Social Nielsen

A more robust mechanism for generating content viewership data

Atharv Vaish
Computing and Mathematical Sciences
California Institute of Technology
Pasadena, CA 91125
avaish@caltech.edu

Arjun Chandrasekhar
Computing and Mathematical Sciences
California Institute of Technology
Pasadena, CA 91125
achandra@caltech.edu

Michael Lauria
Computing and Mathematical Sciences
California Institute of Technology
Pasadena, CA 91125
mlauria@caltech.edu

Anish Agarwal
Computing and Mathematical Sciences
California Institute of Technology
Pasadena, CA 91125
aagarwal@caltech.edu

ABSTRACT

Over the past decade, there has been a continual shift of the consumption of media from traditional sources such as television and theaters to web sources such as Netflix and Hulu. The current method of tracking viewership is flawed in many ways. As of now there is currently no suitable method to measure the web presence of a television show, information that is extremely valuable to advertisers and network executives.

We propose a solution that uses online social media, namely Twitter, to create a more accurate metric for viewership of a television show. We create a classification engine to classify tweets on tracked topics. We test our application against viewership data taken from Nielsen and Netflix on shows airing in 2012 and 2013, and analyze results based on viewership data and market assumptions.

Categories and Subject Descriptors

I.2.7 [**Natural Language Processing**]: Text Analysis; H.2.8 [**Database Applications**]: Data Mining

General Terms

Algorithms, Human Factors

Keywords

Twitter, Betweeness Centrality, Sentiment Analysis, Web Viewership

1. INTRODUCTION

What is it that makes a television show 'successful'? Currently, the basic measure of a television show's success rate is its raw viewership, which is measured by its Nielsen rating. These ratings track how many homes owning a Nielsen box tuned in to watch a television show during its airtime on the channel which it airs. This can be problematic for a few reasons:

Time-shifted Views: Nielsen fails to capture viewers who might watch a show after its original air-date, e.g. viewers who might watch an episode a day or two after its release, viewers who do not start watching and/or catch up on the show until well after it started airing, etc. Because of this, Nielsen box data fails to capture a holistic picture of a show's viewer network.

Web Views: The presence and influence of the Internet is growing rapidly in the fields of media distribution, entertainment and pop culture. Thus it is not surprising that a greater portion of users are taking advantage of the web to watch television shows through episodes made available on the network's own site or sites dedicated for television show viewing such as Hulu, Netflix, etc. Unfortunately this data is not captured at all by the Nielsen ratings, which means the metric is not only incomplete but obsolete, a relic from an era with fundamentally different (and less advanced) methods of spreading and accessing content. This is especially true given the success of shows like House of Cards and the new season of Arrested Development, content which was released through Netflix and never aired on network television, which makes it impossible to be tracked by a Nielsen box.

Viewing Demographic: Nielsen ratings are created through a sample of American families with Nielsen boxes. This sample is no longer indicative of the TV viewing demographic, and it is not very useful for capturing views outside of the family setting. Nielsen fails to capture instances of multiple people viewing a show though a single television, common

occurrences on campuses and other establishments. Nielsen does not capture views from college students or recent graduates, a major demographic that advertisers like to target. Finally, Nielsen gathers little data internationally.

1.1 Objectives

The goal of this project is to develop a new method of generating show-related data that draws from a more representative sample of viewers and provides a more holistic view of who watches which television shows and how successful certain content truly is. Ideally, we want a product that will account for less traditional viewers not captured by Nielsen while still tracking the traditional live television viewers, which Nielsen has a reputation for measuring successfully. Hence, we have designed a product that scans through and classifies social media data to come up with an estimated network of viewers watching each show. There are several advantages to using web data as opposed to Nielsen box data.

Social media, such as Facebook, Twitter, Tumblr, etc. is already a very natural place to post opinions. As previously mentioned, the web is becoming a more and more prevalent method of viewing shows and other media. Our application takes advantage of this one-stop platform for watching and talking about shows.

