A Temporal Topic Model For Noisy Mediums

Rob Churchill, Lisa Singh, and Christo Kirov

Georgetown University
(rjc111)(lisa.singh)@georgetown.edu, ckirov@gmail.com

Abstract. Social media and online news content are increasing rapidly. The goal of this work is to identify the topics associated with this content and understand the changing dynamics of these topics over time. We propose Topic Flow Model (TFM), a graph theoretic temporal topic model that identifies topics as they emerge, and tracks them through time as they persist, diminish, and re-emerge. TFM identifies topic words by capturing the changing relationship strength of words over time, and offers solutions for dealing with flood words, i.e., domain specific words that pollute topics. An extensive empirical analysis of TFM on Twitter data, newspaper articles, and synthetic data shows that the topic accuracy and SNR of meaningful topic words are better than the existing state.

1 Introduction

Enormous amounts of content are being generated on social media, e.g., over 500 million tweets [9] and over 420 million status updates on Facebook [12] are posted daily. One can use topic models to generate meaningful topics from these text streams. Unfortunately, current topic model algorithms have a number of weaknesses. First, the topics generated are often bogged down by noise words or impacted by noise-generating bots, e.g., approximately 15% of Twitter accounts are bots [16]. Second, domain specific corpora often have domain specific words that are not discriminative of topics, but appear across all different topics. For example, current topic models that generate topics about the extremist group ISIS will generate topics that all rank the term `ISIS` (or a variant) amongst the top terms for the topic. While obviously an important term, including it in specific topics about ISIS does not improve the quality of the topics and may make those topics look too similar. Finally, in some domains, the topics themselves change so quickly that current methods have difficulty keeping an up-to-date sketch of the active topics through time.

To begin addressing some of these limitations, we propose the Topic Flow Model (TFM) for monitoring the ebb and flow of topics in noisy text streams, e.g., Twitter, blogs, and online news. TFM identifies groups of content-rich topic words from a semantic graph by finding meaningful subgraphs of words that represent topics, and using relationship strength and frequency of words to determine their importance to different topics through time. This approach also effectively identifies and "drains" flood words, making the top ranked words for the topic more discriminative. **The contributions of this paper are as**

follows: (1) we introduce TFM, a temporal topic modeling algorithm that identifies topics as they emerge; (2) we introduce the concept of flood words and offer solutions to gracefully deal with them, and (3) we conduct an empirical analysis of TFM on Twitter data, newspaper data, and synthetic data, showing its effectiveness for identifying and monitoring changing topic dynamics under different conditions.

2 Related Work

Many topic modeling algorithms have been proposed in the last two decades. The most popular algorithms use probabilistic generative models. Latent Dirichelet Allocation (LDA) [5] and its many variants [15] [17] [3] [11] [2] [14] belong to this group of models, which rely on the assumption that documents are generated following a known distribution of terms. LDA finds the parameters of the topic/term distribution that maximizes the likelihood of the documents in the data set. These models have been successful for longer text documents written by a smaller number of authors that have a fixed vocabulary, i.e., new words (hashtags) are not being created continually. Another direction for research considers methods that have been used in the dimensionality reduction and clustering literature [18][13] [10]. For example, Yan et al. perform topic modeling by applying non-negative matrix factorization to a term correlation matrix [18], an approach that works better on short documents than generative models [18]. A third direction of research uses a semantic graph to identify topics [1][7]. Topic Segmentation [1], for instance, uses an undirected term co-occurence graph and the Louvain modularity algorithm [6] to find topics in a data set. Cataldi et al.'s Emerging Topic Detection (ETD) [7] employs a directed term correlation graph, and uses a double depth-first search to find emerging topics in a temporal topic modeling setting. Finally, some research focuses on post-processing the output of topic models to make them more meaningful [8] [4].

Our work is closest in spirit to Cataldi et al. [7] since we also employ a directed semantic graph and use that graph to identify topics. Our work differs from their work in the following ways; 1) we track all topics through time, as they emerge, persist, and diminish (Cataldi et al. focus on emerging topics), 2) because we are interested in topics through time, we do not regenerate topics at every time step, but instead use the topic knowledge from the previous time step to help determine the changes to existing topics and identify new ones, 3) we employ a new metric, the Energy-Nutrition ratio, to identify the most important terms in the semantic graph and avoid using flood words to build our topics, and 4) we employ a more efficient graph traversal procedure (a constant depth BFS) that identifies more accurate topic terms.

