J Ambient Intell Human Comput (2018) 9:2035–2049
DOI 10.1007/s12652-017-0491-7

ORIGINAL RESEARCH

Tweet and followee personalized recommendations based on knowledge graphs

Danae Pla Karidi[1] · **Yannis Stavrakas**[1] · **Yannis Vassiliou**[2]

Received: 22 December 2016 / Accepted: 11 April 2017 / Published online: 19 April 2017

Abstract Twitter users get the latest tweets of their followees on their timeline. However, they are often overwhelmed by the large number of tweets, which makes it difficult for them to find interesting information among them. In this work, we present an efficient semantic recommendation method that helps users filter the Twitter stream for interesting content. The foundation of this method is a knowledge graph (KG) that can represent all user topics of interest as a variety of concepts, objects, events, persons, entities, locations and the relations between them. Our method uses the KG and graph theory algorithms not yet applied in social network analysis in order to construct user interest profiles by retrieving semantic information from tweets. Next, it produces ranked tweet recommendations. In addition, we use the KG to calculate interest similarity between users, and we present a followee recommender based on the same underlying principles. An important advantage of our method is that it reduces the effects of problems such as over-recommendation and over-specialization. As another advantage, our method is not impaired by the limitations posed by Twitter on the availability of the user graph data. We implemented from scratch the best-known state-of-the-art approaches in order to compare with them and assess our method. Moreover, we evaluate the efficiency and runtime scalability of our method.

✉ Danae Pla Karidi
danae@imis.athena-innovation.gr

Yannis Stavrakas
yannis@imis.athena-innovation.gr

Yannis Vassiliou
yv@cs.ntua.gr

[1] IMIS, Athena RC, Athens, Greece

[2] NTUA, Athens, Greece

Keywords Social recommendation · Content based recommender systems · Knowledge graph · Tweet recommendation · User recommendation

1 Introduction

Twitter is one of the biggest and better known microblogging sites, allowing its users to send and read short messages. Unlike other social media the possible relationships among Twitter users are two, followee or follower. When a user publishes a tweet, it automatically appears on his home page, and on the home pages of his followers (user home pages are also referred to as *timelines*).

Twitter is growing rapidly into one of the most popular social network services. Recent statistics show that more than 550 million users generate more than 300 million tweets every day. This leads to a huge amount of information that can be made readily available, but new problems are introduced as well. Let us think of a user's homepage: it is growing every time a followee tweets, however not all tweets are of interest to the user. Because of this tweet overload, users are often tired of browsing tweets in their homepage, and as a result, they miss interesting tweets.

A solution to this problem is an efficient recommender that helps users filter out uninteresting tweets, and thus, avoid cross passing the interesting ones. Moreover, a similar recommender could recommend followees with similar or relevant interests, resulting in a more interesting timeline. Moving towards this direction, Twitter itself has released the @MagicRecs account that sends personalized recommendations as direct messages to a user, if for example a Tweet is retweeted by several of her followees.

On a different front, knowledge engineers organize human knowledge in an objective way using semantic technologies. In this context, knowledge graphs (KGs) (Ehrlinger et al. 2016) are an instrumental tool for encoding the domains of human knowledge and the relations between them in a way that is manageable by computer systems. It is no coincidence that companies like Google[1], Microsoft[2], IBM, Yahoo, LinkedIn[3], Facebook and Wikimedia promote the development of such structures to improve their commercial services (navigation, search, personalized services, targeted advertising), and to increase the engagement of users (Ferrucci et al. 2010). It is our view that the semantic features of KGs make them suitable for analyzing social network data.

In this work, we present a content based method that uses KGs for (a) personalized tweet recommendation (Pla-Karidi et al. 2016) and (b) personalized followee recommendation (Pla-Karidi 2016). Our method provides an alternative user "timeline" containing tweets that strongly match her interests. Note that these tweets may have been posted by users that are not her followees. This way she will not miss messages that are interesting, even if they are posted by people that she does not follow; at the same time, any irrelevant tweets are filtered out. Furthermore, our method for personalized followee recommendation helps users discover and follow people with similar interests. Both our methods are based on the representation of user profiles as topics of interest (ToIs). In our context, ToIs are nodes of a predefined KG that represent the interests of specific users. Both our recommenders can adapt to cover new topics of interest and reduce the effects of over-specialization and over-recommendation (discussed in Sect. 3.2). As another advantage, our method is not impaired by the limitations posed by Twitter concerning the availability of the user graph data. We implemented from scratch the best-known approaches in order to compare with them. The efficiency of our method outperforms in many cases the state-of-the-art approaches and yields good results in terms of precision and time scalability.

The central idea behind our approach is described in (Pla-Karidi et al. 2016), where we presented our tweet recommendation system. In this paper, we extend that work as follows:

a. we describe in detail our followee recommendation system, for which the intuition has been introduced in (Pla-Karidi 2016),
b. we introduce a common architecture for the tweet recommender and the followee recommender,
c. we emphasize the shared part of the two recommenders, and focus on the use of the KG, which is the cornerstone of our approach,
d. we conduct an experiment to evaluate the followee recommender,
e. we conduct experiments with a large dataset to evaluate the efficiency and the scalability of our approach.

The paper is structured as follows. Section 2 presents an overview of the current state of the art in areas that are related to content and user recommendation in social media networks. Section 3 presents the intuition of our method, while Sect. 4 describes the profiling and recommendation processes in detail. Section 5 presents the experimental evaluation of our model along with a detailed description of the datasets and results regarding both the efficiency and runtime scalability of our method. We conclude the paper in Sect. 6.

2 Related work

A wide variety of tweet and followee recommendation methods can be found in literature. These methods can be grouped into three categories: Collaborative Filtering, Content-Based, and Tweet Ranking methods. In what follows we examine each of those categories.

2.1 Collaborative filtering

Collaborative filtering (CF) methods make use of the community data to build user profiles (Su et al. 2009). The intuition behind these methods is that users that share the same opinion on some topics (interesting, not interesting) tend to have the same opinion on other topics too (user-based CF). Moreover, topics that produce the same opinion from some users tend to receive similar opinions from other users (item-based CF) (Adomavicius and Tuzhilin 2005; Schafer et al. 2007). Both neighborhood-based methods (Sarwar et al. 2001; Shi et al. 2009) and model-based methods (Koren et al. 2008; Rendle et al. 2008) are subcategories of CF that are used widely in tweet recommendation. Neighborhood-based methods recommend items based on the similarity of user of the item neighbors and model-based methods perform recommendation using matrix factorization model or the probabilistic latent factor model. CF methods use user graph data extracted from Twitter like follow and retweet links, to construct a network structure. These methods (Romero et al. 2011; Yang et al. 2012) apply network analysis algorithms to the network structure to find interesting messages. However, the

[1] https://www.google.com/intl/bn/insidesearch/features/search/knowledge.html.

[2] http://blogs.bing.com/search/2013/03/21/understand-your-world-with-bing/.

[3] https://engineering.linkedin.com/blog/2016/10/building-the-linkedin-knowledge-graph.

network construction requires a large volume of link data to be retrieved, stored and analyzed and thus can't be updated in an effective and scalable way when new tweets are published in the stream. Several algorithms and features to be extracted by a user network have been suggested to identify interesting tweets (Lauw et al. 2010). Such a feature is the topology of the followers network (Armentano et al. 2012) that was used to recommend users. Collaborative filtering approaches require each tweet to get instantly feedback from numerous users before being recommended to other users, known as the "cold-start" problem. In (Rendle et al. 2008) authors use a model-based method, which proposes online update rules on a stochastic gradient descent style based on the last example observed. In (Diaz-Aviles et al. 2012) authors propose RMFO-RSV method that maintains a reservoir with a representative set of previously seen data points from the stream, which provides a significant boost in performance compared to the one obtained when only the last example is considered. In (Hong et al. 2013) the authors use Co-Factorization Machines (CoFM) to address the problem of simultaneously predicting user decisions and modeling content in social media by analyzing rich information gathered from Twitter. These methods consider the relationship between tweets and users and the relationship between users and publishers separately. The problem is that two tweets with the exact same text posted by two users will be evaluated differently, although they have the same content thus they are of the same interest to the user! In (Wang et al. 2016) the authors make recommendation with social trust information based on matrix factorization methods.

