MUTUALLY ENHANCING COMMUNITY DETECTION AND SENTIMENT ANALYSIS ON TWITTER NETWORKS

by
William Deitrick

Submitted in partial fulfillment of the requirements for Major Honors in Computer Science

Houghton College, Houghton, New York
May, 2013

Honors Committee

Chair: Dr. Wei Hu, Professor of Mathematics and Computer Science	Signature: _____
Dr. Jill Jordan, Assistant Professor of Mathematics	Signature: _____
Dr. Richard Stegen, Professor of Psychology	Signature: _____

Table of Contents

List of Tables, Graphs, Equations, and Figures

Acknowledgements

I would like to thank Houghton College for providing funding and technical resources for this research. In addition, I would like to thank Dr. Wei Hu; without him this project would not have been possible.

Abstract

The burgeoning use of Web 2.0-powered social media in recent years has inspired numerous studies on the content and composition of online social networks (OSNs). Many methods of harvesting useful information from social networks' immense amounts of user-generated data have been successfully applied to such real-world topics as politics and marketing, to name just a few. This study presents a novel twist on two popular techniques for studying OSNs: community detection and sentiment analysis. Using sentiment classification to enhance community detection and community partitions to permit more in-depth analysis of sentiment data, these two techniques are brought together to analyze four networks from the Twitter OSN. The Twitter networks used for this study are extracted from four accounts related to Microsoft Corporation, and together encompass more than 60,000 users and 2 million tweets collected over a period of 32 days. By combining community detection and sentiment analysis, modularity values were increased for the community partitions detected in three of the four networks studied. Furthermore, data collected during the community detection process enabled more granular, community-level sentiment analysis on a specific topic referenced by users in the dataset.

1. Introduction

The popularity of online social networks (OSNs) has increased dramatically in recent years. Individuals and organizations can now take advantage of a wide array of Web 2.0-powered social networking platforms, including the likes of Facebook, LinkedIn, and Twitter [1]. Though these services vary greatly in both form and function, they are all alike in facilitating the exchange of significant volumes of information among their users. Due to the massive amounts of data that flow through social networks and the relative ease of accessing this data, analysis of social networks has become a research topic of particular interest.

While the study of social networks can take many forms, two popular topics are community detection and sentiment analysis. Distinct groups of entities, or communities, will often form within social networks. Identifying these groups and modeling their dynamic interactions can provide valuable insight across many disciplines, and is the primary goal of community detection [2]. Sentiment analysis, sometimes called opinion mining, provides a means of automatically determining the attitudes or opinions of users via the content they have created [3]. Using sentiment analysis and community detection techniques, previous research has demonstrated the usefulness of information gained from social networks in describing such real-world events and issues as political movements [4], elections [5], and consumer attitudes towards products and services [6]. In this study, both sentiment analysis and community detection will be used to analyze networks from the Twitter OSN.

1.1 The Twitter OSN

Since its inception in 2006, Twitter's overwhelmingly rapid growth has made this service the Internet's fastest growing social networking platform [1]. As a result, Twitter has become a popular target for research efforts. The service's powerful and well-documented Application Programming Interface (API) provides an easy means to obtain content created by Twitter users, and a plethora of libraries exist for many platforms and programming languages to further simplify the data collection process.

Twitter is primarily a microblogging service, allowing its users to post, or "tweet" messages up to 140 characters in length. These messages often contain links to other web content in the form of URLs (usually abbreviated via URL shortening services) and Twitter-specific constructs including hashtags, mentions, replies, and retweets. Hashtags are words or terms denoting a specific topic preceded by the # (pound) symbol, while mentions and replies are a means of referencing other Twitter users by prepending their username with the @ symbol in a message [1]. A retweet allows a user to re-post a tweet created by another user, usually indicating their support for or interest in that tweet's content. Twitter users can "follow" other individuals to receive the messages those users post to Twitter, and anyone following a particular user is denoted as a "follower" of that

user [7]. By default, users' Tweets are publicly accessible, allowing Twitter users open access to each other's content via Twitter's web portal or its API [6].

1.2 Community Detection

Community detection is undoubtedly one of the most popular research topics associated with OSNs. To facilitate community detection, social networks are modeled as mathematical graphs, often referred to as "social graphs", in which vertices (or nodes) represent actors within the network and edges correspond to ties between individuals. Such a network is shown in **Figure 1**. In this image, the coloring of vertices represents possible communities to which each of the nodes in the network may belong. Depending on a network's purpose and the nature of the data being analyzed, its edges may be weighted or unweighted as well as directed or undirected [8].

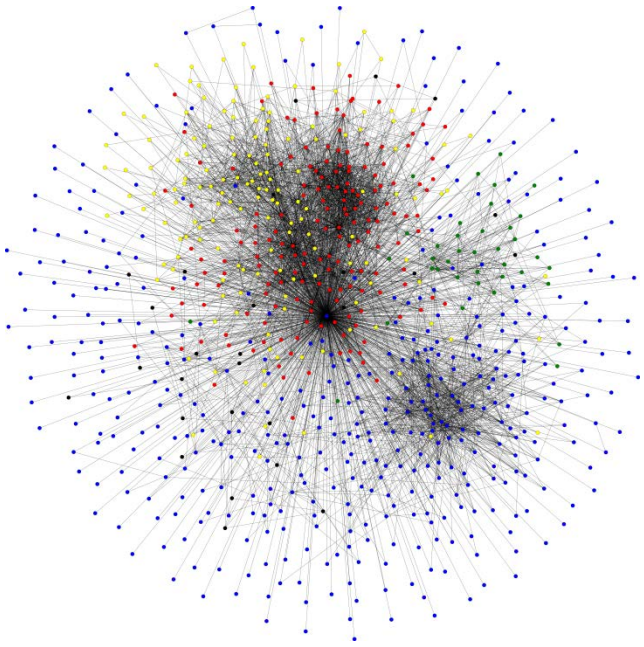

Figure 1: An Example Social Network.

