7-24-2015

Classifying Political Similarity of Twitter Users

William K. Paustian
Ursinus College, wipaustian@ursinus.edu

Classifying Political Similarity of Twitter Users

William Paustian

Nikolai Peralta

Michelle Lu

Advisor: Dr. Akshaye Dhawan

Classifying Political Similarity of Twitter Users

Abstract

The emergence of large scale social networks has led to research in approaches to classify similar users on a network. While many such approaches use data mining techniques, recent efforts have focused on measuring the similarity of users using structural properties of the underlying graph representing the network. In this paper, we identify the Twitter followers of the 2016 presidential candidates and classify them as Democrat, Republican or Bipartisan. We did this by designing a new approach to measuring structural similarity, PolRANK. PolRANK computes the similarity of a pair of users by accounting for both the number of candidates they follow from each party and the specific candidates they follow. To test our algorithm, we crawled a data set of all followers of every presidential candidate in June 2015 and then ran experiments on a random subset of 10% of that data. When tested against similar algorithms, PolRANK outperforms SimRank[1], P-Rank[2] and Cosine-Similarity as it is more efficient when used in large data sets. This efficiency is due to PolRANK's ability to calculate similarity independent of other users. The time complexity of P-Rank is $O(n^4)$ while the time complexity of PolRANK is $O(n^3)$.

1. Introduction

The 2016 presidential election is coming up which means the nation is getting ready to vote on a new president. There are two main parties which participate in these elections: Democrats and Republicans. However not everyone who votes is tightly aligned to one of these political parties. These undecided people are called Independents and we hope to try and classify users on social media who identify as such. In order to do this we have created a similarity measure called **Pol-RANK** which will enable us to compute political leaning similarity scores for a pair of users on Twitter. In this paper we will present our new similarity measure Pol-Rank and compare it to several other similarity measures from the literature. We will show how Pol-Rank classifies users more accurately and with a better time complexity. Our algorithm was based on the same logic and equation structure as P-Rank [2] while editing it to suit our needs and the information we have collected on the Twitter users.

Every node in the graph representing our data set has three aspects to it: the Twitter ID, the Democrats they follow and the Republicans they follow.

The main contributions of this paper are as follows:

- Creation of a new political similarity measure, which could in the future help out and influence other similarity measures.
- Creation of a data set representing presidential candidates and their Twitter followers as of early June 2015.
- Experimental results to show how Pol-RANK measures up to other similarity measures.

2. Related Work:

Methods of calculating the relationship between different objects, similarity measures, have been utilized and evaluated in many applications, such as Structural Similarity, Cosine Similarity, Similarity Rank (Sim-Rank [1]) and Penetrating Rank (P-Rank [2]) and Minimum Similarity (Min-Sim).

Structural similarity is the similarity between two nodes within a graph that accounts for the similarity of their in and out edges and other properties. It is a way to compute the similarity between two nodes by their structure by using the underlying graphs that represent the relationships between their edges. For the remainder of this section we survey existing measures of structural similarity found in the literature. Min-Sim is a similarity measure that calculates the common neighbors between two nodes and then divides that by the cardinality of the minimum out-degree of either node. This is one of the earliest measures of similarity and derives from work in Physics.

$$Sim_{HPI}(X,Y) = \frac{|\Gamma(X) \cap \Gamma(Y)|}{\min\{K_X, K_Y\}}$$

Cosine similarity is a measure of similarity between two vectors that measure the cosine of the angle between two nodes. Therefore, the range of scores is [0, 1]. It is another similarity measure we came across and implemented on our scores. First, it takes the amount of out-neighbors of both nodes and multiplies them within a square root. Then, it computes the amount of out-neighbors that is common between the two nodes. The overall formula is the latter divided by the former that gives us the Cosine Similarity score for those two nodes. This can be represented by the equation shown below. The time complexity of Cosine Similarity is $O(n^2)$.

$$\text{similarity} = \cos(\theta) = \frac{A \cdot B}{\|A\| \|B\|} = \frac{\sum_{i=1}^{n} A_i \times B_i}{\sqrt{\sum_{i=1}^{n} (A_i)^2} \times \sqrt{\sum_{i=1}^{n} (B_i)^2}}$$