In (Yigit et al. 2015) the authors present an extending topology based algorithm for recommending users in Twitter. The proposed algorithm classifies the users according to their friendship relations and constructs a class including user ids to recommend the target user. User actions and user mentions are also used to optimize the results. In (Chen et al. 2012) a collaborative ranking model is proposed, CTR, which considers three major elements on Twitter: tweet topic level factors, user social relation factors and explicit features such as authority of the publisher and quality of the tweet. In (Kim et al. 2011) the authors propose a probabilistic model based on Probabilistic Latent Semantic Analysis (PLSA) to recommend potential followers to users in Twitter. In (Bhattacharya et al. 2014) the authors propose a methodology to infer interests using some user followees (topical experts) and social annotations (collected via the Twitter Lists feature). in (Liu et al. 2016) authors provide followee recommendations by calculating user relevance scores, using neural networks to combine network topology and content of tweets. In (Rodríguez et al. 2016) authors recommend followees, using a fuzzy system that exploits followee similarity along with text similarities. Finally, in (Sharma et al. 2016) authors present GraphJet, a recently-deployed system for real-time content recommendations in Twitter, which is based on a real-time bipartite interaction graph between users and tweets.

2.2 Content based

A common solution to the cold start and complexity problems is to use other information like the textual content of the items to be recommended (Balabanović et al. 1997; Schein et al. 2002). In (Alonso et al. 2010) the authors used crowdsourcing to categorize a set of tweets as interesting or uninteresting and reported that the presence of a URL link is a single, highly effective feature for selecting interesting tweets with more than 80% accuracy. However, this rule may categorize incorrectly an uninteresting tweet (links to meaningless content) as interesting. Content-based methods build profiles by using the user history tweets. Such recommenders are often used in domains where a large amount of textual content is available for each user, such as websites. Recommending interesting tweets using content is not easy, because tweets are limited in size. Previous works in content-based methods mainly recommend tweets to users by using content analysis like Latent Dirichlet Allocation (LDA) or TF IDF metrics to represent user interests. In (Pazzani et al. 1996) the authors first created bag-of-word profiles for individuals from their activities and then chose websites most relevant to the profile of the individual as recommendations. In (Kawamae et al. 2011; Wang et al. 2006) authors conducted topic modeling of temporally-sequenced documents in Twitter and tried to model the topics continuously over time. These approaches learn topic shifts based on word distributions of tweets, while TS-LDA in (Yang and Rim 2014) the model is learning changes based on topic distributions. Another approach, the Labeled-LDA (Ramage et al. 2010), is used to model a tweet using its labeled information, and then built the probability distribution vector of latent topics to represent the content of tweets. Based on similarity between the topic vectors, the incoming tweets are marked as interesting or not interesting.

In (Lu et al. 2012) authors used Explicit Semantic Analysis (Gabrilovich et al. 2007) to construct the user interest profile based on Wikipedia concepts, to re-rank his timeline. However, in Twitter, the content of user tweets is much limited and sparse, so that these explicit terms extracted from history tweets are insufficient to reflect user interests. For example, some latent interests or preferences cannot be characterized in content-based methods (Koren et al. 2008). Another approach to analyzing Twitter that uses topics is TwitterRank, which aims to identify influential micro-bloggers (Weng et al. 2010). This approach leverages LDA by creating a single document from all user

Tweets and then discovering the topics by running LDA over this "document." Again, such an approach has the problems of LDA since the Twitter data is sparse, and the generated topics are based on terms rather than concepts. Most of the time, the twitter activity of a user is insufficient for creating a reliable profile. For this reason, a wide variety of approaches make use of both Content-Based and Collaborative Filtering methods. In (Balabanović et al. 1997) authors proposed to create user profiles not from the contents of tweets of an individual, but from a group of tweets posted by related users. In (Hannon et al. 2010) authors evaluated a range of different profiling and recommendation strategies, based on a large dataset of Twitter users and their tweets, as well as the relationships between them to make useful followee recommendations. In (Elmongui et al. 2015) TRUPI is proposed, a system which combines the user social features and interactions and the history of her tweets and also captures the dynamic level of user interests in different topics to accommodate the change of interests over time. In (Naveed et al. 2011) authors propose a method to predict the probability of a tweet being retweeted based on content features alone. In (IJntema et al. 2010) authors propose Ontology Based recommendations for news recommendations, using a traditional term-based recommender and several semantic-based recommendation algorithms to compare unread news items with the user profile and recommending items with the highest similarity with the user. In (Tajbakhsh et al. 2016) a semantic TF IDF method is proposed, which weighs each message according to two factors: TF-IDF and a semantic similarity measure. In (Subercaze et al. 2016) the authors provide real-time recommendations by building a graph of words. In (Zhao et al. 2016) the authors represent users by user-topics LDA distribution and recommend the top-k similar users by computing hashtag frequencies.

2.3 Tweet ranking

Some recent approaches focus on recommending tweets from the user timeline. In (Duan et al. 2010) authors use a learning-to rank algorithm that uses content relevance, account authority, and tweet-specific features to rank the tweets in the timeline. Other approaches construct a tweet ranking model making use of the retweet behavior of a user. For example, they rank both the tweets and the users based on their likelihood of getting a tweet retweeted (Uysal and Croft 2011).

The amount of information provided by Twitter is so large, that most of the already mentioned algorithms become intractable. Many optimization methods were developed to reduce time complexity. For example, in (Sarwar et al. 2002) the authors applied clustering algorithms to partition user population, built neighborhoods for users from the partition, and considered only those neighborhoods when computing recommendations.

2.4 Comparison with our approach

Our method avoids the efficiency problems of LDA or TF-IDF based methods that are caused by the limited size of tweets. Our approach is using each tweet separately (assigns a topic to each tweet) in contrast to most of the content based methods, which merge tweets and therefore defy the proper granularity for topic extraction. Moreover, our method takes advantage of the KG in order to recommend tweets of *related* topics, while other methods recommend the exact topics that have been found.

In contrast to collaborative filtering techniques, our approach doesn't face the problem of resources availability, due to the fact that it makes no use of the twitter user graph data. This problem is discussed in detail in Sect. 3.2. Whereas some other approaches require a lot of processing time, the overall time needed for our method to construct a new user timeline is minimum and thus it can be implemented as a streaming online service. Finally, comparing with Ontology-based recommenders, we believe that KGs (like Google and Wikipedia Graph) are less complicated than ontologies and thus they provide a stable but also lightweight basis for tweet recommendations.

3 Overview and motivation

In this section we present an intuitive overview of our approach. We also discuss the motivation behind it.

3.1 Overview of our approach

Our approach is based on two principles:

1. The representation of all possible user interests as a hierarchical KG, with more general concepts on top, and more specific concepts as children. Each node corresponds to a ToI, while edges denote the category-subcategory relation between ToIs.

To construct the KG, we opted to use the AlchemyAPI Taxonomy[4] service and its Categories dataset, which is a set of concept categories and subcategories extending up to 5 levels deep. For example, the category "music genres", which is a sub-category of "music", has 17 subcategories, and each of them represents a music genre. This

[4] https://www.ibm.com/watson/developercloud/alchemy-language.html.

