# An Evaluation of Topic Modelling Techniques for Twitter

**Elías Jónsson**
Department of Computer Science
University of Toronto
Toronto, On M5S1A1
elj31@cs.toronto.edu

**Jake Stolee**
Department of Computer Science
University of Toronto
Toronto, On M5S1A1
jstolee@cs.toronto.edu

## Abstract

In this paper, we complete an evaluation of various topic modelling algorithms, and examine their performance when working with Twitter tweets. LDA [1] is an algorithm that is often used when modelling topics within text, and it has been proven to be affective; however, LDA may not necessarily perform well when working with documents that are short in length [2, 3, 4]. We compare LDA to three models which offer potential improvements over the downfalls of LDA when modelling tweets. This includes a variation of LDA, referred to as LDA-U, which aggregates data on a user-basis in an effort to improve the standard LDA model's performance[3]. We also evaluate two other models specifically designed to work with short text: the "biterm topic model" (BTM), and a "word2vec Gaussian mixture model", which models topics as a distribution over words in semantic space [4].

## 1 Introduction

Topic modelling is the task of identifying which underlying concepts are discussed within a collection of documents, and determining which topics each document is addressing. This type of modelling has many applications; for example, topic models may be used for information retrieval (IR) [5], to identify influential individuals on a social media platform [6], or to detect signs of depression [7]. The algorithms traditionally used to tackle the problem of topic modelling include probabilistic latent semantic analysis (pLSA) [8] and Latent Dirichlet allocation (LDA) [1]; however, traditional topic models such as these have typically only been proven to be effective in extracting topics from documents that are at least a few hundred words long, and may not perform well when working with documents of shorter length [2, 3, 4]. This is unfortunate, as resources such as Twitter contain a significant amount of "short text" documents that may provide potentially useful information, if mined properly[1]. In this paper, we examine 3 different approaches to modelling topics within short texts, specifically when dealing with Twitter "tweets". These techniques include aggregating documents by author in an effort to address the downfalls of LDA, the "biterm topic model" (BTM) [3], and a model which clusters word2vec vectors using a Gaussian mixture model (GMM) [4].

## 2 Methods

### 2.1 LDA

As mentioned previously, LDA is a model that is frequently used to capture the set of latent "topics" over documents within a corpus. LDA models the generation of documents within a corpus as

[1]Twitter currently restricts tweets to be at most 140 characters long.

the following process: **1**) A mixture of $k$ topics, $\theta$, is sampled from a Dirichlet prior, which is parameterized by $\alpha$; **2**) A topic $z_n$ is sampled from the multinomial distribution, $p(\theta; \alpha)$, which models $p(z_n = i|\theta)$; **3**) A word, $w_n$, is then sampled (given the topic $z_n$) via the multinomial distribution $p(w|z_n)$. Overall, the probability of a document (or tweet, in our case) "$\mathbf{w}$" containing $n$ words can be described as [1]:

$$p(\mathbf{w}) = \int_\theta \Big( \prod_{n=1}^{N} \sum_{z_n=1}^{k} p(w_n|z_n; \beta) p(z_n|\theta) \Big) p(\theta; \alpha) d\theta \tag{1}$$

Given a corpus of $M$ documents $\mathcal{D} = \{\mathbf{w_1}, ..., \mathbf{w_M}\}$, the EM algorithm can be used to learn the parameters of an LDA model (by maximizing a variational bound on $p(\mathcal{D})$, as seen in Equation 2 below).

$$\log p(\mathcal{D}) \geq \sum_{m=1}^{M} \mathrm{E}_{q_m}[\log p(\theta, \mathbf{z}, \mathbf{w})] - \mathrm{E}_{q_m}[\log q_m(\theta, \mathbf{z})] \tag{2}$$

LDA was used as a baseline for comparison among other specialty models – the implementation of LDA provided by the `gensim`[9] Python library was used to gather experimental data and compared to other models.

