Content-Driven Detection of Campaigns in Social Media

Kyumin Lee, James Caverlee, Zhiyuan Cheng
Department of Computer Science and Engineering
Texas A&M University
College Station, TX 77843
{kyumin, caverlee, zcheng}@cse.tamu.edu

Daniel Z. Sui
Department of Geography
Ohio State University
Columbus, OH 43210
sui.10@osu.edu

ABSTRACT

We study the problem of detecting coordinated free text campaigns in large-scale social media. These campaigns – ranging from coordinated spam messages to promotional and advertising campaigns to political astro-turfing – are growing in significance and reach with the commensurate rise of massive-scale social systems. Often linked by common "talking points", there has been little research in detecting these campaigns. Hence, we propose and evaluate a content-driven framework for effectively linking free text posts with common "talking points" and extracting campaigns from large-scale social media. One of the salient aspects of the framework is an investigation of graph mining techniques for isolating coherent campaigns from large message-based graphs. Through an experimental study over millions of Twitter messages we identify five major types of campaigns – Spam, Promotion, Template, News, and Celebrity campaigns – and we show how these campaigns may be extracted with high precision and recall.

Categories and Subject Descriptors: H.3.5 [Online Information Services]: Web-based services; J.4 [Computer Applications]: Social and behavioral sciences

General Terms: Algorithms, Design, Experimentation

Keywords: social media, campaign detection

1. INTRODUCTION

Social media is inherently a persuasive technology, supporting the rapid insertion of new memes, near instantaneous global reach, and unprecedented leveraging of massive-scale interpersonal connections. On the one hand, many users of social media organically engage with social media to share opinions and interact with friends; on the other, social media is a prime target for strategic influence.

For example, there is widespread anecdotal evidence of "astroturfing" campaigns [3], in which political operatives insert memes into sites like Twitter and Facebook in an effort to influence discourse about particular political candidates and topics. In addition, there are large campaigns of coordinated spam messages in social media [4], templated messages (e.g., auto-posted messages to social media sites from third-party applications announcing a user action, like joining a game or viewing a video), high-volume time-synchronized messages (e.g., many users may repost news headlines to social media sites in a flurry after the news has been initially reported), and so on. In the case of spam and promotion campaigns, the relative openness of many social media sites (typically requiring only a valid email address to register) suggests coordinated campaigns could be a low-cost approach for strategically influencing participants.

User-driven campaigns – often linked by common "talking points" – appear to be growing in significance and reach with the commensurate rise of massive-scale social systems. However, there has been little research in detecting these campaigns. While there has been some progress in detecting isolated instances of long-form fake reviews (e.g., to promote books on Amazon), of URL-based spam in social media, and in manipulating recommender systems [4, 5, 6, 7], there is a significant need for new methods to support web-scale detection of campaigns in social media.

Hence, we focus in this paper on detecting one particular kind of coordinated campaign – those that rely on "free text" posts, like those found on blogs, comments, forum postings, and short status updates (like on Twitter and Facebook). For our purposes, a campaign is a collection of users and their posts bound together by some common objective, e.g., promoting a product, criticizing a politician, or inserting disinformation into an online discussion. Our goal is to link messages with common "talking points" and then extract multi-message campaigns from large-scale social media. Detecting these campaigns is especially challenging considering the size of popular social media sites like Facebook and Twitter with 100s of millions of unique users and the inherent lack of context in short posts.

We explore in this paper several content-based approaches for identifying campaigns from the massive scale of real-time social systems. Concretely, we propose and evaluate a content-driven framework for effectively linking free text posts with common "talking points" and extracting campaigns from large-scale social media. We find that over millions of Twitter messages, the proposed framework can identify 100s of coordinated campaigns, ranging in size up to several hundred messages per campaign.

2. OVERALL APPROACH

In this section, we describe the problem of campaign detection in social media, introduce the data, and outline the metrics for measuring effective campaign detection.

