I, Daniel E Lautzenheiser, hereby submit this original work as part of the requirements for the degree of Master of Science in Computer Science.

It is entitled:
Measuring the Influence of a User on Twitter

Student’s name: Daniel E Lautzenheiser

This work and its defense approved by:

Committee chair: Fred Annexstein, Ph.D.

Committee member: Kenneth Berman, Ph.D.

Committee member: Raj Bhatnagar, Ph.D.
Measuring the Influence of a User on Twitter

A thesis submitted to the
Graduate School
of the University of Cincinnati
in partial fulfillment of the
requirements for the degree of

Master of Science

in the Department of Computer Science
of the College of Engineering & Applied Science

by

Daniel Lautzenheiser

B.S. Xavier University

May 2010

Committee Chair: Fred Annexstein, Ph.D.
Abstract

This thesis focuses on the problem of measuring the influence of a user on Twitter. Twitter is a micro-blogging service and one of the most widely used social media websites in existence today. With its heavily structured nature, Twitter is a near perfect environment in which to observe social interactions and determine the level of influence a given individual exerts on the people who see their content. The data set used in this thesis consists of a very active group of users with widely varying numbers of followers. An algorithm using various publicly available statistics is developed to measure the influence of a given user. In addition an extension of the algorithm to add recursion is detailed, and a description of a real time application that would run the algorithm is presented. The final influence scores are compared against a prominent online influence measuring service called Klout. Through an analysis of the proposed algorithm, this thesis shows that while having the algorithm be complex adds much needed nuance to the scoring process, in the end the amount of followers a Twitter user has is a fairly accurate measurement of that user’s influence.
Table of Contents

1. Introduction .............................................................................................................. 1
   1.1 Twitter’s Growing Influences ..................................................................................... 1
   1.2 Goal of Research ......................................................................................................... 1

2. Related Works ............................................................................................................. 2
   2.1 Published Works .......................................................................................................... 2
   2.2 Twitter Influence Measuring Services ................................................................................ 8
   2.3 Measuring Influence on Twitter .................................................................................... 8

3. Data ..................................................................................................................... 10
   3.1 Data Set Description ................................................................................................................ 10
   3.2 Data Collection Program .............................................................................................................. 11
   3.3 Excluded Target Users .................................................................................................................. 12

4. Scoring Algorithm .................................................................................................... 13
   4.1 Scoring Statistics ....................................................................................................... 13
   4.2 Early Attempts ........................................................................................................... 13
   4.3 The Final Algorithm ...................................................................................................... 14
   4.4 Recursion ...................................................................................................................... 19
   4.5 Proposal for a Real Time Application ..................................................................................... 19

5. Analysis .................................................................................................................. 23
   5.1 iScore ................................................................................................................... 23
   5.2 Versus Followers ................................................................................................................. 25

6. Conclusion ................................................................................................................ 30
List of Tables

1 Results from Cha, Haddadi, Benevenuto, and Gummadi .............................................. 3
2 Thesis Data Set Ranking Results .................................................................................. 4
3 Mentions vs Followers ................................................................................................... 5
4 Unique Mentions vs Followers ...................................................................................... 7
5 Top 10% of users in data set based on Followers (after pruning) ................................. 11
6 Basic statistics used in influence calculation ............................................................... 13
7 uMScore ...................................................................................................................... 15
8 uMSTScore, fScore and iScore Statistics .................................................................... 19
9 iScore vs Klout ............................................................................................................ 23
10 iScore vs Followers ..................................................................................................... 26
11 Klout vs Followers ..................................................................................................... 26
12 Scaled Euler’s vs Followers ....................................................................................... 28
13 Scaled Euler’s vs iScore ............................................................................................ 28
14 Scaled Euler’s vs Klout ............................................................................................. 29
List of Figures

1 Mentions vs Followers Graph .................................................................................................. 5
2 Unique Mentions vs Followers Graph .................................................................................. 6
3 Function for calculating the diminishing return on the uMScore ...................................... 21
4 Main algorithm function with recursion ............................................................................. 22
5 iScore vs Klout Graph ......................................................................................................... 23
6 iScore (Seperated by uMSTScore) vs Klout Graph ............................................................. 24
7 iScore vs Followers Graph .................................................................................................. 25
8 Klout vs Followers Graph ................................................................................................... 26
9 Scaled Euler’s vs Followers Graph ..................................................................................... 27
10 Scaled Euler's vs iScore Graph .......................................................................................... 28
11 Scaled Euler’s vs Klout Graph .......................................................................................... 29
## List of Symbols

<table>
<thead>
<tr>
<th>Symbol</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>aIS</td>
<td>Assumed Influence Score</td>
</tr>
<tr>
<td>fDC</td>
<td>Follower Divide Constant</td>
</tr>
<tr>
<td>fScore</td>
<td>Followers Score</td>
</tr>
<tr>
<td>iScore</td>
<td>Influence Score</td>
</tr>
<tr>
<td>uMScore</td>
<td>Unique Mentions Score</td>
</tr>
<tr>
<td>uMSTScore</td>
<td>Unique Mentions Score to Tweets Score</td>
</tr>
</tbody>
</table>
1 Introduction

1.1 Twitter’s Growing Influences

Over the past six and a half years Twitter has grown from a small side project, designed as a text message sharing service, to one of the most popular social media websites in the world, worth around $10 to $15 billion USD. Twitter currently has over 550 million active users ("Twitter Statistics."). The users range from having no followers, a follower being a person who is subscribed to the user’s Twitter feed, to people with over 45 million followers ("Twitter top 100: Most followers."). With this ever growing number of users, both producers of information and consumers of that information, how than can the influence of a specific user be measured?

