Percolation-Based Topic Modeling for Tweets

Rob Churchill
Georgetown University

Lisa Singh
Georgetown University

ABSTRACT

This paper investigates topic modeling within a noisy domain. The goal is to generate topics that maximize topic coherence while introducing only a small amount of noise. The problem is motivated by the practical setting of short, noisy tweets, where it is important to generate topics containing a larger number of content words than noise words. For the most general version of this problem, we propose a new method, λ-CLIQ. It is a simple variant of the k-clique percolation algorithm that employs for quasi-cliques during graph decomposition and percolation based on λ, a graph property variant. While the topics generated using our base algorithm are highly coherent, they are often contain too few words. To increase topic size, we add a post processing step that augments identified topic words using locally trained embeddings. We show that both λ-CLIQ and λ-CLIQ$^+$ outperform the state of the art in terms of topic coherence on three distinct Twitter data sets.

CCS CONCEPTS

• **Computing methodologies → Topic modeling**.

KEYWORDS

topic modeling, social media, graph model, clique percolation

ACM Reference Format:
Rob Churchill and Lisa Singh. 2020. Percolation-Based Topic Modeling for Tweets. In *Proceedings of KDD Conference (WISDOM'20).* ACM, New York, NY, USA, 8 pages.

1 INTRODUCTION

Since their inception in the early 2000s, topic models have been an important tool for understanding textual content. Originally, they were used to construct and analyze topics within longer documents such as newspaper articles, books, research papers, and journals. Nearly two decades later, the nature of the documents that we would like to employ topic models on has drastically changed, but the nature and underlying assumptions of the topic models have not. Instead of hundreds of well-edited documents each consisting of hundreds of words, we have thousands/millions of documents each consisting of dozens of words.

These short documents are generated on social media sites such as Twitter and Facebook, where millions of posts are shared every day. It was estimated that in 2019 over 500 million tweets were sent per day [12], and over 420 million Facebook status updates were posted daily [21]. Unfortunately, because these documents are not long and rich in organized content, current topic models have difficulty determining coherent topics. The situation is further exacerbated when we focus on a specific domain of social media posts, such as politics, parenting, or sports. Many of the words that define the domain itself are so prevalent that it makes uncovering the individual topics very hard. For instance, in a domain regarding a presidential election, words like 'election,' 'politician,' and 'issue' appear very frequently, but rarely provide additional context to specific topics. These are examples of *flood words* or domain specific noise words [6].

State of the art methods are unable to generate topics with more content rich words than noise words in social media texts. We believe that this is due to the small document length, the varying topic sizes and frequencies, the amount of noise present in such documents, and the prevalence of generic domain words. As our empirical evaluation will demonstrate, the distribution of noise at all word frequencies makes it difficult for probabilistic models to generate coherent (interpretable) topics. We believe this results because state of the art generative models tend to add the most frequent domain words to a large number of topics, leading to generic topics that do not contain sufficient context. In general, probabilistic models tend to perform poorly on Twitter data due to the failure of the inherent assumption that the distribution of content words can be approximated by sampling the relative frequencies of words. Our approach will instead incorporate words and phrases based on a co-frequency graph containing words and phrases that may only be moderately frequent.

At a high level, our goal is to generate topics that maximize topic coherence while introducing only a small amount of noise. We propose a new method, λ-CLIQ , that is a simple variant of the k-clique percolation algorithm [9] typically used for clustering tasks. λ-CLIQ decomposes a co-frequency graph until it contains only small quasi-cliques and then builds (percolates) topics by combining these quasi-cliques if they have high levels of connectivity. While the topics generated using our base algorithm are highly coherent, they often contain too few words and phrases. To increase topic size, we add a post processing step that augments identified topics with words using a locally trained embedding space (λ-CLIQ$^+$). We find that our approach outperforms state of the art models in terms of topic coherence and noise level on three large, domain specific Twitter data sets.

The contributions of this paper are as follows: (1) we propose a new topic modeling algorithm (λ-CLIQ) that uses a co-frequency graph to identify topics by removing weighted edges until only quasi-clique subgraphs remain and recombining subgraphs based on different percolation rules; (2) we propose an extended topic modeling algorithm (λ-CLIQ$^+$) that incorporates a language model to improve topic quality; (3) we compare our proposed algorithm to state of the art methods and show that it has significantly more coherent topics than other methods; and (4) we make our model and other models used in our experiments, along with our evaluation

WISDOM'20, August 2020, San Diego, USA

metrics, available in a Python package intended for further topic modeling research.

The paper is organized as follows. Section 2 provides a background of topic models and their varying approaches. Section 3 describes the components of our model, and how they fit together to produce a topic model. Section 4 shows the results of experiments comparing our model to baseline models on multiple evaluation metrics. Finally, we present conclusions in section 5.

2 RELATED LITERATURE

Many topic models have been proposed over the last two decades, but the majority can be organized into three classes of models - generative models, dimensionality reduction models, and semantic graph models.