#### 2.1.1 Modifying LDA: Aggregating Data

One method that has been used in the past to address the poor performance of LDA on shorter documents is the aggregation of documents into longer "pseudo-documents". This is done in an effort to increase the amount of relevant word "co-occurrences" within a document – if a Twitter user often talks about the same subjects, then combining their tweets will result in an increase in the co-occurrence of the relevant terms. When working with Twitter data, this has often been achieved by combining tweets that were created by the same author [6], or by combining tweets into large documents based on one or more terms shared between them [2]. In our experiments, we concentrate on the former; tweets were aggregated into pseudo-documents based on the author that created them, and a standard LDA model was trained on this modified data set. We denote this variation of the LDA model as "LDA-U" (as was done in [3]).

### 2.2 Biterm Model

Another model initially designed to work specifically with short texts is the "biterm topic model" (BTM) [3]. A graphical representation of this model in comparison to LDA can be seen in Figure 1. The BTM tackles this problem by learning topics over short text by directly modeling the generation of all the unordered word-pairs co-occurring within a document (i.e. biterms) across the corpus. The BTM model is inspired by the idea that topics are groups of correlated words where the correlation is revealed by the word co-occurrence. The BTM model uses that assumption to justify viewing the biterms across the topics as a mixture of various topics. By considering the whole collection of biterms as a mixture of topics, the BTM model is not affected by the same sparsity problem experienced by LDA (which views each document as a mixture of various topics). Figure 1 shows how the BTM uses $phi$ to represent a distribution of topics over the biterms and how $\phi_d$ represents the distribution of topics in a document for the LDA model.

The specific generative process of the corpus in BTM can be described as follows, where $\alpha$ and $\beta$ are Dirichlet parameters.

1. For each topic $z$
   (a) draw a topic-specific word distribution $\phi_z \sim \mathrm{Dir}(\beta)$ for the whole collection
2. Draw a topic distribution $\theta \sim \mathrm{Dir}(\alpha)$
3. For each biterm $b$ in the biterm set $B$
   (a) draw a topic assignment $z \sim \mathrm{Multi}(\theta)$
   (b) draw two words: $w_i, w_j \sim \mathrm{Multi}(\phi_z)$

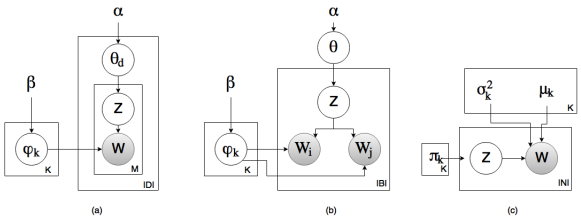


Figure 1: Graphical representations of the LDA (a), BTM (b) and GMM (c) models [3].

Looking at the above procedure and figure 1 it can be derived that

$$P(B) = \prod_{(i,j)} \sum_{z} \theta_z \phi_{i|z} \phi_{j|z} \tag{3}$$

where $\phi$ and $\theta$ are found either with variational inference or sampling methods. In this paper, Gibbs sampling was used to find $\phi$ and $\theta$. After training, $\phi$ and $\theta$ can be found with the following equations

$$\phi_{w|z} = \frac{n_{w|z} + \beta}{\sum_w n_{w|z} + M\beta}$$
$$\theta_z = \frac{n_z + \alpha}{|B| + K\alpha}$$

where $M$ is the number of words in a topic, $|B|$ is the number of biterms, $K$ is the number of topics, $n_{w|z}$ is the number of time the word $w$ is assigned to the topic $z$ and $n_z$ is the number of times a biterm b is assigned to a topic $z$.

One of the obstacles of using BTM is that BTM does not model the document generation process. Therefore, it is not possible to obtain the topic proportion of documents directly during the learning process.

To infer the topics in a document for the BTM, we assume that

$$P(z|d) = sum_b P(z|b)P(b|d) \tag{4}$$

where $P(z|b)$ can be calculated via Bayes' rule

$$P(z|b) = \frac{P(z)P(w_i|z)P(w_j|z)}{\sum_z P(z)P(w_i|z)P(w_j|z)} = \frac{\theta(z)\phi_{i|z}\phi_{j|z}}{\sum_z \theta(z)\phi_{i|z}\phi_{j|z}}.$$

The empirical distribution of biterms in the documents is used to obtain $P(b|d)$. The empirical distribution can be described by the following equation:

$$P(b|d) = \frac{n_d(b)}{\sum_b n_d(b)}$$

where $n_d(b)$ is how many times a biterm b appears in document d.