3 Background and Definitions

A document is an ordered list of terms $(w_1, ...w_k)$, where k is the length of the document. A *topic* T is a set of words believed to describe a theme or subject.

In our models, we will output M topics from a data set of D documents. The set of documents are partitioned into τ time periods. We therefore take as input D_t documents for each $t \in \tau$, and output M_t topics at time t.

Not all words in a document are useful for topic generation. Stop words are obvious words that are frequent, but content-poor. Noise words and spam both detract from topic quality, polluting the topics. Another type of word that pollutes topics is a *flood word*. A flood word is an important domain-specific word occurring so frequently that it is relevant to nearly every topic. For example, suppose we are interested in a data set about the recent presidential election. It would not be surprising if every topic contained Trump and/or Clinton in the top words related to the topic. While clearly relevant, we define these words to be flood words since they are domain-relevant and frequent, but do not add value to potential topics. Therefore, it is imperative to deal with flood words so that they are not the dominant words in every topic generated by our model. To help us keep track of the changing dynamics of words in topics, we now define nutrition, energy, and Energy-Nutrition Ratio.

The *nutrition* of a term is an indicator of how popular a term is in the document collection. More formally, $nutrition(w) = (1-c) + c * tf(w)/tf(w_i^*)$ where w_i^* is the most frequent term in document d_i, $tf(w)$ is the term-frequency of the input word w in d_i, and c is some constant between zero and one. $nutrition(w)_t$ is the sum of these nutritions over D_t in time period t. We then normalize nutrition by $|D_t|$ to account for change in data set size over time.

The *energy* of a term considers the change in nutrition of that term over time. More formally, $energy(w) = \Sigma_{i=1}^{s}(nutrition(w)_t^2 - nutrition(w)_{t-i}^2) \times \frac{1}{i}$ where s is the number of previous time periods before t.

Because energy is a sum of squared differences, it is biased toward higher nutrition terms. A high nutrition term that sees a small change over time intervals might still have a high energy compared to a low nutrition term that has a bigger change relative to its original nutrition. We account for this relative change with the Energy-Nutrition Ratio: $ENR(w) = \frac{energy(w)}{nutrition(w)}$ A term with high nutrition that sees a small change will see a relatively low change in its ENR, whereas a term with low nutrition that sees a large change will see a big change in ENR. We can compare a term's current ENR and previous ENR to decide whether the rate of growth is accelerating, constant, or decelerating. Because energy is a polynomial function of nutrition, it grows and shrinks faster than nutrition. This growth, or lack thereof, is captured in ENR.

Problem Statement: Formally, given τ time intervals, and D_t documents for each $t \in \tau$, find the set of topics M_t and flood words $flood_t$ for each t.

4 Topic Flow Model

4.1 TFM Overview

We now provide an overview of our proposed approach, Topic Flow Model (TFM). The high level algorithm can be found in Algorithm 1. The input to

Algorithm 1 Topic Flow Model (TFM)

1: **Input:** D_t for each $t \in (1, ...\tau)$
2: $\alpha, \beta, \gamma, \delta, \theta$
3: **Output:** M_t, $flood_t$, and L_t for each $t \in (1, ...\tau)$
4: **repeat**
5: $nutrition_t = compute\_nutrition(D_t)$
6: $energy_t = compute\_energy(\alpha, nutrition_t, nutrition_{t-1}, D_t)$
7: $ENR_t = compute\_ENR(nutrition_t, energy_t)$
8: $emerging = select\_emerging\_terms(\alpha, \beta, \gamma, nutrition_t, energy_t, ENR_t, D_t)$
9: $flood_t = identify\_flood\_words(\alpha, nutrition_t)$
10: $C = compute\_term\_correlations(D_t)$
11: $G = create\_term\_correlation\_graph(\delta, energy_t, C)$
12: $M_t = \{\}$
13: **for** $term \in emerging$ **do**
14: $new\_topic = discover\_topic(G, term, \theta)$
15: $M_t = M_t + new\_topic$
16: **end for**
17: **for** $T_i \in M_{t-1}$ **do**
18: $persistent\_topics = identify\_persistent\_topic(G, leaders(T_i), \theta)$
19: $M_t = M_t + persistent\_topics$
20: **end for**
21: $M_t = merge\_topics(M_t)$
22: L_t = identify_leaders(M_t)
23: **until** $t = \tau$
24: return M, $flood$, L