The first class consists of generative models, which rely on the key assumption that documents are generated according to a known distribution of terms. The most prevalent of the generative class is Latent Dirichelet Allocation (LDA) [3], which inspires the vast majority of other generative models. LDA finds the parameters of the topic/term distribution that maximizes the likelihood of documents in the data set over k topics. Among its direct descendants are Hierarchical Dirichelet Process (HDP) [28], Dynamic Topic Models (DTM) [2], Correlated Topic Models (CTM) [14], and Twitter-LDA [35]. Each of these iterations attempts to leverage the key assumption in a different manner to improve upon LDA. HDP [28] removes k from the model's parameters, and attempts to maximize the likelihood of the number of topics in the data set, as well as the likelihood of documents in the data set. DTM [2] attempts to predict topics in future time periods using topics in the current time period. Topics over Time [31] attempts to normalize time in order to perform temporal topic modeling. CTM [14] acknowledges that there likely exists some correlation between topics, and attempts to account for that correlation to produce more coherent topics than LDA. These models have been shown to be successful for data sets consisting of longer text documents written by a small number of authors. Twitter-LDA [35] merges all tweets of a user into a single document in order to get bigger documents, and then runs LDA on those combined documents. While useful when studying user level content, our focus is on data streams associated with hashtags and/or keywords. In that setting, most people only share a small number of tweets.

A new direction of research looks at incorporating more sophisticated NLP techniques and mixture models into generative models. Embedding-based Topic Model (ETM) [23] uses word embeddings to aggregate short texts into long pseudo-texts, and then infers topics from the pseudo-texts. Yan et al. perform topic modeling on pairs of terms with high co-occurrence in their model Biterm Topics [32]. Thompson and Mimno employ probabilistic subsampling in multi-source data sets to avoid source-specific bias in topics [29]. Their approach is aimed at producing a better starting vocabulary to feed into LDA. An approach called lda2vec attempts to mix LDA and word embeddings to produce better topics [18]. A more recent approach uses LDA to get topic embeddings, and uses these embeddings along with pre-trained word embeddings to find topics in short texts [30].

Another type of generative model employs the Dirichlet Multinomial Mixture (DMM), which differs from LDA in that it assumes that each document only has one topic [20]. The DMM has been a key building block to many topic models that attempt to better model data sets containing short documents [16, 19, 24, 34]. SATM [24] assumes that short texts are all part of crumbling longer documents, and attempts to create these longer documents by aggregating short texts. SATM then runs LDA on the larger documents to get the overarching topics, and uses DMM to infer the specific topic of each of the short texts. GSDMM [34] attempts to cluster documents into k topics in a round-robin approach, allowing documents to decide which topic to join by which other documents are most similar to it. Qiang et al. [22] and Li et al. [16] (GPU-DMM) employ approaches using word embeddings to cluster short texts into longer pseudo-texts that are then put through a different generative model. The former uses LDA as its building block, while the latter uses DMM.

The second static class of topic models is that of dimensionality reduction and clustering, focusing on non-negative matrix factorization (NMF). NMF uses matrix multiplication to reduce the dimensionality of the document-word matrix, resulting in an approximation of a topic-word matrix that can be interpreted as a topic set related to the original documents. Shahnaz et al. demonstrated NMF's capabilities as a topic model [27]. Kasiviswanathan et al. combine NMF with online dictionary learning techniques to detect emerging topics in a temporal setting [13]. Yan et al. attempt to detect topics in short texts using NMF in conjunction with a term correlation matrix, as opposed to a term-document matrix, due to the sparsity of short text data [33].

The third class of topic models uses a semantic graph to identify topics. Topic Segmentation (TS) [8], for instance, uses an undirected term co-occurrence graph and the Louvain modularity algorithm [4] to discover topics in a data set. Cataldi et al.'s Emerging Topic Detection [5] leverages a directed term correlation graph paired with a double depth-first search to discover emerging topics in a temporal topic model on social media data. Churchill et al. use a directed term correlation graph in their Topic Flow Model (TFM), but augment it with new metrics and a search algorithm to identify topics as they emerge and diminish [6]. Unlike the first two static classes, the graph-based models do not all work on the same underlying assumptions or algorithms. Their performances are far less correlated than those of the generative and dimensionality reduction classes.

In our evaluation, we will include representative topic models from each class of models. We will show that while the generative models are good at identifying meaningful words/n-grams, they do not put these words/n-grams together into coherent topics. Instead, they spread them across multiple topics because of the prevalence of high frequency noise words and high frequency flood words.

3 APPROACH

In this section, we begin by presenting a high level description of the algorithm. We then go through the details, providing insight throughout.

The λ-CLIQ algorithm, unlike most state-of-the-art topic models, does not immediately begin generating topics as it iterates through documents. Instead, λ-CLIQ attempts to remove noise before generating topics.

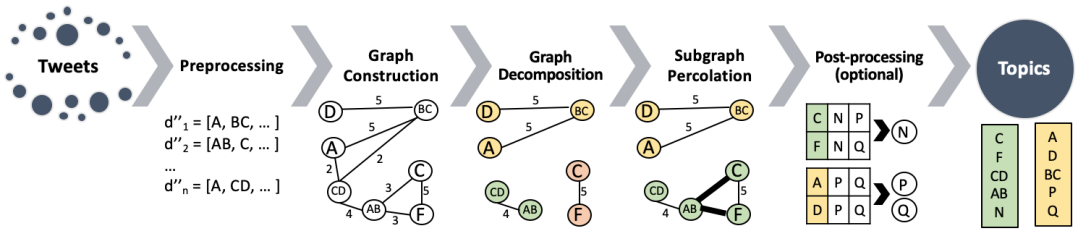

Figure 1: Topic Percolation Algorithm Methodology

Figure 1 shows the overall methodology. Conceptually, the core components of the algorithm can be divided into four subparts:

- Preprocessing - Tokenizing the text, where tokens are words or NGrams (phrases)
- Graph Construction - Creating a weighted co-frequency graph G containing phrase tokens and word tokens that co-occur with NGram tokens.
- Graph Decomposition - Decomposing the graph in rounds based on edge weight in order to find small, well connected quasi-cliques that can be viewed as topic kernels.
- Subgraph Percolation - Adding back edges between topic kernels to build up (percolate) larger communities of quasi-cliques by maintaining λ levels of connectivity among topic kernels, where λ represents the threshold for a graph property or metric.

