# KHYRank: Using Retweets and Mentions to Predict Influential Users

Aron Yu[1], C. Vic Hu[1], and Ann Kilzer[2]

[1]Dept. of Electrical Engineering, The University of Texas at Austin
{aron.yu, cvhu}@utexas.edu
[2]Dept. of Computer Science, The University of Texas at Austin
{akilzer}@cs.utexas.edu

December 8, 2011

**Abstract**

Determining influential users is a widely studied problem in online social networks. In this study, we use real data from Twitter: the social graph and a set of tweets from a five month period. Specifically, we examine retweets and mentions, which indicate a response or citation. We argue that these are better indicators of influence than mere follower count or PageRank. We present KHYRank, an algorithm for determining influence based on the graph structure, and probability of a user mentioning or retweeting another user. Not only does our algorithm successfully find celebrities and content sources, but it also can be efficiently run on a desktop using MATLAB.

## 1 Introduction

We've all heard about viral marketing. Even Forbes.com compared the social graph to "crude oil," describing it as an "unrefined and complex natural resource containing many riches" [20]. Maciej Ceglowski, the developer behind Pinboard.in quipped "Social networks exist to sell you crap. The icky feeling you get when your friend starts to talk to you about Amway, or when you spot someone passing out business cards at a birthday party, is the entire driving force behind a site like Facebook" [2].

Advertisers want to run the most effective campaign using a limited budget. The whole idea behind viral marketing is to target users who will spread the word to their friends. But how do we choose which users to target? Our project is to find a scalable method for finding the most influential users in a social graph. We hypothesize that we can measure an individual's influence by the number of *repeaters* they connect to. By repeaters, we mean those users who redistribute content (Think Twitter's "retweets," Google+ and Facebook's "Share" features).

## 2 Background

### 2.1 Twitter

Twitter is an online social network that specializes in "microblogging," sharing text snippets, or "tweets," of no more than 140 characters. It is a directed graph: you may follow others, subscribing to their tweets, but they may not necessarily follow you. Features of the service include the ability to "retweet," or repeat, verbatim, another user's post. Additionally, users label their tweets via hashtags, short strings beginning with #, which provide some context to the tweet. For instance, a current popular hashtag is "#OcuppyOakland," which quickly provides context to tweets about the current political protests. Another feature is to directly reference users by putting an @ before their screen name. We will refer to this as an "@ mention, (at mention).

Twitter distinguishes itself from other popular social networks in that it is easy to repeat content, either via retweets or repeating a hashtag. Posts are by default public, and twitter is a popular platform for celebrities, brands, and organizations to connect with their followers. Typically, users read tweets in a stream format containing live updates from all the users they follow.

Our study focuses on retweets and mentions. We argue that these metrics are useful in finding influence, as they demonstrate a user responding to a specific piece of information from a source. In the retweet case, the user is just mirroring the source's information to his or her followers. Note that retweets may propagate along a chain of users. For instance, consider when user $A$ is followed by user $B$, who is followed by user $C$. If $B$ retweets $A$'s tweet, then the tweet shows up in $C$'s stream with both the source's name ($A$) and the last link in the chain ($B$). In the @ mention case, the user is typically addressing some topic from a source, or calling them out by name.

### 2.2 Data

We obtained our data from multiple sources:. The first dataset is a set of tweets collected between September 2009 and January 2010 [4]. The data set contains 113,515 Twitter users and 3,844,612 updates from the users. The original test and training set were not well balanced, and contained few overlapping users. We redivided the users by time, so we could use training data from an earlier time period to predict influence in a later time period. The new training set contains data from September through November, 2009, while the new test set contains data from December 2009 and January 2010.

The second dataset is a crawl of the entire Twitter graph from June 2009 [16]. According to the authors, it is a crawl of the entireTwitter site with 41.7 million user profiles and

1.47 billion edges.

The authors of the twitter graph also provided a mapping from Twitter usernames to user ids. Lastly, we also used the info chimps API to obtain mapping information [14]. This was helpful when we parsed usernames from @ mentions and retweets, because most of the data is stored according to user id.

### 2.3 Implementation

We implemented our code in python and MATLAB. Python was used for constructing an adjacency matrix, and parsing the tweet dataset. We combed our tweet data for the retweets and @ mentions. Once we had our data in a suitable format, we used MATLAB for the heavy lifting.

