A Framework for Validating the Merit of Properties that Predict the Influence of a Twitter user

Stefan Räbiger[a,1], Myra Spiliopoulou[b,*]

[a] *Faculty of Engineering and Natural Sciences, Sabanci University, Orhanli, Tuzla, 34956 Istanbul, Turkey*
[b] *Faculty of Computer Science, Otto-von-Guericke University Magdeburg, Universitätsplatz 2, D-39106 Magdeburg, Germany*

Abstract

What characterizes an influential user? While there is much research on finding the concrete influential members of a social network, there are less findings about the properties distinguishing between an influential and a non-influential user. A major challenge is the absence of a ground truth, on which supervised learning can be performed. In this study, we propose a complete framework for supervised separation between influential and non-influential users in a social network. The first component of our framework, the *InfluenceLearner*, extracts a Relation Graph and an Interaction Graph from a social network, computes network properties from them and then uses them for supervised learning. The second component of our framework, the *SNAnnotator*, serves for the establishment of a ground truth through manual annotation of tweets and users: it contains a crawling mechanism that produces a batch of tweets to be annotated offline, as well as an interactive interface that the annotators can use to acquire additional information about the users and the tweets. On this basis, we have created a ground truth dataset of Twitter users, upon which we study which properties characterize the influential ones. Our findings show that there are predictive properties associated with the activity level of users and their involvement in communities, but also that writing influential tweets is not a prerequisite for being an influential user.

Keywords: Identification of Influential Users, Properties of Influential Users, Learning of Influence, Community Mining, Influential Users in Twitter, Influential Users, Influential Tweets, Annotation of Tweets, Mining, Twitter

1. Introduction

The propagation of influence in online social networks has been subject of extensive research, ever since the seminal works of (Domingos & Richardson, 2001; Kempe et al., 2003) on influence propagation in social graphs. Whilst there is a substantial amount of work in identifying influential social graph participants (also known as influentials), there are less findings on identifying the properties which distinguish between influential and non-influential nodes. In this study, we investigate to what extend supervised analysis of a social graph can reveal distinctive properties of influential users. We propose a framework that extracts user attributes that have the potential of predicting a user's influence power, and we use this framework to separate between influential and non-influential Twitter users.

Modeling the spread of influence in social networks is an intensively studied task. Contributions include diffusion models that describe influence propagation, and theoretical findings on how well such a diffusion model can describe reality (Kempe et al., 2003). A major application area for such models is viral marketing, because, as pointed out by Barbieri et al. (Barbieri et al., 2013): "...individuals tend to adopt the behavior of their social peers". They continue with the important statement that "cascades happen first locally, within close-knit communities, and become global "viral" phenomena only when they are able

*Corresponding author
Email address: myra@iti.cs.uni-magdeburg.de (Myra Spiliopoulou)

[1] Work done while being a master student at the Faculty of Computer Science, Otto-von-Guericke Univ. Magdeburg

cross the boundaries of these densely connected clusters of people." (Barbieri et al., 2013). These studies focus on modelling the spread of influence and on finding the persons that have the most influence. But what are the *properties* characterizing these persons? Are there attributes on the activities and writings of a user that indicate her influence?

In this study, we express the problem of identifying influential users as a classification task, and aim to identify the characteristics of such users. Labeled datasets for this task are rare (Bigonha et al., 2012). Moreover, the authors of (Bigonha et al., 2012) are not allowed to give access to the tweet contents due to the terms of service of Twitter. To verify the assumption that tweet content can indicate whether a tweet author is influential, we propose as part of our framework a workflow for the offline creation of a labeled set of tweets and of users who wrote these tweets. We use this workflow to create a labeled dataset that we use for our experiments.

The contribution of our work is then twofold: we propose a supervised learning method and a set of properties for distinguishing between influential and non-influential social graph members, and we also propose a workflow for acquiring labeled data for supervised learning. We study Twitter in our work, because Twitter is a representative of the *who listens to whom* attitude suggested in (Bakshy et al., 2011). However, the workflow we propose allows for building datasets on any social platform to learn a dedicated model on it. As a further by-result of our approach, we make the dataset of our first run of our framework available to other scholars (see section 4 on data access).

The remainder of this paper is structured as follows: in section 2 we discuss related work for detecting influentials in Twitter. In section 3, we introduce our framework and present its learning component that extracts attributes from the social graph to characterize the users of postings, as well as the annotation component for acquiring a labeled dataset. We report on our experiments in section 4. The last section concludes our study.

2. Related Work

There exist various heuristics - based on mentions, replies, followers and followees - that rank users according to their influence (Anger & Kittl, 2011; King et al., 2013; Sun & Ng, 2013; Razis & Anagnostopoulos, 2014). Other heuristics focus on aspects like tweet quality (Kong & Feng, 2011) or utilize alpha centrality which is related to Eigenvector centrality (Overbey et al., 2013). Zhao et al. devise a new measure for influence based on sentiment (Zhao et al., 2014). Sun et al. pursue a more sophisticated approach by building a user and tweet graph to identify influential users (Sun & Ng, 2013). King et al. (King et al., 2013) devise the t-index that denotes the number of times a user's unique tweet has been retweeted to compare the influences users exert on the same topic. Razis et al. (Razis & Anagnostopoulos, 2014) combine the ratio of a user's followers and followees and the ratio of tweets written in a certain period of time into an influence metric. Similarly, Bigonha et al. combine users' sentiment, tweet quality and centrality to obtain an aggregated influence score (Bigonha et al., 2012). In terms of supervised learning, Chai et al. follow a similar approach, but combine attributes related to four categories - activity, centrality, quality and reputation (Chai et al., 2013). Liu et al. extract several attributes known from literature to train an SVM (Liu et al., 2014) and Xiao et al. use attributes related to three different categories in order to find influential users (Xiao et al., 2013). The problem with these approaches is that there is no ground truth on what people consider as influential user to evaluate the approaches on. We address this aspect in our work by building a ground truth on people's perception of influence.

There are also commercial services (including Klout [2], PeerIndex[3], Kred [4]) that assign influence scores to users. However, each such service uses its own, internal/proprietary definition of the term influence. Campo-Ávila et al. (Campo-Ávila et al., 2013) attempt to reverse engineer two of these algorithms (Klout and PeerIndex) and to identify the factors used in these internal definitions of influence. Our intention in this work is not to provide yet another definition of influence (which might be subject of some controversy), but to identify factors that are associated with influence, *when human annotators decide who is influential and who is not*, keeping in mind that humans, in contrast to services, do not have a rigid definition of whom they consider influential.

In a different thread of research, Barbieri et al. study the spread of information in a social network, and point out that cascades are local phe-

[2] www.klout.com (10-30-2014)
[3] www.peerindex.com (10-30-2014)
[4] www.kred.com (10-30-2014)

nomena (Barbieri et al., 2013) that manifest themselves inside close-knit communities; only some of them cross the community borders through nodes that are part of both communities. Wang et al. also assume that influentials need to occur in every community to propagate information across the network (Wang et al., 2010). Inspired by these findings on the role of communities for information propagation, we also take the community structure of the graph into account. However, in contrast to (Wang et al., 2010; Barbieri et al., 2013), our objective is to find the characteristics of influential users and not to point to those users who are influential.