An implementation of BTM was provided by the authors of [3], but an implementation of the model was completed in Python for this paper to further our understanding of the algorithm, and to have full control over the model.

### 2.3 Clustering Word Embeddings

Over the past few years, various "word embedding" language modelling techniques have been proposed, which map words within a corpus to a multi-dimensional vector space. These modelling techniques capture the semantic relationships between all words in a corpus by examining the context in which the words appear. Such word embedding algorithms include word2vec [10, 11], and

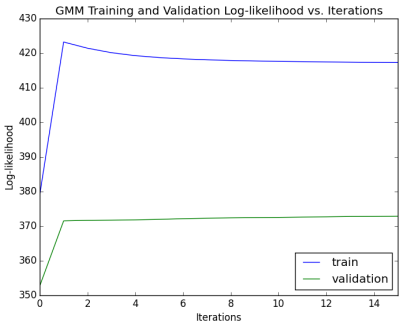


Figure 2: Log-likelihood as EM is run on the GMM (W2V-GMM=$\{$`K`=100, `wlen`=11, `D`=200)$\}$.

GloVe [12]. Recently, various topic models have been proposed which make use of the semantic space created by such word embedding algorithms [13], and may be used as alternatives to traditional topic models. As discussed by Sridhar in [4], one such approach could be to use word2vec (and the "continuous-bag-of-words", or "CBOW" model) to construct a semantic vector space from a corpus, and cluster words in the resulting vector space using a Gaussian mixture model (GMM) [4]. In this report, we will refer to this word2vec and Gaussian mixture models as "W2V-GMM". Much like BTM, W2V-GMM attempts to address the data sparsity issues generally present with models such as LDA by creating a generative model over the whole corpus (rather than a distribution over documents). The intuition behind this model is that the components of the GMM can be viewed as abstract topics within the word embedding's semantic vector space, and each component $z$ within the GMM has a certain probability of generating a word seen in the corpus – $p(w|z)$.

The W2V-GMM model can be configured based on both the underlying word2vec and GMM models. The underlying CBOW word2vec model learns a neural network that is capable of predicting a provided word, given a context - ie. $p(w|c)$ [10, 11]; the length of the window surrounding the provided word ("`wlen`") is configured before training. Furthermore, the word2vec model also learns a latent semantic space of *D* dimensions, which is also set to a fixed size before training begins. As is standard with a GMM, the W2V-GMM can use the EM algorithm to learn its parameterization. Values for $p(w|z)$ (the probability of generating a word, given a specific topic) and $p(z)$ (the GMM "mixing" weights) are learned, and used to calculate $p(z|w)$ (see Equation 5), the posterior probability of a specific topic, given a word. The progression of the train and validation log-likelihood observed during training for a specific configuration of the GMM can be seen in Figure 2. As was done in [4], covariance matrices were assumed to be diagonal for computational purposes.

$$p(z|w) = \frac{p(w|z)p(z)}{p(w)} \tag{5}$$

The author of [4] claims that W2V-GMM is capable of outperforming both LDA and BTM when modelling topics within short documents (such as tweets). To examine the effectiveness of this approach, an implementation of the model outlined in [4] was created using components from the `gensim` [9] and `scikit-learn` [14] Python libraries, and evaluated against the other models. One issue that must be addressed when dealing with word2vec models that are evaluated on held-out corpus data is the situation in which a vector for an unseen word is required. Various techniques have been proposed and analyzed for addressing such terms, referred to "Out-of-Vocabularly" (OOV) words. Such solutions include using character-level (or "subword") language models [15], which can approximate the modelling of unseen words through aggregating models of its sub-components. For the experiments described in this paper, however, the OOV issue was solved by assigning unseen words to random vectors within the semantic space, as is done in [16].

| Model | Hyperparameters |
|---|---|
| LDA | $\{\alpha = 10/K, \beta = 0.01\}$ |
| LDA-U | $\{\alpha = 10/K, \beta = 0.01\}$ |
| BTM | $\{\alpha = 1/K, \beta = 0.01\}$ |
| W2V-GMM | $\{\texttt{wlen} = 11, D = 200\}$ |

Table 1: The optimal hyperparameter settings for each model (for a setting of K topics).