Sim-Rank [1] is a structural similarity measure implemented in information networks. It computes the similarity of context in graph structure by using the in-link relationship between two nodes to compute the similarity between them. It leaves out, however, the out-link relationships between pairs of nodes. The time complexity of Sim-Rank [1] is $O(n^4)$, yet it is not able to fully calculate the similarity between objects Sim-Rank [1] is based off of the idea that, "two objects are similar if they are related to similar objects." Sim-Rank [1] is able to calculate the similarity score by recursively calculating the Sim-Rank [1] scores of nodes that relate to the nodes we are finding the similarity score for. They do not take the out-neighbors into account since they do not find them anywhere near as important as the in-neighbors.

$$s(a,b) = \frac{C}{|I(a)||I(b)|} \sum_{i=1}^{|I(a)|} \sum_{j=1}^{|I(b)|} s(I_i(a), I_j(b))$$

Unlike Sim-Rank [1], P-Rank [2] takes both the in-neighbors and out-neighbors into account for the similarity score.

While P-Rank [2] is also a structural similarity measure that is implemented in information networks, P-Rank [2] makes use of both the in-link (in-neighbor) and out-link (out-neighbor) relationship between nodes in order to better compute the similarity between them. It is based off of the same statement just like Sim-Rank [1] is based off of but they add another statement that justifies the use of out-neighbors: "two entities are similar if they reference similar entities." P-Rank [2] is very similar to Sim-Rank [1] since P-Rank [2] still recursively calculates the scores between the in-neighbors but it calculates the scores for the out-neighbors in order to get a more accurate score. The time complexity of P-Rank [2] is identical to that of Sim-Rank [1] and although it calculates the similarity between objects more thoroughly than Sim-Rank [1], it still cannot fully calculate the similarity between nodes.

$$\begin{aligned} s(a,b) = \lambda \times \frac{C}{|I(a)||I(b)|} \sum_{i=1}^{|I(a)|} \sum_{j=1}^{|I(b)|} s(I_i(a), I_j(b)) \\ + (1-\lambda) \times \frac{C}{|O(a)||O(b)|} \sum_{i=1}^{|O(a)|} \sum_{j=1}^{|O(b)|} s(O_i(a), O_j(b)) \end{aligned}$$

The model for our similarity measure is based on our algorithm's ability to calculate the political similarity of a pair of users independent of other users and therefore, more efficiently. The time complexity of Pol-RANK is $O(n^3)$ as opposed to P-Rank [2] and Sim-Rank's [1] $O(n^4)$. Further, Pol-RANK achieves much better results as it can calculate the similarity between two nodes more extensively.

3. Notation:

Symbol	Reason
Lambda (λ)	Normalizes the entire Democratic and Republican sides of the equation. Originally set to 0.5 but may be changed.
Gamma (γ)	Normalizes each section of the Democratic side and the Republican side. Originally set to 0.3 but may be changed.
D_u, D_v	Holds the number of Democrats *u* and *v* follow respectively.
R_u, R_v	Holds the number of Republican *u* and *v* follow respectively.
$\Sigma\Sigma$ (D_{ui}, D_{vi}), $\Sigma\Sigma$ (R_{ui}, R_{vi})	Calculates the amount of Democrats and Republicans u and v share in common respectively.

4. Pol-Rank

Our objective was to design a pair-wise similarity measure that could compare two twitter users *i* and *j* and assign them a similarity score based on the structure of their connections to the 2016 candidates. Hence, we aimed to design a measure that took into account domain specific information such as party affiliations.

Pol-Rank can be written as:

1. 2.

$$s(u,v) = \lambda \left(\frac{\gamma}{||D_u| - |D_v|| + 1} + \frac{(1-\gamma) \sum_{i=1}^{|D_u|} \sum_{j=1}^{|D_v|} k(D_{ui}, D_{vj})}{(|D_u| + |D_v|)/2} \right) + (1-\lambda) \left(\frac{\gamma}{||R_u| - |R_v|| + 1} + \frac{(1-\gamma) \sum_{i=1}^{|R_u|} \sum_{j=1}^{|R_v|} k(R_{ui}, R_{vj})}{(|R_u| + |R_v|)/2} \right)$$

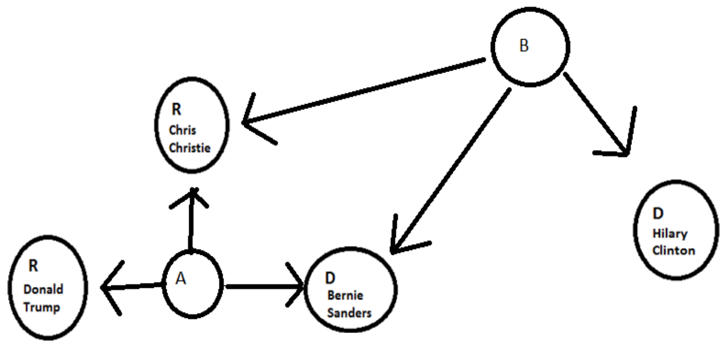
3. 4.