Summarizing, our work differs from other literature on influence in following aspects. Differently from (Bigonha et al., 2012; Anger & Kittl, 2011; King et al., 2013; Sun & Ng, 2013; Razis & Anagnostopoulos, 2014; Kong & Feng, 2011; Overbey et al., 2013; Zhao et al., 2014) and similarly to (Quercia et al., 2011), we do not attempt to find influential users but rather identify the characteristics that separate between influential and non-influential users. To this purpose, we derive properties that reflect social activity, and use them in supervised learning. The learner and the set of properties constitute our first contribution. This set of properties is larger and more elaborate than in (Bigonha et al., 2012; Chai et al., 2013; Liu et al., 2014; Xiao et al., 2013) and, moreover, it is accompanied by an elaborate approach on assessing the ground truth. Indeed, unlike Klout, PeerIndex, Kred and (Bigonha et al., 2012; Cha et al., 2010), we do not provide yet another definition of influence, nor try to reengineer existing definitions, but we rather cover the non-crisp, subjective perception of influence that people have. To this purpose, our approach encompasses a mechanism for the creation of a ground-truth dataset (a seed) of influential and non-influential users through human annotators. This mechanism is our second contribution.

3. Framework

Our framework for identifying characteristics of influentials has two components: the *SNAnnotator* and the *InfluenceLearner*. The former collects data from Twitter to establish a ground truth, which serves as input for the *InfluenceLearner*. The latter is responsible for turning a dataset into graphs, extracting attributes and learning a meaningful model.

3.1. SNAnnotator

Our *SNAnnotator* describes the process of collecting a dataset regarding a specific topic in batch mode, manually labeling and preparing it for attribute extraction with our *InfluenceLearner*.

3.1.1. Offline Dataset Crawl

We collect tweets during a certain period of time and retrieve their authors thereafter. *SNAnnotator* can also operate on multiple topics simultaneously, because it uses hashtags to identify the tweets corresponding to a topic. However, as Cha et al. point out (Cha et al., 2010), a user's influence may vary over topics and change with time. Therefore, we concentrate on learning influence towards single topics. We collect tweets containing the set of predefined hashtags using the Twitter Streaming API[5]. This means only the latest tweets of users related to the topic are collected. Once this process is completed, the metadata of the respective users are obtained through the Twitter Rest API[6]. Since Twitter permits only a limited number of API calls per hour, we decided to collect users afterwards in one go. This helps to avoid potential side-effects. For instance, if we collected multiple tweets of a user, her metadata would have been different all the times, at least in the number of posted tweets. To avoid discrepancies, *SNAnnotator* contains an interface, through which the human annotators can inspect the activities of a user in Twitter also outside the time horizon of the crawl. This allows for more informed decisions on whether a user is influential or not.

In terms of tweets, we only retrieve those written in English to simplify the annotation process for our human annotators. For this purpose, we collect only tweets from users who have chosen the respective language in their Twitter front end, which is detected based on the metadata. This filtering step removes most of the unrelated users and tweets, but not all of them. The remaining unrelated users and tweets are removed during and after the Annotation Process described hereafter.

3.1.2. Annotation Process

We propose two types of labels: we distinguish between tweet and user labels and take into account

[5]https://dev.twitter.com/docs/streaming-apis (10-30-2014)

[6]https://dev.twitter.com/docs/api/1.1 (10-30-2014)

that both types have to be labeled independently from each other. Hence, we develop an offline annotation tool, with which n independent human annotators assign to each tweet and user a label denoting whether the user (resp. the tweet) is influential or not. This tool also takes the independence of labels into account by selecting the next user or tweet to be presented to annotators in a random order.

The decision whether a user (or tweet) is influential is not binary: we rather define five classes for users and tweets, namely "influential", "probably influential", "probably non-influential", "non-influential", "strange". The reason for the first four labels is that some users or tweets seem promising but are not influential yet. The last class "strange" is to be used for tweets written in a language other than English, and tweets that are incomprehensible or are not available online anymore. Twitter users may be classified as "strange" only if their account got suspended/deleted, or if most of their tweets are incomprehensible or not written in English.

We do not provide human annotators with an explicit definition of influence as to when to assign a tweet/user which of the labels. We refrained from defining "influence" explicitly since we want to employ supervised learning to discover characteristic properties of influentials. If we provided a definition, we could have extracted related attributes to build a model yielding 100% accuracy. But this process would not reflect human perception of influence. Thus, we allow human annotators to decide about "influence" for themselves. This procedure introduces noise among the labels due to the subjectivity of human annotators regarding "influence". We try to lessen this problem by combining two strategies: preparing the human annotators with background information and by increasing the number of annotators. Before the annotation process begins, we would like to make sure that all annotators have a basic idea of the procedure. Thus, we prepared an annotation protocol[7], specifying background information about the topic, responsibilities of the individual annotators and suggestions/examples as to how they might define influence for themselves. In the second step, we require each tweet and user to be labeled multiple times, but every time by a different annotator. This strategy helps to decrease the noise across labels and also allows us to assume the representativeness

[7]contact the first author to obtain it

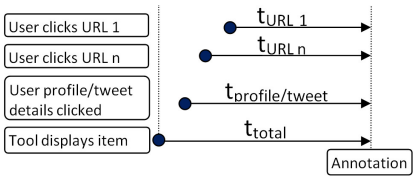

Figure 1: Additional metadata being stored during the annotation of a user and tweet.

of the labels assigned by the human annotators. Although this assumption is difficult to verify, in case of doubt the number of annotators can be increased to get more reliable responses. Note that increasing the number of annotators does not affect our general methodology.

Before assigning the final labels, we remove all tweets and users that were labeled "strange" at least by one annotators. In the last step, we determine the final labels with the help of the median, since the remaining four classes are represented internally as ordinal values. In the remainder of this study, we always refer to those classes, since no more "strange" users and tweets exist in the dataset.

While annotating an item, additional metadata are recorded, which is depicted in Figure 1. The total time t_{total} starts when the human annotator loads a tweet or a user's entry into the *SNAnnotator*; t_{total} stops as soon as the annotator assigns a label. Furthermore, we store how much time each human annotator spends browsing a user's profile ($t_{profile/tweet}$) and for how long a linked URL is visited (t_{URL}). These data allow for learning the labeling behavior of human annotators over time, but are not utilized in this study.

3.1.3. Merging and Cleaning the Annotated Set

Our *SNAnnotator* determines the ground truth for user and tweet labels by selecting the median of the assigned labels by human annotators. Users or tweets that were marked to be removed, are excluded from the dataset, since they could not be labeled by all responsible human annotators a sufficient number of times. All tweets which were written by a deleted user are removed as well. Likewise, users without any more valid tweets are discarded, too. This resulting dataset serves as input for the *InfluenceLearner*.

3.2. InfluenceLearner

Our *InfluenceLearner* learns the characteristic attributes of influential documents and users in a supervised way. To this purpose, we process the original social network and derive from it a set of attributes for the users in the network. Each attribute belongs to either (I) *community structure* which is based on the detected communities, (II) *activity* which relates to how much users involve their followers, (III) *quality* that considers the quality of the tweets, and (IV) *centrality* which incorporates the position of users in the graph. Then, we use the graph data (including tweet labels but not user labels) of the dataset prepared with *SNAnnotator* to learn models on these data, and select the attributes that contribute most to classifier quality. In the last step, we validate those models on the manually labeled users from the dataset of the *SNAnnotator* using two experiments. Since we want to discover predictive attributes, we compare the performance of the classifiers trained on subsets of all attributes with the classifiers learned on the entire set of attributes. Due to the nature of the labels of our dataset, we also analyze whether posting influential tweets is a characteristic property of influential users or not. The experimental setup is described in detail in section 4.