Because we optimize for topic coherence, our algorithm is very selective. To increase the coverage of topics, we also propose an extended version of the algorithm, λ-CLIQ$^+$. It includes a step that uses a pre-computed domain specific language model to augment topics by adding words that are probabilistically similar to those in each topic (Post-processing).

While the overall strategy we propose mirrors that of the k-clique percolation algorithm, we do not identify full cliques, but rather well connected subgraphs. This relaxation is important in sparse domains because there are very few full cliques that occur regularly.

The high level algorithm is presented in Algorithm 1. The inputs are the database (D), a minimum frequency for an n-gram to appear in the co-frequency graph G (min_freq), the increment during graph decomposition (τ), and the minimum threshold for percolation (λ). The output is the set of topics (T). The remainder of this section describes the components of the algorithm and variants that improve the topic quality.

Algorithm 1 λ-CLIQ

1: **INPUT:** D, min_freq, λ, τ
2: **OUTPUT:** T
3: $T = \{\}$
4: D' = preprocess(D)
5: P = compute_freq_phrases(D', min_freq)
6: D'' = replace_words(P)
7: G = construct_cofreq_graph(D'')
8: $m = 1$
9: **while** $|E| > 0$ **do**
10: G = decompose_graph(G, m)
11: tf = remove_topic_fragments(G)
12: $T = T \cup tf$
13: $m = m + \tau$
14: **end while**
15: T = percolate_topics(T, λ)
16: T = remove_small_topics(T))
17: **Return** T

3.1 Preprocessing

The algorithm begins by preprocessing the data D to extract tokens, i.e. words and NGrams (line 4). D' is the preprocessed, tokenized data. We then identify the frequent NGrams P (these are typically phrases) (line 5). They will be a core component of the cofreqency graph (line 5). Finally, for each pre-processed tweet in D', we replace relevant word tokens with the corresponding frequent NGram token to create D'' (line 6). In this way, we give priority to NGram tokens and remove the redundancy of maintaining both the individual words and the NGram in a post. For example, suppose two of the word tokens are `cat` and `dog` and that `cat_dog` is a frequent NGram. Then the *replace_word()* method replaces adjacent tokens `cat` and `dog` with the token `cat_dog`.

3.2 Graph Construction

In noisy environments, we hypothesize that identifying content-rich topics requires us to identify content-rich phrases, or n-grams that appear with regularity. We accomplish this by constructing a graph that contains phrases appearing regularly in the tweet collection and words that appear regularly with those phrases.

Given a document collection D, we will model it as an edge-weighted graph. We represent an edge-weighted graph as a pair (G, w), where $G = (V, E)$ is a graph and $w : E \rightarrow \mathbb{Z}$ is a weight function. V is the set of nodes in G, and E is the set of undirected edges in G. In our graph G, nodes represent phrase or word tokens, edges exist between two phrase tokens or a phrase token and a word token that appear in the same document, and $w(e_{ij})$, the weight on edge e_{ij}, is the number of documents containing both tokens v_i and v_j. G does not contain edges between word tokens as their frequency of co-occurrence overwhelms that of the edges between words and phrases, reducing the importance of phrases in G. Also, since many words appear together by chance on social media, focusing on phrases that occur regularly reduces some of the more 'random' connections that may appear if we have edges between nodes representing words.

Once we identify phrases P that occur at least min_freq times (line 6), the graph construction (line 7) proceeds as follows. A node is created for each phrase in P and each unigram in D'' that occurs with one or more phrases in P. Edges are added between phrases that co-occur and phrases and words that co-occur. The weight of each edge is the frequency of co-occurrence.

3.3 Graph Decomposition Strategies

Our model attempts to break the chains between content and noise words that negatively affect the performance of other topic models, while leaving intact communities of content words. The edges in our graph are weighted with the co-frequency of their respective vertices. The basis of our decomposition strategy is to remove edges in rounds, where all edges of a particular weight m are removed during a single round. In round one, all the edges of weight $m = 1$ are removed. Once the edges are removed, new subgraphs may emerge. Each subgraph or topic fragment (tf) is considered a *topic kernel*, removed from G, and set aside (added to T) until the percolation phase. We continue with new rounds, increasing m by τ until no edges remain (lines 8 – 14).

Our general strategy is based on the K-Clique Percolation algorithm as described in [9]. The main difference is that we relax the clique requirement by identifying *quasi-cliques* (as opposed to cliques) that disconnect from the giant component during edge removal. Because these types of phrase graphs are sparse, full cliques are too restrictive a constraint.

3.4 Graph Percolation Strategies

Given our set of topic kernels $T = tf_1 \dots tf_q$, where q is the number of topic kernels identified during the graph decomposition phase, we want to combine related topic kernels into single topics that are more coherent than the original topic fragments (line 15).

When combining topic kernels, we consider different graph properties or metrics that are possible indicators of high topic cohesion, i.e. high subgraph connectivity. One possible example graph property is subgraph density. We define λ to be the minimum threshold required in order to be a reasonably connected subgraph. Multiple topic kernels can be combined into one topic if the subgraph density of the union is greater than or equal to the subgraph density of the larger topic kernel.