Communities of individuals in a social network can be distinguished using graph clustering techniques. The basic idea of graph clustering is to group similar or associated nodes together. Generally, partitions are created that maximize the number of connections (edges) within a cluster while minimizing connections between clusters [8]. Using graph clustering, previous studies have been able to effectively discover real-world communities on the Twitter OSN such as the Indie Mac developer community studied in [7].

Many graph clustering methods have been applied to community detection on social networks. Hierarchical clustering utilizes hierarchical representations of graphs called dendrograms. These structures provide easy control over clustering resolution, since each level of the hierarchy is effectively a clustering of the graph at a different level of granularity [8]. Though this is a popular method, its shortcomings have stimulated interest in other ideas. One alternative technique, proposed by Newman and Girvan, uses a measure called "betweenness" to discover clusters within a graph structure [9]. Betweenness is a metric applied to the edges within a graph and is defined as the number of shortest paths connecting any two nodes that pass through a given edge [8]. Algorithms based on optimizing edge betweenness that perform well on both real and computer-generated networks have been successfully developed [9]. Another algorithm, known as DENGRAPH, is a density-based clustering algorithm proposed specifically to analyze social network structures. Based on the incremental version of the DBSCAN

algorithm, DENGRAPH provides a key asset in the study of social networks: the ability to handle the constant, dynamic changes in their structure [11]. Still another technique, known as the Label Propagation Algorithm (LPA), is based on the idea of a spreading disease epidemic. With this method, each node in the network is initially assigned a unique label. For every iteration after the initial step, each node is updated to have the label held by the majority of its neighbors (in the case of a tie, a label is picked at random). The effectiveness of this algorithm has been shown by several studies [10]. These examples are but a small sampling of the myriad community detection algorithms that have been proposed over the past few years.

One key aspect of community detection is quantifying the quality, or fitness, of the communities found. Indeed, many community detection algorithms are driven by optimization of one metric or another, such as the well-known modularity metric [8]. Modularity, proposed by Newman and Girvan in [12], measures the fraction of edges within communities minus the expected value in a network with equivalent partitions but random edges. This quantity can be expressed as follows:

$$Q = \sum_{i=1}^{k} \left[\frac{e_i}{m} - \left(\frac{d_i}{2m} \right)^2 \right], \tag{1}$$

where k denotes the number of modules (partitions) created from the network, e_i the number of edges in a module i, d_i the sum the degrees of nodes in a module i, and m the number of edges in the entire network [14]. Modularity can also take edge weights into account with just a slight change to the formula. Despite the popularity of this measure, it has been criticized for such issues as a significant resolution limit [13], stemming from the fact that modularity is oriented towards global optimization [15].

Another method for calculating community fitness, proposed in [16], simply measures the ratio of the internal degree of a community to the total degree of that community:

$$f_G = \frac{k_{in}^G}{(k_{in}^G + k_{out}^G)^\alpha}, \tag{2}$$

where k_{in}^G and k_{out}^G denote the internal and external degrees of the nodes in a module G. The exponent α controls the size of the communities, and ought to be a positive real-valued number [16]. Both modularity and the degree ratio metric will be used in this study to evaluate the results of community detection. These are but two examples of the many methods for evaluating community structures in networks, and further reading on this topic can be found in [17].

In this study, two algorithms were chosen for community detection: the Speaker-Listener Label Propagation Algorithm (SLPA) [18] and the Infomap algorithm [19]. These were chosen because they are able to handle both weighted and directed networks, they both execute relatively quickly on large graphs, and because their operation differs greatly. SLPA is based on the Label Propagation Algorithm (LPA), but offers a critical extension: whereas in LPA each node may contain only a single label, SLPA allows each

node to assume multiple labels. Thus, SLPA is able to detect overlapping communities, such as those that occur in a social network [18]. The implementation of SLPA used for this study was obtained from the creators of this algorithm at their website (https://sites.google.com/site/communitydetectionslpa/). The Infomap algorithm, on the other hand, models information flow in a network using the probabilities of particular random walks within the network. This algorithm was designed for use with biological and sociological networks, and was originally demonstrated on a citation network of publications from the sciences [19]. The iGraph library (http://igraph.sourceforge.net) contains an implementation of this algorithm, which was used with the provided Python interface.

1.3 Sentiment Analysis

In addition to community detection, sentiment analysis has become another popular tool for analysis of social networks. Sentiment analysis is often formulated as a two-step problem, in which it is first necessary to determine whether a given text is subjective or objective. This is known as Subjective/Objective-polarity, or SO-polarity. If it is determined that a text is subjective, it can then be classified according to whether it expresses positive or negative sentiment, which is denoted Positive/Negative-polarity or PN-polarity [20]. Over the past few years, a variety of strategies have been used to perform sentiment analysis on many different types of data.

A common approach to sentiment analysis uses a lexicon of words labeled with their SO or PN polarities. The SentiWordNet lexicon [20] is one such resource that has proven effective in analyzing all manner of text documents, from product reviews [21] to news headlines [22], and has even been used for multilingual sentiment analysis [23]. When using a lexicon such as SentiWordNet, a simple technique involves summing the polarity scores for the words in a document and making a prediction based on the result. While this naïve approach can produce satisfactory results, lexicon-based polarity scores can be used more accurately when coupled with a machine learning algorithm [24].

While lexicon-based methods have been shown to be effective for many types of textual documents, Twitter presents a unique challenge because its informal messages are very short and contain large amounts of slang and misspellings. This reduces the effectiveness of traditional lexicons [3]. As a result, some studies such as [3] and [25] have chosen to use fully or distantly supervised learning to more accurately classify the sentiment expressed in tweets. Fully supervised learning requires manually labeling data to provide input for a machine learning classifier. This is a useful technique, but is extremely time-consuming and often produces training sets that are not of sufficient size to effectively train a classifier [3]. Distantly supervised methods attempt to overcome these issues by automatically labeling training datasets based on “noisy” labels such as emoticons and hashtags [26].