By looking at social media posts, we can gather information not only on how many people watch, but what parts of the show represent the actual points of discussion. In particular, we can gather information about which episodes, characters, actors, plot points, and memes are being discussed and appeal to the viewing crowd, as well as the sentiment of the viewers - do the people who are watching actually like what they are watching? This is the kind of information that adds significant value when compared to simple Nielsen box statistics.

Web data is much more likely to capture time-shifted viewing and so can measure viewership dynamically (as opposed to a Nielsen rating which represents a fixed point in time), giving us the ability to track information about a show over time and see how viewership and sentiment changes, cascades, and so on. Since we can also collect this data quickly, we can also track viewership and sentiment changes live.

With our application, we hope to provide network executives, advertisers, and public viewers with a detailed report on the size, structure, and aggregate sentiment of the network associated with each television show. This application can be used to better understand not only which shows merit marketing money, but also how to market those shows.

This will be extremely useful for all parties involved in television production, as they will be able to better identify the optimal audience to target and adapt the creation and distribution of content to appeal to that group's tastes.

1.2 Previous Commercial and Academic Work

There are two sources of prior related work that are of relevance for our project. There is the current data that Nielsen provides for clients, which we hope to duplicate and extend. There are also academic papers that focus on extracting information and predicting behavior and sentiment from social networks, which provide us with the techniques to obtain the data and power the metrics we hope to provide.

The problem of identifying what people on Twitter are talking about is not a new one. Previous work in the field includes attempts at determining what tweets are event/non-event based (Becker, et. al), identifying micro-memes and conversational tagging (Huang, et. al), and Twitter trend prediction (Shaw, et. al). There have even been past attempts to measure television viewership through social media (Wakamiya, et. al). In the case of the first three papers, our project will build on these past results by not only producing a means of identifying all the Twitter phenomena that these authors wish to identify, but by applying this identification scheme to produce accurate data on television viewership. In the case of the Wakamiya paper, our project will be substantially different in that it takes advantage not only of database properties but also of network properties to generate results.

2. APPROACH

In order to properly classify tweets, we use several modules employing different techniques across graph theory, learning theory, network theory and database theory to produce efficient and accurate results.

2.1 Algorithm Overview

The algorithm operates by taking a list of shows as input, processing each show to derive some sort of ground truth for what words and/or phrases belong to each show, and then combing through the Twitter stream and classifying each tweet as either relevant to a show of interest or not relevant. Additionally, once enough tweets have been accumulated, the program batches several tweets together and performs sentiment analysis on them to determine if each classified tweet is positive, negative or neutral. The result is several database tables that keep track of which tweets were classified for each show, which words and hashtags appeared the most frequently among classified tweets and what rationale the algorithm used for classifying each individual tweet. What follows is a detailed description of the program's control flow and classification algorithm.

2.2 Ground Truth Generation

Given a show name as input, the first step is to look the show up on thetvdb.com (TVDB) to get information on the show's cast, characters, network, and official series name through the TVDB API. The rationale behind this is twofold. First, the character list and series name themselves provide a certain amount of ground truth that can be used in evaluating the value and relevance of a tweet. Second, by looking up the cast and series/network name, we can then perform a Twitter search to look for corpus accounts: the official show account and verified Twitter accounts of the actors on the show. All hashtags produced by these corpus accounts are then taken to be ground truth as well.

We create a table with all the ground truth words, hashtags, and phrases for each show that gives a baseline from which to classify tweets and accumulate more truth. It is important to note that for each word/hashtag/phrase that we add

to the record table for a particular show (SHOWRECORD), we also keep a count of how often that entry appeared in the show Twitter account so we can eventually use it to better predict if a Tweet containing that entry really is about that particular show.

Next, we create a TWEETGRAPH table, which encodes information used to construct a graph object that will model the relationship between hashtags. In particular, it encodes a graph in which each node represents a hashtag and each edge between hashtags indicates that those two hashtags appeared together in a tweet at some point. The weight of each edge is the inverse of the number of times those two tweets have appeared together (meaning all weights are between 0 and 1). Thus hashtags that appear together frequently are connected by a low-cost edge.

