. Wan, H.-J. Zeng, Z. Chen, and W.-Y. Ma. Support Vector Machines Classification with a Very Large-Scale Taxonomy. *SIGKDD Explorations*, 7(1):36–43, 2005.

[22] S. A. Myers, A. Sharma, P. Gupta, and J. Lin. Information network or social network?: The structure of the twitter follow graph. In *WWW Companion Volume*, pages 493–498, 2014.

[23] A. Pal and S. Counts. Identifying Topical Authorities in Microblogs. In *WSDM*, pages 45–54, 2011.

[24] M. Pennacchiotti, F. Silvestri, H. Vahabi, and R. Venturini. Making your interests follow you on twitter. In *CIKM*, pages 165–174, 2012.

[25] M. Potamias, F. Bonchi, C. Castillo, and A. Gionis. Fast shortest path distance estimation in large networks. In *CIKM*, pages 867–876, 2009.

[26] A. D. Sarma, S. Gollapudi, M. Najork, and R. Panigrahy. A sketch-based distance oracle for web-scale graphs. In *WSDM*, pages 401–410, 2010.

[27] P. Symeonidis and E. Tiakas. Transitive Node Similarity: Predicting and Recommending Links in Signed Social Networks. *WWWJ*, 17(4):743–776, 2014.

[28] M. Thorup and U. Zwick. Approximate distance oracles. In *STOC*, pages 183–192, 2001.

[29] K. Tretyakov, A. Armas-Cervantes, L. García-Bañuelos, J. Vilo, and M. Dumas. Fast fully dynamic landmark-based estimation of shortest path distances in very large graphs. In *CIKM*, pages 1785–1794, 2011.

[30] J. Weng, E.-P. Lim, J. Jiang, and Q. He. TwitterRank: Finding Topic-sensitive Influential Twitterers. In *WSDM*, pages 261–270, 2010.

[31] Z. Wu and M. Palmer. Verbs Semantics and Lexical Selection. In *ACL*, pages 133–138, 1994.

[32] X. Yang, H. Steck, Y. Guo, and Y. Liu. On top-k Recommendation Using Social Networks. In *RECSYS*, pages 67–74, 2012.