1.2 Goal of Research

The goal of this research is to develop and demonstrate an algorithm for measuring the level of influence a Twitter user, referred to as a target user in this thesis, has on their followers. Through the use of various publicly available statistics about a target user pulled from Twitter’s public API, a score out of 100 will be calculated for the target user. This score is referred to as the iScore in this thesis. Using this algorithm as a model, a recursive version of the algorithm will be detailed. Finally a real time application running the algorithm will also be presented, as well as the resulting issues from creating and running such an application.
Related Works

2.1 Published Works

Many other works and online services have tackled the complex issue of measuring influence on Twitter, with varying methods and success. A PageRank style measuring system, referred to as TwitterRank, took a non-random selected sample of 6,748 users from Singapore and showed that their dataset was highly reciprocal. This means that most of the Twitter users followed as many people as they had followers. The researchers came to the conclusion that “homophily does exist in the context of Twitter” (Weng, Lim, Jiang, He 2). They go on to explain that, “In the context of Twitter, homophily implies that a twitterer follows a friend because she is interested in some topics the friend is publishing, and the friend follows back because she finds they share similar topical interest” (Weng, Lim, Jiang, He 3). Their data showed that 72.4% of the Twitter users followed 80% or more of their own followers. Using this data and the idea of homophily in Twitter, Weng, Lim, Jiang and He created their PageRank style measuring system, TwitterRank.

Another article disputes the claim of homophily, by analyzing a much larger data set from Twitter (Cha, Haddadi, Benevenuto, Gummadi). Using their nearly complete picture of Twitter’s user base and activity, they concluded that only about 10% of Twitter users were following most of their followers. They looked specifically at three measures: number of followers, number of retweets and number of mentions. Then they ranked their users by these measures and compared the results. It was found that the number of followers does not correlate that well to the number of retweets or the number of mentions when dealing with the top 10% of users in their data set with regards to the number of followers. To make
this comparison they used Spearman’s rank correlation coefficient to measure the correlation between the ranking sets.

$$
\rho = 1 - \frac{6 \sum (x_i - y_i)^2}{N^3 - N}
$$

The variable $x_i$ in the formula above represents the ranking of user i with the first ranking set, and the variable $y_i$ represents the same user’s rank with the second ranking set. The size of the two ranking sets is represented by N. The formula produces a value between -1 and 1, representing how closely the two ranking sets correlate to each other.

For the sake of comparison, the same analysis was performed on the data set used in this thesis. The results are shown below along with the results of the more complete picture of Twitter’s user base from Cha, Haddadi, Benevenuto, and Gummadi. The Top 1% category is excluded from the thesis data set as its only 1 user and thus the correlation value becomes irrelevant. The top 10% and top 1% are based on follower numbers “in the hope that users who get retweeted or mentioned must have some followers” (Cha, Haddadi, Benevenuto, and Gummadi 13).

<table>
<thead>
<tr>
<th></th>
<th>All</th>
<th>Top 10%</th>
<th>Top 1%</th>
</tr>
</thead>
<tbody>
<tr>
<td>Followers vs Retweets</td>
<td>0.549</td>
<td>0.122</td>
<td>0.109</td>
</tr>
<tr>
<td>Followers vs Mentions</td>
<td>0.638</td>
<td>0.286</td>
<td>0.309</td>
</tr>
<tr>
<td>Retweets vs Mentions</td>
<td>0.580</td>
<td>0.638</td>
<td>0.605</td>
</tr>
</tbody>
</table>

*Table 1: Results from Cha, Haddadi, Benevenuto, and Gummadi*
<table>
<thead>
<tr>
<th></th>
<th>All</th>
<th>Top 10% (7 users)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Followers vs Retweets</td>
<td>0.420</td>
<td>0.357</td>
</tr>
<tr>
<td>Followers vs Mentions</td>
<td>0.847</td>
<td>0.464</td>
</tr>
<tr>
<td>Retweets vs Mentions</td>
<td>0.595</td>
<td>0.786</td>
</tr>
<tr>
<td>Followers vs Unique Mentions</td>
<td>0.886</td>
<td>0.607</td>
</tr>
<tr>
<td>Mentions vs Unique Mentions</td>
<td>0.991</td>
<td>0.857</td>
</tr>
<tr>
<td>Retweets vs Unique Mentions</td>
<td>0.593</td>
<td>0.571</td>
</tr>
</tbody>
</table>

*Table 2: Thesis Data Set Ranking Results*

Right away you can see that the correlation values between all pairs in both data sets are similar, but there is a much higher correlation in the thesis data set for followers vs mentions. This shows that the ratio of mentions to followers remains relatively constant in thesis data set regardless of the actual number of mentions and followers. However, this does not hold true for the data set of Cha, Haddadi, Benevenuto, and Gummadi. In the thesis data set when one of the values increases, be it mentions or followers, so does the other value. This means the ranking between the two statistics remains mostly the same.