Once all the tables are constructed, we begin scanning through the Twitter stream and looking for tweets to classify, from which point on they are put through the classification gauntlet.

2.3 Candidate Word List Creation for Tweets

Once a tweet is found, it is then classified by computing its relevance score across all shows and then determining if the tweet is relevant enough to any one show to merit classification. The algorithm works as follows:

1. Parse the tweet into its set of words and hashtags, with all punctuation removed.
2. Remove all stop-list words. The program keeps a global constant list of very commonly used words that should not affect how relevant a tweet is to a certain show or event. Exceptions are made for hashtags (in most cases a word is "hashtagged" to indicate that it is not in fact a mere common noun but rather has some extra meaning or value in context) and words that are part of the ground truth generated by corpus accounts and TVDB information.
3. Create a list of n-grams from the set of words in the tweet. For a given sentence s, an n-gram on s would be a set of n words that occur consecutively in s. For each tweet we generate n-grams from $n = 1$ through $n = 6$ so as to account for not just words but also phrases that might involve a set of multiple words whose relevancy value is greater than the sum of its parts.
4. The final output is a list of words, n-grams, and hashtags. We add this entire list to a table called RECORD, which keeps a count of every word and phrase we have seen in Twitter thus far. This table is used to assess word and phrase frequency in the set of all Twitter data.

Our list is then put through two tiers of classification.

2.4 Tier One Classification (Wordlist)

The score is calculated by dividing the number of times that an n-gram has appeared in either USERRECORD or SHOWRECORD by the number of times that n-gram has appeared in all tweets that have been picked up. The USERRECORD table contains all the words contained in tweets by user accounts related to the show, whereas the SHOWRECORD contains all the words contained in tweets by the primary show account. The precise score x is

$$x = \sum_w (\mathbf{wscore}(w) \times H(w))$$

where

$$\mathbf{wscore}(w) = S\left(\frac{sr[w]}{\mathbf{count}(sr)} \times \frac{\mathbf{count}(r)}{r[w]}\right) + U\left(\frac{ur[w]}{\mathbf{count}(ur)} \times \frac{\mathbf{count}(r)}{r[w]}\right)$$

and H, S, and U are multipliers for hashtags, words contained in SHOWRECORD or USERRECORD.

Essentially, the algorithm prefers words and tags that appear often in connection with a particular show but rarely appear in any other context. On the flip side, it penalizes words and tags that very rarely have appeared in connection to the show or that show up too often in other tweets that aren't necessarily related to the show. Single hashtags are given a significant multiplier to reflect the fact that, based on precedents and general Twitter culture, are more likely to carry extra weight in terms of connecting the tweet to a particular show or event. This score is calculated for every n-gram word derived from a tweet, and then summed to generate the total score for that tweet.

If the tweet's score surpasses the bar set by the dynamic thresholds at the time, the tweet is classified, and all the words, hashtags, and n-grams that were not filtered out by the stop-list are added to SHOWRECORD, while the tweet is recorded (with a timestamp). If a tweet fails to classify via Tier One, it then proceeds to Tier Two.

2.5 Tier Two Classification (Tweetgraph)

If a tweet makes it to this stage in the classification algorithm, the program then proceeds to scan through the TWEETGRAPH table and construct the corresponding graph by mapping each $(hashtag, hashtag, frequency)$ entry in the table to an edge in the graph object. We then calculate the betweenness centrality of each node in the graph.

Betweenness centrality (b) is defined as

$$b = \frac{st(v)}{st}$$

where st is the number of (equally) shortest paths from s to t and $st(v)$ is the number of shortest s-t paths that cross through vertex v. Intuitively, this measure captures a node's importance in terms of how crucial the node is for accessing efficient paths in the graph. On a graph used to model some sort of information network, this is a better measure of influence than pure degree because it captures the role of individual nodes as central figures for information distribution. This makes it an ideal candidate for discerning a word's importance – if a word serves as a strong, central connector of other words related to a show, chances are that word is closely related to the show itself and should carry a strong weight.