## 3 Methodology

We devise a new algorithm to determine influence using retweet and @ mention data. More specifically, we are looking at the number of times a user has been retweeted or been mentioned by his or her followers. The first step is to construct the influence matrix which contains the probability of one user influencing another user for all users in the dataset. The next step is to calculate the influence ' using the influence matrix using different methods. The last step is to simulate the propagation of information in the user network to come up with the simulated influence '. We compare our algorithm's results with traditional influence metrics such as PageRank and degree count.

We define a retweet to be a tweet leading with "RT @user" and an @ mention to be a tweet containing "@user" anywhere besides the beginning of the tweet. Both retweets and @ mentions are subsets of the overall number of tweets by the user.

### 3.1 Influence Matrix Calculation Algorithm

Using the adjacency matrix $\mathbf{A}$, we create two matrices from the training data set. The retweet matrix $\mathbf{R}$ counts the number of times a user in row $i$ retweets about a user in column $j$. The mention matrix $\mathbf{M}$ counts the number of times a user in row $i$ mentions about a user in column $j$. The sum of each row represents the total number of times each user retweeted or mentioned about another user. We use these statistics to compute the Influence matrix $\mathbf{IF}$, which contains probability of a user in row $i$ influencing a user in column $j$. See algorithm 1 for details.

We define the probability of user $i$ influencing user $j$ $P(u_j \text{ influences } u_i)$ to be the weighted sum of the the probability of user $j$ retweeting about user $i$ $P(u_j \text{ rt } u_i)$ and the probability of user $j$ mentioning about user $i$ $P(u_j \text{ m } u_i)$. We define the weights $w_{rt}(j)$ and $w_m(j)$ proportionally to the total number of retweets and mentions by each user. We see here that if user $j$ tends to retweet about $i$ out of all his retweets, then it is more likely

**Algorithm 1**: Influence Matrix Calculation

```
Input: Adjacency matrix $\mathbf{A}$, Retweet matrix $\mathbf{R}$, Mention matrix $\mathbf{M}$
Output: Influence matrix $\mathbf{IF}$
foreach user $u_i$ do
    foreach user $u_j \neq u_i$ do
        $P(u_j \text{ rt } u_i) = \frac{\text{\# of times } u_j \text{ retweeted } u_i}{\text{total \# of retweets by } u_j}$;
        $P(u_j \text{ m } u_i) = \frac{\text{\# of times } u_j \text{ mentioned } u_i}{\text{total \# of mentions by } u_j}$;
        $w_{rt}(j) = \frac{\text{total \# of retweets by } u_j}{\text{total \# of retweets and mentions by } u_j}$;
        $w_m(j) = \frac{\text{total \# of mentions by } u_j}{\text{total \# of retweets and mentions by } u_j}$;
        $\mathbf{IF}_{i,j} = P(u_i \text{ influences } u_j) = w_{rt}(j)P(u_j \text{ rt } u_i) + w_m(j)P(u_j \text{ m } u_i)$;
```

that user $j$ will retweet about user $i$ if user $j$ does decide to retweet about someone. It also follows logically that if user $j$ tends to retweet more than @ mention when passing down a piece of information, more weight should be given to the retweets of user $j$.

It is important to note that user $i$ can only influence user $j$ if user $j$ is a follower of user $i$. In order for user $i$ to have a higher probability of influencing user $j$, user $j$ must be willing to not just pass down information but to pass down user $i$'s information specifically. Figure 3.1 represents a spy plot of the adjacency matrix $\mathbf{A}$ and the computed influence matrix $\mathbf{IF}$ using the first 15,000 users in our dataset. We see that though the users are tightly connected by following one another, the actual influence among the users is rather sparse. Therefore, the influence matrix proves to be a more accurate representation of the influence among the users.

## 3.2 Influence ' Calculation

Using the influence matrix, we now compute the influence ' and rank the users according to their influence level. We implement 3 different computation methods with minor modifications between the methods. The following equations represent the influence ' computation for a single user $u_s$. $T_s$ is the total number of tweets by user $u_s$.

**2-Layer Score:** Looking at the immediate followers of $u_s$ and summing up the probability of $u_s$ influencing each of the followers. We mutiply the result by $T_s$ because in order to be influencial, $u_s$ not only needs to have followers who tends to pass down his content in particular but also need to be able to generate enough content himself.