the algorithm is the set of documents for each time period and a set of tuning parameters that will be described later in this section. For each time interval, our algorithm begins by using nutrition, energy, and the ratio between the two (ENR) to identify emerging terms, terms that have become important in the current time period (line 8). It then creates a directed term-correlation graph (line 11) and identifies the topics from the previous time window that persist in the current time window (line 18). It does this using a double Breadth-First Search (BFS) on the graph. Once all topics have been identified, topics are compared and merged if sufficiently similar (line 21). Leader nodes are chosen for each topic based on their centrality scores within their topic (line 22). TFM outputs a set of topics, topic leaders, and flood words for each time window. The remainder of this section describes the main parts of the algorithm in more detail - identifying emerging terms, building and using the semantic graph, and determining and merging topics. Throughout this section, we will make use of the example presented in Figure 1. In the example, there are three time periods, each containing three documents. The semantic graph, document frequency of each word, and each word's energy are shown for each time period.

4.2 Identifying Important Terms

One of the keys to generating good topics is identifying important terms. We use nutrition, energy, and ENR to aid in this process. After computing these values for each term, we can select the terms that fit within the thresholds of each criterion. We define three separate tuning parameters, one for each value. Flood words, by definition, have the highest nutrition. To avoid adding them to our topics, we set an upper bound for nutrition (α). For energy, we do not want to

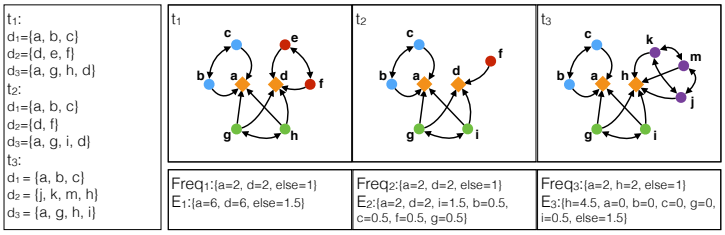

Fig. 1: Example of TFM graphs over time. Orange diamond nodes represent flood words. Topics are color-coded. Red, blue, and green topics emerge in t_1, and persist until t_2 or t_3. The Purple topic emerges in t_3.

consider any words with exceedingly low energy values, so we set a lower bound on energy (β). In figure 2, we show how these initial two thresholds cut swathes of terms from the list of potential emerging terms. Within the set of remaining terms with high energy and high nutrition, we set an ENR threshold (γ) to weed out terms whose growth is low compared to the previous time window.

For our running example, assume $s = 2$, $\alpha = 1$, $\beta = 0$, and $\gamma = 1$. In figure 1, we see that terms `a`, and `d` have the highest frequency in t_1 and t_2, and terms `a` and `h` have the highest frequency in t_3. Looking at nutrition, energy, and ENR, we see that `a`, `d`, and `h` have too high nutrition values and are identified as flood words in different time periods instead of qualifying as emerging terms. At time t_2, the nodes returning from time t_1 have lower energy levels than `i`, leading to i being the only emerging term identified in t_2. As we will show, emerging terms are important because we start traversing the graph for topics from nodes representing emerging terms.

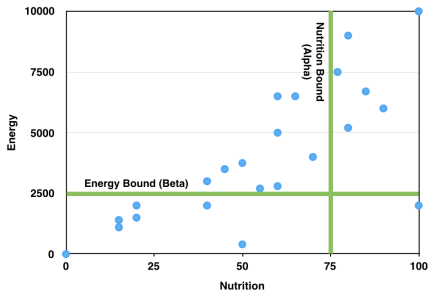

Fig. 2: The effect of α and β thresholds on term selection

4.3 Semantic Graph Construction

In order to determine topics, we construct a directed term-correlation graph. Any word that is not a stop word can be a node in the graph. For edges, we compute asymmetric term correlations [7], and then selectively add edges to our semantic graph based on these correlations. The term correlation $c^t_{k,z}$ of two terms k, z at time t is:

$$c^t_{k,z} = log(\frac{n_{k,z}/(n_k - n_{k,z})}{(n_z - n_{k,z})/(|D_t| - n_z - n_k + n_{k,z})}) \cdot |\frac{n_{k,z}}{n_k} - \frac{n_z - n_{k,z}}{|D_t| - n_k}| \quad (1)$$

where: $n_{k,z}$ is the number of documents in D_t that contain both k and z; n_z is the number of documents in D_t that contain z; n_k is the number of documents in D_t that contain k. The *term-correlation* of two terms is the correlation of the first term to the second term at time t, considering both co-occurrence and individual occurrence of each term. An edge (u, v) in the graph will have a weight equal to the correlation of u to v. In order for an edge to be added to the graph, its weight must be greater than the median term correlation plus δ times the standard deviation of correlations, where δ is a tuning parameter used to control the number of edges in the graph. A higher value of δ will result in a smaller, less connected graph, whereas a zero value will result in a maximally connected graph. Many flood words have high-correlation incoming edges, but because they are connected to so many other terms, their outgoing correlations are well below the median value, preventing these edges from being added to the graph.

In our example, we see the connections between nodes that occur together in a document. As we can see in time period t_1, node `a` has incoming edges with nodes `b` and `c`, because they co-occur in document d_1, and it also has incoming edges from nodes `g`, and `h`, which co-occur with node `a` in document d_3. Note that there are no outgoing edges from `a` or `d` because their correlations to other terms are below the threshold for adding an edge in our example.

4.4 Finding Topics

Using the graph G, we build a set of topics for our current time interval. We define two types of terms to focus on topic discovery: leader terms, and origin terms. A leader term is a term that represents a specific topic, chosen by centrality score. An origin term corresponds to a node from which we can start a topic search in G. As we will describe below, an origin term can be an emerging term or an existing topic leader. There are three main steps for finding topics: identifying persistent topics, discovering emerging topics, and merging similar topics.

Emerging Topics. Starting at an origin, we perform a breadth-first search (BFS) up to the depth limit θ to find other potential terms in the topic. We run a second breadth-first search from any node found during the forward pass, looking specifically for the origin. Since the graph is directed, we are not guaranteed to find a path back to the origin term from every node. If we do find a path during the backward pass, we assume the term is strongly connected to the origin, and include it in the associated topic. Flood words have exceptionally low correlations to other terms, and so even when included in the graph, it is unlikely that they will be included in a cycle from an origin node.

Persistent Topics. To identify topics that have persisted from the previous time period to the current window, we run our double BFS algorithm using the topic's existing leader as the origin, and compare the returned topic terms to the existing set. We compute a *topic distance*: $td_{t_1,t_2} = \frac{min(|t_1 \backslash t_2|, |t_2 \backslash t_1|)}{|t_1 \cap t_2|}$. A smaller distance implies that the topics are more similar. A distance of zero means that one topic is a subset of the other. If the 'new' topic is sufficiently similar to its old self, we keep only the new version. Returning to our example, suppose $\theta = 1$.

From each emerging term, we do a search to a maximum depth of one, where the depth at the root is 0. The resulting topics are $\{b, c\}$, $\{b, c\}$, $\{e, f\}$, $\{e, f\}$, $\{g, h\}$, and $\{g, h\}$, with leader sets $\{b\}$, $\{f\}$, and $\{g\}$, respectively. These are the topics we want to see since they do not include flood words. Duplicates occur because the emerging terms had directed edges in both directions.

Merging Similar Topics. Once we have found a set of topics for our time period, we must decide whether any are similar enough to merge into one. We compare the shared terms of each pair of topics in the set using the distance defined above. If the two topics share enough terms, we merge them by taking the union of their terms, and choosing new leaders. Returning to our example, we find duplicates of each topic because there are multiple emerging terms identified for each emerging topic. Using our method of merging similar topics, we will compare the topic membership of different topics and merge the duplicate topics. Our final result in t_1 is: $\{b, c\}$, $\{e, f\}$, $\{g, h\}$. In t_2 and t_3, topics $\{b, c\}$ and $\{g, h\}$ persist. In t_3, topic $\{j, k, m\}$ emerges.