We calculate betweenness centrality using a modified version of Ulrik Brandes' betweenness centrality algorithm. The original algorithm uses dynamic program to slowly build up the list of shortest paths between each node as well as the list of dependency nodes (the nodes that our fixed-point node utilizes in its shortest paths) and creates a list of dependencies for each node pair. However, Brandes' algorithm is defined on unweighted graphs. Thus, we are using a modification of Brandes' algorithm developed by Tharaka Ahalakoon for weighted graphs.

After calculating betweenness centrality of the current hashtags in `TWEETGRAPH`, we then go through each word in the tweet being classified and see what that hashtag's betweenness centrality is (with 0 being used for the centrality of hashtags that are not nodes in the `TWEETGRAPH`) and divide that centrality by the frequency of occurrences of that hashtag in all tweets seen. We sum up this value for all hashtags in the tweet, and if this sum exceeds the `TWEETGRAPH` threshold we classify the tweet.

2.6 Dynamic Threshold Setting

Dynamic thresholds allow our classification algorithm to correctly classify a tweet within the context of a show. If all tweets had to obtain the same relevance score for each show, either tweets about shows that score lower on average would not be classified, or irrelevant tweets would be classified under high-scoring shows. Instead, each show starts out with a relatively low threshold for both tiers. After a tier classifies ten tweets, its threshold gets updated, and the new threshold is set equal to the score of the tweet at the twentieth percentile of scores for that show, i.e. the score of the eighth tweet in a list of tweets classified by that tier for that show.

The number of tweets required for a threshold update is also changed. The frequency with which the threshold for a show is updated increases if the change in threshold is large, and decreases if it is small in order to stop the algorithm from updating once it has converged upon the correct threshold for a show. This scheme works well, except for one caveat: hashtags. A single relevant hashtag should be enough to classify a tweet. Because of this fact, hashtags are rightfully given more weight in our algorithm. However, placing the threshold at the twentieth percentile of all classified tweets for a show would exclude many important tweets that contain the relevant keywords, just not as hashtags, due to the difference in weight distribution. Because of this, only tweets classified due to the weight of their untagged words rather than by that of their hashtags are included in the dynamic threshold calculations.

If a tweet is classified, then the `TWEETGRAPH` is updated to include all the hashtag pairs in that graph. Specifically, for each hashtag pair $h1, h2$ in the tweet, a tuple $(h1, h2, 1)$ is added to the `TWEETGRAPH`. If the tuple $(h1, h2, k)$ is already in the `TWEETGRAPH`, that tuple is updated to read $(h1, h2, k + 1)$. Essentially, we change the weight of that edge having observed an additional instance of those two hashtags appearing together.

2.7 Sentiment Analysis

Finally, all relevant tweets are recorded in the database and are prepared for sentiment analysis. Once enough new tweets have been classified, we send a batch of tweets to a sentiment analysis classifier to perform bulk analysis. The database is then updated so each tuple containing tweet text also contains the sentiment associated with that tweet, either 0 (negative), 2 (neutral), or 4 (positive).

2.8 Querying Databases

To generate data, we simply provide the shows and let the application run over time. As it accumulates data, the interested party can query the database to find the information of interest, whether it be the accumulated list of relevant words and hashtags in `SHOWRECORD`, the number of tweets classified for each show, or the set of actual tweets classified for each show.

3. METHODS

In order to test our classification algorithms, there were several activities we performed in order to generate meaningful results.

3.1 Choosing Shows to Track

Twitter data was collected on certain days over the May 26th 2013 to June 4th 2013. The data source was the Twitter API that provides application developers with a 1% stream of all tweets on a real-time basis. We tracked Twitter data over 24 hour periods for each day that we chose. The shows tracked could be divided into several categories, which we felt would encompass a diverse collection of television.

Categories 1 and 2 consist of shows that we think Nielsen would do a good job in measuring viewership as they are primarily viewed on televisions. Category 1 focuses on reality TV shows, while Category 2 focuses on live sports. Category 3 consists of niche television. Categories 4 and 5 consists of shows that we think have a much larger online viewing audience that Nielsen does not include in its ratings. Category 4 consists of shows airing on premium channels, whereas Category 5 contains online only content. Our approach is to track the number of tweets for shows in all categories and compare our results with Nielsen's for all categories of shows.