The workflow of our *InfluenceLearner* is as follows. We first derive directed graphs from the original social network reflected in the crawled data, the graph of relations between users and the graph of interactions between them; this step is described in 3.2.1, where we also define the concepts of *Relation Graph* and *Interaction Graph*. Next, we transform each of these two directed graphs into a *line graph*, as explained in 3.2.2. We need these line graphs for the construction of communities that may overlap. We opt for the computation of overlapping communities instead of disjunct ones, because literature (Barbieri et al., 2013) has shown that users tend to participate in more than one community simultaneously. Community computation is described in 3.2.3. From the graphs and communities we derive a list of attributes, which we organise into categories: these attributes will be used as dimension space for learning to separate between influential and non-influential users. The computation of the attributes is described in 3.2.4. On Figure 2, we show this sequence of steps for a tiny example graph.

3.2.1. Building Relation and Interaction Graphs

Similarly to (Bigonha et al., 2012), we distinguish between an Interaction Graph IG and a Relation Graph RG.

Relation Graph. Given the set of users U in a social network, and given that $u_1, u_2 \in U$, we draw an edge (u_1, u_2) iff u_1 is follower of u_2: we use the set of those edges R to build the Relation Graph $RG(U, R)$.

Interaction Graph. Given the set of users U in a social network, and given that $u_1, u_2 \in U$, we draw an edge (u_1, u_2) iff u_1 has retweeted, replied to a tweet of u_2 or otherwise mentioned u_2: we use the set of those edges I to build the Interaction Graph $IG(U, I)$.

The decision to use two graphs goes back to (Huberman et al., 2008), who note that the relationship between followers and followees results in a dense graph; but since individuals truly interact only with a few carefully selected peers, detecting influence is easier in the resulting sparse graph of interactions. In our perception of the original social network and in the two graphs we derive from it, potentially influential nodes are the targets of edges, in the sense that the likelihood of a node being influential intuitively increases with the number of incoming edges to it. It is important to note that in our proposed framework only the largest connected component of IG and RG is considered, because some of the attributes to be extracted, particularly attributes related to *community structure* and *centrality*, are computed per connected component, see Table 1. In the remainder of this work DG denotes an unweighted, directed graph like RG or IG.

3.2.2. Deriving Line Graphs

Our *InfluenceLearner* begins with the method introduced by Evans et al. (Evans & Lambiotte, 2009) for detecting overlapping communities, which occur frequently in social networks. Their main idea is to cluster the social network on the basis of edges instead of nodes, since nodes can belong to more than one community, whereas edges indicate a one-to-one relationship between two entities and therefore no overlaps occur. A node is then assigned to all the communities its adjacent edges belong to. To this purpose, Evans et al. transform a graph G into a line graph $L(G)$, whereby nodes from G are represented as edges in $L(G)$, whereas edges from

G are transformed into nodes in $L(G)$. After a successful transformation any community detection algorithm can be applied to $L(G)$ in order to unveil its community structure. But we need to adapt (Evans & Lambiotte, 2009) as first step, because the original implementation requires too much memory for large graphs. Since the resulting line graph of DG, L(DG), is dense, we reduce memory consumption in our modification.

Our modification [8] takes as input the set of users U, incoming and outgoing edges of DG and transforms DG into its corresponding line graph L(DG) by exploiting the following observation: a node s from DG may only exist in L(DG) if it "connects" two edges in DG. "Connect" in this sense means that one edge is pointing toward s, whereas the other one is going outward from s. In other words, s must have at least one incoming and one outgoing edge in DG. These two edges and s can be transformed into a valid L(DG), i.e. both edges are transformed into nodes which are connected by s in Ł(DG), where s points from the incoming toward the outgoing edge. Figure 3 visualizes the notion of "connected" edges. For instance, in step 1 in DG the red and blue edge are "connected" by node A, as one enters (red) and the other one (blue) leaves A. Thus, we know that A will be present in L(DG), since it "connects" at least one pair of nodes in L(DG), namely the red and blue node, which are the transformed incoming (red) and outgoing (blue) edge from DG. More formally speaking, an edge pair $(i, j), (k, l)$ is "connected" by s if and only if $j = s$ and $k = s$ hold, where $\{i, j, k, l, s\} \in U$. If no self-links are allowed, additionally $i \neq s$ and $l \neq s$ must hold.

We now describe our pseudocode (Algorithm 1). It iterates over all users in DG (line 4) and assumes that the current user s "connects" an edge pair. This holds only true in case s has at least one incoming and one outgoing edge in DG (line 6). Otherwise, s will not be present in L(DG). If s exists in L(DG) as an edge connecting its incoming and outgoing edges from DG, the edge will point from the incoming to the outgoing edge. So far, only one incoming and outgoing edge of s were checked. But all possible combinations of incoming and outgoing edges involving s must be transformed into nodes in Ł(DG) (line 8-9), although it might happen that certain combinations are deemed invalid depending on the scenario (line 10), e.g. self-links may be allowed or not. The weight w of the resulting edge in L(DG) is computed according to $w = 1/\text{out-degree}(s)$ (line 11) and the edge is stored in memory until either the memory limit is exceeded (line 15-16) or the transformation of DG into L(DG) has finished (line 21-22) successfully. In the former case, all edges of L(DG) are written to a file, and the memory is freed again; in the latter case, all edges stored in memory are written to this file as well (line 21-22). Figure 3 illustrates a stepwise example of how our modification transforms DG into L(DG).

Algorithm 1 LineGraphCreator.

```
 1: Input: Users, outgoing edges OE, incoming
    edges IE of unweighted, directed graph DG and
    maximal RAM consumption.
 2: Output: Line graph L(DG).
 3:
 4: for each user in DG do
 5:   s ← user
 6:   if user has outgoing and incoming edges
      then
 7:     // Edge pair: (in, s), (s, out)
 8:     for each in ∈ IE of user do
 9:       for each out ∈ OE of user do
10:         if (in, out) is allowed then
11:           weight w ← 1/out-degree(s)
12:           edge in L(DG) ← (in, out, w)
13:         end if
14:       end for
15:       if allocated RAM is consumed then
16:         store computed edges in file
17:       end if
18:     end for
19:   end if
20: end for
21: if edges in L(DG) exist then
22:   store computed edges in file
23: end if
```

3.2.3. Community Detection on the Line Graphs

For discovering the communities without overlap in L(DG), we choose to employ Louvain method (Blondel et al., 2008), for it is particularly fast and therefore suitable for our large L(RG). We note that influence attributes (see section 3.2.4) are extracted from DG, as overlap needs to be taken into account. In order to identify for each node in DG its (multiple) community memberships, we utilize a mapping

[8] https://github.com/fensta/linegraphcreator (10-30-2014)

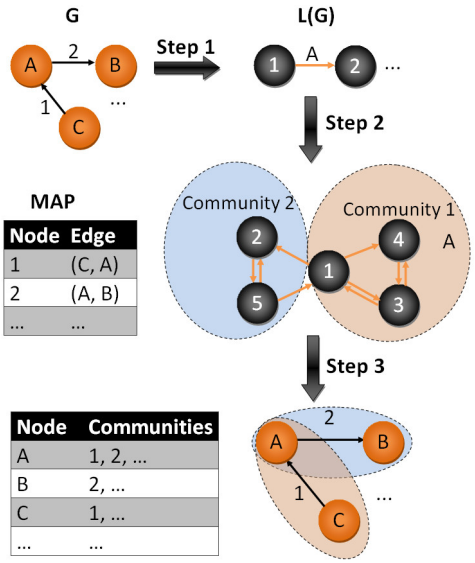

Figure 2: Illustration how we identify overlapping communities in our graphs. In step 1 DG is transformed into L(DG) using Evans' algorithm. In this step MAP is created too: each node in L(DG) is mapped to its respective edge from DG. For instance, node 1 represents edge (C, A) in MAP. In step 2 the Louvain method clusters the nodes of L(DG) (= edges of DG) into disjunct communities. The overlapping communities are then computed in step 3 with MAP: each node of DG is assigned to all communities its adjacent edges belong to. For example, we know node 1 belongs to community 1. This implies the nodes adjacent to (C, A) belong to the same community as this edge. Hence, A and C are part of community 1. Likewise, we see that node 2 belongs to community 2 in L(DG). Similarly, A and B being adjacent to (A, B) belong to community 2. Now A belongs to the two communities 1 and 2 at the same time.