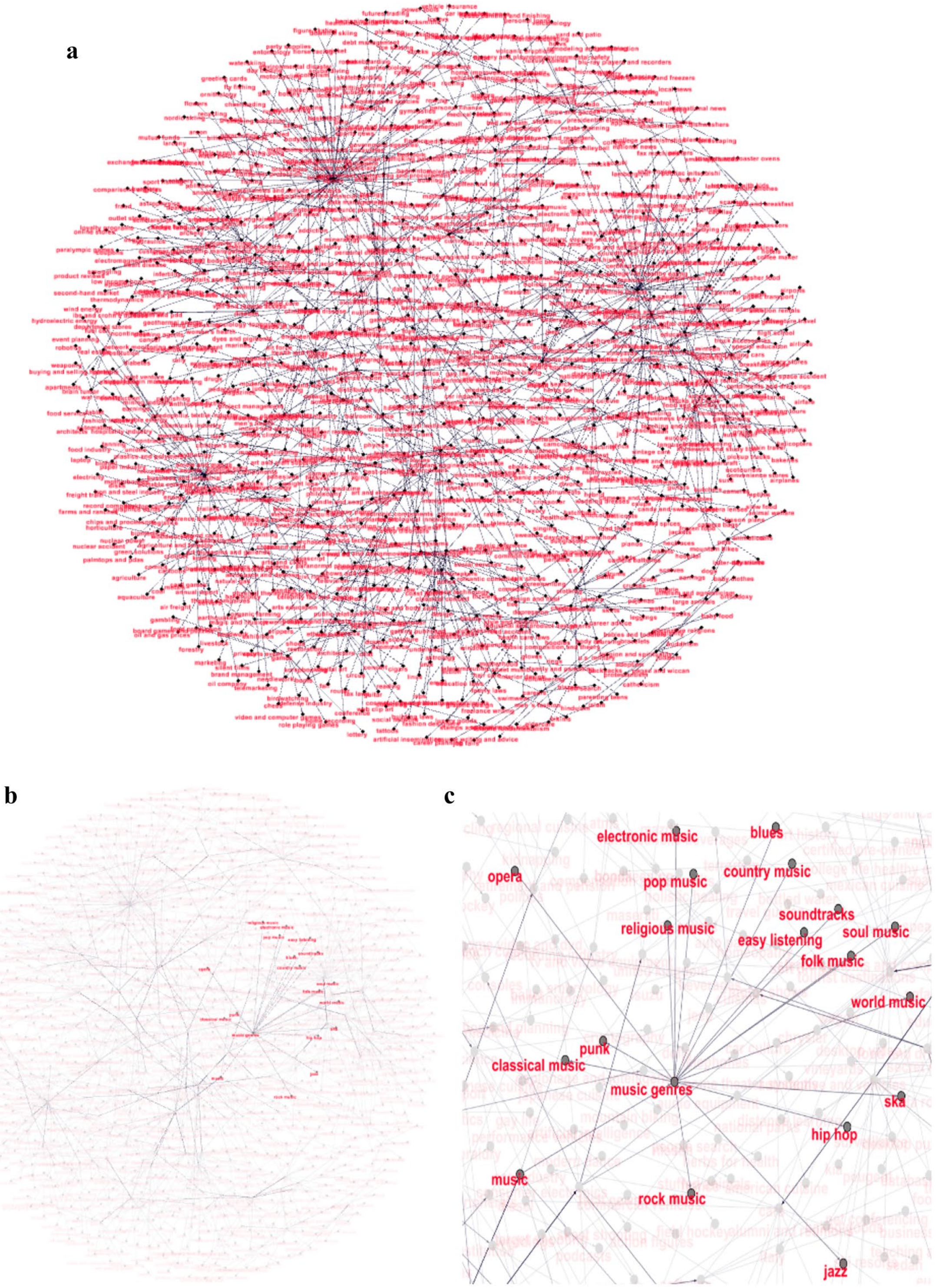


Fig. 1 **a** Knowledge Graph, **b**, **c** Music Genres

example is shown in Fig. 1. Alchemy Taxonomy concepts form a graph $G = (V, E)$, where $V = \{v_i\}$ is the concept set from AlchemyAPI Taxonomy. Each concept v_i is connected with the concept v_j, if and only if these concepts are related in AlchemyAPI Taxonomy Dataset via an edge that belongs to the set $E = \{e_i\}$ of graph edges. All edges are of equal weight. The KG consists of 1092 nodes (concepts) and 1323 edges (concept relations). We use the relation category-subcategory, and the existence of an edge between two concepts is an indicator of semantic relevance. The KG covers the vast majority of concepts that are used in everyday life, therefore it provides a wide knowledge base for our recommender.

2. The representation of the profile of any user as a subgraph of the KG, such that the nodes represent the interests of the specific users (ToIs). Those ToIs are subsequently ranked from the more specific towards the more general.

The construction of user profiles requires to extract a subgraph of the KG, containing the user ToIs. The optimum such subgraph is extracted by using the Steiner Tree (Gilbert et al. 1968) extraction algorithm. Given a graph $G = (V, E)$ and a set $R \subseteq V$, a Steiner Tree is the least cost connected subgraph spanning R. In our method R contains the set of ToIs extracted from user timeline and all edges are of equal length. To compute Steiner trees, we applied the algorithm supplied by the GOBLIN library (http://goblin2.sourceforge.net/), which is based on the heuristic described in (Mehlhorn et al. 1988). The goal of this heuristic is to extract a tree connecting all these ToIs with a minimal sum of costs along its edges. Finally, these ToIs are ranked to avoid recommendations based on very abstract ToIs using a DFS Post order traversal of the tree.

3.2 Motivation of our approach

3.2.1 *Knowledge graph*

A pivotal issue for a recommender is the selection of the criteria, based on which the system will provide recommendations. Most content-based recommenders, which rely on text/keyword similarity, lack in efficiency due to the small size of tweets. Unlike these methods, our approach is based on the semantic relevance between the user interests and the incoming tweets. Choosing semantic relevance as the recommendation criterion has the following advantages. Considering that users read and write content over multiple topics, our intuition is that a recommender should take into account the conceptual associations between these topics, which are—up to some degree—objective and immutable. Our method is based on the representation of user profiles as Topics of Interest (ToIs). Specifically, profiles are constructed upon a predefined structure, the Knowledge Graph. The KG consists of nodes that represent concepts and objects (e.g. events, persons, entities, locations that are potential ToIs), and edges that represent relations between them. In our context, ToIs are nodes of a KG, which represent the interests of specific users. The usage of a KG provides us with a common basis for (a) generating user interest profiles and (b) calculating the semantic relevance between them on the one hand and the incoming tweets on the other. Moreover, KG semantically outperforms the LDA self-topic approaches, as well as term frequency approaches whose efficiency suffers due to the small size of tweets.

3.2.2 *Over-specialization*

A common problem for content-based recommenders is over-specialization. Content-based recommenders suggest items whose similarity scores are high when matched against a user profile. Such approaches, however, restrict the user exclusively to tweets very similar to those already seen by providing recommendations that contain recurrent information and certainly not covering one's range of interests. This problem, called over-specialization, is avoided by our recommender. The intuition is that our recommender provides content covering relevant ToIs as well, ensuring that the recommended tweets span "as much as possible" on the KG, and do not come all from the same node/ToI. To achieve that, we extend the user profile with related ToIs from the KG. The KG contains thousands of nodes, so this extension should respect some constraints. For example, if a node is connecting two or more user ToIs in the KG, then this node is likely itself a ToI. Continuing on this line, the connection of user ToIs in an optimum way leads to a wider profile, exploiting the semantic relations between ToIs. To accomplish this optimum connection, we use the Steiner Tree algorithm, as discussed in the Sect. 4.

3.2.3 *Over-recommendation*

Depending on the type of KG, nodes can represent either exclusively specific objects (events, entities, persons, etc.) or a combination of categories (concepts) and objects. In our approach, as we show in the next section, the KG is a topic taxonomy including topic categories and objects. The recommendation based on a very abstract category results in too many possibly not interesting recommendations. This problem is known as over-reccommendation. For example, let's assume a user is interested in ancient history. A recommender of general topics would recommend tweets about "science", a super-category of history, that include chemistry, computer science, medicine, etc. Ignoring these abstract nodes during user profiling is impossible, since the hierarchy structure of our KG does not allow us to find an optimum subgraph that does not include them. Our approach avoids this effect by ranking the user profile, as we show in Sect. 4.3.