Shows tracked in Category 1 included The Bachelorette. Shows tracked in Category 2 included the NBA playoffs and the NHL playoffs. Shows tracked in Category 3 included Defiance and Warehouse 13. Shows tracked in Category 4 included Game of Thrones, Mad Men, Veep and Doctor Who. Shows tracked in Category 5 included Arrested Development.

3.2 Estimating Viewership from Tweets

To get an estimate for the number viewers based on tweets, we found the ratio of the number of tweets each show got against each other and multiplied all these ratios by a scalar factor to get an estimate for viewing population. This scalar factor was computed using Category 2 viewership data collected before our measurement. Our final scalar factor was 1600 viewers per tweet collected. With our 1% filter, we multiplied by 100.

Figure 1: Sunday May 26th 2013

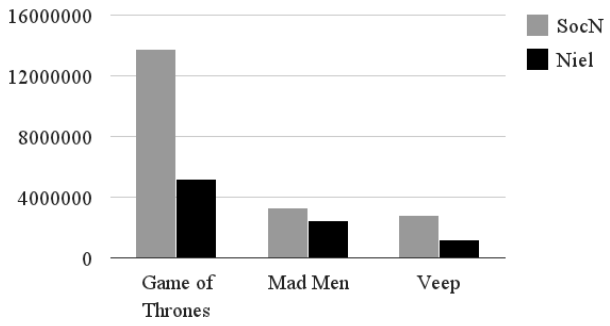

Figure 2: Sunday June 01st 2013

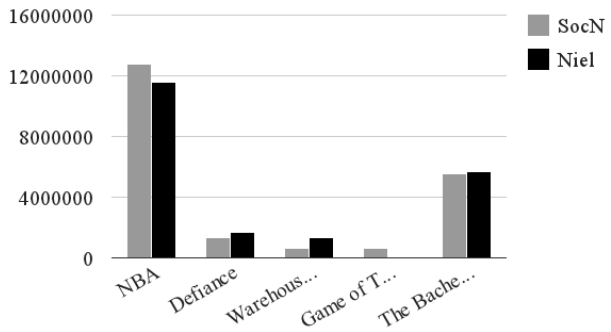

Figure 3: Monday June 02nd 2013

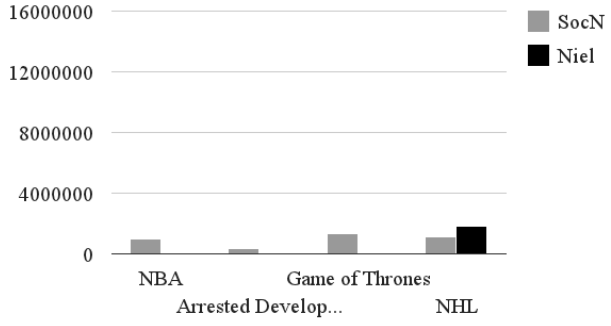

Figure 4: Tuesday June 03rd 2013

4. RESULTS

We tracked five shows from June 1st, 2013 to June 4th, 2013 and three shows on May 26th, 2013.

Gray bar graphs indicate our estimate based on number of tweets. Black bar graphs indicate number of viewers based on Nielsen ratings[1]. Arrested Development is an exception because the source is internet traffic data[2] rather than Nielsen data.

Figure 1 shows viewership estimates for three popular shows on May 26th: Arrested Development, Mad Men and Doctor Who. This was the season premiere of Arrested Development and our estimates matched up well with third party viewership data for Netflix (Arrested Development was only available to watch through Netflix).

Figure 2 shows viewership estimates for three popular shows on June 1st: Game of Thrones, Mad Men, and Veep. This was the day all three shows aired for the first time. Results indicate that our estimates far outstripped Nielsen's. For Game of Thrones, results differed by over a hundred percent.