A metric that is a better indicator of cohesion is Silhouette score. Silhouette score is used to determine the quality of a clustering algorithm. It measures how far apart each cluster is from its closest neighbor. We can adapt the silhouette score for our model's coherence metric. We change the silhouette score slightly to account for our closeness measure, cofrequency. We define $P(x, y)$ of two tokens x and y to be the probability that x and y appear in a document together. Instead of using minimum average distance, we use the maximum average co-frequency of a point with the points in each of the other clusters. So for a token x in topic fragment tf_i, let

$$a(x) = \frac{1}{|tf_i| - 1} \sum_{y \in tf_i, x \neq y} P(x, y) \tag{1}$$

To measure the distance to the closest topic, let

$$b(x) = \max_{j \neq i} \frac{1}{|tf_j|} \sum_{y \in tf_j, x \neq y} P(x, y) \tag{2}$$

Using $a(x)$ and $b(x)$, we can define the silhouette score of token x to be:

$$s(x) = \frac{a(x) - b(x)}{\max a(x), b(x)},\ if\ |tf_i| > 1,\ 0\ if\ |tf_i| = 1 \tag{3}$$

From equation 3.4, we see that the silhouette score is in the range $[-1, 1]$. Higher scores indicate well-clustered, coherent topics, while lower scores indicate poorly clustered topics.

If $\lambda = 0$, we remove all topics with negative silhouette scores in order to keep as much noise out of the merging process as possible. We use a greedy algorithm to combine topics only if the Silhouette score of the larger topic would be improved by merging with the smaller topic.

3.5 Filtering Topics after Percolation

After graph percolation, it is possible for small kernels to persist, having merged with no neighboring kernels. In order to reduce noise, and promote a more coherent topic set, kernels of size two (2-cliques) are removed from the topic set (line 16).

3.6 Optional Post-processing: Augmenting Topics with Domain Language Model

In order to build larger topics that maintain high coherence, we train a Word2Vec embedding model [17]. We then use our locally trained embeddings to add words to each topic after the topics have been found in the graph. We do so by computing the top n closest words to each topic word in the embedding space, and adding those n words to topic t_i, only if their addition maintains a graph property value of at least λ. We refer to this extended algorithm as λ-CLIQ$^+$.

4 EMPIRICAL EVALUATION

In this section, we conduct experiments on three different prelabeled, domain-specific Twitter data sets to demonstrate the accuracy of our model.

4.1 Experiment Setup

Data Sets. Our first data set, which we refer to as the Political data set, contains 764,993 tweets about Donald Trump during the 2016 Presidential Election and Transition Period prior to his inauguration, recorded between August 2016 and December 2016. 5,000 tweets each day were selected at random during this period from all of the tweets that mentioned the candidate. Our second data set, which we refer to as the Parenting data set, contains 953,115 tweets about parenting collected between March 2016 and September 2017. The tweets were collected from the accounts of Twitter users identified as 'parenting authorities' by social scientists studying the affect of social media on parents [26]. The third and final data set, which we refer to as the Covid, contains 883,990 English tweets from between January 16th, 2020 and April 1st, 2020. Tweets were collected if they mentioned hashtags associated with the coronavirus pandemic, including '#covid,' '#coronavirus,' and '#covid19.'

Data Set	|D|	All Tokens	Unique Tokens in D	Unique Tokens in D''
Political	764,993	14,404,344	931,572	380,793
Parenting	953,115	14,658,377	1,562,491	522,797
Covid	883,990	13,766,991	855,734	791,014

Table 1: Data Set Vocabulary Size Before and after Preprocessing

Baseline Algorithms. We test our model against baseline models that represent models from each class of topic models, as discussed in section 2. The models that we test against are Latent Dirichlet Allocation (LDA) [3], Hierarchical Dirichlet Process (HDP) [28], Non-Negative Matrix Factorization (NMF) [13], Topic Segmentation (TS) [8], Dirichlet Multinomial Model (DMM) [20], Self-Aggregated Topic Model (SATM) [24], GPU-DMM [16], and Biterm Topic Model (BTM) [32]. For LDA, TS, and BTM, we used k=20, 100, and 30 topics, respectively. All other baseline models produced their best topic sets at k=50. These values were chosen empircally after testing each model on k-values ranging from 10 to 100. All other parameters were left at their suggested values. [1]. We leave a more comprehensive sensitivity analysis for future work.

Data Preprocessing. The manner in which data is preprocessed can impact topic modeling results. Given the noisy nature of Twitter data, we consider this to be as important to the construction of our model as any of the other steps. We employ the preprocessing pipeline as defined by Churchill et al. [7], and use the following preprocessing steps, in the following order: Remove deleted posts, remove user tags, remove urls, remove punctuation (including hashtag symbol), lowercase text, remove stopwords, stem words, remove words of length less than three.

Deleted posts, user tags, and urls all contribute to noise while adding no rich topic content. We remove these prior to removing punctuation because they are easily identified with punctuation still intact. We remove stopwords, again to reduce noise, and we stem words to reduce the size of the vocabulary and get higher average word frequency. Lemmatization would be a valid replacement for stemming here; however, we choose stemming due to its more extreme truncation, which Churchill et al. showed can actually benefit the performance of topic models on noisy data streams [7]. Finally, we remove short words in order to reduce noise.

The data set statistics are shown in Table 1. $|D|$ represents the number of tweets. The final two columns show the difference in vocabulary size before and after preprocessing. The difference in the starting and ending vocabulary for Covid is signficantly smaller than those of the Political and Parenting data sets because the unprocessed Covid data set did not contain URLs.