5 Empirical Evaluation

This section presents an empirical analysis of TFM and other state of the art methods on a Twitter data set, a newspaper data set, and synthetic data sets.

5.1 Data Sets

For our synthetic data sets, we generate topics from a set of words, assigning each a normally distributed random probability of appearing in a document with that topic. By assigning different probabilities to different words, we are simulating the nature of tweets containing few content-rich, important words mixed in with many less useful words. [1] The synthetic data sets we generated each contain 200 vocabulary items, 500 documents, and seven topics over seven time periods. Our synthetic data sets contain varying levels of flood words, 0%, 1%, 5%, 10%, and 15% of the total number of words in the vocabulary.

The Twitter data set is a daily random sample of 5,000 tweets about Donald Trump from August and September 2016. We have a total of 280,000 tweets split into weekly time periods. The newspaper data set consists of news articles about Trump and Clinton from the Washington Post. The newspaper data set contains 14,269 articles and spans the same time frame as the Twitter data set.

For our data sets, we evaluate the accuracy and quality of topics using recall and Signal to Noise Ratio (SNR). The SNR is the ratio of terms in the approximated topic that belong to the true topic to the terms in the approximated topic that do not belong in the true topic. Let T_{noise} be the set of noise words in topic T, and T_{signal} be the set of signal words in topic T, then $SNR = \frac{T_{signal}}{T_{noise}}$.

[1] Another way to simulate this is to sample from a Zipfian distribution. Our data generator allows for distribution changes. For these experiments, we create a mixture that is noisier and harder to generate topics from than a Zipfian sample.

5.2 Synthetic Data Evaluation

Using our synthetic data sets, we evaluate four methods: TFM, Cataldi et al. [7], LDA [5], and Topic Segmentation (TS) [1]. For the static topic model algorithms, we rerun them in each time period to generate topics. Our settings for TFM are: $\alpha = 4, \beta = 6, \gamma = 1, \delta = 1.5, \theta = 2$. For both LDA and TS, we need to specify the number of topics. We show the results from the best performing number of topics – LDA=7 and TS=10. The results are shown in table 1. The first column shows the fraction of flood words in the data set. In general, TFM performs significantly better across all three metrics at all the different fractions of flood words. It is also interesting to point out that its precision remains high even when the fraction of flood words in the data set is high. This is because it has been designed to avoid generating topics around flood words. In contrast, LDA performs best in terms of recall when the fraction of flood words is low or nonexistent, and best in terms of precision at 5%. The other three methods perform best when the fraction of flood words is set to 5%.

Table 1: Evaluation of synthetic data sets varying the % of flood words

Flood Word %	Metric	LDA	Cataldi	TFM	TS
0%	Precision	0.11	0.00	**0.67**	0.04
	Recall	0.37	0.00	**0.52**	0.20
	SNR	0.95	0.05	**4.10**	0.39
1%	Precision	0.09	0.33	**0.80**	0.00
	Recall	0.30	0.33	**0.37**	0.00
	SNR	1.19	0.51	**5.47**	0.21
5%	Precision	0.23	0.78	**1.00**	0.48
	Recall	0.25	0.63	**0.83**	0.62
	SNR	1.66	0.05	**5.52**	1.38
10%	Precision	0.11	0.00	**0.60**	0.00
	Recall	0.27	0.00	**0.40**	0.00
	SNR	0.50	0.05	**2.48**	0.20
15%	Precision	0.03	0.10	**0.54**	0.00
	Recall	0.10	0.05	**0.42**	0.00
	SNR	0.48	0.22	**2.26**	0.12

5.3 Twitter & Newspaper Evaluation

Researchers at Gallup worked with our research team to semi-manually create a set of popular topics for the presidential election campaign in 2016. For each week we have topics that persist, diminish and emerge. Because that initial topic set was generated without considering tweets from Twitter, we augmented the Gallup topics with appropriate hashtags and other topic words used on Twitter. The average number of topics being discussed each week is 5.5.