For this study, we adopt the fully supervised training approach using multiple datasets to ensure we have a training corpus of sufficient size. The Naïve Bayes classifier from the Natural Language Toolkit (http://nltk.org/), abbreviated NLTK, is used in

conjunction with both unigram (single-token) and bigram (two-token) features to identify the subjective/objective and positive/negative orientation of tweets.

1.4 Purpose of this Research

While previous studies using sentiment analysis and community detection abound, these tasks are generally treated as completely separate issues. This study instead combines the two techniques, investigating the integration of sentiment analysis and community detection on networks from the Twitter OSN. Sentiment analysis is used in addition to other Twitter-specific features including hashtags, mentions, replies, and retweets to enhance community detection. Once community structures have been discovered, the power of combining these techniques is demonstrated by analyzing sentiment information on the community level in one of the dataset's networks.

2. Datasets

Two datasets were used in this study. The first of these was the publicly available Sanders corpus (http://www.sananalytics.com/lab/twitter-sentiment/), a manually-labeled dataset provided for training sentiment classifiers. The second dataset was collected directly from the Twitter API for this study, and contained data relating to several Twitter accounts managed by Microsoft Corporation.

2.1 Sanders Corpus

The Sanders corpus consists of 5,513 tweets manually labeled according to their subject and sentiment. The Twitter API terms of service do not permit direct distribution of the tweets, so a small Python script is provided to download the dataset directly from Twitter. While this corpus contains over 5,500 tweets, some of the tweets appear to be no longer available via the Twitter API and could not be downloaded. Thus, the number of available tweets was reduced to 4,957. Furthermore, only 3,727 of the tweets are labeled according to sentiment as "positive", "negative", or "neutral", while the rest are labeled "irrelevant". All of the tweets labeled "irrelevant" were filtered out, as well as those that were not English (according to the "lang" attribute provided by the Twitter API). Thus, the usable data for this study from the Sanders corpus contained 3,111 tweets.

From the 3,111 tweets extracted from the Sanders corpus, two datasets were created, and later merged with data from the Microsoft dataset described below. The first dataset was used in the training of a subjective/objective Naïve Bayes classifier. All "neutral" tweets were assigned the label "objective", and all positive and negative tweets were assigned the label "subjective". The second dataset was used to train a positive/negative Naïve Bayes classifier, and consisted of the tweets from the Sanders corpus labeled positive or negative. Thus, as can be seen from **Table 1**, 1,028 tweets were subjective while 2,083 were objective. Also, as **Table 2** shows, 484 of the 1,028 subjective Tweets were positive, while 544 were negative.

Table 1. Sanders Dataset: Subjective and Objective Tweets.

Subjective	Objective	Total
1,028	2,083	3,111

Table 2. Sanders Dataset: Subjective Tweets.

Positive	Negative	Total
484	544	1,028

2.2 Microsoft Corpus

The Microsoft corpus collected for this study was downloaded directly from the Twitter API. This dataset was collected in two stages. First, using the Python library Tweepy (http://tweepy.github.com/), the social networks of four Microsoft-sponsored Twitter accounts were crawled: *@technet*, *@windevs*, *@VisualStudio*, and *@Silverlight*.

These four accounts are used by Microsoft to communicate with information technology professionals and developers, and were chosen because they had relatively large numbers of followers but could still be crawled in a timely manner within the rate limits of the Twitter API. All followers and friends of these accounts who were following less than 600 others were collected, creating the social network for each of the four accounts similar to the visualization in **Figure 1**. The limit of following 600 users was imposed similarly to [7] as a means of de-noising and limiting the size of the crawled network. With this limitation in place, the number of users crawled for each of the four accounts is displayed in **Table 3.**

Table 3: Users Crawled for Microsoft Accounts.

@technet	@windevs	@VisualStudio	@Silverlight
1,382	15,559	26,775	18,630

The second stage of data collection involved capturing tweets created by the collected users. This data was harvested using the Java library Twitter4j (http://twitter4j.org/) in conjunction with the Twitter streaming API. Between January 2, 2013 and February 2, 2013, a total of 2,061,789 tweets were collected from the networks of the Microsoft accounts described above. This portion of the dataset provided the additional features described in section 3 that were used to enhance community detection on the four Microsoft networks.

2.3 Combined Datasets

To ensure significant training data was available for sentiment analysis, a portion of the tweets collected in stage two above were withheld as a training set. A total of 3,000 English tweets were randomly selected and removed from the full set of tweets such that they were proportional to the number of tweets collected from each of the four accounts. This set of 3,000 tweets was then manually labeled as "positive", "negative", or "objective" according to the sentiment they expressed. After this was completed, these tweets were split into two datasets just like the Sanders Corpus, with one set of tweets containing the labels "subjective" and "objective" and the other containing the labels "positive" and "negative". These two datasets are described in **Table 4** and **Table 5**, respectively. Finally, the datasets described in **Table 6** and **Table 7** were created for training the two Naïve Bayes classifiers, combining the tweets from the Sanders Corpus and the training tweets from the Microsoft Corpus.

Table 4. Microsoft Dataset: Subjective and Objective Tweets.

Subjective	Objective	Total
940	2,060	3,000

Table 5. Microsoft Dataset: Positive and Negative Tweets.

Positive	Negative	Total
595	345	940

Table 6. Combined Dataset: Subjective and Objective Tweets.

Subjective	Objective	Total
1,968	4,143	6,111

Table 7. Combined Dataset: Positive and Negative Tweets.

Positive	Negative	Total
1,079	889	1,968

The remaining Microsoft tweets were then grouped according to the account with which their author was associated as a friend or follower, and further divided by the day on which they were created. Combined with the social networks created from stage one

of the Microsoft Corpus collection described in section 2.2, these tweets created input for the enhanced community detection described in section 3.

3. Methods

Once data had been collected, initial community detection was performed on the friend/follower networks of all four Twitter accounts in the dataset. After this was completed, sentiment, hashtag, reply, mention and retweet features were computed for each day's data and integrated into the community detection process. This section describes the procedures used to accomplish these tasks, focusing particularly on sentiment analysis.