2.1 Problem Statement

We consider a collection of n participants across social media sites $U = \{u_1, u_2, \ldots, u_n\}$, where each participant u_i may post a time-ordered list of k messages $M_{u_i} = \{m_{i1}, m_{i2}, \ldots, m_{ik}\}$. Our hypothesis is that among these messages, there may exist *coordinated campaigns.*

Given the set of users U, a *campaign* M_c can be defined as a collection of messages and the users who posted the messages: $M_c = \{m_{ij}, u_i | u_i \in U \cap m_{ij} \in M_{u_i} \cap theme(m_{ij}) \in t_k\}$ such that the campaign messages belong to a coherent theme t_k. Themes are human-defined logical assignments to messages and application dependent. For example, in the context of spam detection, a campaign may be defined as a collection of messages with a common target product (e.g., Viagra). In the context of astroturf, a campaign may be defined as a collection of messages promoting a particular viewpoint (e.g., the veracity of climate change). Additionally, depending on the context, a message may belong to one or multiple themes. For the purposes of this paper and to focus our scope of inquiry, we consider as a theme all messages sharing similar "talking points" as determined by a set of human judges.

2.2 Data

To evaluate the quality of a campaign detection approach, we would ideally have access to a large-scale "gold set" of known campaigns in social media. While researchers have published benchmarks for spam webpages, ad-hoc text retrieval, and other types of applications, we are not aware of any standard social media campaign dataset. Hence, we take in this paper a twofold approach: (i) a small-scale validation over hand-labeled data; and (ii) a large-scale validation over 1.5 million Twitter messages for which ground truth is not known.

$\mathbf{CD_{Small}}$: First, we sample a small collection of messages (1,912) posted to Twitter in October 2010. Over this small *campaign dataset* (CD_{Small}), two judges labeled all pairs of the 1,912 tweets as sharing similar "talking points" or not, finding 298 pairs of messages sharing similar "talking points". Based on these initial labels, the judges considered all combinations of messages that may form campaigns consisting of four messages or more, and found 11 campaigns ranging in size from four messages to eight messages. While small in size, this hand-labeled dataset allows us to evaluate the precision and recall of several campaign detection methods.

$\mathbf{CD_{Large}}$: Second, we supplement the small dataset with a large collection of messages (∼1.5 million) posted to Twitter between October 1 and October 7, 2010. We sampled these messages using Twitter's Streaming API, resulting in a representative random sample of Twitter messages. Over this large *campaign dataset* (CD_{Large}), we can test the precision of the campaign detection methods and investigate the types of campaigns that are prevalent in-the-wild. Since we do not have ground truth knowledge of all campaigns in this dataset, our analysis will focus on the campaigns detected for which we can hand-label as actual campaigns or not.

2.3 Metrics

To measure the effectiveness of a campaign detection method, we use variations of average precision, average recall, and the average F_1 measure. The average precision (AP) for a campaign detection method is defined as:

$$AP = \frac{1}{n}\sum_{i=1}^{n} \frac{\max CommonMessages(PC_i, TCs)}{|PC_i|}$$

where n is the total number of predicted campaigns by the campaign detection method, PC is a predicted campaign, and TC is an actual (true) campaign. $MaxCommonMessage$ function returns the maximum of the number of common messages in both the predicted campaign i (PC_i) and each of the actual (true) campaigns (TCs). For example, suppose a campaign detection method identifies a three-message campaign: $\{m_1, m_{10}, m_{30}\}$. Suppose there are two actual campaigns with at least one message in common: $\{m_{30}, m_{38}, m_{40}\}$ and $\{m_1, m_{10}, m_{35}, m_{50}, m_{61}\}$. Then the Precision is $max(2,1)/3 = 2/3$. In the aggregate, this individual precision will be averaged with all n predicted campaigns.

Similarly, we can define the average recall (AR) as:

$$AR = \frac{1}{n}\sum_{i=1}^{n} \frac{\max CommonMessages(PC_i, TCs)}{|TC_j|}$$

where n is the number of the predicted campaigns, and TC_j is a true campaign which has the largest common messages with the predicted campaign i (PC_i). Continuing the example from above, the Recall would be $max(2,1)/5 = 2/5$.