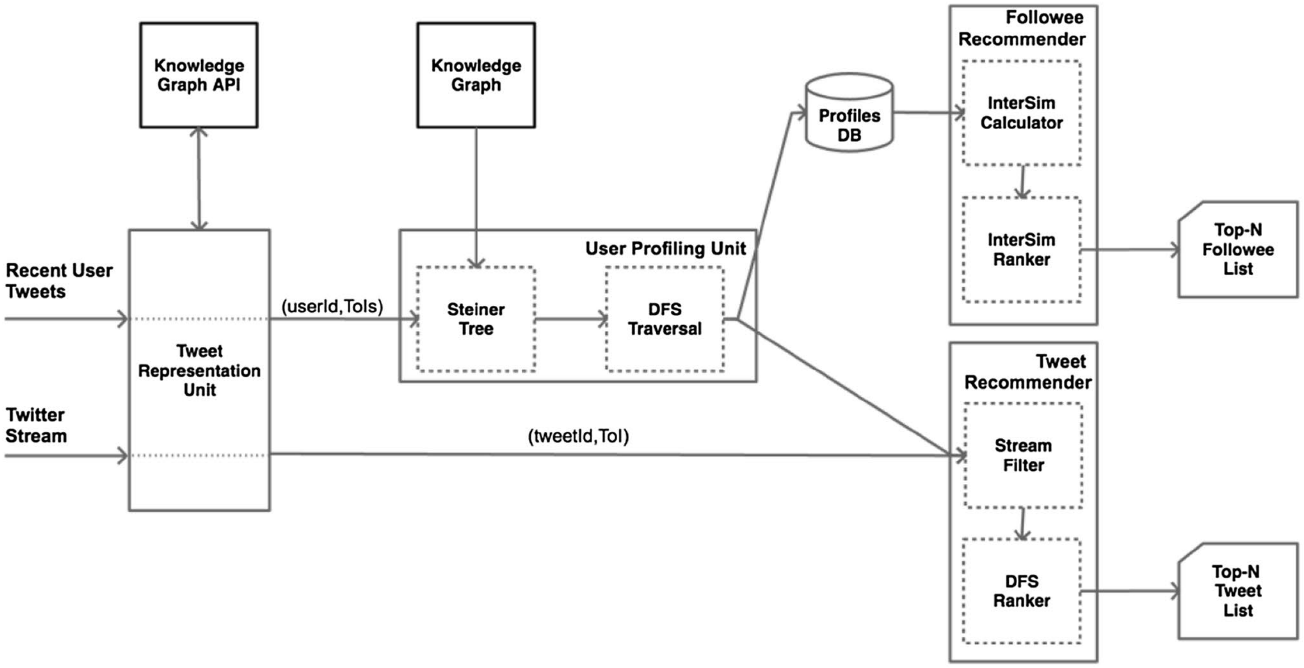


Fig. 2 Tweet and followee recommendations based on knowledge graphs

3.2.4 Followee recommendation

Apart from recommending interesting tweets, another way to improve the user timeline is to recommend followees with similar or relevant interests. Our assumption is that users choose who to follow based on certain criteria, from which the most important is whether the followee posts tweets that are interesting to the user. For example, a person who is interested in sports and politics is likely to follow a person who is interested in these topics too. But a regular Twitter user, apart from some famous individuals, is not widely able to know who shares her interests. Using the same underlying ideas (namely the KG and the Steiner tree) to profile users, we introduce a followee recommendation method that uses a similarity metric, called InterSim, discussed in Sect. 4.4. Additionally, our method can be applied using any KG, since our method's basic principles are not restricted by the specific tools and taxonomies.

3.2.5 Resource availability

Another important advantage of our approach is that we avoid the problem of resource availability. This common problem that many recommenders must face is caused by the cost of the resources (Twitter data) that are necessary for the profiling. The Twitter API poses restrictions regarding the user graph data that are significantly greater than those regarding timeline data (tweets). For example, a recommender can request only 15 friends or followers of a specific account every 15 min, while the restriction for requesting the account's tweets is 1500 every 15 min. Our approach makes no use of Twitter user graph data (friends and followers' relations information) and therefore avoids the problem of resource availability. Instead, we use a set of the user's most recent tweets, which is automatically updated at fixed time intervals. This way our recommender can dynamically adapt to cover new topics of interest that may arise. At the same time our method allows for a lightweight implementation.

4 Tweet and followee recommendation model

In this section, we present in detail our recommendation model, which consists of a common user profiling process and two distinct recommenders: the tweet recommender and the followee recommender. These recommenders provide recommendation lists of tweets and followees, respectively.

4.1 Overall architecture

Figure 2 depicts the overall architecture of our recommendation model, which consists of: the tweet representation unit, the user profiling unit, the followee recommender and the tweet recommender. Those are presented in detail in the following sections.

4.2 Tweet representation unit

The function of this unit is to assign ToIs to tweets. The unit receives as input (a) a set of user tweets and (b) the streaming tweets from Twitter. The output is (a) a list of ToIs for each user and (b) a ToI for each tweet. First, every tweet is pre-processed to remove special characters (emoticons, etc.), and to expand shortened urls. Then, the unit assigns a ToI from the predefined KG to each tweet, using AlchemyAPI's Taxonomy API. The Taxonomy API is an online service for semantic text analysis that assigns concepts from AlchemyAPI Taxonomy Categories dataset (described in Sect. 3.1) to tweets. This service automatically categorizes text and HTML into its most likely topic category from the KG.

4.3 User profiling unit

This unit is responsible for constructing user profiles. The unit receives as input a list of ToIs for each user, which is provided by the Tweet Representation unit. The output is an *extended* ToI list for each user. The unit extends the ToI list by finding related ToIs from the KG using the semantic relations between them. Specifically, this extended list is extracted from the KG using the Steiner Tree algorithm (Sect. 3.2). The intuition behind using Steiner Tree is as follows. As discussed in detail in Sect. 3.1, given a set of ToIs, Steiner Tree is the least cost connected subgraph of KG that contains these ToIs. If a concept is connecting two or more ToIs in the KG, then this concept is likely itself a topic of interest. Although some topics may not be directly related to one's interests, we assume that if a concept belongs to the Steiner Tree of the ToIs, then the likelihood of it being itself a ToI increases.

Next, the extended list is ranked to avoid recommending tweets and followees based on very abstract ToIs. As we discussed in Sect. 3.2, we assume that a user is interested in reading tweets about specific topics. Therefore, our method ranks the user profile using a DFS Post Order Traversal, whose main effect is to explore deeper into the graph, hence promote more specific topics. This traversal requires a root node, which should be of the most abstract level in our KG. The User Profiling unit uses as root node the node of the most abstract level, which is closest to the the node/ToI that has been found the most frequent in the tweets of the user. This ranking method is named TGS-post. Finally, user profiles are stored in a database. A complete profiling example is presented in Sect. 4.5.

We designed and implemented two alternative ranking methods:

- TGS-tf: ToIs are ranked based on the frequency of their occurrences in user tweets.
- TGS-bfs: ToIs are ranked based on the BFS traversal of the Steiner Tree.

4.4 Tweet and followee recommenders

Given the ranked user profiles generated by the User Profiling unit, we now focus on the explanation of the two recommenders:

- The Tweet Recommender receives as input a ToI for each streaming tweet, as assigned by the Tweet Representation module. If a streaming tweet is assigned to a topic that is included in the user profile, then the Stream Filter temporally stores the tweet in a database. The DFS Ranker will then rank the tweets based on the order of ToIs in the user profile (DFS), and will store them in the Top-N Tweet List.
- The Followee Recommender calculates user interest similarity and ranks the stored user profiles. Specifically, for each user profile that is stored in the database, the InterSim Calculator determines the Interest Similarity (InterSim) with all other users. InterSim is calculated by measuring the subgraph overlap between these profiles. Thus, the number of common ToIs between a user and every other profile stored in the database gives us the user interest similarity. Moreover, we use the profile graph diameter as a normalization factor for each calculation. Finally, the InterSim Ranker ranks the profiles in descending order of interest and stores them in the Top-N Followee List.

4.5 A concrete example

As an example, consider Paul Mason, a widely known journalist and broadcaster, who currently works as economics editor at Channel 4 News in UK. We retrieve his timeline, and the Tweet Representation unit assigns a ToI to each tweet. The tweets and the corresponding ToIs are shown in Table 1.