In order to understand how Pol-Rank works as a measure of Political similarity, the equation above can be divided into four parts, two parts for each end of the political spectrum. The sub-parts labeled above as 1 and 2 calculate the Democrat similarity for *u* and *v*. Similarly, the sub-parts labeled 3 and 4 calculate the Republican similarity. Part 1 computes the number of democrat's that *u* and *v* follow in common (similarly Part 3, computes the number of republican's *u* and *v* follow in common). Part 2 computes exactly which democrats do *u* and *v* follow in common. Pol-Rank shares ideas with P-Rank [2] by using P-Rank's [2] use of out—neighbors and in-neighbors but replacing those with Republicans and Democrats. Additionally similarity scores for two users, for each party is computed on two criterion: a. The number of followers *u* and *v* follow from that party, b. The specific users they follow from that party

Pol-Rank utilizes two constants, λ and γ, which help to normalize certain aspects of the equation. λ has a value a 0.5, which in this means that both the similarity for the Republican candidates followed by a pair of users and the similarity for the Democratic candidates followed by the same pair are weighted equally. The constant γ is used to assign a weight to the importance given by Pol-Rank to the *number* of candidates followed by two users *i* and *j* from a given party vs. the *specific* candidates followed from a given party. The two summations account for an exact match in specific candidates followed. The two summations are used to calculate the common Democrats and Republicans between the two nodes. Gamma has a value of 0.3, which shows that we attribute a greater importance to the specific candidates that two users follow in common than the number of candidates they follow in common for each party.

There are two different scenarios that our similarity measure can deal with: two nodes that are similar and two nodes that are dissimilar. Let's suppose that there exists two nodes *a* and *b* are completely similar and *c* and *d* that are dissimilar. When Pol-Rank comes into contact with *a* and *b,* part 1 will become 0.3 since both nodes share the same amount of Democrats along with 1 divides 0.3. Part 2 will calculate the common Democrats between *a* and *b,* divide that by adding together the total number of Democrats they both follow, which will be divided by 2. That section is then multiplied by (1- γ) and given more importance over part 1 since the common candidates shared between *a* and *b* are more important than the amount. Part 3 and part 4 are almost similar to part 1 and 2, respectively. The aspects that change is that D_u and D_v are replaced by R_u and R_v respectively. When part 1 and 2 are calculated, they are multiplied by λ and part 3 and 4 are multiplied by (1- λ) which will normalize both sides and make sure they are both given equal importance. This will calculate out to be 1 since both nodes have the same amount of Democrats and Republicans followed and the same people in those respective parties. If two nodes are dissimilar like nodes *c* and *d,* they calculated the same way but will produce and different score depending on the people they follow.

Let's look at a snapshot of a graph and how Pol-Rank would calculate the similarity between them:



Consider the two nodes *A* and *B* both of which follow the same amount of types of Republican and Democratic candidates. Below shows how Pol-Rank will calculate the scores. The following equation evaluates to 1.

$$s(u,v) = 0.5\left(\frac{0.3}{||1| - |1|| + 1} + \frac{(1\text{-}0.3) * 1}{(|1| + |1|)/2}\right) +$$
$$(1\text{-}0.5)\left(\frac{0.3}{||2| - |2|| + 1} + \frac{(1\text{-}0.3) * 2}{(|2| + |2|)/2}\right)$$

P-Rank [2] has a section that recursively computes the similarity scores of any in-neighbors and out-neighbors of the two nodes. Instead of doing that for our equation we decided to keep the double summation and use that to compute the common Republican and Democratic followers between the two nodes.

Pol-Rank scores are within the interval (0, 1]. A score of 1 for two users *i* and *j* represents nodes that are entirely similar in both the number of Republicans and Democrats followed as well as the specific candidates followed by *i* and *j*. A score of 0 represents nodes that are very dissimilar however it will never reach 0 because this would mean that both nodes do not follow any political candidates. Such Twitter users are not a part of the data set we gathered.