Finally, we can combine precision and recall as the average F_1 measure (AF):

$$AF_1 = \frac{2 * AP * AR}{AP + AR}$$

An effective campaign detection approach should identify predicted campaigns that are composed primarily of a single actual campaign (i.e., have high precision) and that contain most of the messages that actually belong to the campaign (i.e., have high recall). A method that has high precision but low recall will result in only partial coverage of all campaigns available (which could be especially disastrous in the case of spam or promotional campaigns that should be filtered). A method that has low precision but high recall may identify nearly all messages that belong to campaigns but at the risk of mislabeling non-campaign messages (resulting in false positives, which could correspond to mis-labeled legitimate messages as belonging to spam campaigns).

3. CONTENT-DRIVEN CAMPAIGN DETECTION

To detect coordinated campaigns, we explore in this paper several content-based approaches for identifying campaigns. Our goal is to identify methods that can balance both precision and recall for effective campaign detection. We primarily consider a graph-based framework, where we model messages in social media as a *message graph.* Each node in the message graph corresponds to a message; edges correspond to some reasonable notion of content-based correlation between messages, corresponding to pairs of messages with similar "talking points." Formally, we have:

Definition 1 (Message Graph). *A message graph is a graph $G = (V, E)$ where every message in M corresponds*

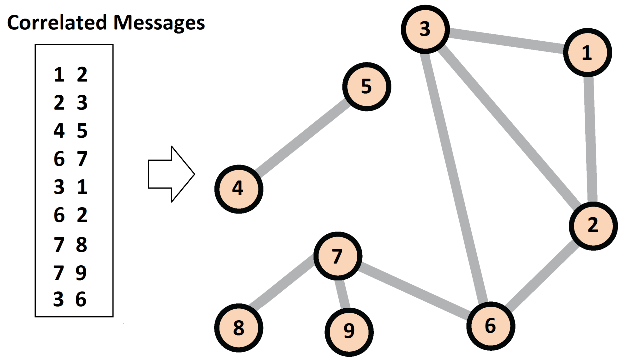

Figure 1: The campaign message graph

to a vertex m_{ix} in the vertex set V. An edge $(m_{ix}, m_{jy}) \in E$ exists for every pair of messages (m_{ix}, m_{jy}) where corr(m_{ix}, m_{jy}) $> \tau$, for a measure of correlation and some parameter τ.

A message graph which links unrelated messages will necessarily result in poor campaign detection (by introducing spurious links). Traditional information retrieval approaches for document similarity (e.g., cosine similarity, KL-divergence) as well as efficient near-duplicate detection methods (e.g., Shingling [1], I-Match [2] and SpotSigs [8]) have typically not been optimized for the kind of short posts of highly-variable quality common in many social media sites (including Facebook and Twitter). Hence, we shall investigate experimentally several possible approaches for determining pairwise message correlation which guides the formation of the message graph.

Given a message graph, we propose to explore three graph-based approaches for extracting campaigns:(i) loose extraction; (ii) strict extraction; and (iii) cohesive extraction. Experimentally, we compare these graph-based approaches versus a traditional k-means clustering approach and reach poor results for clustering as compared to the graph methods. For now, we focus our attention on extracting content-driven campaigns via graph mining.

3.1 Loose Campaign Extraction

The first approach for content-driven campaign detection is what we refer to as *loose campaign extraction*. The main idea is to identify as a logical campaign all chains of messages that share common "talking points". In this way, the set of all loose campaigns is the set of all maximally connected components in the message graph:

DEFINITION 2 (LOOSE CAMPAIGN). *A loose campaign is a subgraph $s = (V', E')$, such that s is a maximally connected component of G, in which s is connected, and for all vertices m_{ix} such that $m_{ix} \in V$ and $m_{ix} \notin V'$ there is no vertex $m_{jy} \in V'$ for which $(m_{ix}, m_{jy}) \in E$.*

As an example, Figure 1 illustrates a collection of 10 messages, edges corresponding to messages that are highly correlated, and the two maximal components (corresponding to loose campaigns): {1, 2, 3, 6, 7, 8, 9} and {4, 5}. Such an approach to campaign detection faces a critical challenge, however: not all maximally connected components are necessarily campaigns themselves (due to long chains of tangentially-related messages). For example, a chain of similar messages A–B–C–...–Z, while displaying local similarity properties (e.g., between A and B and between Y and Z) will necessarily have low similarity across the chain (e.g., A and Z will be dissimilar since there is no edge between the pair, as in the case of messages 9 and 1 in Figure 1). In practice, such maximally connected components could contain disparate "talking points" and not strong campaign coherence.