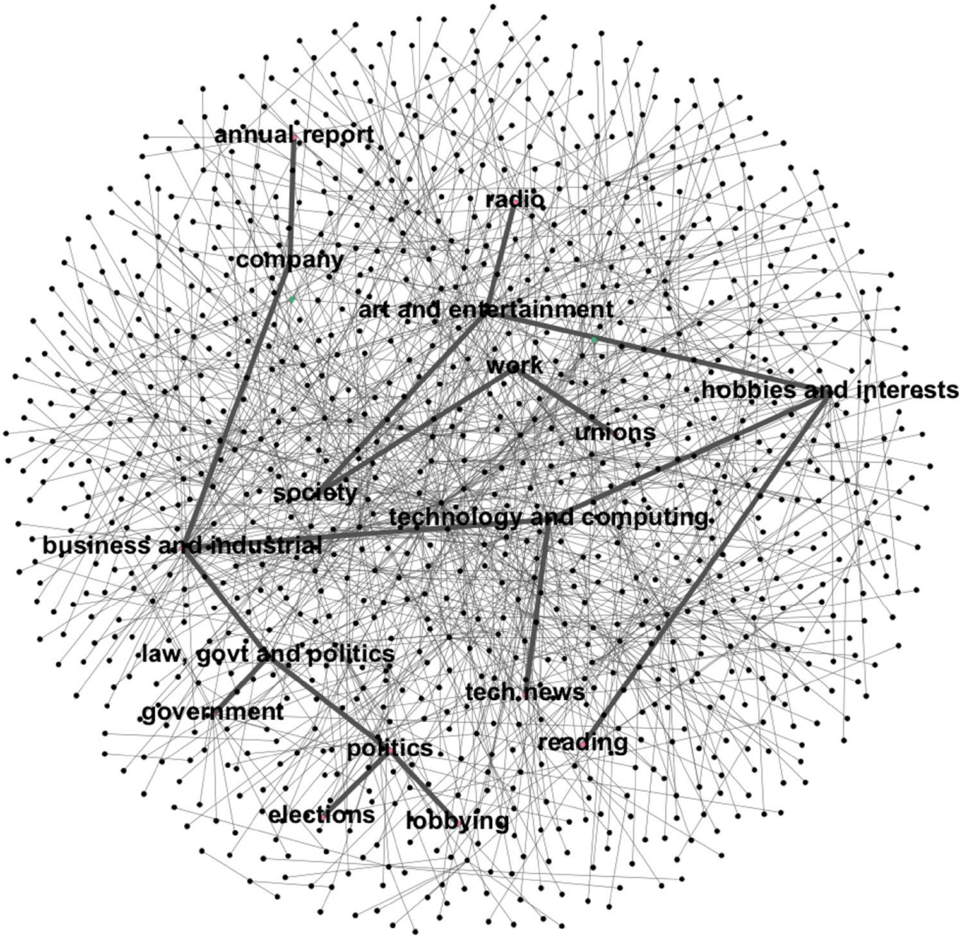
The list of assigned ToIs is: annual report, politics, radio, reading, lobbying, government (three instances), tech news, business and industrial, elections, unions.

Next, the User Profiling unit applies the Steiner Tree using as input this list of ToIs. The resulting Steiner Tree is depicted in Fig. 3 and consists of the following ToIs: *annual report, radio, business and industrial, company, art and entertainment, hobbies and interests, reading, law, govt and politics, government, politics, society, work, unions, technology and computing, tech news, elections, lobbying*. Afterwards, the unit ranks these ToIs using a DFS post order traversal of the tree. As described in Sect. 4.3, the

Table 1 Paul Mason Tweet Representation

Tweets	ToIs
"Reglingnoics"! https://t.co/q5lozpnd4l in other respects "had so many strange ideas"! Greekdept	Annual report
According to Regling I had "strange ideas" http://t.co/Qwp4ww8d9 H Read them here &amp; compare them to Mr Regling's	Politics
Tonight on BBC One's Question Time http://t.co/wghmmx4grh	Radio
Il Fatto Quotidiano interview on M. Renzi's 'comment' &amp; my reply http://t.co/fhfmidntzd	Reading
Interviewed by POLITICO on the 3rd Bailout, Schauble-Merkel, the Eurozone &amp; the refugee crisis http://t.co/hpuyrmvggw	Lobbying
DER SPIEGEL: Complicit in Corruption: How German Companies Bribed Their Way to Greek Deals http://t.co/9cdikw4ahn	Government
Ne pas manquer vendredi soir le prochain live de Mediapart: deux heures avec https://t.co/xtktgivrba	Tech news
The lenders are the real winners in Greece http://t.co/ltjtyvu5zw	Government
We would like to see you again fighting together for Syriza victory...Your division it's a defea…	Business and industrial
The double purpose of these elections http://t.co/phl26kcmf2	Elections
Paul Krugman on Greece's 3rd mou: an agreement designed to fail http://t.co/hhwiw5fgop	Government
Yanis Varoufakis—'Left should beware of friends who fear confronting the rich' http://t.co/posnmlqviq	Unions

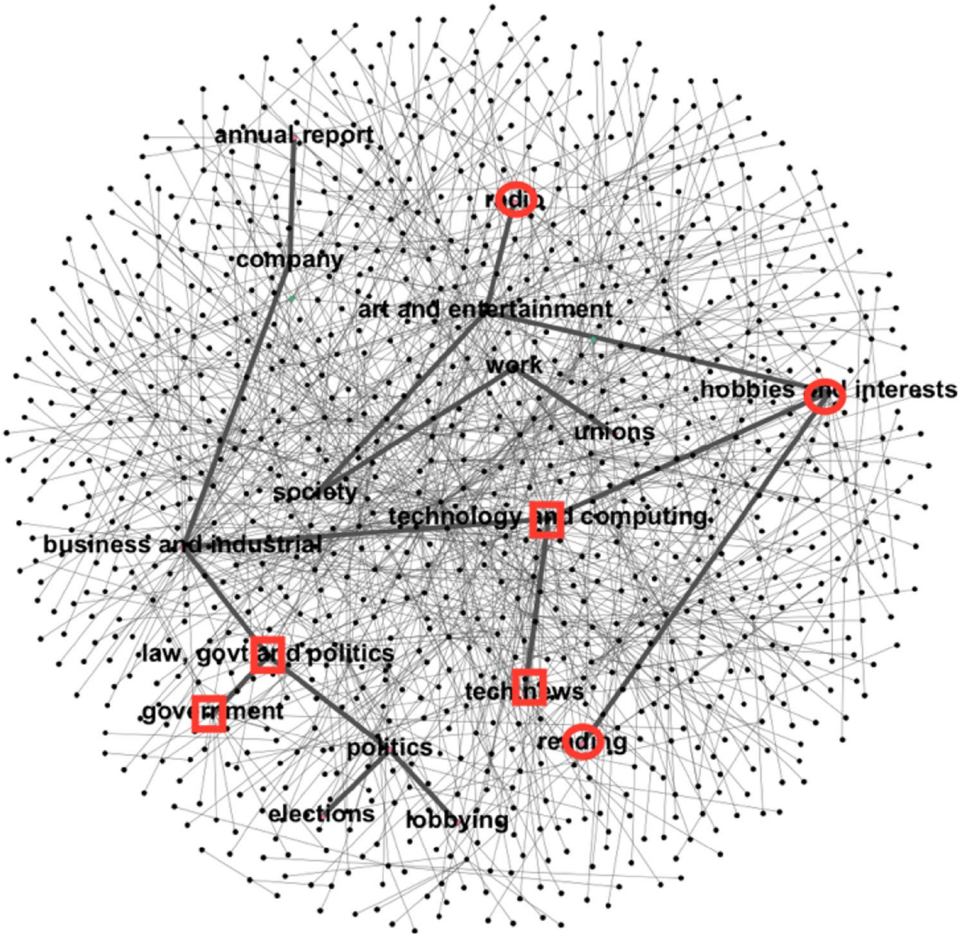
Fig. 3 Paul Mason profile



root node is chosen based on the frequency of the ToIs in the tweets of the user. In this example, the most frequent ToI is the topic "government". Thus, according to Alchemy Taxonomy, the nearest abstract ToI in the Steiner Tree is the topic "law, govt and politics", which is chosen as the root node.