3.1 Network Construction and Basic Community Detection

To perform community detection on the friend and follower networks, representations of these networks were created as directed graphs with weighted edges. Edges were created according to the friend and follower relationships within the four networks, and assigned a weight value of one. This resulted in the networks described in **Table 8**. The number of vertices in each network was equivalent to the number of accounts in **Table 3**. The number of edges, however, was much higher, as every Twitter account was associated with many connections to other individuals. Crawled from the *@VisualStudio* account, the largest network had had 258,538 edges, while the smallest, from the *@technet* account, contained only 4,834 edges. After the networks had been constructed, both the Infomap and SLPA algorithms were run to perform initial community detection before any additional features were added.

Table 8. Networks Created from the Microsoft Dataset.

Account	Vertices	Edges
@Technet	1,382	4,834
@Windevs	15,559	65,718
@VisualStudio	26,775	258,538
@Silverlight	18,630	127,983

3.2 Updating the Networks with Additional Features

Three types of features were used to augment the results of SLPA and Infomap on the initial friend/follower networks. These included: replies, mentions and retweets; hashtags; and sentiment classification of tweets. These features were computed for all of the 32 days in the dataset. Then, they were used to iteratively increment edge weights in the four social networks, and community detection was repeated on the networks using edge weights updated with each day's data. Variations of this technique were attempted in order to determine optimal performance, such as cumulatively maintaining edge weight updates or resetting the network to the initial friend/follower network after computing communities with each day's data.

3.2.1 Reply, Mention, Retweet, and Hashtag Features

The first and most intuitive feature included as a supplementary feature for community detection was the presence of replies, mentions, and retweets in tweets

referencing other users. The Twitter API conveniently encodes this information in the "entities" section of the data it returns describing tweets, trivializing the extraction of these features. Whenever a reply, mention, or retweet referencing another user in the social network was found in a given day's data, the weight of the edge from the mentioning user to the one mentioned was incremented by one (assuming an edge from the first to the second user existed). The second supplementary feature for community detection was the presence of hashtags in tweets. Whenever two users mentioned the same hashtag in one or more of their tweets from a given day, the weights of any existing edges between those two users were incremented by one. Again, this process was fairly trivial, as Twitter's API also encodes the hashtags used in each tweet in the "entities" section of the metadata describing tweets.

3.2.2 Sentiment Features

While calculating the features described above was relatively straightforward, computing sentiment features was much more involved. This section describes the steps used to train the subjective/objective (SO) and positive/negative (PN) Naïve Bayes classifiers, the accuracy achieved on the training set with these classifiers, and the integration of sentiment features with community detection.

The first step towards training both the SO and PN Naïve Bayes classifiers was converting tweet text into a set of features suitable for input to the appropriate classifier. Both unigram and bigram features were used to train each classifier, with bigram features helping to account for cases when words in a sentence were negated (i.e. preceded by the word "not").

Before unigram and bigram features were created from tweets, tweet text was first preprocessed according to techniques inspired by a previous study [25]. First, all characters in the tweet were converted to lowercase. Then, all hashtags were replaced with "twitterhashtag", retweet designations ("RT") were removed, and usernames were replaced with "twitterusername". Similarly, URLs were replaced with "twitterurl". Then, tweet text was split into individual word tokens, from which a list of unigram and bigram features was created. While these procedures were applied for both the SO and PN classifiers, additional preprocessing was found to improve the accuracy of the PN classifier. For this classifier, repeated punctuation was replaced with the punctuation symbol and a plus sign (i.e. "!!!" would be replaced with "!+"). Additionally, sentence punctuation following words was split into separate individual tokens, and non-sentence punctuation (such as parenthesis and quotation marks) was removed. Stopwords from the NLTK Stopwords Corpus (with the exception of the tokens "don", "no", "s", "t", "not" and "nor") were also removed from the set of tokens representing each tweet for the PN classifier.

Once preprocessing was completed, the combined SO and PN datasets from section 2.3 were used to train the NLTK Naïve Bayes classifiers. Ten-fold cross-validation on the training data was used with both classifiers to approximate their accuracy. The SO classifier achieved an average accuracy of 70% across the ten folds, while the average accuracy of the PN classifier was 79%. The top 20 most informative

features as selected by the two Naïve Bayes classifiers are shown in **Table 9** and **Table 10**, along with their associated labels. Note that, in both of these tables, bigram features are displayed as two tokens enclosed in parenthesis. Several of the tokens identified as most informative in both sets of features are highly domain-specific. This is due to the fact that the Sanders Corpus focuses particularly on tweets relating to Apple and Android.

With construction of the SO and PN classifiers complete, sentiment features were added as supplementary features to community detection. Similarly to the previous two types of supplementary features, sentiment features were used to increment edge weights within the social network, which in turn influenced the performance of the Infomap and SLPA community detection algorithms.

Several steps were involved with updating edge weights based on sentiment. First, each tweet in a given day's data was classified as either subjective or objective. Then, any tweet classified as objective was further classified as positive or negative. Once sentiment classification was complete, hashtags were used to ensure sentiment about unrelated topics was not used to update the network. Thus, edge weights in the network were updated as follows: whenever two users posted a tweet with the same sentiment classification containing the same hashtag, the weights of any edges connecting those users were incremented by one. This effectively could have allowed one tweet to cause two edge updates between two users. If a user's tweet was classified as subjective and assigned the same subjective label (positive or negative) as another user who tweeted the same subjective sentiment and a common hashtag, two edge updates would be made: one for the subjective classification and another for the shared positive or negative classification.

Table 9. Top 20 Most Informative Subjective/Objective Features.