3.2 Strict Campaign Extraction

A natural alternative is to constrain campaigns to be maximal cliques, what we call *strict campaigns*:

DEFINITION 3 (STRICT CAMPAIGN). *A strict campaign $s' = (V'', E'')$ in a message graph $G = (V, E)$, in which $V'' \subseteq V$ and $E'' \subseteq E$, such that for every two vertices m_{ix} and m_{jy} in V'', there exists an edge $(m_{ix}, m_{jy}) \in E''$ and the clique cannot be enlarged by including one more adjacent vertex (corresponding to a message in M).*

To identify these strict campaigns, we can first identify all loose campaigns – by identifying all maximally connected components over the message graph, we can prune from consideration all singleton messages and are left with a set of candidate campaigns. Over these candidates, we can identify the strict campaigns through maximal clique mining. However, discovering all maximal cliques from a graph is an NP-hard problem (i.e., the time complexity is exponential). Finding all maximal cliques takes $O(3^{n/3})$ in the worst case where n is the number of vertices [9]. Over large graphs, even with parallelized implementation over MapReduce-style compute clusters, the running time is still $O(3^{n/3}/m)$ in the worst case, where n is the number of vertices and m is the number of reducers [11].

And there is still the problem that even with a greedy approximation, strict campaign detection may overconstrain the set of campaigns, especially in the case of loosely-connected campaigns. Returning to the example in Figure 1, the maximal cliques {1, 2, 3} and {2, 3, 6} would be identified as strict campaigns, but perhaps {1, 2, 3, 6, 7} form a coherent campaign even though the subgraph is not fully-connected. In this case the strict approach will identify multiple overlapping campaigns and will miss the larger and (possibly) more coherent campaign. In terms of our metrics, the expectation is that strict campaign detection will favor precision at the expense of recall.

3.3 Cohesive Campaign Extraction

Hence, we also consider a third approach which seeks to balance loose and strict campaign detection by focusing on what we refer to as *cohesive campaigns*, which relaxes the conditions of maximal cliques:

DEFINITION 4 (COHESIVE CAMPAIGN). *Given a message graph G = (V,E), a subgraph G' is called a **cohesive campaign** if the number of edges of G' is close to the maximal number of edges with the same number of vertices of G'.*

The intuition is that a cohesive campaign will be a dense but not fully connected subgraph, allowing for some variation in the "talking points" that connect subcomponents of the overall campaign. There are a number of approaches mining dense subgraphs and the exact solution is again NP-hard in computation complexity, so we adopt a greedy approximation approach following the intuition in [10]. The approach to extract cohesive campaigns requires a notion of maximum co-clique $CC(m_{ix}, m_{jy})$ for all neighbors:

Definition 5 (Maximum co-clique: $CC(m_{ix}, m_{jy})$). *Given a message graph G = (V,E), the maximum co-clique $CC(m_{ix}, m_{jy})$ is the (estimated) size of the largest clique containing both vertices m_{ix} and m_{jy}, where $m_{jy} \in V$ and m_{jy} is a neighbor vertex of m_{ix} (i.e., they are connected).*

Considering all of a vertex's neighbors, we define the largest of the maximum co-cliques as $C(m_{ix})$:

Definition 6 ($C(m_{ix})$). *Then, $C(m_{ix})$ is the largest value between m_{ix} and any neighbor m_{jy}, formally defined as $C(m_{ix}) = max\{CC(m_{ix}, m_{jy}), \forall m_{jy} \in Neighbor(m_{ix})\}$.*

With these definitions in mind, our approach to extract cohesive campaign is as follows:

1. Estimate each vertex's $C(m_{ix})$: In the first step, our goal is to estimate the C values for every vertex in a candidate campaign which indicates the upper bound of the maximum clique size the vertex belongs to. Starting at a random vertex m_{ix} in s, we compute the maximum co-clique size $CC(m_{ix}, m_{jy})$, where $m_{jy} \in V'$ and m_{jy} is a neighbor vertex of m_{ix}. Then, we compute $C(m_{ix})$. We insert m_{jy} into a priority queue and sort all m_{jy} by $CC(m_{ix}, m_{jy})$. Next, we greedily advance to the m_{jy}, which has the largest $CC(m_{ix}, m_{jy})$ among all m_{jy}, and remove it from the queue. Finally, we compute $C(m_{jy})$. We repeat this procedure for every vertex in the candidate campaign. At the conclusion of this procedure, we have an estimated $C(m_{ix})$ for every vertex.