The final ranked profile of Paul Mason is: *government, elections, lobbying, politics, annual report, company, tech*

Fig. 4 User profile overlap



news, reading, unions, work, society, radio, art and entertainment, hobbies and interests, business and industrial, law, govt and politics.

Next, the tweet recommender receives the streaming tweets and the corresponding ToIs and filters those that are included in the user profile. For example, let us assume the following ToIs for a stream of tweets:

- ToI of tweet 1: radio
- ToI of tweet 2: statistics
- ToI of tweet 3: elections

First, the Tweet Recommender filters out tweet 2, since "statistics" does not belong to the user profile. Then, the Tweet Recommender promotes tweet 3 over tweet 2, because "elections" has higher priority than "radio" according to the ranked user profile. Hence, the recommendation list is: (1) tweet 3 and (2) tweet 1.

For a followee recommendation example, we picked two random Twitter users, user1 and user2, with equal sized profiles and profiled them. The KG's nodes that represent the overlap between Mason and user1 are surrounded by rectangles and the common nodes between Mason and user2 by circles, as shown in Fig. 4. Thus, InterSim(Mason, user1) = 4 and InterSim(Mason, user2) = 3, which means that user1 will take a higher place in the followee recommendation list.

5 Experimental evaluation

In order to evaluate our tweet and followee recommenders, we conducted two offline evaluation tests presented in Sects. 5.1 and 5.2. The results of the tweet recommendation method were compared with the most popular state-of-the art methods. Furthermore, we conducted a large scale in-depth evaluation test using a large real-life dataset along with a runtime test of the proposed method. The results are shown in Sect. 5.3, proving our method's stability and runtime scalability.

5.1 Comparison with other approaches

In order to evaluate our tweet recommender, we conducted an offline evaluation test and compared it with the most popular state-of-the art methods. For a set of users, we gathered their most recent tweets, constructed their profiles, and recommended tweets based on the approach described in Sect. 4. Our user dataset was constructed crawling tweets and retweets from "The Twitter 100" users of 2012[5] list. This is a list of Britain's most influential users of 2012 based on PeerIndex that measures interactions across the web to help users understand their impact in social media. First, we constructed user profiles as follows:

- Crawl twelve most recent tweets of the user (twelve was chosen to avoid scalability and info availability issues, due to Twitter API Rate limits)
- Assign a topic (ToI) to every tweet, as described in Sect. 4.2
- Extract the Steiner Tree from the Knowledge Graph containing the ToIs, as described in Sect. 4.3
- Execute a DFS traversal of the Steiner Tree as described in Sect. 4.3

The resulting tree is the user profile. Finally, our dataset consists of a hundred users and their profiles, which are represented as vectors of ToIs ordered according to the DFS traversal. Subsequently, we constructed a test set in order to evaluate our method. We decided to build this test dataset out of the users' retweets, because we assume that when a user retweets a post, he is most likely interested in it. Then we assigned a ToI to each retweet, as described in Sect. 4.2.

The test dataset consists of the most recent retweets crawled from the users' timelines (500 retweets) and their Topics of Interest. The test process was made in the following stages for each user in the first dataset:

- Get user profile
- For each ToI in the vector (beginning from the first) get all retweets of the same ToI from the test dataset
- Store them in the recommendation list
- Continue from stage 2 for the next ToI in the profile vector

This way we manage to rank all retweets from test dataset according to profile and store them in the recommendation list. Subsequently, we computed three performance measures: precision-at-k, mean average precision and overall accuracy. Precision-at-k corresponds to the precision (information retrieval performance measure) calculated in the first k recommendations in the recommendation list. As relevant elements we consider the retweets made by the user. We conducted experiments from k = 1 to 10. The results are shown in Tables 2 and 3.

Table 2 Precision-at-k

k	Precision-at-k
1	0.236559139785
2	0.195652173913
3	0.195652173913
4	0.16847826087
5	0.154347826087
6	0.143115942029
7	0.143115942029
8	0.126358695652
9	0.115942028986
10	0.105434782609

Table 3 Mean Average Precision and Overall Accuracy

Mean average precision (MAP@10)	0.157973978161
Overall accuracy	0.988854305118

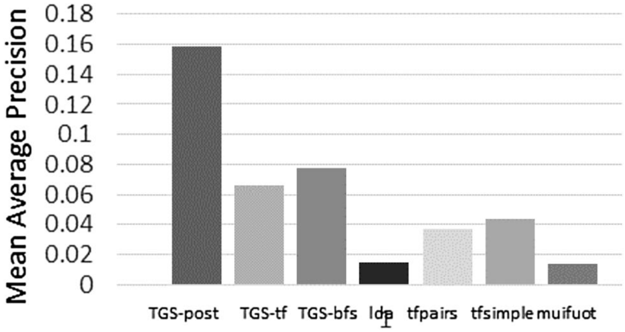


Fig. 5 Precision

Figure 5 depicts the mean average precision comparison between our method, TGS-post, our alternative rankings, TGS-tf and TGS-bfs and four state-of-the art approaches. Most approaches do not provide detailed instructions regarding their implementation. For the comparison to be fair, we implemented four popular methods, namely a simple LDA (lda), two TF-IDF (tfsimple and tfpairs, i.e. single word and word pairs) and a collaborative filtering approach [muifuot (Pennacchiotti et al. 2012)]. As we can see, our method reaches a mean average precision score of 15.7% and outperforms the other methods.

Figure 6 depicts the overall accuracy comparison between these methods. As we can see, our method reaches an overall accuracy score of 98.8% and outperforms the state-of-the art methods.

Both efficiency metrics show that our recommender can successfully retrieve the interesting (retweeted) and not

[5] The Twitter 100: Britain's titans of the Twittersphere - http://www.independent.co.uk.

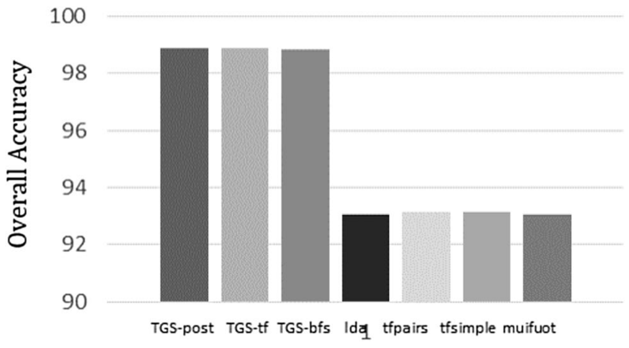


Fig. 6 Accuracy

Table 4 Precision-at-k

k	Precision-at-k
1	0.14315353
2	0.16457286
3	0.18903036
4	0.19455253
5	0.20481928
6	0.21046443
7	0.21572796
8	0.22233202
9	0.22504622
10	0.23011994

Table 5 Mean Average Precision

Mean average precision (MAP@10)	0.199981913

interesting tweets (true positive and true negative results), but still recommends some tweets that were not retweeted by the user (false positives). We can also observe that our model outperforms in terms of precision and accuracy the most common state-of-the art methods (lda, tfidf-simple, tfidf-word pairs, muifuot) mentioned in Sect. 2.

5.2 Followee recommendation experiment

In order to evaluate our followee recommender, we conducted an offline evaluation test based on the dataset presented in Sect. 5.1. First, we constructed user interest profiles as in Sect. 5.1. Subsequently, we crawled the followees of all 100 users and used it as ground truth to compare with our recommender results. Finally, we used this dataset as input to our recommender. The evaluation process was made in the following stages for each user in the dataset:

- Calculate Interest Similarity (InterSim), as described in Sect. 4.4, between the user and every other user in the dataset.
- Rank recommended users in descending order of InterSim
- Store top-k recommended users in a recommendation list

This way we managed to rank all users from the dataset according to the user profile and computed precision-at-k and mean average precision (MAP@10). Precision-at-k corresponds to the precision (information retrieval performance measure) calculated in the first k recommendations in the recommendation list. As relevant elements we consider user's ground truth followees made by the user (e.g. all ground truth followees that exist in recommendation list are considered as true positives). We conducted experiments from k = 1 to 10. The results are shown in Tables 4 and 5.