Feature	Label
fucking	subjective
(ios, 5)	subjective
liked	subjective
(customer, service)	subjective
(twitterusername, video)	subjective
:(	subjective
totally	subjective
birthday	subjective
usage	subjective
(can't, wait)	subjective
(twitterusername, ios)	subjective
:-)	subjective
phone!	subjective
wtf	subjective
awesome!	subjective
(twitterusername, thanks)	subjective
customer	subjective
itunes	subjective
followers	objective
trouble	subjective

Table 10. Top 20 Most Informative Positive/Negative Features.

Feature	Label
:)	positive
hate	negative
;)	positive
awesome	positive
itunes	negative
issues	negative
:-)	positive
(?, twitterhashtag)	negative
won't	negative
fuck	negative
sandwich	positive
fucking	negative
sucks	negative
(ice, cream)	positive
cream	positive
battery	negative
issue	negative
else	negative
(?, twitterusername)	negative
(cream, sandwich)	positive

3.3 Enhanced Community Detection and Sentiment Analysis

To enhance the basic community detection described in section 3.1, the three supplementary feature types were used to cumulatively update edge weights in the social network. The social networks for each Microsoft account were updated according to the features in their tweets from each day in the dataset, and community detection using both SLPA and Infomap was performed again after features from each day's data had been included.

As the network was updated community detection was repeated, and the calculated features and detected communities were stored for further analysis. This provided for in-depth analysis of sentiment information uncovered when calculating sentiment features for the network as described in section 4.

4. Results and Analysis

The three types of supplementary features significantly increased modularity in the Infomap output for three of the four Microsoft communities. This section describes and analyzes the results of community detection on these networks and shows how the sentiment features computed to enhance community detection provided even more insight when paired with community detection results. In this way, it is shown that community detection and sentiment analysis can be mutually supportive, each providing information to enhance the other.

4.1 Community Detection

While community detection was performed using both the Infomap and SLPA algorithms, best results were achieved using Infomap with cumulatively maintained edge updates from each day's data. Thus, in **Table 11**, community counts, modularity values,

Table 11. Community Detection Results.

			@technet				@windevs			@VisualStudio			@Silverlight	
Date	*Comm-unities*	*Modu-larity*	*Degree Ratio*	*C.*	*M.*	*D.R.*	*C.*	*M.*	*D.R.*	*C.*	*M.*	*D.R.*
Initial	51	0.3462	0.4891	532	0.1839	0.4006	1141	0.3144	0.3357	445	0.2377	0.4131
2-Jan	51	0.3446	0.4889	536	0.1870	0.3992	1146	0.3107	0.3349	428	0.2385	0.4203
3-Jan	52	0.3523	0.4848	548	0.1879	0.3946	1142	0.2985	0.3346	579	0.3180	0.3722
4-Jan	51	0.3632	0.4838	538	0.1967	0.3974	1145	0.2961	0.3348	587	0.3216	0.3677
5-Jan	51	0.3597	0.4863	549	0.2036	0.3937	1118	0.2967	0.3376	574	0.3243	0.3709
6-Jan	52	0.3612	0.4834	543	0.2061	0.3938	1133	0.2933	0.3355	591	0.3270	0.3671
7-Jan	53	0.3652	0.4716	543	0.2112	0.3956	1141	0.3004	0.3352	596	0.3283	0.3649
8-Jan	51	0.3705	0.4853	563	0.2209	0.3891	1134	0.2992	0.3335	594	0.3309	0.3629
9-Jan	52	0.3860	0.4776	561	0.2257	0.3889	1135	0.2978	0.3332	603	0.3319	0.3608
10-Jan	52	0.3806	0.4878	559	0.2333	0.3888	1147	0.3012	0.3319	593	0.3333	0.3657
11-Jan	53	0.3886	0.4782	554	0.2396	0.3902	1147	0.2893	0.3289	603	0.3394	0.3621
12-Jan	52	0.3932	0.4827	561	0.2418	0.3883	1121	0.2925	0.3336	625	0.3376	0.3548
13-Jan	52	0.3903	0.4804	562	0.2455	0.3893	1117	0.3035	0.3353	614	0.3416	0.3569
14-Jan	54	0.3926	0.4670	567	0.2480	0.3883	1136	0.2998	0.3324	623	0.3371	0.3528
15-Jan	55	0.3883	0.4644	559	0.2539	0.3899	1145	0.3000	0.3290	620	0.3398	0.3556
16-Jan	54	0.4106	0.4707	557	0.2662	0.3923	1153	0.3013	0.3303	624	0.3369	0.3516
17-Jan	55	0.4019	0.4708	559	0.2688	0.3915	1153	0.3047	0.3290	628	0.3420	0.3522
18-Jan	54	0.4024	0.4640	547	0.2720	0.3932	1128	0.3096	0.3318	621	0.3443	0.3529
19-Jan	57	0.4023	0.4568	564	0.2734	0.3894	1144	0.3089	0.3305	630	0.3504	0.3518
20-Jan	56	0.4061	0.4639	564	0.2719	0.3878	1137	0.3103	0.3302	614	0.3449	0.3542
21-Jan	57	0.4078	0.4568	567	0.2737	0.3891	1158	0.3097	0.3276	621	0.3542	0.3548
22-Jan	57	0.4093	0.4579	553	0.2893	0.3941	1159	0.3128	0.3274	634	0.3574	0.3546
23-Jan	59	0.4077	0.4522	567	0.2886	0.3898	1142	0.3141	0.3298	644	0.3559	0.3490
24-Jan	58	0.4141	0.4561	562	0.2935	0.3923	1172	0.2985	0.3255	636	0.3566	0.3497
25-Jan	59	0.4150	0.4510	578	0.2950	0.3870	1168	0.3007	0.3250	639	0.3557	0.3505
26-Jan	58	0.4151	0.4560	572	0.2989	0.3901	1165	0.3017	0.3245	644	0.3555	0.3499
27-Jan	58	0.4145	0.4548	561	0.3055	0.3917	1176	0.3028	0.3237	625	0.3607	0.3523
28-Jan	58	0.4186	0.4558	570	0.3045	0.3892	1178	0.3060	0.3234	629	0.3631	0.3548
29-Jan	59	0.4192	0.4518	580	0.3048	0.3850	1160	0.3073	0.3249	645	0.3631	0.3492
30-Jan	60	0.4260	0.4507	574	0.3038	0.3882	1221	0.3210	0.3164	646	0.3577	0.3479
31-Jan	59	0.4281	0.4524	573	0.3096	0.3866	1161	0.3163	0.3251	665	0.3604	0.3434
1-Feb	60	0.4294	0.4503	564	0.3116	0.3897	1171	0.3177	0.3230	652	0.3610	0.3480
2-Feb	60	0.4287	0.4501	577	0.3129	0.3853	1182	0.3163	0.3215	659	0.3615	0.3428