Constructing the Phrase Model. Once our data set has been cleaned, we approximate phrases by creating NGrams up to 4-grams from the remaining clean text. Other more traditional means of identifying phrases, such as Named Entity Recognition and Noun Phrase detection, are less attractive options for social media data because they rely on accurate part-of-speech tagging and well-formed sentences to produce accurate results. We then replace the component words of NGram with the respective NGrams within each document. We identify frequent NGrams using the Natural Language Tool Kit (NLTK) [1]. We empirically evaluated our parameters and determined that setting the minimum frequency of an NGram to be 256 is reasonable for all of our data sets. The higher the minimum frequency, the faster the construction of the Phrase model, but the smaller the vocabulary. Too small a vocabulary is problematic, so balancing graph construction cost and vocabulary size is important.

Training the Word Embeddings. In order to find closely related words for improving topics, we train a Word2Vec model using CBOW on our data set [17]. A skip-gram model would be an acceptable substitute for CBOW here. We train our model with 100 features, using the Gensim Python library [25].

Algorithm Parameters. We evaluate two variants of the λ-CLIQ: using density (λ-CLIQ$_D$) and using Silhouette (λ-CLIQ$_S$) during percolation. We also evaluate λ-CLIQ$^+$. While we conducted an extensive sensitivity analysis, we present only the parameters for the best results in the paper. For the Political data set, $\tau = 1$, and the embedding distance threshold = 0.5. For the Parenting data set, $\tau = 50$, and the embedding distance threshold = 0.25. For the Covid data set, $\tau = 1$, and the embedding distance threshold = 0.25. These parameters remain the same for all three algorithm variants.

4.2 Quantitative Analysis

In this section, we perform a quantitative analysis of our model's performance against the baseline models. We consider two metrics, topic coherence and topic diversity, that together attempt to address how well models can isolate meaningful topics in the data sets.

Topic Coherence. To measure a model's ability to capture meaningful topics, we employ normalized pointwise mutual information (NPMI) [15]. NPMI measures how closely related two words are based on their relative cofrequencies. Along with its variants, NPMI has been used to evaluate topic coherence in many recent topic modeling papers [10, 11, 16, 22, 24]. We can sum the pairwise NPMI scores for all of the tokens in a topic to get that topic's NPMI score. Recall, for a pair of tokens (x, y), we define the probability of them appearing together in a document as $P(x, y)$. We can use these probabilities to compute the NPMI of a topic $t \in T$ as follows:

$$NPMI(t) = \frac{\sum_{x,y \in t} \frac{\log(\frac{P(x,y)}{P(x)P(y)})}{-\log(P(x,y))}}{\binom{|t|}{2}} \tag{4}$$

NPMI scores are in the range $[-1, 1]$, with negative scores indicating low mutual information between topic tokens, and positive scores indicating high mutual information. A high NPMI indicates high topic coherence. A high topic coherence score indicates a topic model that is capable of placing topic words together in a meaningful way.

Topic Diversity. To measure how successful topics are at finding unique topics, we use topic diversity. Topic diversity is the fraction of unique words in the top 20 words of all topics in a topic set [11]. Low diversity indicates that topics are being flooded with the same few words, while high diversity indicates that a model is able to efficiently isolate topics and topic words.

Comparison. Neither topic coherence nor topic diversity are indicators of a good topic set individually. However, together they can give a better notion of whether a topic model that is producing

[1] We use the Mallet implementation of LDA

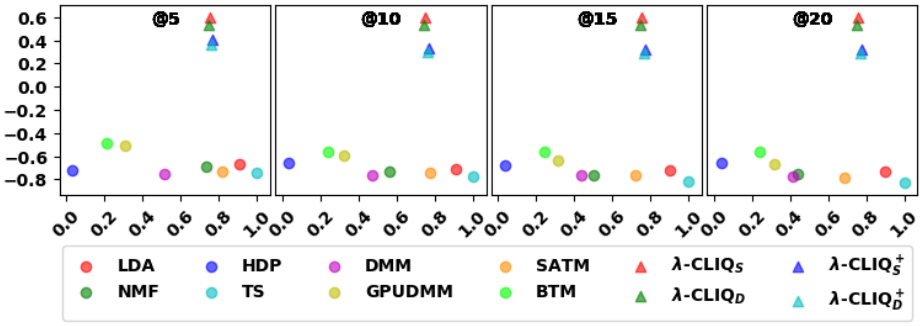

Figure 2: Coherence compared with Diversity for each Model on the Covid data set.

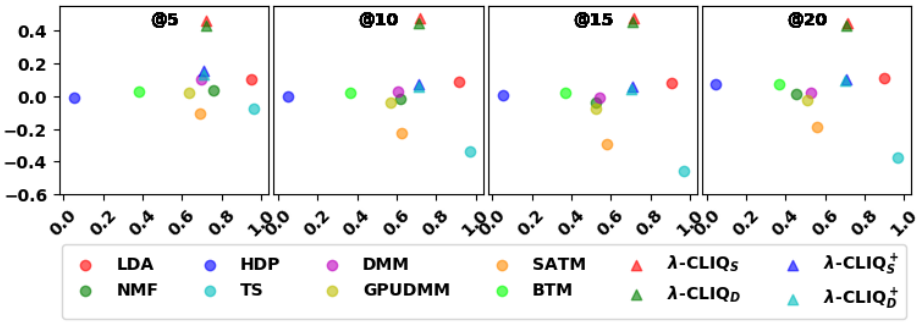

Figure 3: Coherence compared with Diversity for each Model on the Parenting data set.