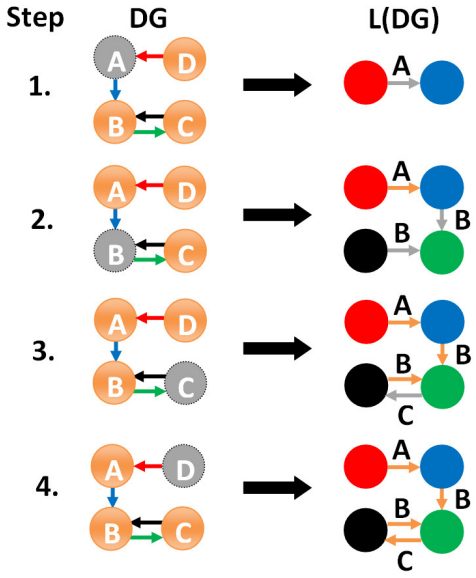

Figure 3: Illustration of the stepwise execution of LineGraphCreator to transform DG (left) into L(DG) (right). The edges of DG have the same color as the resulting nodes in L(DG), as edges in DG turn into nodes in L(DG). The nodes' colors in DG and resulting edges in L(DG) share identical colors for the same reason since nodes from DG are transformed into edges in L(DG). We omit to display edge weights in L(DG) due to readability. But they are computed according to 1/out-degree(n), where n is the node currently being processed (indicated by the grey color).
In step 1, node A is processed. It has an incoming (red) and an outgoing edge (blue), so it exists in L(DG). More precisely, in L(DG) both edges are turned into nodes and are connected by A which is now converted into an edge. Its direction depends on the involved edges in DG: it points from the incoming to the outgoing edge. The edge weight would be 1/out-degree(A) = 1. Steps 2 and 3 follow the same reasoning. Since node B has two incoming edges, two edges are added in L(DG) in a single step. Only in step 4 node D is skipped as it does not have any incoming edges. The algorithm has now processed all nodes in DG.

MAP, which we create while transforming DG into L(DG). In MAP, each edge e from DG, existing in L(DG) as a node n, is assigned to n. Hence, each entry in MAP is a pair $< n, e >$. Figure 2 illustrates the whole process of discovering overlapping communities including the use of MAP.

3.2.4. Computation of the Influence Attributes

The attributes we extract are based on four categories:

1. *Community*: We extract attributes related to community structure because Barbieri et al. state in (Barbieri et al., 2013) that communities connected more frequently to the outside world foster the spreading of messages in the network, and therefore enable word-of-mouth propagation.
2. *Activity*: In (Cha et al., 2010) the authors report that influentials gain their importance over time through constant involvement of their followers. Thus, we assume influentials are active and engage their followers into interactions.
3. *Quality*: In (Kong & Feng, 2011) the contributors explain that tweets with high quality characterize influentials.
4. *Centrality*: In (Bigonha et al., 2012) the authors state that well-connectedness in the network is typical for influentials. Therefore, influentials are to some extend "central" and possess more power than non-influentials.

Hereafter, we introduce the attributes we propose for each of these four categories (cf. Table 1). All

attributes are computed for each user separately, once for the Relation Graph and once for the Interaction Graph.

No.	Attribute name	Category
1	Inter-community edges	Community
2	# Connected communities	Community
3	# Edges from influentials	Community
4	# Overlaps	Community
5	Potential audience	Community
6	Influence	Activity
7	# Interactions	Activity
8	# Tweets	Activity
9	TFF (Bigonha et al., 2012)	Quality
10	User readability (Bigonha et al., 2012)	Quality
11	User quality (Kong & Feng, 2011)	Quality
12	In-degree (de Nooy et al., 2005)	Centrality
13	Closeness (de Nooy et al., 2005)	Centrality
14	Vertex betweenness (de Nooy et al., 2005)	Centrality
15	Eigenvector centrality (de Nooy et al., 2005)	Centrality
16	Edge betweenness (Girvan & Newman, 2002)	Centrality

Table 1: Attributes to be extracted per user.

Community

This category encompasses following attributes:

1. *Inter-community edges* measures the overall strength of a user's connections to adjacent communities in *DG*. This attribute is calculated as the sum over all inter-community edges with weight w being taken from L(*DG*) per user u_1. If u_1 has an outgoing edge to a user from a different community, w of the edge connecting both users in L(*DG*) has to be added. We note that in practice w is computed as $w = 1/\text{out-degree}(u_1)$ in *DG* like during the creation of the line graph because if u_1 exists in *MAP*, we know that she is present in L(*DG*), too. For this attribute multiple edges to the same community contribute to the sum.
2. *# Connected communities* counts how often the user is connected to other communities in DG. We calculate it like Inter-community edges, but now each outgoing edge contributes equally to the end result, favoring in *RG* users with many followees and in *IG* users with many interactions.
3. *# Edges from influentials* counts how often influentials have outgoing edges to the current user in *DG* as this direction is more meaningful. For instance, in *RG* it is easy for a user to follow an influential one, whereas the opposite case indicates that a non-influential user is also "interesting" to a certain extent.
4. *# Overlaps* counts to how many different communities a user belongs to in *DG*. Note that it is never larger than *# Connected communities*. For example, consider a user who participates in two communities X and Y; then *# Overlaps* = 2 for this user. But if the user is linked to 8 users of X and 3 users of Y, then *# Connected communities* = 11 for this user.
5. *Potential audience* measures how many individuals the user can reach. This is the sum of the sizes of the communities which the user is member of normalized by the total number of users in *DG*.

Activity

This category encompasses following attributes:

1. *# Interactions* regarding the topic counts all replies, retweets and mentions among collected users using the available metadata.
2. *# Tweets* regarding the topic, which was selected using appropriate hashtags before the crawl, is the number of collected tweets by this user.
3. *Influence* selects the median of all tweet labels collected from that person. A tweet label is an integer value between one and four. Higher values imply more influential tweets. As users rarely publish influential tweets, averaging the labels is not as effective as choosing the median. We also experimented with selecting the third quartile instead, but the quality of the final models remained unaffected.

Quality & Centrality

These categories encompass the following attributes:

1. *TFF* (Twitter Followee Follower ratio) is the ratio of a user's followers over her followees. If a user has more followers than users she follows, it suggests that she is to some degree "interesting". Thus, a larger *TFF* value indicates more likelihood to be influential users. Instead of using the follower and followee counts from our collected dataset, we employ those retrieved from Twitter.
2. *User readability* employs the average Flesh-Kincaid Grade Level (FKGL) score over all tweets composed by the respective user. FKGL describes how many years of education are required for an individual to understand the

given text. For this purpose, FKGL takes a tweet's words, syllables and sentences into account. Before calculating the respective FKGL value, we pre-process each tweet by replacing hashtags, URLs, popular emoticons, "RT" (indicating retweets) and user names by appropriate placeholder texts. Otherwise, the FKGL scores could be biased. For instance, assuming we have the same tweet twice and the only difference between both is the mentioned user within the tweet, FKGL would depend on the length of the mentioned user name. But if every user name is replaced with the same placeholder text, FKGL yields identical results for both tweets, which is more intuitive.