5. Experimental Result

5.1 Experimental Setup

In order to evaluate the performance of Pol-RANK and compare it to other similarity measures in the literature, we needed to collect data on 2016 presidential candidates and their followers on Twitter. To do this, we looked at two different Twitter APIs, *Twython [3]* and *Twitter4j [4]*, which are written in Python and Java respectively. Both of these APIs worked well and gave us the data we needed, however, we ran into a problem with rate limiting. Rate limiting is Twitter's way of controlling how people use the API; they set a limit on how much data people can gather. When using *Twitter4j [4]*, we were limited to grabbing 180 followers every 15 minutes; *Twython* [3] allowed us to gather 5,000 followers every minute. Therefore, in order to crawl the Twitter feeds of candidates and grab the Twitter ID's of their followers, we needed to gather the followers and then put our code to sleep for 1 minute for *Twython* [3]. After discovering *Twython* [3], we discontinued our work with *Twitter4j [4]* and used Twython [3] exclusively as we were able to gather more data in a shorter period of time using paging. We read the data in from the Twitter pages of the candidates and outputted that data to text file. Another problem we had with *Twitter4j [4]* was our encounters with different types of errors. Some of the errors were a result of Twitter accounts (usually bots) being shut down in the time between when we grab the ID of a follower and when we finished making our query. Although *Twitter4j [4]* was unable to handle these errors, *Twython* [3] handled the errors with no problems.

After gathering all the data we needed, we began to research the different similarity measures that could evaluate accurate similarity scores between Twitter users. We looked at P-Rank [2], Sim-Rank [1], Cos-Sim, Min-Sim and wrote them into code using a graph library called the Stanford Network Analysis Platform (SNAP [5]) in C++. We then implemented various similarity measures and computed scores on followers of presidential candidates. We noticed that these existing similarity measures were not evaluating accurate scores. Additionally, P-Rank [2] and Sim-Rank [1] were very slow due to the significant time complexity of those algorithms ($O(n^4)$). Subsequently, we decided to create our own similarity measure, Pol-RANK that utilizes domain specific information. After designing Pol-RANK, we wrote it into SNAP [5] and experimented extensively. With the results of our experiments, we could see that the scores Pol-RANK evaluated were clearly a better representation than that of other similarity measures. Pol-RANK evaluated the scores at a significantly faster rate than Sim-Rank [1] and P-Rank [2] as we simplified much of the computation. The results of these experiments are presented in the following sections.

With our data, we then created text files that housed all of the data by candidate. In order to test our data, we created groups of three candidates and merged them into many different groups. Finally, we plotted scores from both Pol-RANK and P-Rank [2] to demonstrate the greater level of accuracy from our similarity measure of Pol-RANK contrasted against P-Rank [2].

After we collected all the data we needed from Twitter, we decided to separate candidates into groups of 3. For example we had sampled 10% of Martin O'Malley, Paul Cruz and Rick Perry as one group of candidates. And within that group of 3 we separated that into seven more groups by seeing who followed just each of the candidates and a combination of any of the candidates and made those into their own groups. Once that was figured out, we ran Pol-Rank on each of the seven groups and gathered those scores into their own text files. Afterwards we ran nodes from one group with nodes from another group to see how different the scores were between them. The results of these tests will be explained in more detail in Section 5.

5.2 Results

In this section, we will report on some of the experimental data that we have created by using the Pol-RANK formula. After we crawled the follower's list of each 2016 Presidential Candidate and parsed the directed graphs, we were able to apply our new algorithm to these sets. In Figures 1 through 4, we plotted some of the scores that were calculated through Pol-RANK. We decided to focus on certain combinations of 3 candidates and a portion of the data we gathered is showcased in these figures. The reason for grouping the followers of candidates in groups of 3 was to try and test out our similarity measure and the other similarity measures and see how they ran using nodes that are similar and nodes that were dissimilar. Groups of 3 were used because larger groups would substantially increase the run time for P-Rank [2]. We did this in order to test out our similarity measure against nodes that follow the same type and amount of candidates. If our similarity measure is correct, every pair of nodes that follow the same candidates should have a score of 1. Along with testing Pol-RANK with similar nodes, we also used it with dissimilar nodes. We ran our measure against nodes that were dissimilar to each other and acquired a mix of different scores as we predicted we would. Since Figure 1 has the most diverse set of nodes compared to the others in Figures 1 - 4, we were given more varied scores. The less varied the nodes were, the less varied the scores. The scores went as low as 0.65 but never higher than 1 and therefore land well within the range of Pol-RANK's [0, 1] range. Figure 2 has the scores from two different nodes and hence created two different types of scores (which still stayed in the range of [0,1]).