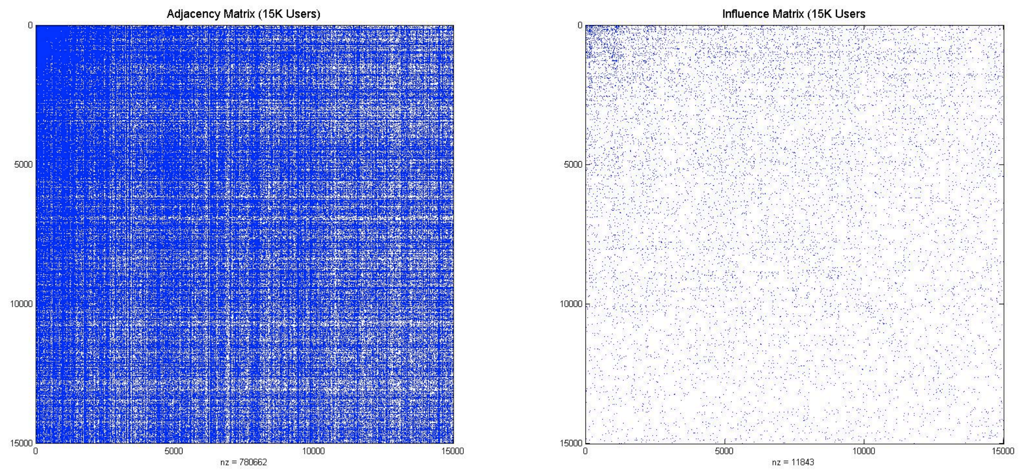

Figure 1: Connection Graph for 15,000 Users

| Intersection | N |
|---:|:---|
| 4 | 10 |
| 33 | 100 |
| 352 | 1000 |
| 4044 | 10000 |
| 4725 | 13161 |

Figure 2: Intersection between the training and test sets of the top N most retweeted users. Note there are only 13161 retweeted users in the set.

$$\text{L2 Score}(u_s) = T_s \left[ \sum_{k \in \text{followers}} P(u_s \text{ influences } u_k) \right]$$

**3-Layer Score:** Looking at the immediate followers of $u_s$ and the followers of the followers. We weight down the probabilities of the followers of $u_s$ influencing their individual followers as it is one degree of separation further away from $u_s$. This method can be seen as performing L2 Score twice with the results being weighed down as the degreee of separation increases.

$$\text{L3 Score}(u_s) = T_s \left[ \sum_{k \in \text{followers}} P(u_s \text{ influences } u_k) \left[ w_{follow} \frac{\text{L2 Score}(u_k)}{\text{\# of followers of } u_k} \right] \right]$$

**3-Layer Score (modified):** Same as the L3 Score but does not include $u_s$ as one of the followers of the followers. It is possible that $u_s$ and one of his followers are mutually following each other. In this case, the normal L3 Score will include the influential effect of the followers on $u_s$, which would in turn be used to calculate the score for $u_s$. Therefore, this method removes the follow-back effect.

$$\text{L3 Score}(u_s) = T_s \left[ \sum_{k \in \text{followers}} P(u_s \text{ influences } u_k) \left[ w_{follow} \frac{T_k \sum_{\ell \in \text{followers of } u_k} P(u_k \text{ influences } u_\ell)}{\text{\# of followers of } u_k \text{ excluding } u_s} \right] \right]$$

## 4 Results

### 4.1 Experiment 1

How well does influence in the past predict influence in the future? For this experiment, we divide our data into two parts. The training set contains data from September through November, 2009, while the test set contains data from December 2009 and January 2010.

We see that there is a lot of change in the retweeted and mentioned users. Around 33% of users retweeted in the first time period were also retweeted in the second time period.

| Intersection | N |
|---:|:---|
| 6 | 10 |
| 25 | 100 |
| 274 | 1000 |
| 5049 | 10000 |
| 13557 | 31543 |

Figure 3: Intersection between the training and test sets of the top N most mentioned users. Note we only count the 31543 mentioned users for whom we also have recorded tweets.

### 4.2 Experiment 2: Simulation

In this experiment, we use the adjacency matrix **A** and the influence matrix **IF** mentioned in the Methodology section to simulate the actual progression of information within the network. After building our network with the adjacency matrix, we weigh each edge within the network according the influence matrix.

For simulation, we start with a state where no one in the network knows about this new piece of information. We then give this information to a seed user and allow him or her to tweet about this information. To measure the influence of the seed user, we cound the number of users who know about this information at the end of a designated time iterations. To smooth out the results, we averaged the results over multiple runs. For data analysis, we used 5 time iterations and averaged the results over 100 iterations. We run the experiment with each user taking turns to be the seed user in order to compute the simulated score for each user.