Accuracy & SNR For this empirical evaluation, we compare TFM, Cataldi et al. (Cataldi) [7], LDA [5], HDP [15], Topic Segmentation (TS) [1], Topics over Time (ToT) [17], and Naïve Graph Properties on the Trump Twitter data set. The Naïve Graph Properties method attempted to discover topics using graph invariants, including degree, betweenness centrality, and eigencentrality. We show only the best graph invariant results in this analysis. We set the number of topics for LDA to 12 and 24 and for TS to 10 and 20. Using these parameter settings for LDA and TS led to fewer noise words than other settings.

We present our findings in two forms: the number of ground truth topics identified in each time interval in figure 3(a) (x-axis = time period, y-axis = number of ground truth topics identified), and the average signal to noise ratio of identified topics in figure 3(b) (x-axis = time period, y-axis = SNR). In order

for a ground truth topic to be considered accurately identified by a model, the model must output a topic with an SNR of at least 0.5 in reference to that ground truth topic. We see that the best performing algorithms are TFM and Cataldi, followed by different variants of LDA. LDA does identify a significant number of topics. TS identified one ground truth topic at time zero, but failed to identify any others, while HDP and Naïve Graph Properties identified no ground truth topics. TS's failure to identify more topics stemmed from the large number of noise words that it picked up in comparison to the number of ground truth words when generating topics. HDP's failure to identify ground truth topics seems to be a result of overfitting of the topics. In each of its topic sets, every topic contained almost the exact same terms. Notice that for both TFM and Cataldi, a warm-up period is needed - neither perform well in the first time period. In terms of SNR, TFM has the highest SNR of all the methods. Cataldi and LDA are comparable to TFM in time periods in which they find the same number of topics.

We tested the best performers, TFM, Cataldi, and LDA on the Washington Post data set, using the same ground truth topics used for the Trump Twitter data set since we were interested in topics related to Trump. For TFM, we used the following parameter values: $\alpha = 10, \beta = 2, \gamma = 2, \delta = 1.5, \theta = 2$. We present our accuracy results in figure 3(c). TFM outperformed Cataldi and LDA, identifying nearly every topic in every time interval except for the first. In figure 3(d), we see that it has a high SNR across time windows. Cataldi identified one or more topics in every time window except for the first, while both LDA options find one topic in the third time window. The SNR of TFM was consistent across all time windows. In the last time window, Cataldi had a higher average SNR.

Finally, due to space limitations, we cannot present a sensitively analysis for all the different parameters. However, we pause to mention that small variations of α, β, and γ do not impact the identified important terms significantly. Large differences, on the other hand, do (as we will show in the next subsection). In terms of the semantic graph, keeping θ low (around 2) reduces noise in the discovered topics and improves efficiency. δ impacts the number of edges in G. We have found that while the number of edges decreases significantly when δ increases, the accuracy of selected topics does not decrease for our data sets.

5.4 TFM Flood Words Evaluation

We now present two cases that demonstrate the graceful handling of flood words.

Flood Word Retainment. The α parameter controls the removal of flood words prior to generating G. In this experiment, we consider the case of setting $\alpha = \infty$, removing no flood words. When doing this on the Trump tweets data set, we find that TFM only identifies two topics as emerging, `hillary` and `#debatenight`. In both cases, no other words are identified as significant terms in those topics. This suggests that the two terms were so much more frequent than any other term that no other term could be reasonably assumed to be emerging relative to these terms. When α is smaller, these two terms are correctly labeled as flood words and over two dozen unique topics are identified. The debate topic is still identified even though the associated hashtag is a flood word. We pause