The highest correlation for all pairs, however, is between Mentions and Unique Mentions. While not all that significant when it comes to measuring influence, it does highlight an interesting point. The point being that most users that post any number of mentions, referred to in this thesis as a mentioning user, tend to post the same number of mentions as every other mentioning user on average. Also it can be concluded that an increase in users mentioning a target user will result in a very predictable increase in total mentions of that target user.
The top 10% of users in the thesis data set are somewhere between the correlation values of all pairs and the top 10% in the Cha, Haddadi, Benevenuto, and Gummadi data set. The major exception to this being the retweets vs mentions correlation which is very high in the thesis data set. This means that the ratio of retweets to mentions is remaining relatively constant. According to the paper, a high number of mentions indicates that the
mentioning users have a large interest in involving themselves in the lives of the target users. “The most mentioned users were mostly celebrities. Ordinary users showed a great passion for celebrities, regularly posting messages to them or mentioning them, without necessarily retweeting their posts” (Cha, Haddadi, Benevenuto, and Gummadi 4). Since the thesis data set consists entirely of YouTube personalities, which could be considered internet celebrities, the large number of mentions some of them get would be in agreement with the research paper. However, the high correlation of retweets to mentions for the top 10% in the thesis data set also indicates that those target users deliver information deemed important enough to retweet in addition to mentioning. According to Cha, Haddadi, Benevenuto, and Gummadi, a high number of retweets normally indicated the user was a content aggregation service, a businessman, or a news site.

![Unique Mentions vs Followers Graph](image)

\[ y = 0.0522x + 288.24 \]

\[ R^2 = 0.667 \]

*Figure 2: Unique Mentions vs Followers Graph*
<table>
<thead>
<tr>
<th>Followers</th>
<th>Unique Mentions</th>
<th>Correlation</th>
</tr>
</thead>
<tbody>
<tr>
<td>Max</td>
<td>216254</td>
<td>Max 15707</td>
</tr>
<tr>
<td>Min</td>
<td>70</td>
<td>Min 9</td>
</tr>
<tr>
<td>Average</td>
<td>38463.59</td>
<td>Average 2294.93</td>
</tr>
<tr>
<td>Median</td>
<td>19603</td>
<td>Median 1034</td>
</tr>
</tbody>
</table>

*Table 4: Unique Mentions vs Followers*

The biggest discrepancy between all pairs and the top 10% of the thesis data set is a decrease in the correlation of followers vs unique mentions. That means that the top 10% of users based on number of followers had less unique mentions per follower than the rest of the data set. This reiterates a point made in the research paper, that the number of followers cannot be used by itself as an indicator of the influence a specific target user holds.

Another article by Bakshy, Hofman, Mason and Watts, explored the idea of tracking influence through the monitoring of links posted on Twitter. Their data set consisted of nearly 1.6 million seed users who had posted a bit.ly URL. It also contained a subset of Twitter’s follower tree containing around 56 million users and 1.7 billion edges, with edges being follower and friend connections on Twitter. Using the timestamps of the tweets containing a specific bit.ly URL and their near complete follower graph of Twitter, they were able to create a directional graph indicating how the bit.ly URL propagated through Twitter. With this information they were able to assign influence scores to the different users along the chain based on how many people tweeted the URL because of them.
2.2 Twitter Influence Measuring Services

With Twitter’s API being open for anyone to use, there are many services out there that will evaluate a Twitter user and produce an influence score. This includes services like Klout and Marketing Grader. Klout gathers data from Twitter, Facebook, LinkedIn and other social networking sites, and from this data it calculates a score out of 100 to represent a user’s influence. In this thesis the Klout scores of the users in the data set are used for comparison with the scoring algorithm presented below. While the exact details of how the Klout score is calculated are not publicly available, a general overview of their scoring process is laid out on their website. It is stated that the Klout score relies on “the ratios of reactions you generate compared to the amount of content you share” (“How It Works.”). This same philosophy is used in the scoring algorithm presented below with the number of mentions to the number of tweets. Also stated is that Klout “also consider[s] factors such as how selective the people who interact with your content are. The more a person likes and retweets in a given day, the less each of those individual interactions contributes to another person’s Score” (“How It Works.”). This indicates a PageRank approach to the scoring, which would require a large amount of data to calculate a score.

2.3 Measuring Influence on Twitter

The article by Anger and Kittl titled Measuring Influence on Twitter, serves as the primary inspiration for the work in this thesis. They look at several “Twitter functionalities and performance indicators” to determine their final influence scoring. These indicators include the Follower/Following Ratio ($r_f$), the Retweet and Mention Ratio ($r_{RT}$), and the Interactor Ratio ($r_i$) (Anger, Kittle 1). The Follower/Following Ratio indicates the level of interest a target user’s followers have in what they have to say without the target user
having to show interest back towards their followers. The Retweet and Mention Ratio is the number “of tweets that are amplified or lead to a communicative action” between the target user and another Twitter user divided by the number of tweets made by the target user (Anger, Kittle 4). The Interactor Ratio is the number of individual users who retweet the target user’s tweets or mention the target user divided by the total amount of followers. Both the $r_{RT}$ and the $r_i$ are used in part in the algorithm presented below for calculating the influence of a target user.