and degree ratios for the Infomap algorithm are given (abbreviated in three of the four columns as “C.”, “M.”, and “D.R.” respectively). These values are shown for the initial community partitions computed by Infomap for all four Microsoft accounts and for each of the 32 days in the dataset. As can be seen from the data in **Table 11**, modularity values increased significantly for the *@technet*, *@windevs*, and *@Silverlight* accounts from the initial network to the final February 2 network. The greatest increase in modularity was in the network from the *@windevs* account, with the modularity value for the partitioning increasing from 0.1839 to 0.3129. This suggests that more meaningful communities were uncovered by updating network edge weights with the three supplementary feature types. Increasing weight values of edges connecting nodes with common features allowed community partitions that were likely more representative of the real world interactions in these networks to be discovered.

Interestingly, unlike the other three accounts, the modularity values for the *@VisualStudio* account did not noticeably increase when the supplementary features were used. While the reason for this is not immediately clear, there are several possible explanations for these results. The *@VisualStudio* network is significantly larger than any of the other networks, with twice as many edges as the next largest network tested. Furthermore, the average degree of nodes in the *@VisualStudio* network, 9.66, is much higher than any of the other networks (the next highest, from the *@Silverlight* network, is 6.87). Thus, considering the relatively large number of edges in the *@VisualStudio* network, updates to edge weights based on the three supplementary features may not have been sufficient to significantly alter the community detection output.

Also, while modularity tended to increase with the inclusion of additional features, the degree ratio values decreased as edge weights were updated. While this may at first seem counterintuitive, it is actually to be expected. The modularity metric takes edge weights into account, while the degree ratio does not. Thus, since the number of distinct communities detected increased as additional days’ features were included (while maintaining the same internal edge structure), the degree ratio scores decreased.

4.2 Sentiment Analysis

As demonstrated in section 4.1, the three types of supplementary features helped discover communities exhibiting stronger real-world interaction. However, just as sentiment analysis helped facilitate enhanced community detection, community detection also served to enhance sentiment analysis. Correlating sentiment information with detected communities permits more in-depth analysis of sentiment information from the level of entire networks down to single communities.

Sentiment analysis data was computed and stored as a part of the community detection process for each of the four Microsoft accounts, which simplified further analysis of this data. In **Figure 2**, sentiment statistics are presented for each Microsoft account over the 32 days in the dataset. Each of the four curves shown in these figures represents a percentage of the total tweets from each day classified as subjective, objective, positive, or negative. As can be seen from each of these figures, the majority of tweets from each of these networks were classified as objective, with the *@technet*

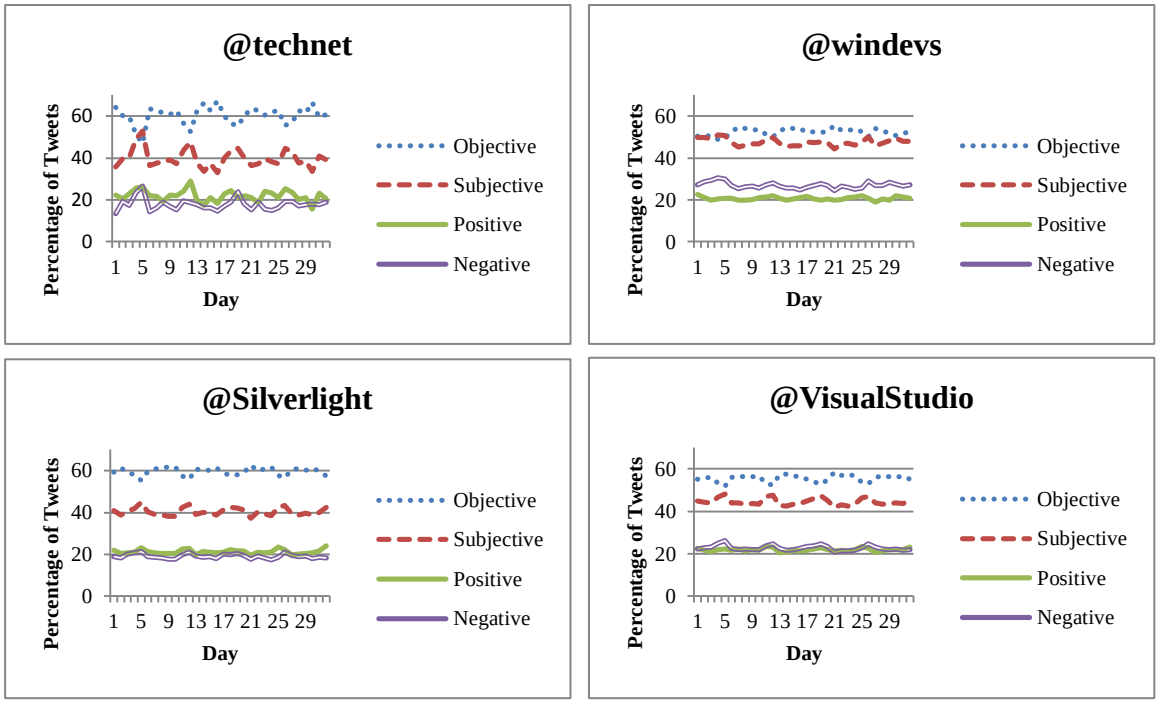

Figure 2. Overall Sentiment Statistics for Microsoft Networks.

network exhibiting the highest ratio of objective (average of 60% per day) to subjective (average of 40% per day) tweets. In three of the four networks, the percentages of positive and negative tweets were relatively equal. In the *@windevs* network, however, the objective tweets were primarily negative. Of the objective tweets produced by this network, 44% per day on average were positive, while 56% were classified as negative.