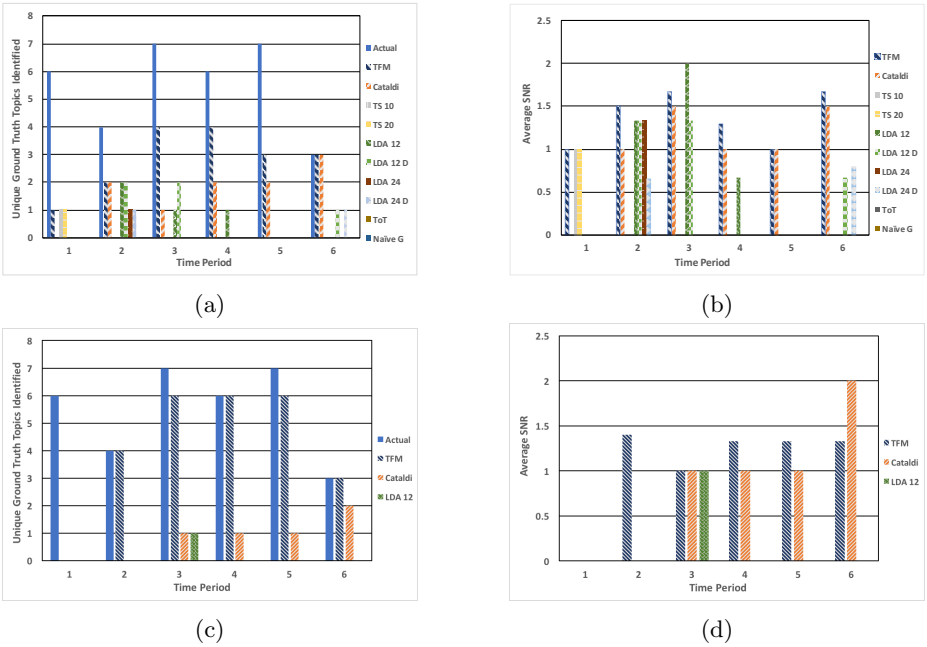

Fig. 3: a) Trump Twitter: Ground Truth Topics Identified by each method b) Trump Twitter: SNR of each method c) Newspaper: Ground Truth Topics Identified by each method d) Newspaper: SNR of each method

to point out that it is typical for some flood words to be in the graph, but the most extreme to not be. For the Trump tweet data set, the degree of the average flood word in G is 935 and its average correlation is 0.29, while the degree for the average topic word is 24 and the average correlation is 1.18. This highlights the importance of not focusing topic discovery on flood words.

TFM vs. Cataldi Emerging Terms. For this case study, we compare the emerging terms of TFM and Cataldi on the Trump tweet data set for two different weeks, September 4th and September 11th. Figure 4 shows the TFM emerging terms, the TFM flood words, and the Cataldi emerging terms as a Venn diagram. This figure highlights a few interesting findings. First, the emerging terms of Cataldi are all flood words returned by TFM, except for one word, `tax`. Second, most of the flood words identified by TFM are very general domain words, e.g. `president`, `campaign`, `policy`, `people`. These words are content-poor within the domain because they cross a large number of more meaningful, content-rich topics. There are a few terms that are more content-rich, `Putin` being the most notable. In these cases, either the flood word was an emerging term in the previous time window, so the topic has already been discovered, or the term rose so rapidly that its presence in G would cause it to be the center of a topic that is really a cluster of smaller topics merged into one.

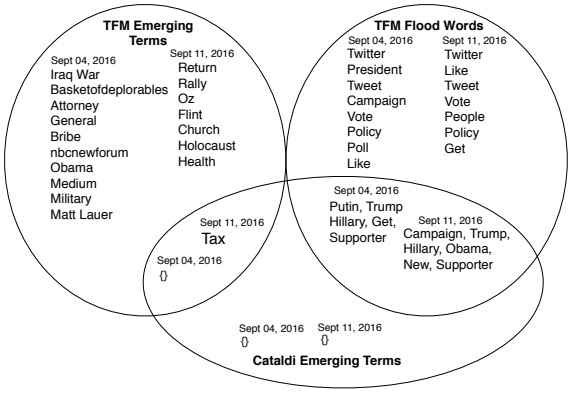

Fig. 4: Distribution of Terms in Trump Data Set, 09/04/2016 and 09/11/2016

5.5 Execution Time Comparison: TFM & Cataldi et al.

In this experiment, we compare execution times of TFM & Cataldi on graphs of similar size. These experiments were run on an Ubuntu 16.04 machine with a 4.00 GHz processor and 16GB of memory. Table 2 shows the results. We set the edge inclusion parameters to levels such that the number of edges in the respective graphs are similar. We tested algorithms on a graph size of 350,000 edges, 190,000 edges, and 110,000 edges. The specific numbers were chosen because the former was the approximate number of edges seen using TFM's optimal parameter settings, the latter was the optimal for Cataldi, and the middle gave reasonable, albeit worse, results for both models, with respect to topic quality. Table 2 shows that TFM's execution time is significantly smaller as the number of edges increases. This occurs because the DFS used in Cataldi requires traversal of a larger number of paths than the constant depth search used by TFM.