3. *User quality* is determined by averaging the tweet quality values over all collected tweets by a user. Tweet quality is based on the assumption that the more often a tweet is retweeted, the higher its quality is. If a tweet is re-posted by a different user, a score of 0.5 is assigned to this tweet. If the retweet includes comments, another 0.5 is added to the score. The lowest value is zero, but there is no upper limit.
4. *In-degree* of a node: a higher in-degree intuitively implies a higher likelihood of an individual to be influential in both of our graphs.

Attributes $13 - 15$ are calculated in a straightforward manner.

4. *Edge betweenness* we add up the scores per user whenever she is part of an edge - be it as source or target node.

3.2.5. Learning MaxClassifier & DirectClassifier

The *InfluenceLearner* derives two sets of labels which are assigned to the users as their respective ground truth. One of which is based on tweet labels, the other one on user labels from the collected ground truth dataset that was established using the *SNAnnotator*. Specifically, the *DirectLabel* is derived directly from the user labels in that it utilizes the user labels of the ground truth dataset. In contrast, the *MaxLabel* is derived indirectly from the tweet labels of a user; it is computed as the highest label in the ground truth dataset that human annotators assigned to any of her tweets. For each, we build separate models on the *RG* and *IG* leading to four classifiers in total. The *DirectClassifier* learns from the *DirectLabel*, and correspondingly *MaxClassifier* learns from *MaxLabel*. More precisely, the *DirectClassifierRG* and *MaxClassifierRG* are trained on the *RG*, while the *DirectClassifierIG* and *MaxClassifierIG* are trained on the *IG*. But besides using different user labels for building the classifiers, the inputs for the *MaxClassifierRG* and *DirectClassifierRG* (*MaxClassifierIG* and *DirectClassifierIG*) remain identical; so the same attributes extracted from the *RG* (*IG*) are given to the appropriate classifiers.

The dataset obtained by using our *SNAnnotator* exhibits a highly skewed distribution, because influentials occur naturally more rarely than non-influentials. We remedy this problem by using a combination of SMOTE (Synthetic minority oversampling technique) (Chawla et al., 2002) and a cost-sensitive matrix. At first, the number of influentials in the dataset is doubled by employing SMOTE. The resulting skewed distribution is balanced by setting up an appropriate cost matrix, such that the costs of misclassifying the minority class results in a proportionally large penalty.

3.2.6. Correlation between Influential Tweets and Influentials Users

Due to our ground truth, we want to examine whether influential tweets serve as indicator for influential users. To this purpose, we compare the classifiers with respect to the type of label (*MaxLabel*, *DirectLabel*) and graph (*RG*, *IG*), i.e. we compare *DirectClassifierRG* with *MaxClassifierRG* and *DirectClassifierIG* with *MaxClassifierIG*. For analyzing these correlations, we calculate Spearman's coefficient ρ and Pearson's r. We employ both to detect any potential relationships because the former can detect non-linear relationships, while the latter can detect linear ones. Since ρ requires a ranking among instances, we rank them on the confidence of the classifier when it assigns a label. In this experiment, we use for the statistical tests the following null hypothesis: "influential users do not correlate with influential tweets": i.e. users may be influential no matter whether their tweets are.

3.2.7. Finding the most Predictive Attributes

Since our main goal is to identify attributes distinguishing influential users from non-influential ones, we need to evaluate how well this subset performs with respect to a standard. In our case, we compare the classifiers learned on the subset of the most predictive attributes with those trained on the full set of attributes. We do this separately for all four classifiers. More specifically, we first find the subset of the most predictive attributes by applying

Correlation-based Feature Extraction (CFS) to the attributes shown in Table 1 for each of the four classifiers. The ranking of the attributes in these subsets is established by ordering the attribute scores that were assigned according to the Chi-Squared method (χ^2), which evaluates the worth of an attribute according to a χ^2 test with respect to the user labels. In the next step, the classifiers based on these selected attributes are trained according to section 3.2.5. We then compare their performances with the classifiers learned on all attributes to find out if the detected attribute subset is predictive or not. We select CFS and χ^2 in our study because they were applied successfully in the past, e.g. in (Liu et al., 2002).

4. Experimental Validation

This section describes results from applying the proposed framework.

4.1. Dataset

We used *SNAnnotator* to crawl data from Twitter and have them manually annotated[9]. In particular, we harvested all tweets on "Amazon Kindle" that were posted within one week. Afterwards, we retrieved the authors of these tweets within a single day to avoid possible side-effects. The topic "Amazon Kindle" was identified using the hashtags #KINDLE, #Kindle and #kindle and all tweets are written in English. After removing invalid tweets and users, nine different annotators labeled the remaining ones manually. In the end, each tweet and user was labeled by three different human annotators. After removing 310 users and 1497 tweets, since they were at least once assigned the label "strange", we select for the remaining tweets and users the median of the three annotator labels as final labels.

Finally, we merge the classes "influential" and "probably influential", and the classes "non-influential" and "probably non-influential" because the human annotators turned out to be too reluctant to assign the labels "influential" and "non-influential". A study of why this is so, is beyond the scope of this paper. But merging the classes introduces noise, because it is more difficult to distinguish between "probably non-influential" and "probably influential" than between "influential" and "probably non-influential" or between "non-influential" and "probably influential". This makes our learning task more challenging.

An overview of our final dataset is given in Table 2. It is obvious that *RG* is much denser than *IG*. We also depict the number of influentials in both graphs depending on the labels we use: if we select them based on *DirectLabel* there exist 250 influentials. In contrast, there are only 161 influentials in *IG*, but compared to the number of users existing in this graph, the fraction of influentials has nearly doubled. The same observation holds for influentials if we determine them based on *MaxLabel*. Only 15 out of 92 influentials based on *MaxLabel* are also considered influential in terms of *DirectLabel*. Hence, only very few influentials have posted influential tweets. It seems that constructing *IG* automatically removes more non-influentials than influentials. However, some of the influential users are also removed from *IG* because we did not collect any interactions between them and other collected users. In *RG* there are 35741 tweets, but only 131 of them are influential. In *IG* only 25800 tweets exist, but the fraction of influential tweets is as in *RG*.

	RG	*IG*
Users	4437	1685
Edges	550931	5471
Influentials (*DirectLabel*)	250 (5.6%)	161 (9.6%)
Influentials (*MaxLabel*)	92 (2.0%)	74 (4.4%)
Influential tweets	131 (0.4%)	113 (0.4%)

Table 2: Final statistics for *RG* and *IG*. In *IG* exist 25800 tweets, whereas *RG* encompasses 35741 tweets.

4.2. Experiments

For building our four different classifiers on *RG* and *IG*, we select Weka (Hall et al., 2009) as platform. After evaluating the performance of different classifiers (SVM, Decision Tree, Naive Bayes and Bayesian network) in terms of ROC curves in preliminary experiments, we selected a Bayesian network from Weka using 10-fold cross-validation for building our classifiers.

[9]Interested scholars can acquire our dataset; please contact the first author. The dataset cannot be placed into a public page, because tweet content cannot be published by us (Twitter regulation). However, interested scholars can retrieve the tweet IDs from the first author, then obtain the corresponding tweets directly from Twitter, and then link the tweets with the recorded data of our dataset.)

4.2.1. Influential Tweets vs. Influential Users

In our first experiment, we analyze the correlation between influential tweets and influential users, i.e. that users may be influential no matter whether their tweets are. We compare *DirectClassifierRG* with *MaxClassifierRG* (see 2nd column of Table 3) and *DirectClassifierIG* with *MaxClassifierIG* (3rd column of Table 3) using ρ and Pearson's r (first column of 3).