While looking at the overall sentiment trends for each of these accounts is interesting, this information is ultimately of limited usefulness. A multitude of topics are discussed by many different groups of people in each of these networks. Realistically, it would be much more helpful to consider sentiment expressed by specific groups of people or about specific topics.

To facilitate a more narrowed analysis, popular tweet topics in the dataset were identified using hashtags. **Table 12** shows the top ten most popular hashtags appearing in the entire dataset and the number of times each was used. Four out of these ten hashtags ("windowsphone", "wpdev", "wp", and "wp8") reference Microsoft's smartphone platform, Windows Phone, indicating that this was a popular topic during the period the dataset was collected. Thus, to demonstrate the power of combining sentiment analysis with community detection, Windows Phone was chosen as a topic for deeper analysis. To understand overall sentiment towards Windows Phone, sentiment classifications were tallied for tweets containing the above four hashtags from the networks associated with

Table 12. Top Ten Hashtags.

Hashtag	Count
tech	6892
windows8	5463
microsoft	4871
windowsphone	4188
wpdev	3788
wp	3332
wp8	2906
getglue	2407
android	2260
fb	2258

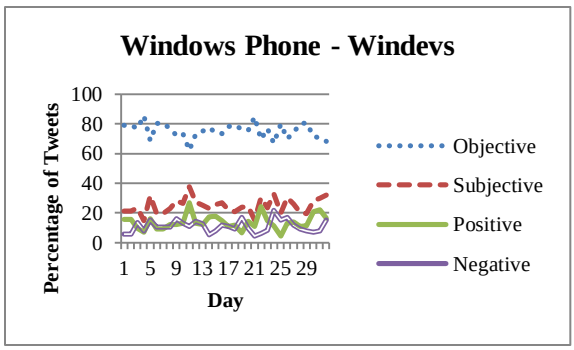

Figure 3. Windows Phone Sentiment - @windevs.

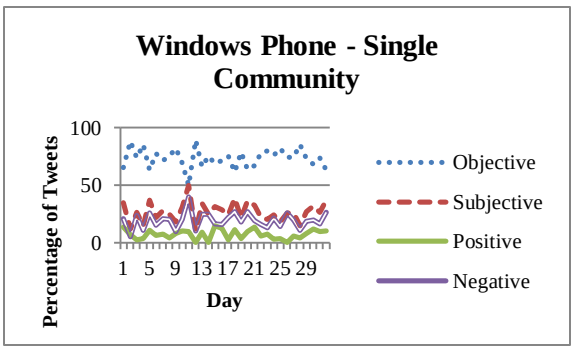

Figure 4. Windows Phone Sentiment - Largest Community from @windevs.

each Microsoft account. **Figure 3** shows percentages of tweets pertaining to Windows Phone expressing each of the four sentiment labels from one of the Microsoft networks, *@windevs*. In the *@windevs* network, the majority of tweets from each day pertaining to Windows phone were classified as objective (73% on average), and of the subjective tweets there was a fairly even split of positively (12% on average) and negatively (14% on average) classified tweets. **Figure 3** represents sentiment scores pertaining to the entire *@windevs* network. But, by combining community detection and sentiment analysis results, an even more granular perspective becomes available. **Figure 4** displays Windows Phone-related sentiment classification percentages for the largest detected community in the *@windevs* network, and reveals a significantly different trend than that of the *@windevs* network as a whole. While the majority of tweets produced by this community are still objective (an average of 73% per day), of the subjectively classified tweets only 7% per day on average are positive, while 19% on average are negative. Thus, the largest community detected within the *@windevs* network exhibited significantly more negative sentiment towards Windows Phone than the network as a whole. This demonstrates how community data was able to enhance sentiment analysis by permitting a more granular view of sentiment from a specific community within the larger *@windevs* network.

5. Conclusion

Due to the rising popularity of online social networks, analysis of OSNs has become the focus of many recent research efforts. Two common topics are community detection and sentiment analysis, which examine the structure and content of social networks. Though community detection and sentiment analysis are usually treated as separate issues, this research integrates the two and demonstrates how these techniques can be used to enhance each other.

The publicly available Sanders Sentiment Corpus was used to provide data for this study, in addition to four Microsoft-related social networks downloaded directly from the Twitter API. While the Sanders Corpus was relatively small, the Microsoft dataset was significantly larger. Overall, the combined Microsoft dataset contained the friend and follower networks of 62,346 Twitter users and 2,061,789 of their tweets, collected over a period of 32 days.

Community detection was performed on the friend/follower networks of the four Microsoft accounts using the SLPA and Infomap algorithms. This community detection was then enhanced with three types of additional features: replies, mentions, and retweets; hashtags; and sentiment classifications. The sentiment classifications were derived using two Naïve Bayes classifiers trained with the Sanders dataset and a small portion of hand-labeled tweets from the Microsoft dataset. These three feature types were calculated from each day's tweets, and were applied to the four networks in the dataset by increasing edge weights between network nodes.

Using the three supplementary feature types to enhance community detection improved modularity values in the Infomap output on three of the four networks studied. The most dramatic change in modularity was in the *@windevs* network, with modularity increasing from 0.1839 to 0.3129. Furthermore, combining sentiment classifications and community groupings permitted more in-depth analysis of sentiment data from the same *@windevs* network, which was illustrated by examining sentiment directed towards Microsoft's Windows Phone. Thus, this study takes the novel approach of combining community detection and sentiment analysis, demonstrating that these techniques can be used in a mutually informative way with each enhancing the other.

References

[1] A. H. Wang, "Don't follow me: Spam detection in Twitter," *Proceedings of the 2010 International Conference on Security and Cryptography (SECRYPT)*, Piraeus, 26-28 July, 2010, pp. 1-10.