Figure 4.2 demonstrates 2 example information flow in a 6-user network. A black arrow from 1 to 2 means that user 2 is following user 1, which is determined by the adjacency matrix. When we give the information to user 1, user 1 starts to influence the network. In example 1, all the users except user 4 know about the information in 2 time iterations. Since user 2 failed to influence user 4 and user 4 is not following anyone else, there is no way for user 4 to know about this information at this point. In example 2, all the users know about the information in 3 time iterations. Even though user 1 failed to influence user 3 in the first iteration, user 2 managed to influence user 3 in the next iteration, which allow user 3 to continue influencing his or her followers. Therefore, we see that information flow through different paths in the graph, and a user may be influenced by different neighbors.

We compare the simulated score to the calcuated scores from our algorithm. The L2 score turns out to yield the best result for our experiment. It is also relevant to mention that the computation speed of the KHYRank algorithm is relatively high. Using 10,000 users, it takes about 11 seconds to compute the influence scores and 2 minutes to run the simulation. Using 15,000 users, it takes about 26 second to compute the influence scores and 3.5 minutes to run the simulation. Using 20,000 users, there is a significant increase in computation time. This is because our test machine has reached its RAM limit and started using SWAP to account for the extra memory usage. We perform our experiment on a 64-bit Windows 7 machine with Intel Core i5-760 QuadCore 2.8GHz and 8GB of RAM, running 32-bit MATLAB.

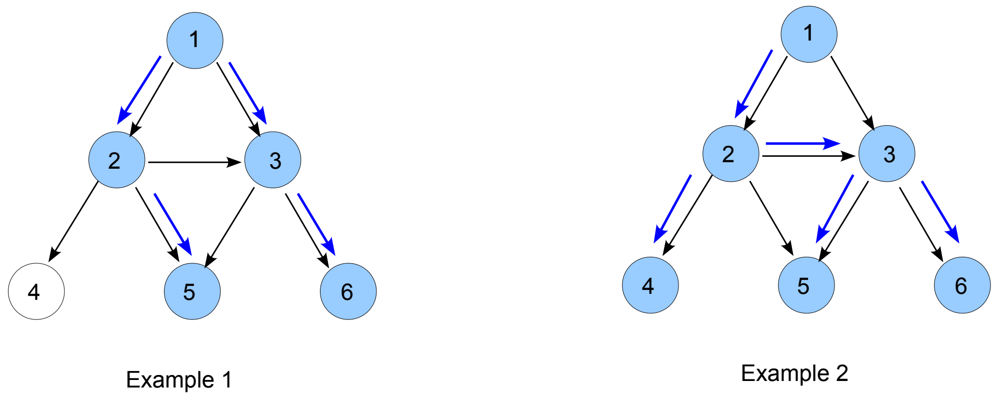

Figure 4: Examples of information flow in a 6-user network.

### 4.3 Experiment 3

In this experiment, we compare the top 150 influential users ranked by each of our algorithms. We investigate whether or not the algorithms give similar results. A venn diagram of the data is plotted in figure 4.3. We observe Degree count and PageRank are more similar to one another, while the L2 algorithm is somewhat different. This is consistent with the findings of Cha et. al [3], who noted that degree count had little to do with number of mentions or retweets.

## 5 Discussion

As shown in figure 4.3, our KHYRank algorithm lists many celebrities (Tim O'Reilly, Felicia Day, Chris Brogan) and content sources (The New York Times, The Onion, Wired) as the most influential users. By examining these users' feeds, we notice that they post frequent tweets with interesting content.