Anger and Kittle use these indicators to calculate their own influence score, or social networking potential (SNP) as they call it. Using the $r_{RT}$ and the $r_i$, they sum them up and divide by 2 to calculate the SNP. They ran this scoring against the top 10 Twitter users in Austria based on the number of followers, with some interesting results. The number 2 target user based on number of followers was the number 1 user based on SNP, while the number 1 target user, ArminWolf, was 4th based on SNP. When comparing their final SNP scores with the Klout scores of they found that the Klout scores lined up with the number of followers ranking, placing ArminWolf with the highest Klout score. They go on to mention that they specifically avoided placing a large emphasis on the number of followers, even though many online influence measuring services do. This is because “quantity does not equal quality and a small audience of engaged users is worth more than a large audience of less active users” (Anger, Kittle 4).
3 Data

3.1 Data Set Description

The thesis data set consists of the Twitter accounts belonging to the YouTube channels that are part of the Polaris network. The Polaris network is made up of 96 YouTube channels as of this writing, and when data collection started 75 of those channels had Twitter accounts. For 5 weeks in August and September of 2013, a custom built program collected the tweets and mentions of each target user. Mentions of a target user are defined as any tweet that contains the @ symbol followed by a target user’s screen name somewhere in the tweet. The tweets of each target user and their profile information, which used the Timeline portion of the Twitter API, were collected about 2.4 times every 15 minutes. This limit was due to Twitter limiting the number of calls to any individual part of their API to 180 calls per 15 minutes per IP address. Mentions used the Search portion of the Twitter API, and thus were on a separate limit, but do to the vastly greater amount of data each call collected only about 90 calls could be completed in 15 minutes. Each call collected 50 tweets and profile data of the users who posted the tweets. These limits were not really an issue for the purposes of this thesis as the data set was not that large, but they would cause significant issues for any type of real time application using the algorithm presented below for calculating influence. This problem is explored more fully in section 4.

The data set was selected for use in this research because all of the target users are influencers. In the strictest way, this means that the followers to friends ratio is heavily skewed towards followers. In other words the target users in the data set are not at all reciprocal with their followers. The target users’ follower counts range from 70 all the way up to 1.46 million, but friend counts ranging only from 0 up to 485. They share
characteristics with celebrities in that they have a high number of mentions due to people wanting to interact with them, but also they produce a steady stream of news that people follow them for, mainly tweets about new videos they have posted.

<table>
<thead>
<tr>
<th></th>
<th>Followers</th>
<th>Friends</th>
<th>Retweets</th>
<th>Mentions</th>
</tr>
</thead>
<tbody>
<tr>
<td>Totalbiscuit</td>
<td>216254</td>
<td>107</td>
<td>22557</td>
<td>38442</td>
</tr>
<tr>
<td>Deadloxx</td>
<td>163456</td>
<td>74</td>
<td>6142</td>
<td>15107</td>
</tr>
<tr>
<td>day9tv</td>
<td>163184</td>
<td>315</td>
<td>5867</td>
<td>7400</td>
</tr>
<tr>
<td>InTheLittleWood</td>
<td>161454</td>
<td>485</td>
<td>19852</td>
<td>18168</td>
</tr>
<tr>
<td>HuskyStarcraft</td>
<td>119221</td>
<td>149</td>
<td>8613</td>
<td>5914</td>
</tr>
<tr>
<td>lomadia</td>
<td>106322</td>
<td>218</td>
<td>1386</td>
<td>3516</td>
</tr>
<tr>
<td>JesseCox</td>
<td>105836</td>
<td>65</td>
<td>7163</td>
<td>17915</td>
</tr>
</tbody>
</table>

Table 5: Top 10% of users in data set based on Followers (after pruning)

3.2 Data Collection Program

The data collection program cycled through the users continuously, making calls to the Twitter API acquiring the users’ tweets, profile info and their most recent mentions. Due to strict limitations of 150 requests per hour put on the Twitter API when I started my research, the program was originally designed as a message passing parallel program. The setup consisted of a master server, using a php script and a mysql database, and any number of clients, written in C#. Each client upon start up would check in with the master server, and the master server would assign target users to the clients for data collection. The client would then collect the target user’s tweets, profile info and recent mentions. After which the data was transmitted back to the master server, in XML format, where it was processed and stored in a database for later analysis. If a client did not complete the data
collection for their assigned target user within a certain period of time the target user would be reassigned to another client. Once all of the target users had their data collected, the master server would start over with the assignments. Twitter eventually loosened up on their restrictions, allowing 180 requests per 15 minutes. Due to the small size of the test data set it became unnecessary to use anymore than a single computer in this setup for data collection. However, for future work and testing the system could be easily expanded to collect larger data sets simply by adding more clients.

3.3 Excluded Target Users

In addition to certain YouTube channels being excluded from the data set due to a lack of a Twitter account, 4 others channels that had accounts were also excluded. The top 3 target users based on follower counts had 1.46 million, 850 thousand, and 576 thousand, while the rest of the users had less than 217 thousand followers. This presented a massive gap between the top 3 target users and everyone else, skewing the data set results. This was particularly apparent when attempting to show the data in any form of graph involving the number of followers. Therefore, those top 3 target users were removed from the data set. One other target user was also removed due to Klout not being able to find them and give a score. Since Klout is being used in this thesis as the primary statistic for comparison of the final scoring, the user needed to be removed.
4 Scoring Algorithm

4.1 Scoring Statistics

From the data set collected several statistics are available for each user to be scored upon, including the number of followers, the number of tweets, the number of mentions and the number of unique mentions. The number of unique mentions is defined as the number of users who mentioned the target user in the time period in question.