Table 2: Execution Time Comparison

# Edges	TFM	Cataldi
350,000	43s	5 days
190,000	39s	2 hrs
110,000	37s	28s

6 Conclusion

In this paper, we introduce the Topic Flow Model, and demonstrate its abilities to not only identify emerging topics, but to track those topics through time. We introduce the notion of flood words, and demonstrate how their graceful handling is integral to identifying concise topics in noisy data such as tweets, and even in less noisy data. We compare Topic Flow Model to state of the art topic modeling algorithms and show that it identifies topics more accurately with less noise than other methods. In future work, we plan to design extensions of this model for other types of text, understand the impact of pre-processing on topic model algorithms, and develop methods for reducing noise in topics.

Acknowledgements. This work was supported by the Massive Data Institute (MDI) at Georgetown University.

References

1. de Arruda, H.F., da Fontoura Costa, L., Amancio, D.R.: Topic segmentation via community detection in complex networks. CoRR (2015), http://arxiv.org/abs/1512.01384
2. Bhadury, A., Chen, J., Zhu, J., Liu, S.: Scaling up dynamic topic models. WWW, SIAM (2016)
3. Blei, D.M., Lafferty, J.D.: Dynamic topic models. ICML, IEEE (2006)
4. Blei, D.M., Lafferty, J.D.: Visualizing topics with multi-word expressions. ArXiv e-prints (2009)
5. Blei, D.M., Ng, A.Y., Jordan, M.I.: Latent dirichlet allocation. Journal of Machine Learning Research **3**, 993–1022 (Mar 2003)
6. Blondel, V.D., loup Guillaume, J., Lambiotte, R., Lefebvre, E.: Fast unfolding of communities in large networks. Journal of Statistical Mechanics (10), P10008 (2008)
7. Cataldi, M., Di Caro, L., Schifanella, C.: Emerging topic detection on twitter based on temporal and social terms evaluation. MDM-KDD, ACM (2010)
8. Chang, J., Boyd-Graber, J., Wang, C., Gerrish, S., Blei, D.M.: Reading tea leaves: How humans interpret topic models. NIPS, AAAI (2009)
9. InternetLiveStats: Twitter usage statistics. http://www.internetlivestats.com/twitter-statistics/, accessed: 2017-05-05
10. Kasiviswanathan, S.P., Melville, P., Banerjee, A., Sindhwani, V.: Emerging topic detection using dictionary learning. CIKM, ACM (2011)
11. Lafferty, J.D., Blei, D.M.: Correlated topic models. In: NIPS. pp. 147–154. AAAI (2006)
12. Noyes, D.: The top 20 valuable facebook statistics - updated may 2017. https://zephoria.com/top-15-valuable-facebook-statistics/, accessed: 2017-05-05
13. Shahnaz, F., Berry, M.W., Pauca, V., Plemmons, R.J.: Document clustering using nonnegative matrix factorization. Inf. Process. Manage. pp. 373–386 (Mar 2006)
14. Sleeman, J., Halem, M., Finin, T., Cane, M., et al.: Modeling the evolution of climate change assessment research using dynamic topic models and cross-domain divergence maps. Symposium on AI for Social Good, AAAI (2016)
15. Teh, Y.W., Jordan, M.I., Beal, M.J., Blei, D.M.: Hierarchical dirichlet processes. Journal of the American Statistical Association **101**(476), 1566–1581 (2006)
16. Varol, O., Ferrara, E., Davis, C.A., Menczer, F., Flammini, A.: Online human-bot interactions: Detection, estimation, and characterization. ICWSM, AAAI (2017)
17. Wang, X., McCallum, A.: Topics over time: A non-markov continuous-time model of topical trends. KDD, ACM (2006)
18. Yan, X., Guo, J., Liu, S., Cheng, X., Wang, Y.: Learning topics in short texts by non-negative matrix factorization on term correlation matrix. SDM, SIAM (2013)