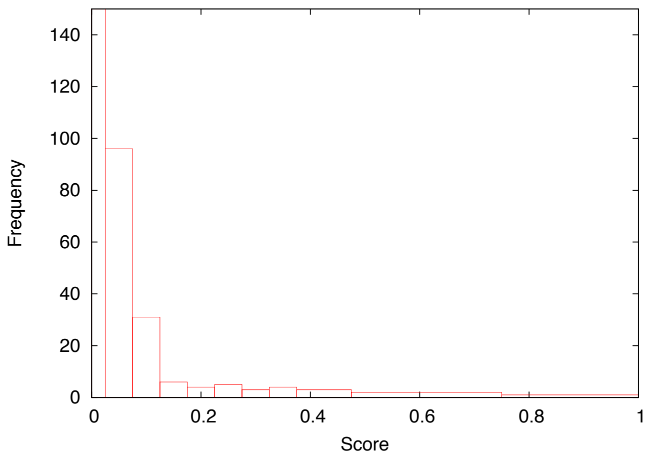

Figure 5: L2 Algorithm

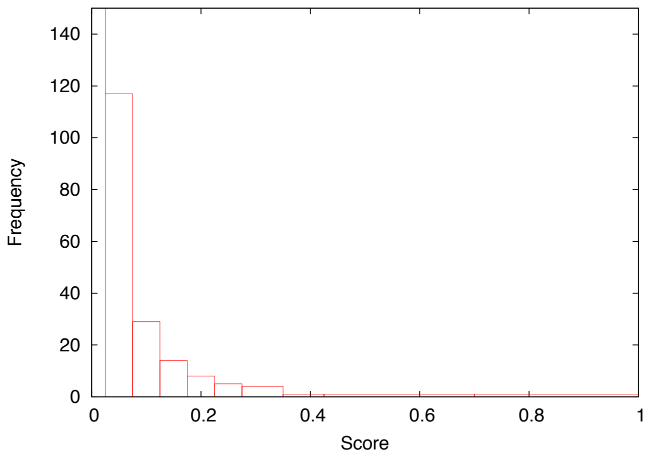

Figure 6: L3 Algorithm

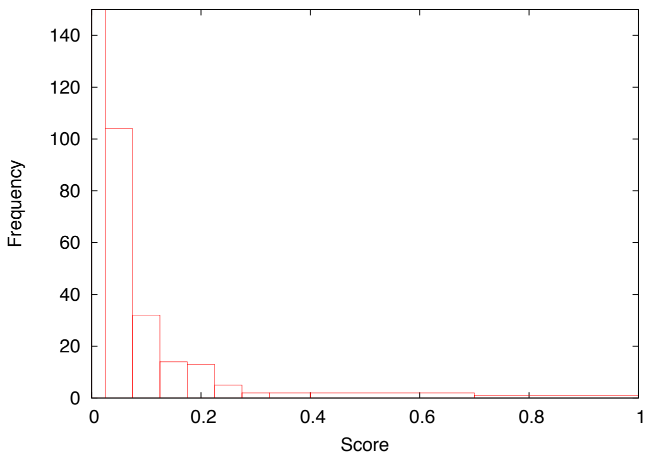

Figure 7: L3 Modified Algorithm

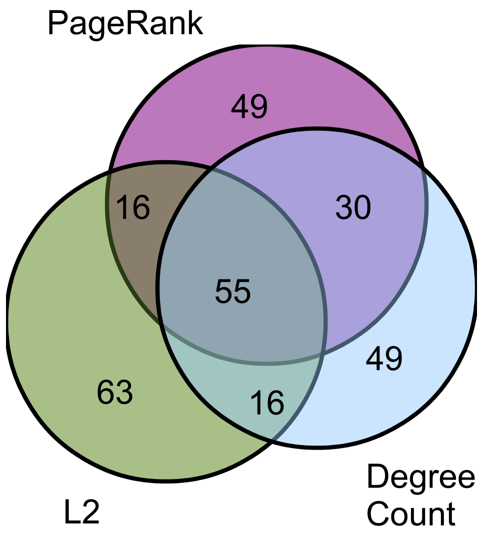

Figure 8: We examined the top 150 most influential users on the 15K dataset as ranked by our L2 algorithm, PageRank, and degree count.

| Rank | KHYRank | PageRank | Degree Count |
|---|---|---|---|
| 1 | Chris Brogan | Barack Obama | Barack Obama |
| 2 | Wil Wheaton | Twitter | Kevin Rose |
| 3 | The New York Times | Kevin Rose | Zappos.com CEO -Tony |
| 4 | Tim O'Reilly | The Onion | Twitter |
| 5 | John Gruber | Wil Wheaton | Chris Brogan |
| 6 | Dave Winer | Zappos.com CEO -Tony | The Onion |
| 7 | The Onion | Tim O'Reilly | Wil Wheaton |
| 8 | Wired | Leo Laporte | Jason Calacanis |
| 9 | iJustine | Laughing Squid | Leo Laporte |
| 10 | Felicia Day | Jason Calacanis | Chris Pirillo |

Figure 9: Ranks Comparison

The histograms are unsurprising: the normalized scores show exponential decay. However, we observe that the L3 Algorithm has a higher distribution of low scores. The L2 and L3 Modified distributions are very similar.