[2] T. Falkowski, A. Barth and M. Spiliopoulou, "Studying community dynamics with an incremental graph mining algorithm," *Proceedings of the 14th Americas Conference on Information Systems (AMCIS 2008)*, Toronto, 14-17 August, 2008, pp. 1-11.

[3] K. Liu, W. Li and M. Guo, "Emoticon Smoothed Language Models for Twitter Sentiment Analysis," *Proceedings of the Twenty-Sixth AAAI Conference on Artificial Intelligence*, Toronto, 22-26 July, 2012, pp. 1678-1684.

[4] A. Burns and B. Eltham, "Twitter Free Iran: an Evaluation of Twitter's Role in Public Diplomacy and Information Operations in Iran's 2009 Election Crisis," *Record of the Communications Policy and Research Forum 2009*, Sydney, 19-20 November, 2009, pp. 322-334.

[5] D. Gayo-Avello, P. T. Metaxas and E. Mustafaraj, "Limits of Electoral Predictions Using Twitter," *Proceedings of the International Conference on Weblogs and Social Media (ICWSM) 2011*, Barcelona, 17-21 July, 2011, pp. 490-493.

[6] B. J. Jansen, M. Zhang, K. Sobel and A. Chowdury, "Twitter Power: Tweets as Electronic Word of Mouth," *Journal of the American Society for Information and Technology*, Vol. 60, No. 11, 2009, pp. 2169-2188.

[7] M. van Meeteren, A. Poorthuis and E. Dugundji, "Mapping Communities in Large Virtual Social Networks," *Proceedings of the First International Forum on the Application and Management of Personal Electronic Information*, Cambridge, 12-13 October, 2009.

[8] S. E. Schaeffer, "Graph Clustering," *Computer Science Review*, Vol. 1, No. 1, 2007, pp. 27-64.

[9] M. Girvan and M. E. J. Newman, "Community Structure in Social and Biological Networks," *Proceedings of the National Academy of the Sciences of the United States of America*, Vol. 99, No. 12, 2002, pp. 7821-7826.

[10] I. X. Y. Leung, P. Hui, P. Lio and J. Crowcroft, "Towards Real-time Community Detection in Large Networks," *Physical Review E*, Vol. 79, No. 6, 2009, pp. 066107.

[11] T. Falkowski, A. Barth and M. Spilioupoulou, "Dengraph: A Density-based Community Detection Algorithm," *Proceedings of the IEEE/WIC/ACM International Conference on Web Intelligence*, Silicon Valley, 2-5 November, 2007, pp. 113-115.

[12] M. E. J. Newman and M. Girvan, "Finding and Evaluating Community Structure in Networks," *Physical Review E*, Vol. 69, No. 2, 2004, pp. 026113.

[13] S. Fortunato and M. Barthelemy, "Resolution Limit in Community Detection,"

Proceedings of the National Academy of Sciences of the United States of America, Vol. 104, No. 1, 2007, pp. 36-41.

[14] B. H. Good, Y. de Montjoye and A. Clauset, "Performance of Modularity Maximization in Practical Contexts," *Physical Review E*, Vol. 81, No. 4, 2010, pp. 046106.

[15] A. Lancichinetti, F. Radicchi, J. J. Ramasco and S. Fortunato, "Finding Statistically Significant Communities in Networks," *PLoS ONE*, Vol. 6, No. 4, 2011, pp. e18961.

[16] A. Lancichinetti, S. Fortunato and J. Kertész, "Detecting the Overlapping and Hierarchical Community Structure in Complex Networks," *New Journal of Physics*, Vol. 11, No. 3, 2009, pp. 033015.

[17] M. Newman, "Networks: An Introduction," 1st Edition, Oxford University Press, Inc., New York, 2010.

[18] J. Xie, "Agent-Based Dynamics Models for Opinion Spreading and Community Detection in Large-Scale Social Networks," Ph.D. Thesis, Rensselaer Polytechnic Institute, Troy, 2012.

[19] M. Rosvall and C. T. Bergstrom, "Maps of Random Walks on Complex Networks Reveal Community Structure," *Proceedings of the National Academy of the Sciences of the United States of America*, Vol. 105, No. 4, 2008, pp. 1118-1123.

[20] A. Esuli and F. Sebastiani, "SENTIWORDNET: A Publicly Available Lexical Resource for Opinion Mining," *Proceedings of the 5th Conference on Language Resources and Evaluation*, Genoa, 24-26 May, 2006, pp. 417-422.

[21] A. Hamouda and M. Rohaim, "Reviews Classification Using SentiWordNet Lexicon," *The Online Journal on Computer Science and Information Technology*, Vol. 2, No. 1, 2011, pp. 120-123.

[22] F. Chaumartin, "UPAR7: A Knowledge-Based System for Headline Sentiment Tagging," *Proceedings of the 4th International Workshop on Semantic Evaluations*, Prague, 23-24 June, 2007, pp. 422-425.

[23] K. Denecke, "Using SentiWordNet for Multilingual Sentiment Analysis," *Proceedings of the IEEE 24th International Conference on Data Engineering Workshop*, Cancún, 7-12 April, 2008, pp. 507-512.

[24] B. Ohana and B. Tierney, "Sentiment Classification of Reviews Using SentiWordNet," *Proceedings of the 9th IT&T Conference*, Dublin, 22-23 October, 2009.

[25] A. Pak and P. Paroubek, "Twitter As a Corpus for Sentiment Analysis and Opinion Mining," *Proceedings of the International Conference on Language Resources and Evaluation*, Malta, 19-21 May, 2010, pp. 1320-1326.

[26] M. Speriosu, N. Sudan, S. Upadhyay and J. Baldridge, "Twitter Polarity Classification with Label Propagation Over Lexical Links and the Follower Graph," *Proceedings of the First Workshop on Unsupervised Learning in NLP*, Edinburgh, 30 July, 2011, pp. 53-63.