Figures 3 and 4 had the same similarity scores, as the nodes that were evaluated were extremely similar. By looking at the Figures 3 and 4, you can see that they have the same value for their Pol-Rank scores. This is due to the nodes in Figure 3 all following the same amount and type of Democrats and Republicans. When Pol-Rank evaluated their scores, it notices that and gives them a score of 1. Since each node is virtually the same, they all get a score of 1 when compared to other nodes within the data set. The same is can be said for the nodes in Figure 4. Since they all contain the same amount and types of Republicans, each comparison will get a score of 1. Each node had one outgoing edge (candidate that they were following), and Pol-RANK calculated their scores to be 1 since all the other nodes they were evaluated against were the exact same type of node. When Pol-RANK is compared to P-Rank [2] in terms of node type, P-Rank [2] is unable to create the correct scores that we need. Instead of creating one score as shown in Figure 4, Figure 5 contains two different scores: 1 and 0.2. This does not accurately reflect the scores that should be present based on the nodes being compared, as since the nodes all follow the same Republicans, the scores should all be 1. Therefore, P-Rank [2] is incompetent in creating the scores needed opposed to Pol-RANK's efficiency.

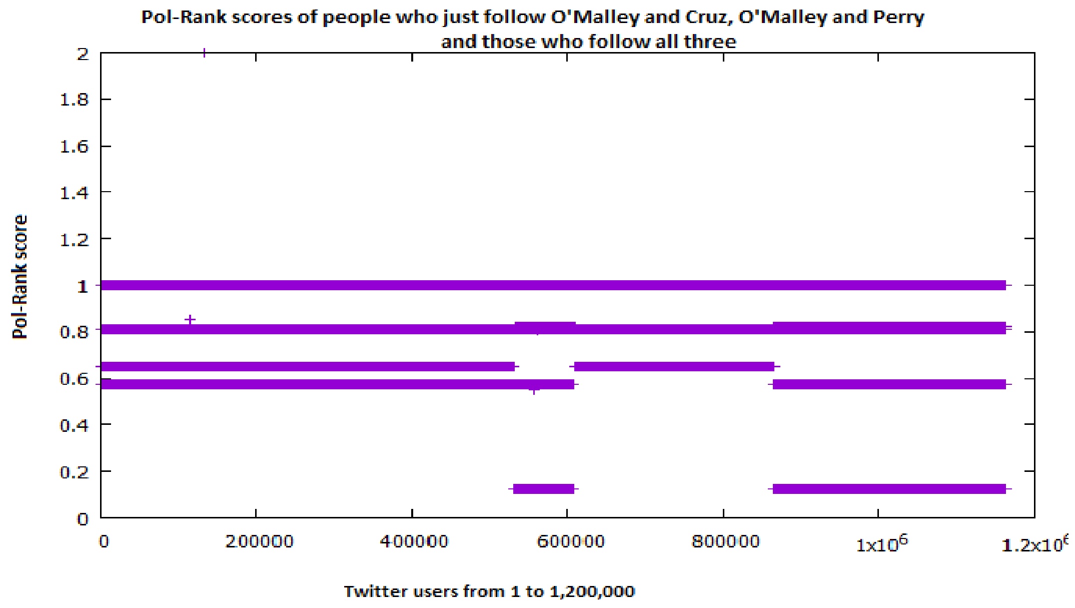


Figure 1 shows the POL-Rank scores between twitter users follow Martin O'Malley and Cruz with users who follow O'Malley and Perry and those who follow all three of the candidates.

Figure 1 shows the different scores being calculated when we compare three types of nodes who follow different candidates. Since the nodes being calculated belong to at most one group then there will exits more than one score. As you can see above, there exist different scores. The range is [0.1, 1] with a majority of the nodes having a score of 1.