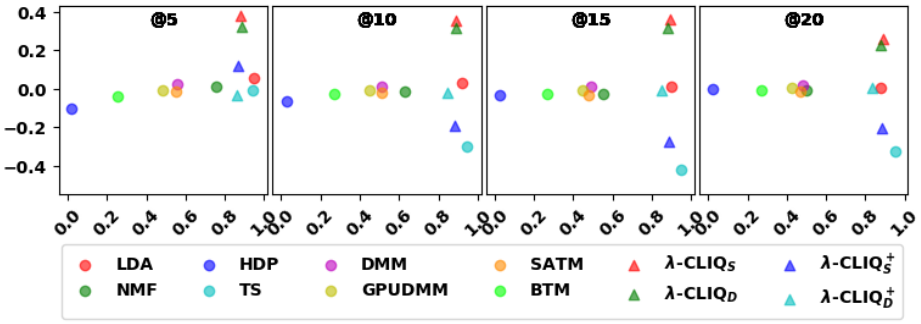

Figure 4: Coherence compared with Diversity for each Model on the Political data set.

good topics. Topic coherence measures the meaningfulness of individual topics; however it does not detect the meaningfulness of the entire topic set. When coupled with topic diversity, which measures whether or not topics are being repeated or mixed together, topic coherence becomes useful. A topic model with good coherence and diversity scores is likely a better model than one with only a high score in one of the two categories. We present this relationship in Figures 2, 3, and 4.

Each plot compares the coherence score (y-axis) with the diversity score (x-axis) of each model for the top-x words in each topic. The number of top words is indicated at the top of each plot (@5, 10, 15, 20). The baseline models are represented by circles, whereas the variations of our model are represented by triangles. By visualizing the results in this way, we can easily tell which models produce the best topics according to coherence and diversity. The best methods having high coherence and high diversity would be in the top-right corner of each plot.

The first observation is that λ-CLIQ$_S$ and λ-CLIQ$_D$ are closest to the upper right corner for all data sets across all top word sizes. In all data sets, we see that λ-CLIQ$_S$ produces slightly better coherence than λ-CLIQ$_D$. The extended variations have a lower coherence than their counterparts without embedding augmentation. This is due to the nature of embedding augmentation. Adding more words to a topic can result in lower NPMI due to the added words not

λ-CLIQ $_S$	λ-CLIQ^+_S	LDA	HDP	NMF	TS	DMM	GPUDMM	BTM	SATM
julian$assang	**julian$assang**	elect	donald	show	**hillari**	**hillari**	donald	donald	campaign
wikileak	**wikileak**	**russia**	**hillari**	realiti	media	**wikileak**	**email**	peopl	report
clinton$campaign	**clinton$campaign**	**putin**	support	lose	still	fake	**hillari**	trump	return
trump$campaign	trump$campaign	**email**	vote	**email**	actual	**leak**	accus	**clinton**	call
manag	manag	report	**clinton**	ahead	candid	**email**	**wikileak**	campaign	press
	snowden	**russian**	peopl	**leak**	press	**clinton**	**clinton**	think	**hillari$clinton**
	hillaryemail	releas	elect	world	blame	break	break	media	attack
	modi	return	obama	real	corrupt	donald	media	american	trump
	podestaemail	white$hous	presid	return	**email**	democrat	video	**email**	time
	podesta	prove	media	record	sick	video	**leak**	show	scandal
	dncleak	**wikileak**	trump	late	deplor	report	fake		hide
	clinton$camp	**leak**	campaign	crowd	creat	media	report		health
	robbi	offici	said	night	**wikileak**	plan	call		**leak**
	mook	confirm	call	video	tcot	campaign	stori		watch
	lewandowski	evid	think	respect	liar	elect	miss		
	donald$campaign	hide	make	daili	**hillaryclinton**	accus	**russia**		
	donald$trump$campaign	told	time	today	slam	expos	attack		
	payrol	**hack**	america	tonight	push	stori	show		
		msnbc	need	1	**clinton$camp**	2016	tape		
		deni	women	**wikileak**	wake	support	claim		

Figure 5: A Topic about the leak of Hillary Clinton's emails via Wikileaks, captured by each topic model. Bolded words are highly related to the topic.

having high co-frequencies with all of the original topic words. This is best visualized in the Political data set results, where the NPMI of λ-CLIQ^+_S drops drastically from the @5 to @10 plot. This indicates that the set of words added by embedding augmentation contained significant amounts of noise. Recall that the other two data sets were set to a lower embedding augmentation threshold, and their NPMI scores do not drop as significantly. For the extended variants, λ-CLIQ^+_S still edges out λ-CLIQ^+_D, with the exception of the Political data set.

The best baseline model overall is LDA, which has coherence competitive with the other models, and a much higher diversity than other baselines with high coherence. In the Covid data set, the baseline models all have very poor coherence, while our models get the highest coherence scores for any data set. It's unclear as to why the baseline models have such poor performance, but it is possible that the poor performance stems from the high frequency of generic words (flood words) about coronavirus. If most people talk about how coronavirus affects their everyday life, then it is possible that more generic words, that share little mutual information with all other words, are found to be the top words by the baseline models. The proliferation of generic words in the topics would explain the low NPMI scores. Another reason that Covid could be a more difficult data set than the other two is its larger vocabulary size, meaning that cofrequencies will on average be lower.

In the Parenting data set, we see a reasonable comparison between our models and LDA. The extended variants produce topics with a similar coherence, but lower diversity, while the non-augmented variations get a much higher coherence and similar diversity to that of their augmented counterparts.

With the non-augmented variants of our model producing better coherence scores than our extended variants, a natural question is why bother with the embedding augmentation at all? We address this with our qualitative analysis.