2. Cohesive campaign extraction: Given the estimated $C(m_{ix})$ for every vertex in a candidate campaign, by considering the order in which the greedy algorithm in Step 1 encounters each vertex, we can consider consecutive neighbors as potential members of the same coherent campaign. Intuitively, the $C(m_{ix})$ values should be high for vertices in dense subgraphs but should drop as the algorithm encounters nodes on the border of the dense subgraph, then rise again as the algorithm encounters vertices belonging to a new dense subgraph. We identify the first vertex with an increasing $C(m_{ix})$ over its neighbor as the initial boundary of a cohesive campaign. We next include all vertices between this first boundary up to and including the vertex with a $C(m_{ix})$ value larger than or equal to some threshold (= the local peak value * λ). By tuning λ to 1, the extracted cohesive campaigns will be nearly clique-like; lower values of λ will result in more relaxed campaigns (i.e., with less density). We repeat this procedure until we extract all cohesive subgraphs in the candidate campaign.

The output of the cohesive campaign extraction approach is a list of cohesive campaigns, each of which contains a list of vertices forming a cohesive subgraph.

4. EXPERIMENTAL STUDY

In this section, we explore campaign discovery over social media through an application of the framework to messages sampled from Twitter. We begin by examining how to accurately and efficiently construct the campaign message graph, which is the critical first step necessary for campaign detection. We find that a short-text modified Shingling-based approach results in the most accurate message graph construction. Based on this finding, we next explore campaign detection methods over the small hand-labeled Twitter dataset, before turning our sights to analysis of campaigns discovered over the large (1.5 million messages) Twitter dataset.

4.1 Message Graph Construction

Recall that each node in the message graph corresponds to a message; edges correspond to some reasonable notion of "relatedness" between messages corresponding to human-labeled similar "talking points". Our first goal is to answer the question: can we effectively determine if two messages are correlated (i.e., algorithmically determine if they share similar "talking points") across hundreds of millions of short messages for constructing the message graph in the first place? This step is critical for accurate message graph formation for discovering campaigns.

Using the small *campaign dataset* (CD_{Small}), we consider the 298 pairs of messages sharing similar "talking points" (as determined by human judges) as the ground truth for whether an edge should appear in the message graph between the two messages. We can measure the effectiveness of a message correlation method by precision, recall, and F_1.

We investigate the identification of correlated messages through a comparative study of five distinct techniques: unigram-based overlap between messages, edit distance, and three representative near-duplicate detection algorithms (Shingling [1], I-Match [2], SpotSigs [8]). Near-duplicate detection approaches have shown great promise and effectiveness by web search engines to efficiently identify duplicate web content, but their application to inherently short messages lacking context is unclear.

In our experiment, we see that the Shingling approach performs the best, with an $F_1 = 0.81$. To improve the performance of the Shingling approach with Jaccard coefficient, we propose as a measure of correlation the overlap coefficient ($corr_{overlap}(A, B) = \frac{|A \cap B|}{min(|A|, |B|)}$). With the overlap coefficient, we get $F_1 = 0.88$. In the further experiments, we use the Shingling approach with overlap coefficient.

4.2 Campaign Detection over Small Data

In the previous experiment, we evaluated several approaches to measuring message correlation. Now we turn our attention to evaluating campaign detection methods. We begin in this section with the small data set (which recall allows us to measure precision and recall against ground truth) before considering the large data set.

Over the hand-labeled campaigns in CD_{Small}, we apply the three graph-based campaign extraction methods: (i) loose; (ii) strict; and (iii) cohesive, over the message graph generated via the best performing message correlation method identified in the previous section. We also compare campaign extraction using a fourth approach based on text clustering. For this non-graph-based approach, we consider k-means clustering, where each message is treated as vector with 10K bag-of-words features, weighted using TF-IDF, with Euclidian distance as a distance function. We vary the choice of k value, and report the best result.