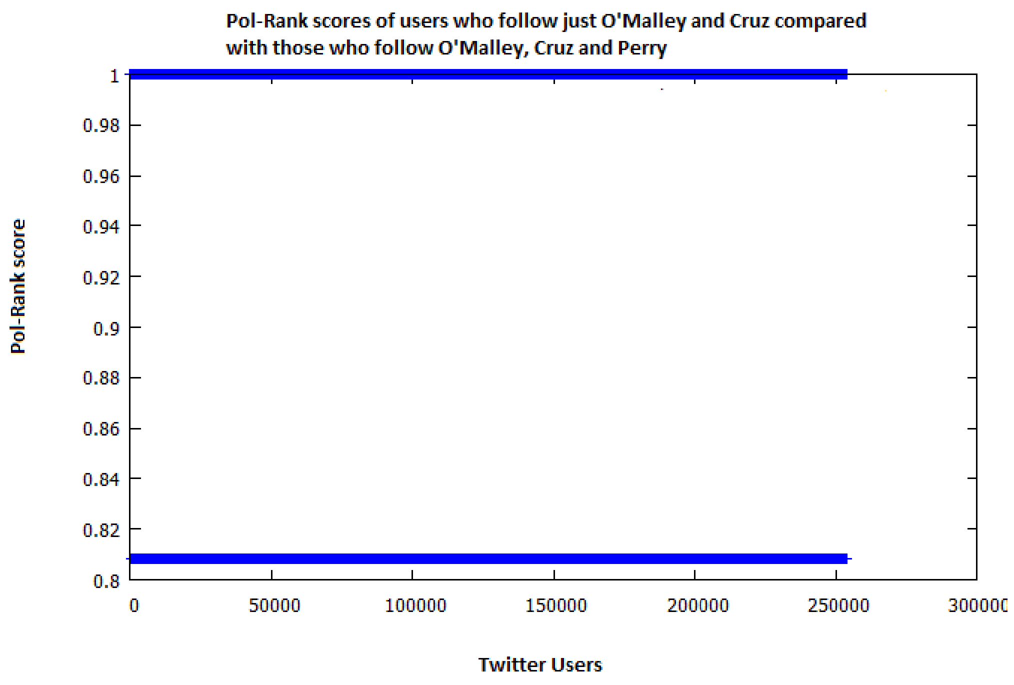


Figure 2 shows the scores between those who follow Martin O'Malley and Ted Cruz and those who follow O'Malley, Cruz and Perry.

Figure 2 shows the Pol-Rank scores of people who follow O'Malley/Cruz and people who follow all three candidates. Since there exist two different types of nodes within this comparison then there should be two types of scores. This is because if a node that follows O'Malley/Cruz or O'Malley/Cruz/Perry is compared with the same type of node, the score will be 1. If a node is compared with a different type of node, an O'Malley/Cruz node with an O'Malley/Cruz/Perry node, then that should be a score less than 1.

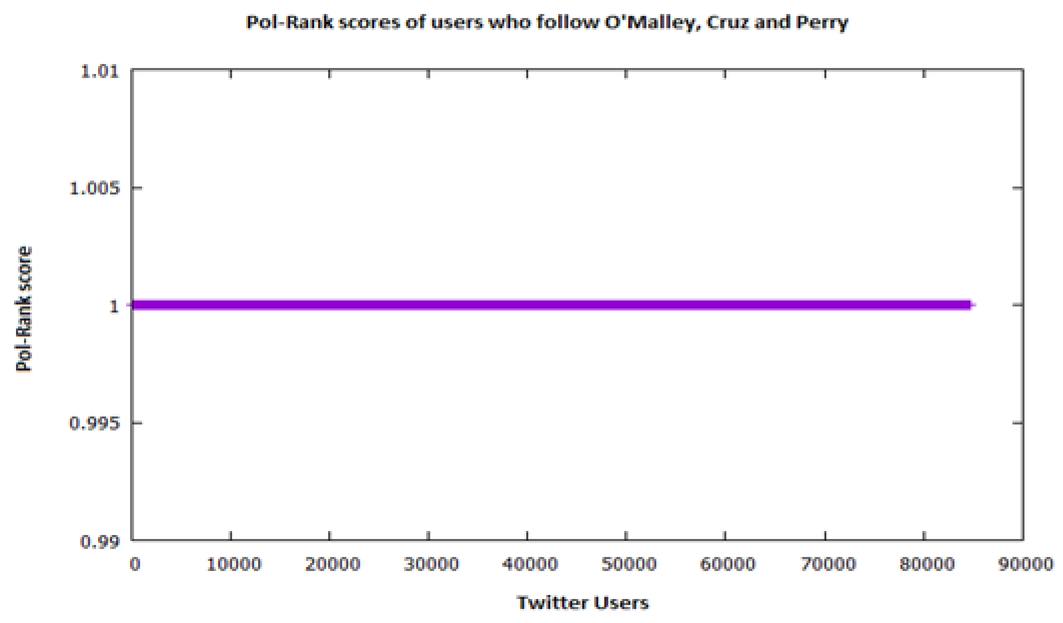


Figure 3 shows the score of people who follow O'Malley, Cruz and Perry.