<table>
<thead>
<tr>
<th></th>
<th>Followers</th>
<th>Tweets</th>
<th>Mentions</th>
<th>Unique Mentions</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Min</strong></td>
<td>70</td>
<td>20</td>
<td>20</td>
<td>9</td>
</tr>
<tr>
<td><strong>Max</strong></td>
<td>216,254</td>
<td>3117</td>
<td>38442</td>
<td>15707</td>
</tr>
<tr>
<td><strong>Average</strong></td>
<td>38,463.59</td>
<td>397.85</td>
<td>5251.07</td>
<td>2294.93</td>
</tr>
</tbody>
</table>

*Table 6: Basic statistics used in influence calculation*

4.2 Early Attempts

Early attempts as creating a scoring algorithm focused around two statistic ratios in particular, the mentions to unique mentions ratio and the mentions to tweets ratio. The mentions to unique mentions ratio allows for the inclusion of not only the raw number of mentions a user receives but also the number of users that it took to get those mentions. To make the ratio usable for calculating an influence score out of 100 the ratio was reversed, with the number of unique mentions being divided by the number of mentions. This guaranteed that the resulting ratio was bounded between 0 and 1. Reversing the ratio in this manner also reflected the idea that having more users mentioning the target user a few times resulted in more overall influence than a few users mentioning the target user many
times each. For example, a target user with 10 unique mentions and 100 overall mentions would get a score of 0.10, meaning on average each mentioning user mentioned the target user 10 times. While a target user with 50 unique mentions and 100 overall mentions would get a score of 0.50, with each mentioning user averaging 2 mentions.

The mentions to tweets ratio was used to show just how much content the target user needed to produce in order to acquire the amount of responses they had. To make the ratio usable in a score out of 100, it was reversed. This resulted in a value between with a minimum of 0 and no upper bound, here being one of the major issues with this scoring system. If a user produced more tweets than mentions their ratio would be greater than one, otherwise their ratio would be bound between 0 and 1. The closer the ratio is to 0, the fewer tweets were required to elicit the resulting number of mentions. To make the ratio useful in the actual scoring, it was subtracted from 1.

Once the ratios has been transformed into useful states they were added together, divided by 2 and multiplied by 100. The result was a score out of 100 indicating the influence of the user. However, for users with more tweets than mentions, their score could end up being negative. Due to that issue, this scoring system was abandoned in favor of the more complex scoring algorithm presented below.

4.3 The Final Algorithm

The algorithm presented here focuses heavily on mentions, similar to the first scoring formula attempt, as mentions are real tangible evidence of the mentioning user being influenced by the target user. However, the first issue you run into with using mentions is that not every mention will gain the target user the same amount of influence. This is due to a single user mentioning the target user several times, hence where the
number of unique mentions comes into play. Using the unique mentions to total mentions ratio makes it so you can count each mention equally, however even this does not give you the whole picture of the target user’s influence through mentions. Just because a user mentions the target user once, it does not mean they have been completely influenced and no more influence can be gained from a second, third or even fourth mention from the same mentioning user. This is why for the algorithm presented here utilizes a new statistic referred to as the uMScore, or unique mentions score, which is derived from the mentions. The uMScore is calculated by totaling up the number of mentions from each unique mentioning user, and then a diminishing return is applied to that number to get the mentioning user’s individual uMScore. These individual uMScores are then totaled to create the final uMScore for the target user. Below is the mathematical formula for calculating the uMScore.

\[
\text{uMScore} = \sum_{i=1}^{n} \left( 1 + \sum_{j=1}^{f(x_i)-1} e^{-0.4 \times j} \right)
\]

\[f(x_i) = \text{Number of mentions of the target user made by user } x_i\]

\[n = \text{Number of unique users who mentioned the target user}\]

<table>
<thead>
<tr>
<th></th>
<th>uMScore</th>
</tr>
</thead>
<tbody>
<tr>
<td>Min</td>
<td>12.96</td>
</tr>
<tr>
<td>Max</td>
<td>23,195.77</td>
</tr>
<tr>
<td>Average</td>
<td>3234.72</td>
</tr>
</tbody>
</table>

*Table 7: uMScore*
Now the uMScore by itself is not sufficient to give a complete picture of a target user’s influence, as it does not take into account the amount of information produced by the target user in order to get the mentions necessary to achieve their uMScore. Therefore in the scoring process we look at the uMScore per tweet. This will help us distinguish between a target user who tweets 1000 times and gets 1000 mentions and a target user who tweets 1 time and gets 1000 mentions. The second user clearly has more influence over their followers than the first user does, and the influence scores for the two users will reflect this.

The last part of the scoring algorithm looks at the number of followers the target user has acquired. While this statistic on its own is not all that informative it is still important. If the algorithm only focused on the uMScore to tweets ratio, a user with 5,000 followers could end up with the same score as a user with 1,000,000 followers. There is still a big enough difference between the number of followers of the two users that their overall influence should vary significantly if their uMScore to tweets ratios is the same or close to each other’s ratios. However, there is an issue with using the follower count, as the number of followers has no upper bound and can vary over a large range. A difference in the number of followers is also more significant when those numbers are low. The influence difference between a user with 100 followers and a user with 600 followers should be more significant than the influence difference between a user with 1,600,000 followers and a user with 1,600,500 followers. To that end the formula for calculating Euler’s number is used to simulate the diminishing returns on the number of followers to be used in the score. This portion of the score is referred to here as the fScore. The fScore formula utilizes a constant called the follower divide constant (fDC) that is used to prevent the fScore from heavily tending towards 1. The values of the fDC that produce the best results, specifically for the thesis data set, are between 5,000 and 15,000. For the purposes of this thesis the fDC is
set to 6216, which was determined by optimizing the constant to best correlate the final influence scores to the Klout scores that are being used for comparison. The mathematical formula for the calculation of the fScore is as follows:

$$fScore = \left(1 + \frac{1}{\frac{followerCnt}{fDC}}\right)^{\frac{followerCnt}{fDC}}$$

$$followerCnt = \text{The number of followers of the target user} \quad fDC = \text{The follower divide constant.}$$