Table 1 presents the experimental results of the four cam-

Table 1: Effectiveness Comparison of Campaign Detection Approaches

Approach	NumC	F_1	Precision	Recall
Loose	12	0.962	0.986	0.940
Strict	12	0.906	0.907	0.904
Cohesive	11	**0.963**	0.977	0.950
k-means	5	0.89	1	0.805

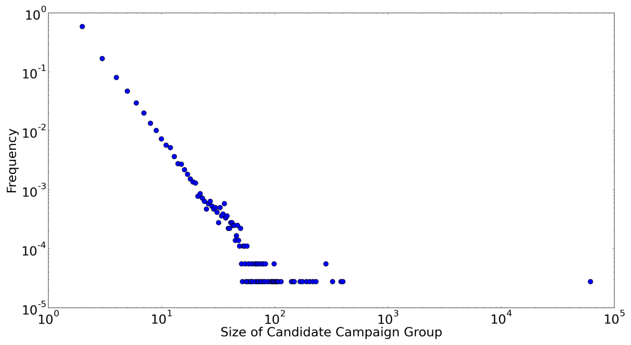

Figure 2: Size of Candidate Campaigns

paign detection approaches. The cohesive campaign detection approach found 11 campaigns ($NumC$) like the ground truth, but missed a message in two campaigns. The strict approach found 12 campaigns, missed one message in a true campaign, and divided a true campaign to two predicted campaigns because the approach due to the strict campaign rule (all nodes in a campaign should be completely connected). The loose approach found 12 campaigns, one of which is not an actual campaign (false positive) and some predicted campaigns contain dissimilar messages due to long chains. The k-means clustering algorithm found only 5 campaigns. Overall, the cohesive and strict approaches outperformed the loose and cluster-based approaches. In practice, the ideal approach should return the same number of campaigns of the ground truth in order to reduce post-labeling time and to further analysis. In this perspective, the cohesive approach may be preferred over the strict approach.

4.3 Campaign Detection over Large Data

We next examine campaign extraction from the large Twitter data set, CD_{Large}. Can we detect coordinated campaigns in a large message graph with 1.5 million messages? What kind of campaigns can we find? Which graph technique is the most effective to find campaigns?

Message Graph Setup: Based on the best message graph construction approach identified in the previous section, we generated a message graph consisting of 1.5 million vertices (one vertex per message). Of these, 1.3 million vertices are singletons, representing messages without any correlated messages in the sample (and hence, not part of any campaign). Based on this sample, we find 199,057 vertices have at least one edge; in total, there are 1,027,015 edges in the message graph.

Identifying Loose Campaigns: Based on the message graph, we identify as *loose campaigns* all of the maximally connected components, which takes about 1 minute on a single machine (relying on a breadth-first search with time complexity $O(|E| + |V|)$. Figure 2 shows the distribution of the size of the candidate campaigns on a log-log scale. We

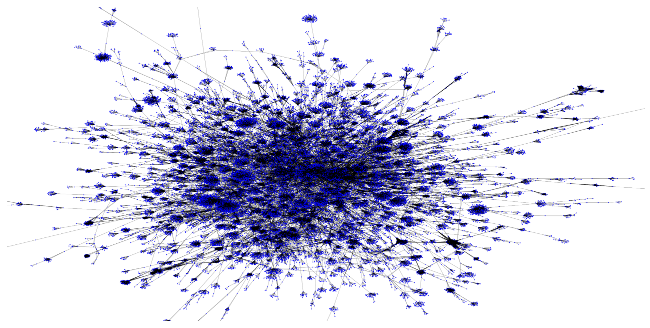

Figure 3: Candidate with 61,691 Vertices

see that the candidate campaign sizes approximately follows a power law, with most candidates consisting of 10 or fewer messages. A few candidates have more than 100 messages, and the largest candidate consists of 61,691 messages. On closer inspection, the largest candidate (as illustrated in Figure 3) is clearly composed of many locally dense subgraphs and long chains. Examining the messages in this large candidate, we find many disparate topics (e.g., spam messages, Justin Bieber retweets, quotes, Facebook photo template) and no strong candidate-wide theme, as we would expect in a coherent campaign.