## 6 Related Work

The search for influential users is a widely-studied topic.

**Twitter Network**

Cha et al. collected a twitter graph and tweet dataset, examining aspects such as in degree, retweets, and mentions. They note the difficulty in defining influence, nothing that "there is no tangible way to measure such a force, nor is there a concrete definition of what influence means, for instance, in the spread of news" [3]. They develop several interesting conclusions. For instance, high node degree does not necessarily correlate with influence. Content is the main factor in determining whether a tweet gets retweeted, while a users' "name value" influences the number of mentions they get. Typically, influential users focus on a specific topic, and generate intelligent, perceptive content.

**Defining Influential Users**

To find the influential users in a social network, we need to understand and quantify some underlying properties such as centrality, communicability, and betweenness measures on a general matrix, and the framework that Estrada et al. put together meets this purpose perfectly [9]. By using the matrix exponentials of a network's adjacency, they introduced the *subgraph centrality* of node i $\left(I + A + \frac{A^2}{2!} + \cdots + \frac{A^k}{k!} + \cdots\right)_{ii} = (exp(A))_{ii}$[8] and *communicability* between node i and j $(exp(A))_{ij}$ [6]. Furthermore, to quantify the influence of a node, *betweenness* was introduced to describe the change of communicability when the node is removed, which is defined as

$$\frac{1}{(N-1)^2-(N-1)}\sum_{i=1}^{N}\sum_{j=1}^{N}\frac{exp(A)_{ij}-exp(A-E(r))_{ij}}{exp(A)_{ij}}$$

where $i \neq j, i \neq r, j \neq r$, A-E(r) is the adjacency matrix when all edges connected to node r are removed[7].

**Studying the Spread of Influence**

As pointed out in a research paper by Kempe et al. at Cornell University [15], selecting the most influential users in a social network is a NP-hard optimization problem, and even a provable approximation for efficient algorithms can be challenging to solve. Inspired by two diffusion models of interacting particles, *Linear Threshold*[12] and *Independent Cascade Models*[11], where each node can be turned on as *active* from *inactive* based on its neighbor weights and activating threshold, the team used a natural greedy hill-climbing approach related to [5] to show that the performance is guaranteed to be at least 63% (1-1/e) of optimal, which significantly out-performs other algorithms targeting high-degree or "central" nodes. The group further generalized the proved strategies to more realistic marketing scenario, where marketing actions are no longer simplified as making a single node active, but rather have stochastic effects to increase a subset of nodes' probability of becoming active (and each individual has different response to each actions.) To maximize the expected size of the final active set, they applied the hill-climbing algorithm on the expected revenue with respect to a vector of investment actions.

**The SPINE Algorithm**

When given a social graph and a list of actions propagating through it, Mathioudakis et al. designed the SPINE algorithm to find the 'backbone' of the network through the use of the independent-cascade model [18]. The algorithm proved to be an effective pre-processing step for solving influence-maximization problems. The effectiveness of SPINE came from its ability to reduce computation speed significantly while giving up little accuracy. Its main applications included propagation characterization, feed ranking, and viral marketing.

**PageRank**

Page el al. [19] developed the famous PageRank algorithm behind the web search engine Google to measure and rank the importance of web contents relevant to user queries. Let $E(u)$ be a vector over a source of rank corresponding to a set of web pages. Then $R$, the PageRank, is defined as a vector satisfying

$$R(u) = c\sum_{v\in B_u}\frac{R(v)}{N_v} + cE(u) = c(A + E \times \mathbf{1})R$$

Where $N_u = |F_u|$ is the count of links pointed from page $u$, $c$ is the normalizing factor such that the total rank of all pages is constant. $A_{u,v} = 1/N_u$ if $u$ links to $v$, $A_{u,v} = 0$ otherwise. The PageRank algorithm is solely based on the web linkage structure, regardless of the contents and other metadata. It emphasized on backlinks authority, meaning that more important pages have higher voting weights against the average pages. It is useful for information retrieval and finding clusterings in a web graph.