In Table 3, we depict the values of the two coefficients and the p-values in parentheses (significance level is 0.05). All coefficients are very low, hence we cannot negate the null hypothesis. There are following possible explanations for that. First, the human annotators decide subjectively whether a user is influential and whether a tweet is influential, so the null hypothesis may be due to the annotators' behavior. Moreover, the human annotators have little available information on tweets (just the tweet content) in contrast to the information on users (all tweets of each user, personal information and external URLs), so the labels of the tweets may have been assigned in a less well-informed way. Finally, there *are* users, e.g. celebrities, for which we suspect that they have many followers and many discussions on their tweets, because of inherent properties of the users rather than because of the tweets' contents. To investigate this further, there is need for data where celebrities can be identified and treated separately.

	RG	*IG*
ρ	-0.004 (0.813)	-0.023 (0.352)
r	-0.022 (0.144)	0.001 (0.952)

Table 3: Results for Pearson's r and Spearman's coefficient ρ when comparing the *MaxClassifierRG* with the *DirectClassifierRG* (left) and the *MaxClassifierIG* with the *DirectClassifierIG* (right). In parentheses are the p-values.

4.2.2. Comparison of ROC Curves

In Figure 4, the ROC curve of each final classifier is plotted against four baseline methods. The random baseline "B-Random" chooses the label "non-influential" proportional to the ratio between influentials and non-influentials depending on which combination of learner and graph is being used. The remaining three baselines per classifier are chosen according to their quality and the number of incorporated categories: the best ones in terms of ROC curve utilizing attributes related to one, two and three categories out of {community, activity, quality, centrality}. The remaining ones are not plotted in order to facilitate readability. The performance of all four classifiers *MaxClassifierIG*, *MaxClassifierRG*, *DirectClassifierIG*, *DirectClassifierRG* shows an improvement over the random baseline.

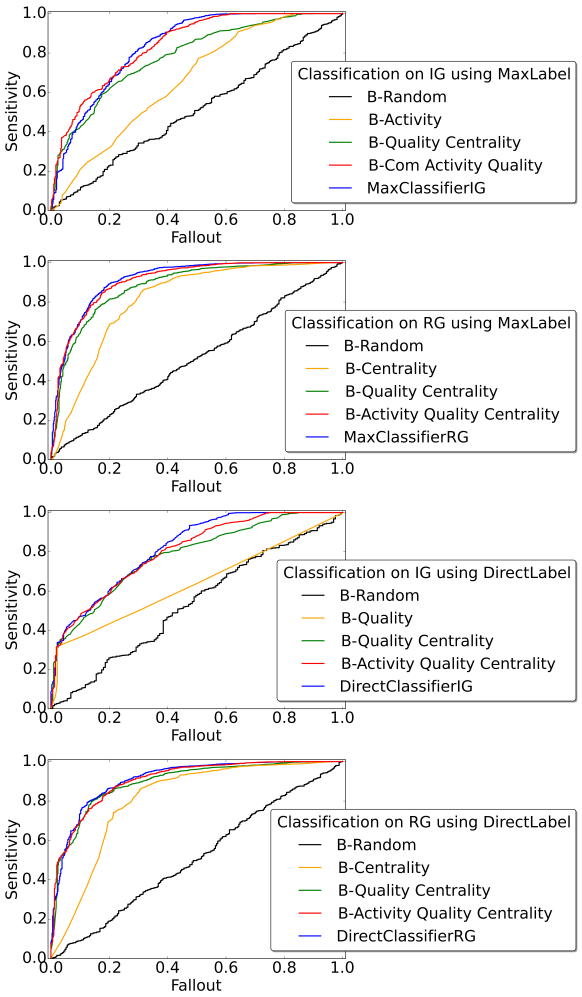

Figure 4: ROC curves of final models compared with a random baseline ("B-Random") and the best baselines capitalizing attributes related to one, two and three out of four categories. $Fallout = 1 - Specificity$.

As it can be seen from Figure 4, the more categories are added, the better is the performance of the classifier. This is visible when comparing the random baseline with the classifier trained on attributes related to a single category. But also when utilizing more categories in the classifiers, the ROC curves improve. However, the improvement satu-

rates after considering three categories, since the ROC curves of the classifiers incorporating three categories approach our final classifiers, although they perform slightly worse. The only exception is *MaxClassifierIG*, where the baseline utilizing no attributes related to quality outperforms the final model. We therefore assume that less extracted attributes are sufficient to detect influentials. This is further supported by the observation that the baselines utilizing only attributes related to three instead of four categories are also able to compete with the final classifiers for the *RG*.

We see in Figure 4 that the performance of the classifiers on *RG* is higher than the performance of the corresponding classifiers on *IG*. This indicates that separation between influential and non-influential tweets is easier on the Relation Graph RG than on the Interaction Graph *IG*. This finding is in contrast to the finding of (Bigonha et al., 2012; Huberman et al., 2008), who also collected tweets on a single topic and associated them to Twitter users (we used the same approach, so the experiments are comparable): they concluded that Interaction Graphs represent influence better than Relation Graphs do. A possible explanation for the superiority of Relation Graphs may be the construction of Interaction Graphs: if the crawled dataset contains no interactions of a user, then this user is not in *IG* but is present in *RG*, hence *RG* contains more information that can be exploited for supervised learning. This construction caveat can only be amended (in an unbiased way) if we copy the whole Twitter network; however, this is not feasible, because new tweets are added continuously to Twitter, so sampling is inevitable.

4.3. Predictive Attributes

In Table 4, we depict the most predictive attributes, where the predictive power was computed according to CFS and χ^2. The attributes in this table achieve the same predictive performance as the complete set of indicators in 3.2.4 [10].

When we look at the most predictive attributes, it turns out that *Influence*, *# Interactions* and *TFF* are utilized across all four learners. Furthermore, when comparing the *DirectClassifierRG* with *MaxClassifierRG* and the *DirectClassifierIG* with *MaxClassifierIG*, we find that the selected attributes are identical and only their rankings differ slightly. Most notably, *Influence* ranks highest over all four learners on average, which suggests that median tweet labels do correlate with influential users to some extent. But this relationship is weak and not statistically significant(cf. section 4.2.1).

Now we also want to answer the question: what happens if no tweet labels exist? The rationale is that it is time-consuming to obtain a second set of labels from human annotators. Since *MaxClassifierRG* and *MaxClassifierIG* cannot be built without the existence of tweet labels, we discard both learners from our experiment. Therefore, we build the *DirectClassifierRG* and *DirectClassifierIG* by following the steps described in this paper, but now we exclude *Influence* from the attributes. When comparing the ROC curves of these classifiers, it turns out that their performance is comparable with the respective best baselines incorporating attributes related to three categories, which are depicted in Figure 4. Due to space limitations we do not include the ROC curves, but only the most discriminatory attributes for the identification of influentials in Table 5. In the *IG # Tweets* replaces *Influence* and the remaining attributes are identical as opposed to the learners incorporating this attribute. Likewise, *Influence* is replaced by *In-degree* in the *RG*.

In fact, if we remove any single attribute of Table 4 from the learners, their quality remains almost constant. Both observations suggest that all ranked attributes in Table 1 exhibit a low correlation with user and tweet labels. In other words, no single attribute is discriminatory by itself. But once we remove a subset of those attributes, the ROC curves deteriorate. This claim indicates that the differences between influentials and non-influentials are very subtle, and therefore the attributes for identifying influentials should be manifold. For example, in our dataset the top-ranked attributes come from attributes related to at least three different categories. Hence, it seems promising to add further categories about influential users instead of looking for more attributes related to the proposed categories.