Figure 3 shows pairwise scores for only one type of node: a user that only follows O'Malley, Cruz and Perry. Since there is only one type of nodes then any two nodes that are being compared will calculate a Pol-Rank score of 1, which is shown above.

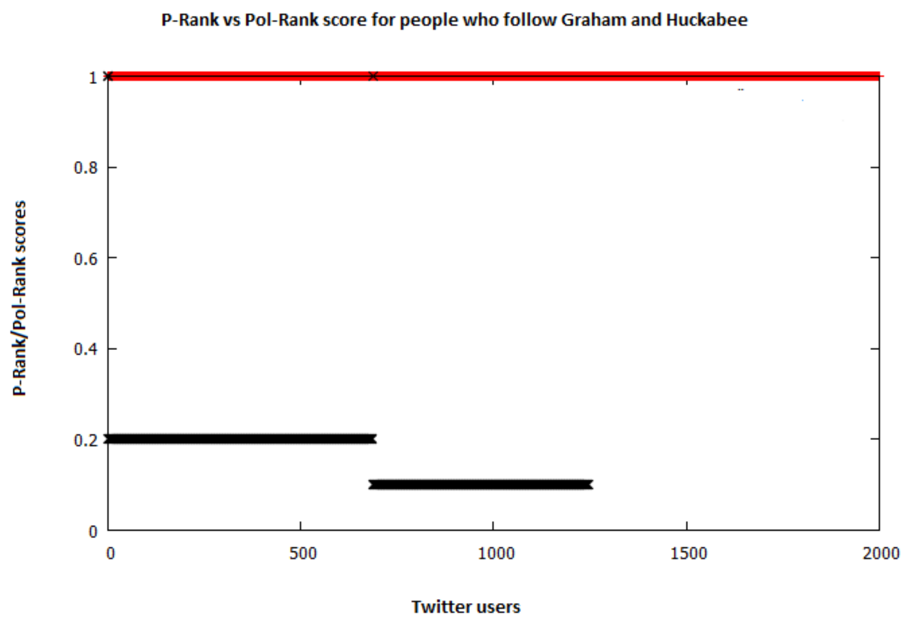


Figure 4 shows a comparison between the Pol-Rank scores and the P-Rank [2] scores of people who follow Graham and Huckabee

Figure 4 shows something very different in comparison to Figures 1 through 3. In order to show how Pol-Rank is more accurate than P-Rank [2], Figure 4 shows a side-by-side comparison between the two similarity measures with people who follow Graham and Huckabee. Looking above you can see that P-Rank [2] produces three different values: 0.1,0.2, and 1. Since the nodes being calculated are follow the exact same number of Republicans and same common Republicans therefore their score be 1. However, P-Rank [2] is not able to do that since it is not build to be able to do that. Therefore from Figure 4, you can see that Pol-Rank is a more accurate measure than P-Rank [2].

5.2 Exploring connectivity

There is a general belief in the political world that democrats are more popular with the younger demographics in America. Younger users also tend to embrace social media more readily. Hence, we were interested in investigating whether the edge demonstrated by democrats' with younger followers translated into a greater ability for them to reach more people. This can be important when a candidate is trying to reach an audience larger than their direct followers (for example through *retweets* on Twitter or *shares* on Facebook). To validate this hypothesis, we gathered one Democrat and two Republican data sets and ran experiments to analyze the number of followers' that a candidates followers had.

We crawled 10% of the followers of the candidates and also crawled their followers' followers as well and put this data into csv files. We did this through the Twitter API using *Twython* [3] and *Twitter4j [4]* implementations. Starting with the Twitter ID of a given presidential candidate, we were able to grab the IDs of all of his/her followers and subsequently, a list of the followers' followers as well. We then computed the size of the list and plotted it into graphs. An example line from one of the csv files is displayed here. "1897788080; Ali Shamsedin; 43" The first value (1897788080) is the Twitter ID of the candidates' follower, the second value (Ali Shamsedin) is the name that the account was registered with and the third and last value (43) is the amount of followers the follower has. Using this information, we were able to generate a list of the amount of followers each follower of the presidential candidate has and, consequently, the total *reach* of the presidential candidate.