In our experiment, we applied the PageRank algorithm in our twitter graph and compared the results with our KHYRank algorithm. We define the adjacency matrix $A$ as following: entry $A_{ij} = 1$ if user $j$ is following user $i$; otherwise, $A_{ij} = 0$. Then, we obtained the PageRank matrix $R$ through the relation[1]

$$\mathbf{R}(t+1) = d\mathbf{M}\mathbf{R}(t) + \frac{1-d}{N}\mathbf{1}$$

and solve for

$$\mathbf{R} = (\mathbf{I} - d\mathbf{M})^{-1}\frac{1-d}{N}\mathbf{1}$$

where $\mathbf{M} = (K^{-1}A)^T$, $K$ is the matrix with outdegrees in diagonal and zeros in non-diagonal entries. Unsurprisingly, the results we got from this approach are very similar to the those we got from degree count, which are both solely based on the linkage structure in the network.

**Optimization Algorithms**

To approximate a massive graph of networks data, Savas and Dhillon[21] provided a faster and better performance framework using clustered low rank matrices. While preserving essential information structure of a massive graph, their algorithm partitions nodes into clusters, computes low rank approximations independently, and combines each low rank approximations to obtain the approximation of the entire graph. The approximation errors were derived to have deterministic bounds using a stochastic algorithm introduced by Halko et al.[13] The clustering approach was experimented and shown to outperform traditional approximation methods in both speed efficiency and memory usage significantly.

**Community-based approaches**

Wang et al. took a community-based approach to finding the influential nodes by observing information diffusion [22]. Their research targeted the mobile social network as it modeled the spread of information on a large scale. Communities within a given network were identified through a modified version of the current community detection algorithms. The changes included the addition of information diffusion as a main factor, the handling of weighted directed graphs, and the ability to modify the threshold of community decision boundaries. Instead of mining every single communities in the network, they allowed the mining decision to be dynamically chosen by the algorithm. The selection of algorithm for finding top-K influential user is trivial. This new algorithm provided provable performance guarantee just like previous algorithms. When tested on a network with 15 times more user nodes than previous testing, the new algorithm proved to be orders of magnitude faster than the leading greedy algorithm, while maintaining a small error margin.

**Game Model**

The dynamics of friendship and information sharing in a social network fit into Galeotti et al.'s model of Network Games [10]. Here, players are connected in a graph. Players can either act or not act. In a game of *strategic substitutes*, a neighbor's action replaces the need for the player to act. In the *strategic complements* game, a neighbor's action gives the player greater incentive to act, while a neighbor's inaction gives the player less incentive to act. Viral marking in a social network displays features of both strategic complements and substitutes. For instance, if Alice shares content, her friend Bob may feel less need to duplicate content, since typically some of their friends (including Charlie) overlap. Perhaps Charlie only needs to hear the campaign once to be influenced. However, we might also imagine that Charlie would be more interested in the campaign if many of his friends were sharing information about it.

**Using topic information to determine influence**

A study by Weng et al. discovers high reciprocity for following in Twitter. They also extend the PageRank algorithm into TwitterRank, an algorithm for finding influence. Unlike other approaches, TwitterRank considers the topics on which users tweet when determining influence. The authors observe that twitter users often follow others because of a shared topic of interest. They also argue that degree count is a weak indicator of influence. They demonstrate that their algorithm outperforms follower count and traditional PageRank [23].

**Order of information adoption**

Lee et al. also build on the PageRank algorithm, adding a component that considers the order of information adoption in Twitter. They study information diffusion on a real dataset, and learn that topics have a short lifespan. Topics initially spread quickly, then die down in popularity. They conclude that "early tweets spread better" [17].

**Modeling message flow**

Ye and Wu modeled message flow using a tree structure. They also examine content with multiple sources, as well as study a dataset of 500K messages relating to Michael Jackson's death, which is famous for breaking on Twitter before traditional news outlets. They learn that most replies happen soon after the original message. Furthermore, they observe that messages can propagate far through the graph. [24]

## 7 Conclusion

We have built our own algorithm for predicting influence on Twitter, using real data from mentions and retweets. We compared it to traditional influence algorithms such as PageRank and degree count, and find that our algorithm does well at finding celebrities and content sources.

Please see `http://khyrank.cvhu.org` for implementation details.

## 8 Acknowledgements

We would like to thank Joe Reisinger for providing a dataset for our preliminary work. We would also like to thank Charlie Schmidt and his belated cat Fatso. You are a source of inspiration.

## References

[1] A. Arasu, J. Novak, A. Tomkins, and J. Tomlin. Pagerank computation and the structure of the web: Experiments and algorithms. In *Proceedings of the Eleventh International World Wide Web Conference, Poster Track*, pages 107–117, 2002.