Identifying Strict Campaigns: To refine these candidates, one approach suggested in Section 3 is *strict campaign detection*, in which we consider only maximal cliques as campaigns (in which all message nodes in a subgraph are connected to each other). While maximal clique detection may require exponential time and not be generalizable to all social message datasets, in this case we illustrate the maximal cliques found even though it required ∼7 days of computation time (which may be unacceptable for campaign detection in deployed systems). Considering the top-10 strict campaigns discovered in order of size: [559, 400, 400, 228, 228, 227, 227, 217, 217, 214], we find high overlap in the campaigns discovered. For example, the 2nd and 3rd strict campaigns (each of size 400) have 399 nodes in common. Similarly, the 4th, 5th, 6th, 7th, and 10th strict campaigns have over 200 nodes in common, suggesting that these five different strict campaigns in essence belong to a single coherent campaign (see Figure 4). This identification of multiple overlapping strict campaigns – due to noise, slight changes in message "talking points", or other artifacts of short messages – as well as the high cost of maximal clique detection suggests the cohesive campaign detection approach may be preferable.

Identifying Cohesive Campaigns: We next applied the *cohesive campaign extraction* approach to the set of candidate campaigns corresponding to maximal connected components. We assign λ to 0.95 and use the CSV tool [10] for an efficient implementation of computing each vertex m_{ix}'s $C(m_{ix})$ by mapping edges and vertices to a multidimensional space. Although computing $C(m_{ix})$ of all vertices takes $O(|V|^2 \log |V| 2^d)$ where d is a mapping dimension, the performance for real datasets is typically sub-quadratic. Like the candidate campaign sizes, the distribution of the size of the cohesive campaigns follow a power law. Since the cohesive campaign extraction approach can isolate dense subgraphs, we see that the large 61,691 message candidate has been broken into 609 sub-components. Compared to

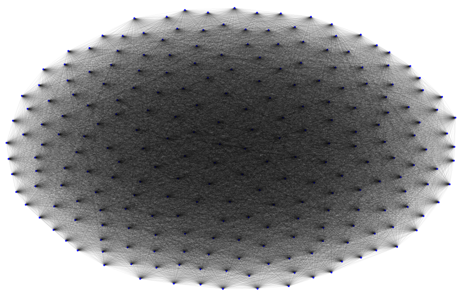

Figure 4: An Example Dense Subgraph Campaign: Strict Campaign Detection Identifies 5 Different Maximal Cliques; Cohesive Campaign Detection Identifies a Single Coherent Campaign

Table 2: Top-10 Largest Campaigns

Msgs	Users	Talking Points
560	34	Iron Man 2 spam
401	390	Facebook photo template
231	231	Support Breast Cancer Research (short link)
218	218	Formspring template
203	197	Chat template (w/ link)
166	166	Support Breast Cancer Research (full link)
165	154	Quote "send to anyone u don't regret meeting"
153	153	Justin Bieber Retweets
145	31	Twilight Movie spam
111	111	Quote "This October has 5 Fridays ..."

strict campaign detection, the cohesive campaign extraction approach required only 1/7 the computing time on single workstation.

Examining the top-10 campaigns (shown in Table 2) we see that the cohesive campaign detection approach overcomes the limitations of strict campaign detection by combining multiple related cliques into a single campaign (recall Figure 4). The biggest campaign contains 560 vertices and is a spam campaign. The "talking point" of this campaign is an Iron Man 2 promotion of the form: "#Monthly Iron Man 2 (Three-Disc Blu-ray/DVD Combo + Digital Copy) ... http://bit.ly/9L0aZU", though individual messages vary the exact wording and inserted link.