# 3 Experiments

## 3.1 Data Collection and Preprocessing

The data used in these experiments was collected via the Twitter Search API, which allows for past tweets to be collected based on a set of input keywords. 1,350 English Tweets from 100 different search queries were collected and preprocessed. This included search queries such as: {"blockchain", "Sony", "San Francisco", "Dropbox", "jQuery", "Xbox", "Star Wars" and "Kardashian"}. Tweets were preprocessed in a similar fashion to what was done in [3, 4] by completing the following steps: Tweets were tokenized and made lowercase; emoticons, URLs, and HTML tags, and common English "stopwords" were stripped; infrequent words (which occurred less than 1 time in the entire corpus) and extremely short tweets (with less than 2 words) were removed; and words were lemmatized using the NLTK Python library's default POS tagger [17]. After the data was preprocessed and filtered, the tweets were divided into an 80/20 training and testing data split. This resulted in a data set of 129,530 tweets total, which was divided into 103,624 training and 25,906 testing tweets. When aggregating data into pseudo-documents for the LDA-U model, it was found that, on average, each user had 11.24 documents within the corpus, and aggregating tweets caused the average document length to grow from 8.46 to 12.79 words.

## 3.2 Selecting Model Hyperparameters

After data was collected and preprocessed, multiple versions of each of the LDA, LDA-U, BTM, and W2V-GMM models were trained in an effort to evaluate the optimal hyperparameter settings for each model. Grid search was performed over various hyperparameter settings for each model, and final model configurations were selected based on the model perplexity/likelihood of the held-out set of "test" tweets. For the LDA and LDA-U models, the following hyperparameter settings were evaluated (for a setting of "K" topics): $\alpha$={1/K, 10/K, 50/K, 100/K, 250/K}, $\beta$={0.001, 0.01, 0.1, 0.5, 1.0}. For the BTM model, the following hyperparameter settings were examined: $\alpha$={1/K, 50/K, 100/K}, $\beta$={0.001, 0.01, 0.5}. For the W2V-GMM model, the following hyperparameter settings were evaluated: $D$={50, 100, 200}, $\texttt{wlen}$={11, 13, 15, 17} (similar to what was done in [4]). For each model, values of $K$ from the set {10, 50, 100, 200} were evaluated. The set of hyperparameters $K \times \alpha \times \beta$ was then evaluated for the LDA, LDA-U, and BTM models, and the set $K \times D \times \texttt{wlen}$ was then evaluated for the W2V-GMM model. The most optimal hyperparameter settings can be seen in Table 1. These values were used when comparing the various models.

## 3.3 Evaluation

After the optimal hyperparameter settings were selected for each model, experiments were run to compare the effectiveness of each model. For each experiment, the optimal models (as seen in Table 1) were chosen for a setting of K=100 topics, as it is equal to the number of search queries that were used when the data was collected. The resulting topics from the models often corresponded to the search queries used to retrieve the data (or mixture of related search queries). A few topics were chosen randomly from the resulting topics of the W2V-GMM model, as seen in Table 2, and the most similar BTM topics can be seen in Table 3. Most of those topics are easy to interpret, for example the first topic is about the earthquake in Ecuador where approximately 654 people died. The second topic contains terms that are all related to the English soccer Premier League. The third and fourth topic are though little harder to interpret for the W2V-GMM model because it seems like

| W2V-GMM | | | |
|---|---|---|---|
| Topic 1 | Topic 2 | Topic 3 | Topic 4 |
| earthquake | tottenham | #blackholes | rob |
| aftershock | spurs | portal | #sexy |
| ecuador | arsenal | snape | manuka |
| remain | mauricio | failure | kardashian |
| devastate | leicester's | #stephenhawking | #nude |
| #ecuador | #epl | military | #hot |
| tremor | pochettino | #bernie | #real |
| 646 | title | ukrainian | kim |
| magnitude | match | escape | #leaked |
| 130 | arsene | buchanan | blac |

Table 2: Ten most probable words shown for four different topics predicted by W2V-GMM.

| BTM | | | |
|---|---|---|---|
| Topic 1 | Topic 2 | Topic 3 | Topic 4 |
| ecuador | leicester | stephen | kanye |
| toll | title | hawk | west |
| earthquake | win | black | birthday |
| death | league | hole | celebrate |
| 650 | ranieri | could | girl |
| rise | city | via | others |
| top | premier | universe | miami |
| 654 | tottenham | portals | kourtney |
| ecuador's | say | history | kim |
| people | tell | another | photo |

Table 3: Ten most probable words shown for four different topics predicted by BTM.