[1]http://tvbythenumbers.zap2it.com

[2]http://aim.proceranetworks.com/2013/05/28/arrested-development-netflix-launch-frozen-bananas-and-ostriches-for-all/

Figure 3 shows viewership estimates for five shows on June 2nd. Two extremely popular shows, the NBA playoffs and The Bachelorette, aired live on television along with two niche sci-fi shows: Defiance and Warehouse 13. Our estimates matched to within ten percent of Nielsen's.

Figure 4 shows viewership estimates for five shows on June 3rd. NHL aired live. Our estimates matched within twenty percent of Nielsen's for NHL and we estimated some time-shifted viewing of shows airing on Sunday and Monday that Nielsen has no data for.

Figure 5 shows viewership of Game of Thrones from 1st to 4th June based on our algorithm and Nielsen. Nielsen only has estimates for 1st June while we estimate significant viewership of Game of Thrones over subsequent days as well.

5. ANALYSIS

We can analyze our results using our category descriptions as defined earlier.

5.1 Twitter as a Predictor of Viewership

The three shows we tracked in Category 1 and Category 2 (the NBA, The Bachelorette and the NHL) have a high correlation between Nielsen's results on the number of television viewers and our estimate of viewers though number of tweets. Furthermore, our estimate for viewership of Ar-

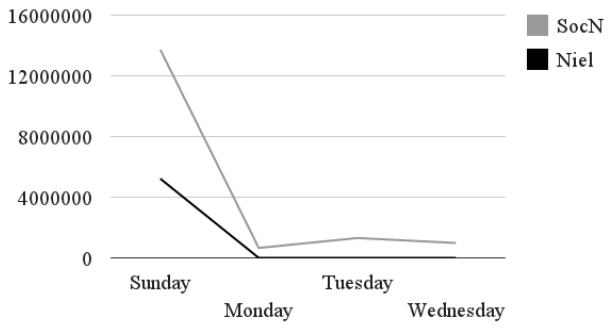

Figure 5: Game of Thrones Viewership

rested Development matched up closely with internet traffic data for the show on Netflix, which is a good indication of accuracy – such traffic can be tracked precisely and Arrested Development can only be watched through a Netflix account. Piracy for Arrested Development over the period we tracked was negligible.

Therefore, for the shows that we hypothesized we would correlate well with other sources, our estimates matched up well. This shows that the number of tweets we classify as relevant to a show could potentially be a strong indicator of viewership.

5.2 Outperforming Nielsen

For shows such as Game of Thrones, Mad Men and Veep, which have viewers who are more likely to consume media online, our estimated viewership was much higher than that predicted by Nielsen. This was especially true for Game of Thrones, which is available for viewers to watch on its network's (HBO) own site, "HBO GO". We also recognized a much larger viewership than Nielsen for Game of Thrones, which correlates with our expectations given that the college demographic that watches (and often pirates) Game of Thrones is not counted at all by Nielsen.

Our algorithm also manages to capture time-shifted viewing (shown in **Figure 5**), giving a more holistic picture for the viewership of a show compared to Nielsen. Our estimates for shows with large online viewership will most likely be valuable to network executives trying to gauge the performance of certain television shows that Nielsen can neither estimate viewership accurately on a given day nor capture time-shifted viewing.

6. FUTURE DIRECTIONS

At our current stage, we have demonstrated for a large classification of shows our classification engine is effective at matching tweets to shows. We have an indication that we are getting more accurate viewership data for these shows than what Nielsen can provide. However, there are several directions where we believe we can make further improvements.

6.1 Further Applications

Extension to Other Social Media: Currently our algorithm focuses on Twitter, because this medium has the largest volume of content. In future, we would like to mine other social networks (Facebook, Tumblr, etc.) in order to improve our ratings.

Classifying Non-Television Related Events: While our project aims to measure viewership of television shows, we believe the potential of our algorithm is wider in scope. For one, we can use the algorithm to look at viewership of television events that are not necessarily 'shows' - for example, during the NBA Eastern Conference Finals we deployed our algorithm to track tweets related to the show 'NBA' with good results (**Figure 2**).