To be able to compare the final influence score with other influence scores, such as Klout, it needed to be a score between 0 and 100. This presented a problem for the uMScore to tweets ratio, as the majority of the users in the data set ended up with ratios greater than 1 with no upper bound. To turn this unbounded number into something usable for the scoring process, the ratio was reversed. Instead of taking the uMScore divided by the number of tweets, the score takes the number of tweets divided by the uMScore and subtracts that from 1. This modified score is referred to as the uMSTScore, defined by the following formula:

$$uMSTScore = 1 - \frac{\text{ Tweets}}{uMScore}$$

This guarantees an upper bound of 1, and for any user that had a uMScore to tweets ratio greater than or equal to 1 there was a lower bound of 0. For any user with a uMScore to tweets ratio less than 1, they would have a uMSTScore less than 0 with no lower bound. It is at this point that two separate scoring formulas are needed, one for the positive uMSTScores and one for the negative uMSTScores. For the negative uMSTScores the
algorithm again uses the idea of diminishing returns to transform the uMSTScore into a score out of 100. The formula utilizes a constant called the assumed influenced score or aIS, which represents the assumed final influence score, the iScore, of users with an uMSTScore of 0. The aIS constant was optimized alongside the fDC, the follower divide constant, against the correlation of the iScores to the Klout scores. For this data set the aIS was optimized to 86. The formula for the iScore of users with a negative uMSTScore is as follows:

\[
iScore = aIS + \left( aIS \times (-1 + e^{(-1 \times fScore \times |uMSTScore|)}) \right)
\]

\[
aIS = \text{The assumed iScore of users with a USTScore of 0}
\]

For users with a positive uMSTScore, we do not need to use a formula with diminishing returns as the uMSTScores are bounded between 0 and 1. For these users a different approach was taken. The aIS constant is utilized as an upper bound for the iScore of users with negative uMSTScores, so for users with positive uMSTScores the aIS is treated as the lower bound with 100 being the upper bound. These bounds hold true for the users with positive uMSTScores until the fScore is applied. To prevent the scores from heavily tending towards 100, they are weighted towards the aIS by squaring the uMSTScore. The formula for the iScore for users with a positive uMSTScore is as follows:

\[
iScore = \left( (uMSTScore^2 \times (100 - aIS)) + aIS \right) \times fScore
\]
Table 8: uMSTScore, fScore and iScore Statistics

<table>
<thead>
<tr>
<th></th>
<th>uMSTScore</th>
<th>fScore</th>
<th>iScore</th>
</tr>
</thead>
<tbody>
<tr>
<td>Min</td>
<td>-2.78</td>
<td>0.39</td>
<td>29.32</td>
</tr>
<tr>
<td>Max</td>
<td>0.99</td>
<td>0.99</td>
<td>97.62</td>
</tr>
<tr>
<td>Average</td>
<td>0.57</td>
<td>0.82</td>
<td>79.11</td>
</tr>
</tbody>
</table>

4.4 Recursion

Now that the basis of the scoring algorithm is setup, the idea of recursion can be applied to the algorithm. The most logical place for recursion to occur is during the calculation of the uMSTScore, in which we increase the individual uMSTScore of the mentioning user by a percent equal to their calculated iScore. This means at most the score can be doubled. It also means that the algorithm without the recursion is the same as the algorithm with recursion given that all of the mentioning users have an iScore of 0. To achieve an iScore of zero you must have have a uMSTScore of 0 or no followers. As a result of this behavior, the iScore can be calculated either with no recursion, partial recursion or full recursion and it will never overestimate the final iScore. This idea is critical to the proposed real time application to calculate iScores.

4.5 Proposal for a Real Time Application

There are several challenges in turning the algorithm into a real time application that could be used on any target user. First and foremost is the amount of data needed to fully calculate a user’s iScore. At minimum it requires the number of tweets, all mentions, and the number of followers of the target user in the given time period. For any target user with only a few mentions this is not a problem, as all of this information could be collected in as
few as 2 requests to the Twitter API. However, if the target user has a large amount of mentions it could take significantly longer, as Twitter caps the number of mentions you can pull with a single request to 200. There is also the issue of how far back to go with data collection. If you do not go back far enough you will not get an accurate picture of the target user’s influence, but if you go back too far you run into a couple of problems. First, the influence score will become too heavily weighted by the target user’s past influence. Secondly, the amount of data you would need to collect will continue to grow, which will make any real time application very impractical. For the proposed application, data will be collected for the past week. This will allow for the influence score to reflect the current influence of the target user over their past influence, as well as keep the amount of data that needs to be collected low. It will, however, go back far enough so that it should not vary wildly from day to day.

The real issues arise when the application tries to do the recursive part of the algorithm, as with more popular target users we could be looking at tens of thousands of mentions in that time period. In the thesis data set the largest number of mentions of a target user over the 5 week period was 171,507 mentions, which is about 34,000 mentions per week. To calculate the iScore of that user recursively even with just one level of recursion would take several days due to the limitations of placed on calls to the Twitter API. This is where the idea of partial recursion comes into play. The application would start by collecting the data for the target user, but not for any of the mentioning users. It would then calculate an initial iScore from that data, assuming that all mentioning users had an iScore of 0. This number would be displayed in the application and marked as an initial score. From there the application would begin gathering data for the mentioning users, calculating their iScores and one by one updating the target user’s iScore with the new information. This updated score would be displayed in the application and marked as partial
recursion with a percentage to indicate how many of the mentioning users have been scored. This would allow for a quick score to be displayed, as well as allow the application user to wait for a more accurate score if so desired. Given enough time the application would complete a full level of recursion as which point it would display in the application that full recursion has been completed for 1 level and would update the iScore of the target user appropriately.