4.3 Qualitative Analysis

Similar to clustering, determining which topics are the 'best' does not always map to the ones with the highest quantitative evaluation. Therefore, we show a qualitative case study comparing a very popular topic from the Political data set across all models. We chose a topic from the Political data set specifically because λ-CLIQ^+_S gets a poor coherence score in that data set. The topic that we chose pertains to the scandal surrounding the leak of Hillary Clinton's private emails, orchestrated by Russia and published by Wikileaks.

Figure 5 shows this topic as it was captured by each topic model in our experiments. For the sake of space, we limit each topic to the top twenty words, although some are shorter. In this figure, the words most relevant to the topic (as decided by comparing to expert-labelled topics) are bolded. We can see from a glance that the topic model with the most relevant words is λ-CLIQ^+_S. It contains ten bolded words out of eighteen, indicating that it successfully isolated a significant number of relevant topic words. From a qualitative perspective, having email and wikileaks is important for this topic. Four out of eight of the baseline models have those two words. LDA, TS, DMM, and GPUDMM all produce reasonable topics, but each model allows much more noise than λ-CLIQ^+_S.

A disadvantage of the generative baseline models (LDA, HDP, DMM, GPUDMM, BTM, SATM) compared to the percolation-based models is that their statistical nature dissuades the inclusion of NGrams in the most likely words of a topic. Because the models hinge on high frequency and high co-occurrence between tokens, unigrams are much more likely to have a higher weight than NGrams, which are much less frequent. In the percolation-based models, we see far more NGrams in each topic, many of which are relevant topic words.

Comparing coherence scores. Returning to the question of why we bother using embedding augmentation, let's consider the topic generated by λ-CLIQ^+_S. In this example, it is clear exactly which words came from the embedding augmentation, and which came from the standard model. We can see that while the first three words in the topic from λ-CLIQ$_S$ capture important facets of the topic, the words from the embedding augmentation provide important context such as *hillaryemail*, *podestaemail*, *dncleak*, and *lewandowski*. The 'words' containing *email* are hashtags that were deconstructed in the preprocessing phase, and give the context of what was released on Wikileaks. Along with *lewandowski*, they also provide the subjects of the scandal, presidential candidate Clinton and her campaign executives. *dncleak* provides further

context about relevant events associated with the scandal during the campaign period.

When evaluating topic models, quantitative evaluation alone is not sufficient. Conducting both a qualitative and a quantitative analysis gives us more insight into the quality of topics generated by different models. By doing so, we are able to see the value of adding embedding augmentation to our model.

5 CONCLUSIONS

In this paper we propose a new topic model that is a variant of the k-clique percolation algorithm used in clustering. Using a cofrequency graph we identify topics by removing weighted edges until only quasi-clique subgraphs remain and recombine subgraphs using different percolation rules. We also propose an extended version of our model that incorporates a language model to improve the quality of topics. We show through an extensive empirical analysis that our model produces more coherent topics than other state of the art models, while maintaining a high level of topic diversity. We find that although some state of the art models can produce very high topic diversity scores, their topic sets are not as coherent as those produces by our model. We conduct a qualitative analysis of topics and find that our model is capable of producing coherent topics that are less noisy than those produced by other topic models. Finally, we publicly release our model as well as others used in experiments as a Python package to help advance topic modeling research on social media data sets such as Twitter.[2]

While λ-CLIQand λ-CLIQ$^+$can produce coherent topics in social media data with less noise than other state of the art models, they can still be improved upon. Integrating online computation of topics and NGrams would allow this class of models to more easily scale to larger data sets. Adding a temporal aspect to λ-CLIQ would allow one to track coherent, low noise topics throughout time, instead of statically. Both of these innovations are necessary for successful topic modeling in the fast-evolving landscape of contemporary social media.

Acknowledgements. This work was supported by the National Science Foundation grant numbers #1934925 and #1934494, and by the Massive Data Institute (MDI) at Georgetown University.