More generally, we could determine the impact of events in the news, NGOs, politicians, and celebrities. It may even be possible to predict the outcome of elections, and trends in the stock market.

6.2 Natural Language Processing

Proper Noun Identification: Many shows have titles that are also common nouns. Distinguishing when a noun is proper or common in a given context is crucial for scoring the relevancy of tweets correctly – even if we have perfect weights, if the algorithm can easily confuse Friends the TV show with friends the common noun, it will be inaccurate.

Machine Learning and Sentiment Analysis: Currently our program uses a pre-existing sentiment analysis library; however there may be value in designing a sentiment analysis suite specific to tweets related to television. In any case, a necessary next step is to update the algorithm so that after performing batch-sentiment analysis it performs aggregation analysis to determine the overall polarity of the show. That is, it should assess not only how many people are talking about a show but also how the people who watch that show feel about that show, and to what degree do they feel positively or negatively about that show. A show with a small but vocal group of fans could be more valuable to advertisers than a show with a large but apathetic fan base.

Meme Tracking: Currently we can see which words, characters, and hashtags (which might include short phrases) get the most mentions, but if we can go a step further and identify memes and inside jokes, it would be incredibly valuable to those looking to design ad campaigns around a show.

6.3 Algorithm Improvements

Do certain shows appeal to very similar networks? If so, perhaps advertising during such shows would be more fruitful if the two shows' campaigns were paired or merged. Such an improvement can help with cross advertising to audiences that watch similar shows.

The network associated with a show might be small, but if it has some properties (high clustering, etc.) that make it conducive to quick and complete content dissemination, then perhaps that network might be more profitable than its size suggests.

Even if the network directly associated with a show is small, perhaps its user-base consists of highly influential users through

whom an ad campaign could reach a broader network than the raw numbers might suggest.

What is the correlation between the shows someone watches and the shows watched by his followers/friends? If a strong correlation exists, we can take advantage of this property to track people who most likely view a show even if they do not explicitly tweet about it.

6.4 Other Improvements

Advanced Network Analysis: Our show tracks what people are tweeting about, but it does not pay much attention to the people who make those tweets. In the future we want to keep track of this and perform analysis on the network of users associated with each show.

Graphical User Interface: Currently our program works via the command line. A future step would be to add a user-friendly interface with customizable options for how to go about collecting and outputting the data, what ground truth to use, etc.

7. ACKNOWLEDGMENTS

We would like to thank our mentor for this project, Professor K. Mani Chandy, for guiding us through the process and for funding some of our expenses. We would also like to acknowledge Professors Adam Weirman and Steven Low for the support provided for the project. Finally, we would like to acknowledge John Lilley from IMSS, who helped set up our high performance computing infrastructure for data analysis.

8. BIBLIOGRAPHY

Becker, Hila, Mor Naaman, and Luis Gravano. "Beyond trending topics: Real-world event identification on Twitter." *Proceedings of the Fifth International AAAI Conference on Weblogs and Social Media (ICWSM'11)*,2011.

Huang, Jeff, Katherine M. Thornton, and Efthimis N. Efthimiadis. "Conversational tagging in Twitter." *Proceedings of the 21st ACM conference on Hypertext and hypermedia.* ACM, 2010.

Chen, George H., Stanislav Nikolov, and Devavrat Shah. "A Latent Source Model for Online Time Series Classification." *arXiv preprint arXiv:1302.3639* (2013).

Wakamiya, Shoko, Ryong Lee, and Kazutoshi Sumiya. "Towards better TELEVISION viewing rates: exploiting crowd's media life logs over Twitter for TELEVISION rating." *Proceedings of the 5th International Conference on Ubiquitous Information Management and Communication.* ACM, 2011.

Brandes, Ulrik. "A faster algorithm for betweenness-centrality*." *Journal of Mathematical Sociology 25.2* (2001): 163-177.

Alahakoon, Tharaka. *Path centrality: A new centrality measure in networks.* Diss. University of South Florida, 2010.