Even this approach however is not completely practical, as previously stated for popular target users it would take several days to complete full first level recursion. For target users with a smaller amount of mentions this process would go much faster, but it would still take some time. In the end this is only one level of recursion and further levels would take exponentially more time, especially if one of the mentioning users had a large amount of followers and mentions. Another issue with going more than one level of recursion deep is that you will begin to see loops that would inflate the score out of control. At that point you would need to implement a similar system to PageRank in which the amount of influenced transferred decays over time to prevent loops from effecting the final scoring. Below is the pseudocode for the recursive version of the algorithm:

```plaintext
function calcDeminishingScore (value v)
    decayRate = -0.4 //Controls the rate of deminishing returns. Makes max score ~3
    score = 10

    for i = 1 to v
        score += e^(decayRate * i)
    endfor

    return score
```

*Figure 3: Function for calculating the diminishing return on the uMScore*
function scoreUser(userToScore uts)

mList = list of mentions from last week for user uts
mCounts = hash table of user -> number of mentions

for each mention m in mList
    if posting user pu of m is in mCounts
        add 1 to value for pu in mCounts
    else
        add user pu to mCounts with a value of 1
    endif
endfor

uMScore = 0 //Unique Mentions Score for user uts

for each user u in mCounts
    uScore = calcDeminishingScore(value from mCounts for user u)
    uMScore += (uScore * (scoreUser(u) + 1)) //Recursive call.
endfor

count = number of followers of user uts
tweetCount = number of tweets

if uMScore equals 0 OR count equals 0
    return 0
endif

followerDivideConstant (fDC) = 6216 // Determined by optimizing variable
assumedInfluenceScore (aIS) = 86 // Determined by optimizing variable

fScore = (1 + (1 / (followerCnt / fDC))) ^ (followerCnt / fDC)

uMSTScore = 1 – (tweets / uMScore)

if uMSTScore is greater than or equal to 0
    iScore = ((uMSTScore ^ 2) * (100 – aIS)) * fScore
else
    iScore = aIS + (aIS * (-1 + e ^ (-1 * fScore * uMSTScore)))
endif

return userScore

Figure 4: Main algorithm function with recursion
The program begins by taking a username as input, referred to as uts in the pseudocode. From there it gathers a list of all mentions and mentioning users of the target user from the last week. It loops through this list counting how many mentions each unique mentioning user made in that time. It then uses these counts to calculate the uMScore, by applying diminishing returns to each of the mentioning users' counts. This is where the recursive call comes in to play. When calculating each of the mentioning users' section of the uMScore you multiply their part by 1 plus their influence score divided by 100. This means if they have an influence score of 0, their part of the uMScore will only count once, but if they have an influence score of 100 it will count twice. From there the number of followers and tweets the target user has is gathered. If either the uMScore is 0, aka no one mentioned them, or they have 0 followers their final influence score will be 0 and no further calculation is needed. Now two different values need to be calculated, the mentions and tweets score (uMSTScore), which utilizes the uMScore, and the follower score (fScore). The fScore is a number between 0 and 1 that as the number of followers approaches infinity it approaches 1. The uMSTScore is simply the number of tweets over the uMScore all subtracted from 1. The final formula to calculate the influence score is than determined by whether the uMSTScore is positive or negative.
5  Analysis

5.1  iScore

Analysis of scoring algorithm presented here, shows that with a little complexity a coherent influence scoring algorithm can be created, but at the same time could use a lot more tweaking. One of the major concerns is that even though iScores of target users with

![iScore vs Klout Graph](image)

*Figure 5: iScore vs Klout Graph*

<table>
<thead>
<tr>
<th>Klout</th>
<th>iScore</th>
<th>Correlation</th>
</tr>
</thead>
<tbody>
<tr>
<td>Max</td>
<td>81</td>
<td>Max 97.62</td>
</tr>
<tr>
<td>Min</td>
<td>25</td>
<td>Min 29.32</td>
</tr>
<tr>
<td>Average</td>
<td>57.1831</td>
<td>Average 79.11</td>
</tr>
<tr>
<td>Median</td>
<td>57</td>
<td>Median 83.43</td>
</tr>
</tbody>
</table>

*Table 9: iScore vs Klout*
positive uMSTScores were weighted towards the aIS, which was 86, many of the users still had iScore very close to 100. This indicates that some more subtlety is needed in the scoring process. While this algorithm focused very heavily on mentions, including the number of retweets of each of the target user’s tweets might have helped to solve this issue. However, Twitter does not currently have a tool in their API to see who retweeted a particular tweet. This leaves us with just a raw number, removing any possibility of the recursive call involving the retweets.

As you can see in the graphs above, the resulting iScores correlated very strongly to the collected Klout scores. Even in some cases almost perfectly matching. Such is the case with target user BenjaGPDK who had a Klout score of 25 and an iScore of 29.32. In figure 6
you can see that there is some overlap between the iScores of the target users with positive uMSTScores and those with negative uMSTScores. The highest iScore for a target user with a negative uMSTScore was 85.45 for JohnTarrJr, who had an uMSTScore of -0.01 and an fScore of 0.57. The lowest iScore for a target user with a positive uMSTScore was 52.12 for Sonic_Paradox, who had an uMSTScore of 0.79 and an fScore of 0.55.