As we can see, our method reaches a mean average precision score of 19.99%, indicating that our recommender can efficiently recommend followees. However, due to Twitter API Rate limits, our dataset is limited in 100 users, while the ground truth dataset contains all true followees. Therefore, we expect that precision-at-k would be higher, if the dataset was more complete.

5.3 Experiment with a large dataset

In the previous section, we presented the evaluation results of our method using a 100-user dataset. This dataset that was used, because the previous experiment aimed to compare our method to the state-of-the art, concerned the top-100 active users in one country. A larger dataset would make it impossible to test any collaborative filtering method, due to Twitter's API Rate limits regarding followee data. This small dataset raised questions regarding the efficiency of our method in less biased and more real-life datasets. Furthermore, we did not have a measure regarding the runtime of the proposed method. In this section, we answer these questions by evaluating our tweet recommender using a large and not biased dataset. The evaluation of our followee recommender would require a large amount of user user graph data (followers and followees), which we were not able to collect due to Twitter's API Rate limits. Finally, we modified the KG by removing some nodes that are constantly miss-assigned by AlchemyAPI. For example, every tweet that could not be assigned by the API to any other category was finally miss-assigned to category "social network", because of the tweet's metadata or url.

5.3.1 Dataset

Our Dataset consists of all public tweets (~1% sample of all tweets) that were posted from 24 March until 23 April

2012, which were collected using the Public Streaming Twitter API. We requested only stream tweets written in the English language. During these 31 days of tweet crawling, we managed to store in compressed Json text files 146887375 tweets (48,96 GB on disk). However, our dataset contained duplicate tweets and tweets without text or url content, thus tweets that cannot be analyzed by AlchemyAPI Taxonomy. For this reason, a short preprocessing stage was added to remove duplicate and text-empty tweets from the Dataset. To accomplish both preprocessing and further data manipulation needs, we imported the data into a PostgreSQL database.

5.3.2 Forward chaining validation and data integration

To test the efficiency of our tweet recommender, we choose as ground truth test dataset the data retrieved from users' retweets, because we assume that when a user retweets a post, he is most likely interested in it. Moreover, since our recommender is meant to provide real-time recommendations based on past data (older user tweets), cross validation could be problematic (e.g. interest changing, emerging events, new ToIs etc.). Forward chaining evaluation method can model the situation at prediction time. Following this method, we divided the Dataset in 5 subsets (four sets of 6 days tweets each and one of seven-days tweets) respecting the chronological order of their publication, to form our train datasets. Thus, our training sets are:

- TrainSet [1]: tweets and retweets from day1–day6
- TrainSet [2]: tweets and retweets from day7–day12
- TrainSet [3]: tweets and retweets from day13–day18
- TrainSet [4]: tweets and retweets from day19–day24
- TrainSet [5]: tweets and retweets from day25–day31

According to Forward Chaining evaluation, each test set should consist of the retweets from the directly consecutive set. Thus, our test sets are:

- TestSet [1]: retweets from day1–day6
- TestSet [2]: retweets from day7–day12
- TestSet [3]: retweets from day13–day18
- TestSet [4]: retweets from day19–day24
- TestSet [5]: retweets from day25–day31

The fourfold forward chaining evaluation consists of the following steps:

- Fold 1: TrainSet [1], TestSet [2]
- Fold 2: TrainSet [1, 2], TestSet [3]
- Fold 3: TrainSet [1, 2 and 3], TestSet [4]
- Fold 4: TrainSet [1, 2, 3 and 4], TestSet [5]

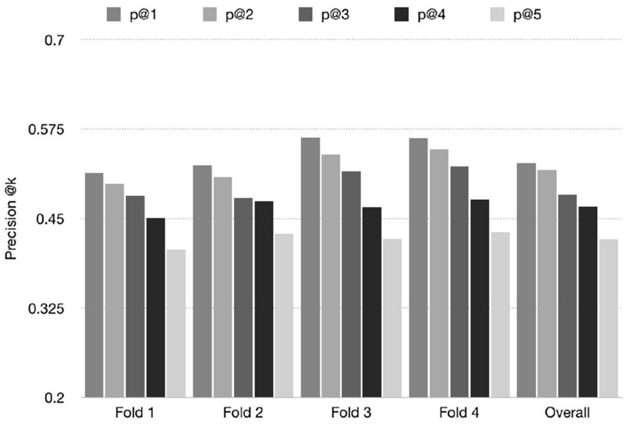


Fig. 7 Precision-at-k

For each training set we constructed a new database table. These tables contain the streaming data in a reorganized way. Rows represent users, who were active during the set's time period. Each row contains a user-id, a list of the tweets he posted during this time. Our Tweet Representation unit adds an extra column containing the user interest profile (sorted ToI list) based on the tweets he posted during that period. For our experiment, we used only users that have posted three or more tweets (and/or retweets) during this month. Finally, our train sets contain 200,039 unique users and their interest profiles. Each test set is a separate table where each line represents a retweet that is posted in chronological order, because we wanted to simulate the streaming tweets entering our recommender. Each line contains the tweet-id, the user-id that posted this retweet and a ToI assigned by the Tweet Representation module.

The results of our experiment from k = 1 to 5 are shown in Fig. 7. Finally, we conducted an overall evaluation, where the training set contains all users' tweets (no retweets) and the test set contains all users' retweets.

As we can observe, our method reaches an average precision score of 48,4% while the overall accuracy is 98.9%. This means that our recommender can successfully retrieve the interesting (retweeted) and not interesting tweets (true positive and true negative results), but still recommends some tweets that were not retweeted by the user (false positives). This could be caused, besides our recommender's weaknesses, since not all retweets from the test set are seen by all users, because they are not following every other user in the network. Hence, they cannot have retweeted something they could not have seen in their timeline, even if they are interested in it.

We can also observe that our method shows even better results in terms of efficiency, when tested on a larger dataset. This could happen because, the new dataset is more consistent regarding time since the tweets and retweets were published within one month. Furthermore, it concerns

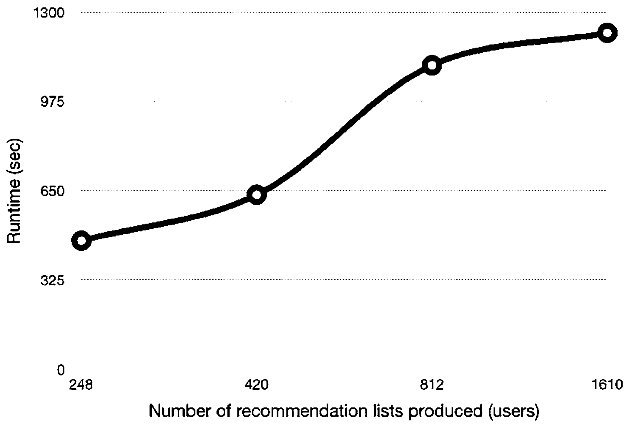


Fig. 8 Recommender runtime

many users increasing the possibility of having users that are followers-followees with each other in it (the possibility of retweeting content from one's followees is higher than from the random stream). Finally, the preprocessing stage that was described in Sect. 5.3.1 along with the modification in the KG, have enabled the method to provide even better recommendations.

5.3.3 Runtime testing

Time scalability is very important in recommender systems, especially if they are designed for real-time use. Therefore, we tested our method for the first 1610 users to estimate our methods runtime scalability. Specifically, we measured the time needed for user profiling, tweet filtering and top-5 ranking for 0 to 1610 users. As shown in Fig. 8, our method can provide quick recommendations, revealing an opportunity to develop an online real-time tweet recommender application.

6 Conclusions

In this work, we presented a content based method for personalized tweet and followee recommendation. This method is based on conceptual relations between users' topics of interest (ToIs). The method takes advantage of the objective relation between the ToIs of a user and a Knowledge Graph. We have shown that the recommendations based on these relations can reduce the effects of over-recommendation and over-specialization problems, and can be used to capture the dynamic change of these interests too in a scalable way. The efficiency of the proposed method outperforms in many cases the previous state-of-the art works, and exhibits even better results both in terms of precision and time scalability when tested on a larger dataset.