[2] Maciej Ceglowski. The social graph is neither, November 2011.

[3] Meeyoung Cha, Hamed Haddadi, Fabricio Benevenuto, and Krishna P. Gummadi. Measuring user influence in twitter: The million follower fallacy. In *Proc. International AAAI Conference on Weblogs and Social Media (ICWSM)*, 2010.

[4] Z. Cheng, J. Caverlee, and K. Lee. You are where you Tweet: A content-based approach to geo-locating Twitter users. In *In Proceeding of the 19th ACM Conference on Information and Knowledge Management (CIKM)*, 2010.

[5] P. Domingos and M. Richardson. Mining the network value of customers. *Seventh International Conference on Knowledge Discovery and Data Mining*, 2001.

[6] E. Estrada and N. Hatano. Communicability in complex networks. *Phys. Rev. E*, 77(2007)(036111), 2007.

[7] E. Estrada, D. J. Higham, and N. Hatano. Communicability betweenness in complex networks. *Phys. A*, 388(2009):764–774, 2009.

[8] E. Estrada and J. A. Rodríguez-Velázquez. Subgraph centrality in complex networks. *Phys. Rev. E*, 71(2005)(056103), 2005.

[9] Ernesto Estrada and Desmond J. Higham. Network properties revealed through matrix functions. *SIAM Rev*, page 52:696714, 2010.

[10] Andrea Galeotti, Sanjeev Goyal, Matthew O. Jackson, Fernando Vega-Redondo, and Leeat Yariv. Network games. *Review of Economic Studies*, page 77, 2010.

[11] J. Goldenberg, B. Libai, and E. Muller. Using complex systems analysis to advance marketing theory development. *Academy of Marketing Science Review*, 2011.

[12] M. Granovetter. Threshold models of collective behavior. *American Journal of Sociology*, pages 85(6):1420–1443, 1978.

[13] N. Halko, P. G. Martinsson, and J. A. Tropp. Finding structure with randomness: Stochastic algorithms for constructing approximate matrix decompositions. *Tech. re*, 2009.

[14] Infochimps. Twitter screen name, user id, search id mapping api.

[15] David Kempe, Jon Kleinberg, and Eva Tardos. Maximizing the spread of inuence through a social network. *ACM SIGKDD international conference on Knowledge discovery and data mining*, page 137146, 2003.

[16] Haewoon Kwak, Changhyun Lee, Hosung Park, and Sue Moon. What is Twitter, a social network or a news media? In *WWW '10: Proceedings of the 19th international conference on World wide web*, pages 591–600, New York, NY, USA, 2010. ACM.

[17] Changhyun Lee, Haewoon Kwak, Hosung Park, and Sue Moon. Finding influentials based on the temporal order of information adoption in twitter. In *Proceedings of the 19th international conference on World wide web*, WWW '10, 2010.

[18] Michael Mathioudakis, Francesco Bonchi, Carlos Castillo, Aristides Gionis, and Antti Ukkonen. Sparsication of inuence networks. *ACM SIGKDD international conference on Knowledge discovery and data mining*, 2011.

[19] Lawrence Page, Sergey Brin, Rajeev Motwani, and Terry Winograd. The pagerank citation ranking: Bringing order to the web. Technical Report 1999-66, Stanford InfoLab, November 1999. Previous number = SIDL-WP-1999-0120.

[20] Venkatesh Rao. The social graph as crude oil (go ahead, build that yasn!), October 2011.

[21] Berkant Savas and Inderjit S Dhillon. Clustered low rank approximation of graphs in information science applications. *SIAM International Conference on Data Mining*, 201.

[22] Yu Wang, Gao Cong, Guojie Song, and Kunqing Xie. Community-based greedy algorithm for mining top-k inuential nodes in mobile social networks. *16th ACM SIGKDD international conference on Knowledge discovery and data mining*, page 10391048, 2010.

[23] Jianshu Weng, Ee-Peng Lim, Jing Jiang, and Qi He. Twitterrank: finding topic-sensitive influential twitterers. In *Proceedings of the third ACM international conference on Web search and data mining*, WSDM '10, 2010.

[24] Shaozhi Ye and S. Felix Wu. Measuring message propagation and social influence on twitter.com. In *Proceedings of the Second international conference on Social informatics*, SocInfo'10, 2010.