[10] We have also applied our classification algorithms on various subsets of the complete set of attributes and found again that a selection of attributes from different categories yields better results than many attributes from the same category. We omit the results due to lack of space.

Rank	*RG*	*IG*
1	*Edge betweenness*	*Influence*
2	*Influence*	*In-degree*
3	*# Tweets*	*# Edges from influentials*
4	*Eigenvector centrality*	*# Interactions*
5	*# Interactions*	*TFF*
6	*TFF*	

Rank	*RG*	*IG*
1	*Influence*	*In-degree*
2	*Edge betweenness*	*Influence*
3	*# Tweets*	*Edges from influentials*
4	*Closeness*	*# Interactions*
5	*# Interactions*	*TFF*
6	*TFF*	

Table 4: Subsets of most discriminatory attributes for *DirectClassifier* (top) and *MaxClassifier* (bottom) ordered with respect to their χ^2 score.

Rank	*RG*	*IG*
1	*Edge betweenness*	*In-degree*
2	*In-degree*	*Edges from influentials*
3	*# Tweets*	*# Interactions*
4	*Eigenvector centrality*	*# Tweets*
5	*# Interactions*	*TFF*
6	*TFF*	

Table 5: Subsets of most discriminatory attributes when *Influence* is removed for *DirectClassifier* ordered with respect to their χ^2 score.

4.4. How do our Influence Classification Results compare to Influence Scores by Commercial Services?

In our framework we concentrate on one topic only because a user may be influential with respect to one topic but non-influential with respect to another (Cha et al., 2010). Services like Klout, Kred and PeerIndex provide a global score that aggregates influence over many topics. Although they also deliver a score for our specific topic "'Amazon Kindle', we do not compare our results with them since they combine information from multiple social networks like Facebook and Twitter. The topic-specific scores are computed with respect to a more abstract topic like "books" instead of only considering the domain "Amazon Kindle". This tampers the scores to a certain degree depending on the domain. Furthermore, it is unclear whether the global scores affect the topic-specific influence scores somehow as these information are not publicly accessible. In other words, these services have a very different definition of influence and a very different basis for computation (more than one social networks) so that a comparison of influential users for the topic we studied makes no sense.

5. Conclusions

In this paper, we presented a framework for the identification of properties that characterize influential users. In the following, we discuss our contributions and identify shortcomings and issues that require further research.

5.1. Contributions

The literature on identifying influential users is vast, but the identification of properties peculiar to influential users has only recently moved to the focus of scholars. The task of discerning such properties is impeded by the fact that there is no agreed-upon definition of the term "influential user". There are commercial services that have their own proprietary definition of this term and, accordingly, assign influence scores to users. We do not study whether one of these definitions is better than the others, but rather prefer to adhere to a notion of influence that is closer to what people, not service providers, perceive as "influential user". Accordingly, our approach encompasses a crawling and annotation tool, the *SNAnnotator*, with which human annotators can inspect tweets and Twitter users, and decide which of these users are influential. Our *SNAnnotator* incorporates a mechanism that aggregates the votes given by the human annotators for each user; this mechanism compensates for the fact that different human annotators have different (implicit) perceptions of what constitutes *influence*.

The *SNAnnotator* is our first contribution. It is a self-standing tool that can be used for the construction of training datasets on user influence. For our experiments, the *SNAnnotator* was used by a small number of independent human annotators. However, the tool can be used by an arbitrary number of independent human annotators, as is usual in the crowdsourcing context. More precisely, scholars can

invoke the crawling component of the *SNAnnotator* to acquire tweets and associated information, and then recruit human annotators who will use the annotation component to label the data.

Our second contribution is the *InfluenceLearner*, a mining method that computes a large set of properties/attributes, which reflect the behavior and community participation of social network users, and then uses these properties to separate between influential and non-influential users. This contribution is twofold: it consists of (i) the algorithms that process the social network and derive the properties from it in a 3-step workflow, and (ii) of the set of attributes itself.

(i) The workflow of the *InfluenceLearner* combines and extends the methods of (Evans & Lambiotte, 2009; Blondel et al., 2008) for the construction of overlapping communities and the computation of the properties/attributes on them. We have extended the algorithm of Evans et al. (Evans & Lambiotte, 2009) so that it requires a constant amount of memory for creating line graphs to discover overlapping communities; this allows us to process also larger graphs.

(ii) The set of properties is more comprehensive than the sets of properties considered in earlier works (Bigonha et al., 2012).

Scholars can apply our *InfluenceLearner*on any labeled training set, and check whether the properties we propose contribute to separation. This also contributes to an independent validation of our experiments (if a similarly acquired training dataset is used), and to testing the potential of our set of properties on training datasets that reflect different definitions of the term "influential user".

Our third contribution concerns the findings from our experimental results. First, we found the Relation Graph permits a better identification of influential users than the Interaction Graph; this finding disagrees with earlier literature (Bigonha et al., 2012; Huberman et al., 2008), where a superiority of the Interaction Graph was found – for *unsupervised learning*. Our explanation is that the Relation Graph incorporates more data that can be used for *supervised learning* than the Interaction Graph, namely data on users that had no interactions inside the collected snapshot but are influential members of their communities nonetheless. This indicates that the Relation Graph may be preferable for supervised learning, albeit the Interaction Graph might be more advantageous in unsupervised scenarios. Hence, when analyzing influence, it is useful to build and exploit both graphs.

Further, our experimental results show that the differences between influential and non-influential users are subtle. There is no single attribute that suffices for separation. This indicates that the separation between influential and non-influential users requires considering many aspects of user behavior - the categories in which we organized the attributes. Our results also show that it is not necessary to consider all categories, although more than one category must be considered, and that it is not necessary to consider all attributes per category. This is of advantage in terms of execution overhead, since the computation of some of the attributes is resource consuming. However, the identification of the best subset of such attributes is still an open issue; such a set is most likely depending on the definition of the term "influential user".

Our experimental results also indicate that it is difficult to derive a definition of "influential user" that reflects the perceptions of human annotators *and* can be reflected in a set of derived properties. Our approach has the advantage that it accounts for subjective views of different people on who is an influential user. Moreover, our framework allows scholars to test the role of different properties for training datasets that adhere to different definitions of "influence".

A final contribution of our work is the dataset we produced with the *SNAnnotator* to validate our *InfluenceLearner*. As explained in 4.1, we make this dataset available to interested scholars. It can be used for validation of further learning algorithms, beyond our *InfluenceLearner*. It can also be used to perform analyses on how humans annotate, since it contains detailed recordings of the activities of the human annotators.

5.2. Limitations & Future Work

We have validated our *InfluenceLearner* on only one topic and corresponding dataset. The graph corresponding to this dataset has the same statistics as the graphs of other known Twitter datasets (e.g. power law holds for follower/followee relationship); the subject of this dataset was popular but not the most popular one, i.e. the subject is not an outlier. Hence, we expect that our dataset is typical for the behavior of Twitter users, when they

comment on popular subjects. Moreover, the instructions to our human annotators did not take the topic into account, nor were these annotators so chosen as to be peculiarly familiar with the subject of the dataset. Thus, also the annotation process can be expected to be typical for any rather popular Twitter subject. Nonetheless, experiments on further training datasets are needed to demonstrate the transferability of our results, concerning the predictive power of properties and the superiority of the Relation Graph over the Interaction Graph on arbitrary popular subjects. A study of less popular subjects would also be interesting in this context.