Based on a manual inspection of the identified campaigns, we categorize the campaigns into five categories:

- *Spam campaigns*: These campaigns typically post duplicate spam messages (changing @username with the same payload), or embed trending keywords; often with a URL linking to a malware website, phishing site or a product website. Example: "Want FREE VIP, 100 new followers instantly and 1,000 new followers next week? GO TO `http://alturl.com/bpby`".
- *Promotion Campaigns*: Users in these campaigns promote a website or product. Their intention is to expose it to other people. Example: "FREE SignUp!!! earn $450 Per Month Do NOTHING But Getting FREE Offers In The Mail!! `http://budurl.com/PPLSTNG`".
- *Template Campaigns*: These are automatically-generated messages typically posted by a third-party service. Example: "I'm having fun with @formspring. Create an account and follow me at `http://formspring.me/xnadjeaaa`".
- *News Campaigns*: Participants post recent headlines along with a URL. Example: "BBC News UK: Rwanda admitted to Commonwealth: Rwanda becomes the 54th member of the Commonwealth g.. `http://ad.vu/nujv`".
- *Celebrity Campaigns*: Users in these campaigns send messages to a celebrity or retweet a celebrity's tweet. Example: "@justinbieber please follow me i love youuu<3".

Some of these campaigns are organic and the natural outgrowth of social behavior, e.g., a group of Justin Bieber fans retweeting a message, or a group posting news articles of interest. On closer inspection, we observe that many of the less organic campaigns (e.g., spam and promotion campaigns) are driven by a higher ratio of messages to participants. For example in Table 2, the Iron Man 2 spam campaign consists of 560 messages posted by only 34 different participants. In contrast, the Justin Bieber retweet campaign consists of 153 messages posted by 153 different participants.

5. CONCLUSION AND FUTURE WORK

In this paper, we have investigated the problem of campaign detection in social media. We have proposed and evaluated an efficient content-driven graph-based framework for identifying and extracting campaigns from the massive scale of real-time social systems. Based on the success of the system we are extending this work to incorporate adaptive statistical machine learning approaches for isolating artificial campaigns from organic campaigns. Do we find that strategically organized campaigns engage in particular behaviors that make them clearly identifiable? Our results suggest that campaigns are not necessarily "invisible" to automated detection methods. We are also interested in exploring if campaigns are centralized around common types of users or are they embedded in diverse groups. How early in a campaign's lifecycle can a strategic campaign be detected with high confidence? Do we find a change in campaign membership and detection effectiveness after it reaches a critical mass? These challenges motivate our continuing research.

6. REFERENCES

[1] A. Z. Broder, S. C. Glassman, M. S. Manasse, and G. Zweig. Syntactic clustering of the web. *Comput. Netw. ISDN Syst.*, 29(8-13):1157–1166, 1997.

[2] A. Chowdhury, O. Frieder, D. Grossman, and M. C. McCabe. Collection statistics for fast duplicate document detection. *ACM Trans. Inf. Syst.*, 20(2):171–191, 2002.

[3] L. Films. (astro)turf wars. `www.astroturfwars.com`, 2011.

[4] H. Gao, J. Hu, C. Wilson, Z. Li, Y. Chen, and B. Y. Zhao. Detecting and characterizing social spam campaigns. In *Proceedings of the 10th annual conference on Internet measurement (IMC)*, 2010.

[5] N. J. Hurley, M. P. O'Mahony, and G. C. M. Silvestre. Attacking Recommender Systems: A Cost-Benefit Analysis. *Intelligent Systems, IEEE*, 22(3):64–68, 2007.

[6] S. K. Lam and J. Riedl. Shilling recommender systems for fun and profit. In *WWW*, 2004.

[7] E. P. Lim, V. A. Nguyen, N. Jindal, B. Liu, and H. W. Lauw. Detecting product review spammers using rating behaviors. In *CIKM*, 2010.

[8] M. Theobald, J. Siddharth, and A. Paepcke. Spotsigs: robust and efficient near duplicate detection in large web collections. In *SIGIR*, 2008.

[9] E. Tomita, A. Tanaka, and H. Takahashi. The worst-case time complexity for generating all maximal cliques and computational experiments. *Theor. Comput. Sci.*, 363:28–42, 2006.

[10] N. Wang, S. Parthasarathy, K.-L. Tan, and A. K. H. Tung. Csv: visualizing and mining cohesive subgraphs. In *SIGMOD*, 2008.

[11] B. Wu, S. Yang, H. Zhao, and B. Wang. A distributed algorithm to enumerate all maximal cliques in mapreduce. In *Proceedings of the Fourth International Conference on Frontier of Computer Science and Technology*, 2009.