REFERENCES

[1] Steven Bird, Ewan Klein, and Edward Loper. 2009. *Natural Language Processing with Python.* O'Reilly Media.
[2] David M Blei and John D Lafferty. 2006. Dynamic topic models. In *23rd international conference on Machine learning (ICML)*. 113–120.
[3] David M. Blei, Andrew Y. Ng, and Michael I. Jordan. 2003. Latent Dirichlet Allocation. *Journal of Machine Learning Research* 3 (3 2003), 993–1022.
[4] Vincent D Blondel, Jean loup Guillaume, Renaud Lambiotte, and Etienne Lefebvre. 2008. Fast unfolding of communities in large networks. *Journal of Statistical Mechanics: Theory and Experiment* 10 (2008).
[5] Mario Cataldi, Luigi Di Caro, and Claudio Schifanella. 2010. Emerging Topic Detection on Twitter Based on Temporal and Social Terms Evaluation. In *ACM KDD Workshop on Multimedia Data Mining*.
[6] Rob Churchill, Lisa Singh, and Christo Kirov. 2018. A Temporal Topic Model for Noisy Mediums. In *Pacific-Asia Conference on Knowledge Discovery and Data Mining (PAKDD)*.
[7] Rob Churchill, Lisa Singh, and Josh Pasek. 2019. The Impact of Preprocessing for Building Topic Models Using Short, Noisy Text. In *Under Review*.
[8] Henrique Ferraz de Arruda, Luciano da Fontoura Costa, and Diego R. Amancio. 2016. Topic segmentation via community detection in complex networks. *Chaos* 26 (2016).
[9] Imre Derényi, Gergely Palla, and Tamás Vicsek. 2005. Clique percolation in random networks. *Physical review letters* 94, 16 (2005), 160–202.
[10] Adji B Dieng, Francisco JR Ruiz, and David M Blei. 2019. Topic modeling in embedding spaces. *arXiv preprint arXiv:1907.04907* (2019).
[11] Adji B. Dieng, Francisco J. R. Ruiz, and David M. Blei. 2019. The Dynamic Embedded Topic Model. *CoRR* abs/1907.05545 (2019).
[12] InternetLiveStats. 2019. Twitter Usage Statistics. http://www.internetlivestats.com/twitter-statistics/. Accessed: 05-05-19.
[13] Shiva Prasad Kasiviswanathan, Prem Melville, Arindam Banerjee, and Vikas Sindhwani. 2011. Emerging Topic Detection Using Dictionary Learning. In *ACM International Conference on Information and Knowledge Management*.
[14] John D Lafferty and David M Blei. 2006. Correlated topic models. In *Advances in Neural Information Processing Systems (NIPS)*. 147–154.
[15] Jey Han Lau, David Newman, and Timothy Baldwin. 2014. Machine reading tea leaves: Automatically evaluating topic coherence and topic model quality. In *Conference of the European Chapter of the Association for Computational Linguistics*. 530–539.
[16] Chenliang Li, Haoran Wang, Zhiqian Zhang, Aixin Sun, and Zongyang Ma. 2016. Topic Modeling for Short Texts with Auxiliary Word Embeddings. In *ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR '16)*. 165–174.
[17] Tomas Mikolov, Kai Chen, Greg Corrado, and Jeffrey Dean. 2013. Efficient estimation of word representations in vector space. *arXiv preprint arXiv:1301.3781* (2013).
[18] Christopher E. Moody. 2016. Mixing Dirichlet Topic Models and Word Embeddings to Make lda2vec. *CoRR* abs/1605.02019 (2016).
[19] Dat Quoc Nguyen, Richard Billingsley, Lan Du, and Mark Johnson. 2015. Improving topic models with latent feature word representations. *Transactions of the Association for Computational Linguistics* 3 (2015), 299–313.
[20] Kamal Nigam, Andrew Kachites McCallum, Sebastian Thrun, and Tom Mitchell. 2000. Text classification from labeled and unlabeled documents using EM. *Machine learning* 39, 2-3 (2000), 103–134.
[21] Dan Noyes. 2019. The Top 20 Valuable Facebook Statistics - Updated May 2017. https://zephoria.com/top-15-valuable-facebook-statistics/. Accessed: 2017-05-05.
[22] Jipeng Qiang, Ping Chen, Tong Wang, and Xindong Wu. 2016. Topic Modeling over Short Texts by Incorporating Word Embeddings. *CoRR* abs/1609.08496 (2016).
[23] Jipeng Qiang, Ping Chen, Tong Wang, and Xindong Wu. 2017. Topic modeling over short texts by incorporating word embeddings. In *Pacific-Asia Conference on Knowledge Discovery and Data Mining*. Springer, 363–374.
[24] Xiaojun Quan, Chunyu Kit, Yong Ge, and Sinno Jialin Pan. 2015. Short and sparse text topic modeling via self-aggregation. In *International Joint Conference on Artificial Intelligence*.
[25] Radim Řehůřek and Petr Sojka. 2010. Software Framework for Topic Modelling with Large Corpora. In *LREC 2010 Workshop on New Challenges for NLP Frameworks*. 45–50.
[26] R Ryan, P. E. Davis-Kean, L. Bode, J Kruger, Z. Mneimneh, and L. Singh. 2020. The new Dr. Spock: Analyzing information provided by parenting-focused Twitter accounts. In *[Unpublished paper under review]*.
[27] Farial Shahnaz, Michael W. Berry, V.Paul Pauca, and Robert J. Plemmons. 2006. Document Clustering Using Nonnegative Matrix Factorization. *Information Processing Management* (3 2006), 373–386.
[28] Yee Whye Teh, Michael I Jordan, Matthew J Beal, and David M Blei. 2006. Hierarchical Dirichlet Processes. *J. Amer. Statist. Assoc.* 101, 476 (2006), 1566–1581.
[29] Laure Thompson and David Mimno. 2018. Authorless Topic Models: Biasing Models Away from Known Structure. In *COLING*.
[30] Jiamiao Wang, Ling Chen, Lu Qin, and Xindong Wu. 2018. ASTM: An Attentional Segmentation Based Topic Model for Short Texts. In *IEEE International Conference on Data Mining (ICDM)*. IEEE, 577–586.
[31] Xuerui Wang and Andrew McCallum. 2006. Topics over Time: A non-Markov Continuous-time Model of Topical Trends. In *ACM International Conference on Knowledge Discovery and Data Mining (KDD)*.
[32] Xiaohui Yan, Jiafeng Guo, Yanyan Lan, and Xueqi Cheng. 2013. A Biterm Topic Model for Short Texts. In *International Conference on World Wide Web (WWW '13)*. 1445–1456.
[33] Xiaohui Yan, Jiafeng Guo, Shenghua Liu, Xueqi Cheng, and Yanfeng Wang. 2013. Learning topics in short texts by non-negative matrix factorization on term correlation matrix. In *SIAM International Conference on Data Mining (SDM)*.
[34] Jianhua Yin and Jianyong Wang. 2014. A dirichlet multinomial mixture model-based approach for short text clustering. In *ACM SIGKDD international conference on Knowledge discovery and data mining*. ACM, 233–242.
[35] Wayne Xin Zhao, Jing Jiang, Jianshu Weng, Jing He, Ee-Peng Lim, Hongfei Yan, and Xiaoming Li. 2011. Comparing twitter and traditional media using topic models. In *European Conference on Information Retrieval (ECIR)*.

[2] http://datalab.cs.georgetown.edu/code/