Our *InfluenceLearner* can be applied on datasets that adhere to any definition of "influential user". In this study, we have evaluated on a dataset that reflects the diffuse perception of influence that one would expect from distinct and independent human beings. Applying our *InfluenceLearner* on other datasets would deliver insights on how the set of properties we propose can describe other, proprietary definitions of influence. This is an appealing task for future work. However, we are more interested in capturing the perception of influence among humans, so we want to investigate ways of extending the *SNAnnotator* to build training sets that are more insensitive to the attitude of each single human annotator. Indeed, the low number of human annotators inevitably affects the generalizability of our results. Next to the already mentioned option of crowdsourcing, we are interested in studying active learning strategies, e.g. by selectively sampling among the ratings of the human annotators (Donmez et al., 2009). We plan to supplement this strategy by using information about how the annotators rating behavior changed over time; to this purpose, we can exploit the additional metadata recorded by the *SNAnnotator* during the annotation process. To this purpose, we will interpret each human annotator as a single classifier to predict the labels of fellow annotators.

Our experimental results indicate that the properties we use are correlated. However, we have not investigated how the properties correlate with respect to the outcome (influential / non-influential user). Building a minimal set of predictive properties is important for computational reasons; the computation of some properties is resource-consuming. This is an issue for future work.

Last but not least, we also want to improve the quality of our learned model by extracting sentiment-related attributes, as sentiment is known to correlate with influential users (Bigonha et al., 2012).

References

Anger, I., & Kittl, C. (2011). Measuring influence on twitter. In *Proceedings of the 11th International Conference on Knowledge Management and Knowledge Technologies* (p. 31). ACM.

Bakshy, E., Hofman, J. M., Mason, W. A., & Watts, D. J. (2011). Everyone's an influencer: quantifying influence on twitter. In *Proceedings of the fourth ACM international conference on Web search and data mining* (pp. 65–74). ACM. doi:10.1145/1935826.1935845.

Barbieri, N., Bonchi, F., & Manco, G. (2013). Cascade-based community detection. In *Proceedings of the sixth ACM international conference on Web search and data mining* (pp. 33–42). ACM. doi:10.1145/2433396.2433403.

Bigonha, C., Cardoso, T. N., Moro, M. M., Gonçalves, M. A., & Almeida, V. A. (2012). Sentiment-based influence detection on Twitter. *Journal of the Brazilian Computer Society*, (pp. 1–15).

Blondel, V., Guillaume, J., Lambiotte, R., & Lefebvre, E. (2008). Fast unfolding of communities in large networks. *Journal of Statistical Mechanics: Theory and Experiment*, *2008*, P10008. doi:10.1088/1742-5468/2008/10/P10008.

Campo-Ávila, J. d., Moreno-Vergara, N., & Trella-López, M. (2013). Bridging the gap between the least and the most influential twitter users. *Procedia Computer Science*, *19*, 437–444.

Cha, M., Haddadi, H., Benevenuto, F., & Gummadi, P. K. (2010). Measuring user influence in twitter: The million follower fallacy. *ICWSM*, *10*, 10–17.

Chai, W., Xu, W., Zuo, M., & Wen, X. (2013). Acqr: A novel framework to identify and predict influential users in micro-blogging. *PACIS 2013 Proceedings. Paper 20*, .

Chawla, N. V., Bowyer, K. W., Hall, L. O., & Kegelmeyer, W. P. (2002). Smote: Synthetic minority over-sampling technique. *Journal of Artificial Intelligence Research*, *16*, 321–357.

Domingos, P., & Richardson, M. (2001). Mining the network value of customers. In *KDD '01: Proceedings of the 7th ACM SIGKDD international conference on Knowledge discovery and Data Mining* (pp. 57–66). New York, NY, USA: ACM. doi:10.1145/502512.502525.

Donmez, P., Carbonell, J. G., & Schneider, J. (2009). Efficiently learning the accuracy of labeling sources for selective sampling. In *Proceedings of the 15th ACM SIGKDD international conference on Knowledge discovery and data mining* (pp. 259–268). ACM. doi:10.1145/1557019.1557053.

Evans, T., & Lambiotte, R. (2009). Line graphs, link partitions, and overlapping communities. *Physical Review E*, *80*, 016105. doi:10.1103/PhysRevE.80.016105.

Girvan, M., & Newman, M. E. (2002). Community structure in social and biological networks. *Proceedings of the National Academy of Sciences*, *99*, 7821–7826.

Hall, M., Frank, E., Holmes, G., Pfahringer, B., Reutemann, P., & Witten, I. H. (2009). The weka data mining software: an update. *ACM SIGKDD Explorations Newsletter*, *11*, 10–18. doi:10.1145/1656274.1656278.

Huberman, B., Romero, D. M., & Wu, F. (2008). Social networks that matter: Twitter under the microscope. *First Monday*, *14*.

Kempe, D., Kleinberg, J., & Tardos, É. (2003). Maximizing the spread of influence through a social network. In *Proceedings of the ninth ACM SIGKDD international conference on Knowledge discovery and data mining* (pp. 137–146). ACM. doi:10.1145/956750.956769.

King, D., Ramirez-Cano, D., Greaves, F., Vlaev, I., Beales, S., & Darzi, A. (2013). Twitter and the health reforms in the english national health service. *Health policy*, *110*, 291–297.

Kong, S., & Feng, L. (2011). A tweet-centric approach for topic-specific author ranking in micro-blog. In *Advanced Data Mining and Applications* (pp. 138–151). Springer.

Liu, H., Li, J., & Wong, L. (2002). A comparative study on feature selection and classification methods using gene expression profiles and proteomic patterns. *Genome Informatics Series*, (pp. 51–60).

Liu, N., Li, L., Xu, G., & Yang, Z. (2014). Identifying domain-dependent influential microblog users: A post-feature based approach. In *Twenty-Eighth AAAI Conference on Artificial Intelligence*.

de Nooy, W., Mrvar, A., & Batagelj, V. (2005). *Exploratory social network analysis with Pajek* volume 27. Cambridge University Press.

Overbey, L. A., Paribello, C., & Jackson, T. (2013). Identifying influential twitter users in the 2011 egyptian revolution. In *Social Computing, Behavioral-Cultural Modeling and Prediction* (pp. 377–385). Springer.

Quercia, D., Stillwell, D., Michal Kosinskil, & Crowcroft, J. (2011). Our twitter profiles, our selves: Predicting personality with twitter. *Proceedings of the 3rd IEEE Conference on Social Computing (SocialCom) 2011*, .

Razis, G., & Anagnostopoulos, I. (2014). Influencetracker: Rating the impact of a twitter account. *arXiv preprint arXiv:1404.5239*, .

Sun, B., & Ng, V. T. (2013). Identifying influential users by their postings in social networks. In *Ubiquitous Social Media Analysis* (pp. 128–151). Springer.

Wang, Y., Cong, G., Song, G., & Xie, K. (2010). Community-based greedy algorithm for mining top-k influential nodes in mobile social networks. In *Proceedings of the 16th ACM SIGKDD international conference on Knowledge discovery and data mining* (pp. 1039–1048). ACM. doi:10.1145/1835804.1835935.

Xiao, C., Zhang, Y., Zeng, X., & Wu, Y. (2013). Predicting user influence in social media. *Journal of Networks*, *8*, 2649–2655.

Zhao, K., Yen, J., Greer, G., Qiu, B., Mitra, P., & Portier, K. (2014). Finding influential users of online health communities: a new metric based on sentiment influence. *Journal of the American Medical Informatics Association: JAMIA (2014)*, .

Highlights

1. Classification algorithm separating between influential & non-influential users
2. Identification of properties that characterize influential users
3. Use of community structure to separate influential & non-influential users
4. Interactive annotation tool for labeling users & tweets regarding influence
5. Workflow for building a ground truth for influential users & tweets