References

Adomavicius G, Tuzhilin A (2005) Toward the next generation of recommender systems: a survey of the state-of-the-art and possible extensions. IEEE Trans Knowl Data Eng 17(6):734–749

Alonso O et al (2010) “Detecting uninteresting content in text streams”. SIGIR crowdsourcing for search evaluation workshop

Armentano MG, Godoy D, Amandi A (2012) Topology-based recommendation of users in micro-blogging communities. J Comput Sci Technol 27(3):624–634

Balabanović M, Shoham Y (1997) Fab: content-based, collaborative recommendation. Commun ACM 40(3):66–72

Bhattacharya P et al (2014) Inferring user interests in the twitter social network. In: Proceedings of the 8th ACM Conference on recommender systems. ACM

Chen K et al (2012) Collaborative personalized tweet recommendation. In: Proceedings of the 35th international ACM SIGIR conference on Research and development in information retrieval. ACM

Diaz-Aviles, E et al (2012) What is happening right now... that interests me?: online topic discovery and recommendation in twitter. In: Proceedings of the 21st ACM international conference on Information and knowledge management. ACM

Duan Y et al (2010) An empirical study on learning to rank of tweets. In: Proceedings of the 23rd International Conference on Computational Linguistics. Association for Computational Linguistics

Ehrlinger L, Wöß W (2016) Towards a Definition of Knowledge Graphs. SEMANTiCS

Elmongui HG et al (2015) TRUPI: twitter recommendation based on users' personal interests. Computational linguistics and intelligent text processing. Springer International Publishing, New York, pp 272–284

Ferrucci D, Brown E, Chu-Carroll J, Fan J, Gondek D, Kalyanpur AA, Lally A, Murdock JW, Nyberg E, Prager J et al (2010) Building Watson: an overview of the DeepQA project. AI magazine 31(3):59–79

Gabrilovich E, Markovitch S (2007) Computing semantic relatedness using wikipedia-based explicit semantic analysis. IJCAI 7

Gilbert EN, Pollak HO (1968) “Steiner minimal trees”. SIAM J Appl Math 16(1):1–29

Hannon J, Bennett M, Smyth B (2010) Recommending twitter users to follow using content and collaborative filtering approaches. In: Proceedings of the fourth ACM conference on recommender systems. ACM

Hong L, Doumith AS, Davison BD (2013) Co-factorization machines: modeling user interests and predicting individual decisions in twitter. In: Proceedings of the sixth ACM international conference on Web search and data mining. ACM

IJntema W et al (2010) Ontology-based news recommendation. In: Proceedings of the 2010 EDBT/ICDT workshops. ACM

Kawamae N (2011) Trend analysis model: trend consists of temporal words, topics, and timestamps. In: Proceedings of the fourth ACM international conference on web search and data mining. ACM

Kim Y, Shim K (2011) Twitobi: A recommendation system for twitter using probabilistic modeling.” Data Mining (ICDM), 2011 IEEE 11th International Conference on. IEEE

Koren Y (2008) Factorization meets the neighborhood: a multifaceted collaborative filtering model. In: Proceedings of the 14th ACM SIGKDD international conference on knowledge discovery and data mining. ACM

Lauw HW, Ntoulas A, Kenthapadi K (2010) Estimating the quality of postings in the real-time web. In: Proceeding of SSM conference

Liu Y et al (2016) A User Adaptive Model for Followee Recommendation on Twitter. International Conference on Computer

Processing of Oriental Languages. Springer International Publishing

Lu C, Lam W, Zhang Y (2012) Twitter user modeling and tweets recommendation based on wikipedia concept graph. Workshops at the twenty-sixth AAAI Conference on artificial intelligence

Mehlhorn K(1988) A faster approximation algorithm for the Steiner problem in graphs. Inf Process Lett 27(3):125–128

Naveed N et al (2011) Bad news travel fast: a content-based analysis of interestingness on twitter. In: Proceedings of the 3rd International Web Science Conference. ACM

Pazzani MJ, Muramatsu J, Billsus D (1996) Syskill and Webert: identifying interesting web sites. AAAI/IAAI, Vol 1

Pennacchiotti M et al (2012) Making your interests follow you on twitter. In: Proceedings of the 21st ACM international conference on information and knowledge management. ACM

Pla-Karidi D (2016) From user graph to topics graph: towards twitter followee recommendation based on knowledge graphs. Data Engineering workshops (ICDEW), 2016 IEEE 32nd international conference on. IEEE

Pla-Karidi D, Stavrakas Y, Vassiliou Y (2016) A personalized tweet recommendation approach based on concept graphs. The 13th IEEE International conference on ubiquitous intelligence and computing (UIC)

Ramage D, Dumais ST, Liebling DJ (2010) Characterizing microblogs with topic models. ICWSM 10:1–1

Rendle S (2011) Factorization machines with libFM. ACM Trans Intell Syst Technol (TIST) 3(3):57

bib id="bib2">Rendle S, Schmidt-Thieme L (2008) Online-updating regularized kernel matrix factorization models for large-scale recommender systems. In: Proceedings of the 2008 ACM conference on recommender systems. ACM

Rodríguez FM, Torres LM, Garza SE (2016) Followee recommendation in Twitter using fuzzy link prediction. Expert Syst 33(4):349–361

Romero DM et al (2011) Influence and passivity in social media. Machine learning and knowledge discovery in databases. Springer, Berlin, pp 18–33

Sarwar B et al (2001) Item-based collaborative filtering recommendation algorithms. In: Proceedings of the 10th international conference on world wide web. ACM

Sarwar BM et al (2002) Recommender systems for large-scale e-commerce: Scalable neighborhood formation using clustering. In: Proceedings of the fifth international conference on computer and information technology. Vol 1

Schafer JH et al (2007) Collaborative filtering recommender systems. The adaptive web. Springer, Berlin, pp 291–324

Schein AI et al (2002) Methods and metrics for cold-start recommendations. In: Proceedings of the 25th annual international ACM SIGIR conference on research and development in information retrieval. ACM

Sharma A et al (2016) GraphJet: real-time content recommendations at twitter. Proc VLDB Endow 9(13):1281–1292

Shi Y, Larson M, Hanjalic A (2009) Exploiting user similarity based on rated-item pools for improved user-based collaborative filtering. In: Proceedings of the third ACM conference on recommender systems. ACM

Su X, Khoshgoftaar TM (2009) A survey of collaborative filtering techniques. Adv Artif Intell 2009:4

Subercaze J, Gravier C, Laforest F (2016) Real-time, scalable, content-based Twitter users recommendation. Web Intelligence 14(1):17–29. doi:10.3233/WEB-160329

Tajbakhsh MS, Bagherzadeh J (2016) Microblogging Hash Tag Recommendation System Based on Semantic TF-IDF: Twitter use case. Future internet of things and cloud workshops (FiCloudW), IEEE international conference on. IEEE

Uysal I, Croft WB (2011) User oriented tweet ranking: a filtering approach to microblogs. In: Proceedings of the 20th ACM international conference on Information and knowledge management. ACM

Wang M, Ma J(2016) A novel recommendation approach based on users' weighted trust relations and the rating similarities. Soft Comput 20(10):3981–3990

Wang X, McCallum A (2006) Topics over time: a non-Markov continuous-time model of topical trends. In: Proceedings of the 12th ACM SIGKDD international conference on Knowledge discovery and data mining.ACM

Weng J et al (2010) Twitterrank: finding topic-sensitive influential twitterers. In: Proceedings of the third ACM international conference on web search and data mining. ACM,

Yang, MC, Rim HC (2014) Identifying interesting Twitter contents using topical analysis. Expert Syst Appl 41(9):4330–4336

Yang MC et al (2012) Finding interesting posts in twitter based on retweet graph analysis. In: Proceedings of the 35th international ACM SIGIR conference on research and development in information retrieval. ACM

Yigit M, Bilal E Karahoca A (2015) Extended topology based recommendation system for unidirectional social networks. Expert Syst Appl 42(7):3653–3661

Zhao F et al (2016) A personalized hashtag recommendation approach using LDA-based topic model in microblog environment. Future Generation Comput Syst 65:196–206