5.2 Versus Followers

Two of the major issues with the iScore are that it relies a little bit too much on the number of followers, and that the fScore is handled very differently in the two formulas for the iScore. As demonstrated above with JohnTarrJr and Sonic_Paradox, a target user with a negative uMSTScore can have a far higher iScore than a target user with a positive

![iScore vs Followers Graph](image)

\[ y = 0.0002x + 70.225 \]

\[ R^2 = 0.4518 \]

*Figure 7: iScore vs Followers Graph*
<table>
<thead>
<tr>
<th>Followers</th>
<th>iScore</th>
<th>Correlation</th>
</tr>
</thead>
<tbody>
<tr>
<td>Max</td>
<td>216254</td>
<td>Max 97.62</td>
</tr>
<tr>
<td></td>
<td></td>
<td>0.672172092</td>
</tr>
<tr>
<td>Min</td>
<td>70</td>
<td>Min 29.32</td>
</tr>
<tr>
<td>Average</td>
<td>38463.59</td>
<td>Average 79.11</td>
</tr>
<tr>
<td>Median</td>
<td>19603</td>
<td>Median 83.43</td>
</tr>
</tbody>
</table>

*Table 10: iScore vs Followers*

```
y = 0.0001x + 51.441
R² = 0.5407
```

<table>
<thead>
<tr>
<th>Followers</th>
<th>Klout</th>
<th>Correlation</th>
</tr>
</thead>
<tbody>
<tr>
<td>Max</td>
<td>216254</td>
<td>Max 81</td>
</tr>
<tr>
<td></td>
<td></td>
<td>0.735351068</td>
</tr>
<tr>
<td>Min</td>
<td>70</td>
<td>Min 25</td>
</tr>
<tr>
<td>Average</td>
<td>38463.59</td>
<td>Average 57.1831</td>
</tr>
<tr>
<td>Median</td>
<td>19603</td>
<td>Median 57</td>
</tr>
</tbody>
</table>

*Table 11: Klout vs Followers*
uMSTScore, even though there is a larger difference in the uMSTScores and the fScores are very similar. This is an issue that needs to be addressed in future work.

An interesting result came out of the over dependency on follower, however, in that the iScore followed almost exactly the same curve created by the formula for Euler's number when scaled to 100. This is shown in figure 7. Since the correlation between Klout and the iScore is so high and the graph of Klout vs Followers, shown in figure 8, had a similar shape it became necessary to see how close the two measures actually came to matching up with the formula for Euler's number. The formula, called Scaled Euler’s here, is shown below in figure 9. The results of the Scaled Euler’s versus the iScore and Klout are shown in figures 10 and 11 respectively.

![Figure 9: Scaled Euler's vs Followers Graph](Image)

\[ y = 0.0003x + 61.67 \]
\[ R^2 = 0.5982 \]
Followers | Scaled Euler's | Correlation
--- | --- | ---
Max | 216254 | Max | 96.46 | 0.773408688
Min | 70 | Min | 37.66 |
Average | 38463.59 | Average | 72.81 |
Median | 19603 | Median | 76.12 |

Table 12: Scaled Euler's vs Followers

![Figure 10: Scaled Euler's vs iScore Graph](image)

Table 13: Scaled Euler's vs iScore

<table>
<thead>
<tr>
<th>iScore</th>
<th>Scaled Euler's</th>
<th>Correlation</th>
</tr>
</thead>
<tbody>
<tr>
<td>Max</td>
<td>97.62</td>
<td>Max</td>
</tr>
<tr>
<td>Min</td>
<td>29.32</td>
<td>Min</td>
</tr>
<tr>
<td>Average</td>
<td>79.11</td>
<td>Average</td>
</tr>
<tr>
<td>Median</td>
<td>83.43</td>
<td>Median</td>
</tr>
</tbody>
</table>
As you can see in figure 10, the iScore puts a massive emphasis on the number of followers, especially for those users with positive uMSTScores. While Klout does put as large emphasis on the number of followers, it is clear that they put more on other parts of the scoring process than the iScore does. One thing to keep in mind, however, when dealing with the Klout score in comparison to the iScore is that since Klout is proprietary, there is no way to tell what the data set used to calculate the Klout score even contained. It
could have gone back further than 5 weeks, or it could have included past scores in some fashion. So part of the difference between the iScore and the Klout score must be contributed to the difference in the data set in addition to the obvious difference in how the scores are calculated.

One final issue with the iScore is due to the small size of the data set and that the two constants, the follower divide constant and the average influence score constant, were optimized to maximize the correlation between the iScore and Klout score. This could have resulted in the algorithm for the iScore being overly tailored to fit with this data set in particular. The only way to test if this hypothesis is true would be to gather a larger data set and run the same process. Once the results were in, compare the two constants to see how similar their values are to the values used with the thesis data set.

6 Conclusion

The approach presented in this thesis for calculating the influence score of a Twitter user, while more complex than other influence scoring methods, does provide a more detailed approach to the problem. Although like other methods it uses mentions as its primary statistic, it takes into account that not every mention will garnish the target user the same amount of influence. It also allows for a recursive approach that most methods lack. However, it does have much room for improvement. Specifically, in that it relies too heavily on followers, requires too much raw data to be practical on a large scale, does not handle recursion past one level deep very well, and it does not incorporate retweets. These areas would need to be the focus of any future work on this algorithm, with the data requirement being first and foremost the largest issue to tackle.
Bibliography