Figure 1: Rand Paul

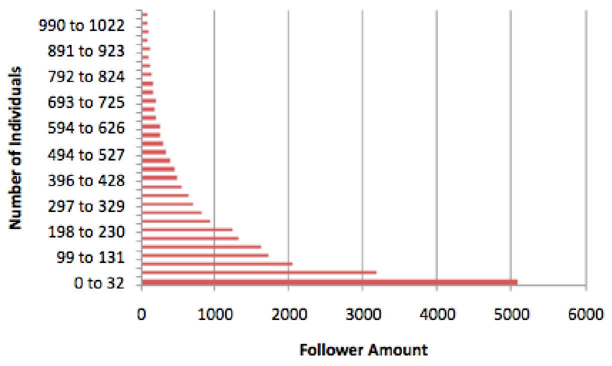


Figure 2: Ted Cruz

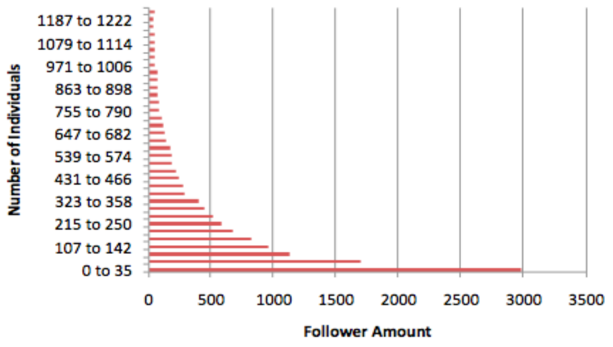
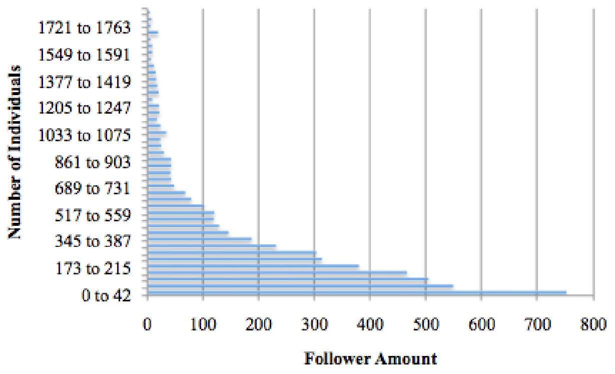


Figure 3: Lincoln Chaffee



The data shown in Figure's 1-3 validates our hypothesis. The Republicans Rand Paul and Ted Cruz (Figure 1 and Figure 2) had a lesser reach than the Democrats Lincoln Chaffee and

(Figure 3). The increment values of Rand Paul and Ted Cruz are 32 and 35 opposed to the larger values of 42 for Chaffee.

6. Conclusion and Future Work

In this paper we propose a new similarity measure specifically designed for scoring Twitter users based on the political candidates they follow. Using the Penetrating-Rank (P-Rank) similarity measure as the basis for equation, we were to come up with our new similarity measure Pol-Rank. Unlike P-Rank, instead of taking the in-neighbors and the out-neighbors of a node into account we take the Democrats and Republicans into account. Pol-Rank has advantages over other similarity measures. It has a lower time complexity than most of the other similarity measures we encountered and in comparison to the other similarity measures it produces much more accurate scoring with information that we gathered from each user. We also got rid of some of the more complicated mathematics that P-Rank was computing which also allowed us to. In order to test out our measure, we took some of our users from the candidates that we crawled and used our similarity measure on groups of three. We gathered from the scores that Pol-Rank produced accurate scoring based on whom the users were following than other measures.

References

[1] G. Jeh and J. Widom. SimRank: a measure of structural-context similarity. *In Proceedings of the eighth ACM SIGKDD conference (KDD'02)*, pages 538–543. ACM, 2002.

[2] P. Zhao, J. Han, and Y. Sun, "P-Rank: a comprehensive structural similarity measure over information networks," in Proceedings of the ACM 18th International Conference on Information and Knowledge Management (CIKM '09), pp. 553–562, November 2009.

[3] Twython, Python Twitter API, https://github.com/ryanmcgrath/twython

[4] Twitter4J, Java Twitter API, http://twitter4j.org/en/index.html

[5] SNAP, Stanford Network Analysis Platform, http://snap.stanford.edu/snap/

[6] Jones, Jeffrey M. "Young Americans' Affinity for Democratic Party Has Grown." Gallup.com Gallup, 28 Mar. 2014. Web. 23 July 2015