#### 3.3.1 Intra-Cluster and Inter-Cluster Distance

As described in Section 3, the tweet data was collected based on a set of input keywords. Comparing the generated topic clusters to the pre-defined search query clusters can give insight into how well a topic model is grouping documents. This is similar to what is discussed in [3], except the authors made use of hashtag data (rather than search query data). The intuition behind this performance measure is that a "good" topic model should cluster documents that are from similar "search query" groups into topic clusters that are tightly packed and spread out from each other. As defined in [3], we can represent a document as a vector of topic probabilities, given the document. In other words, for a set of $k$ topics $\{z_1, \ldots, z_k\}$, we can define a tweet $d_i$ as a vector in the form seen in Equation 6. Using these vectors, we can then measure the "distance" between two tweets in this space using the "Jensen-Shannon" divergence measure, as defined in Equations 7 and 8 (where $D_{KL}$ is the KL divergence measure) [3].

$$d_i = [p(z_1|d_i), \ldots, p(z_k|d_i)] \tag{6}$$

$$dis(d_i, d_j) = \frac{1}{2} D_{KL}(d_i||m) + \frac{1}{2} D_{KL}(d_j||m) \tag{7}$$

$$m = \frac{1}{2}(d_i + d_j) \tag{8}$$

The authors of [3] then go on to introduce two "distance scores" based on $dis(d_i, d_j)$, which represent the average (intra) distance between tweets inside the pre-defined "search query" groups, and the average (inter) distance between the tweets in different pre-defined groups. They define the

| Model | H-Score |
|---|---|
| W2V-GMM | 0.9524 |
| LDA | 0.60597 |
| LDA-U | 0.54834 |
| BTM | 0.41111 |

Table 4: H-score for all four models.

"intra-cluster" and "inter-cluster" distance measure as seen in Equations 9, 10. In these equations, $C$ refers to any pre-defined "search query" tweet group taken from the data set.

$$IntraDis(C) = \frac{1}{K}\sum_{k=1}^{K}\left[\sum_{\substack{d_i,d_j \in C_k \\ i \neq j}} \frac{2dis(d_i,d_j)}{|C_k|\,|C_k - 1|}\right] \tag{9}$$

$$IntraDis(C) = \frac{1}{K(K-1)}\sum_{\substack{C_k, C_{k'} \in C \\ k \neq k'}}\left[\sum_{d_i \in C_k}\sum_{d_j \in C_{k'}} \frac{dis(d_i,d_j)}{|C_k|\,|C_{k'}|}\right] \tag{10}$$

The authors then use these two "inter-cluster" and "intra-cluster" measure to compute a final "$H$" score, as seen in Equation 11.

$$H = \frac{IntraDis(C)}{InterDis(C)} \tag{11}$$

The $H$ score defined in Equation 11 was calculated using the LDA, LDA-U, BTM, and W2V-GMM topic models, and their values can be seen in Table 4. As can be seen from the table, the BTM model has the lowest H-score and, therefore, we can conclude that the BTM model generates topics that correspond closely to the predefined search queries used to retrieve each tweet from the corpus. However, the W2V-GMM model has significantly higher H score –this may imply that the topics generated by the W2V-GMM mixes words from different Twitter search queries. This can be seen in topic 3 in table 2, where this topic is mixed with words related to Stephen Hawking and the U.S. election.

### 3.4 Coherence Scores

Evaluating topic models is often viewed as a difficult task. Sometimes, a measure such as the $H$ score is used, or the perplexity of held-out data is evaluated against generated models and used to analyze the relative performance of competing topic models. However, studies have been performed on the correlation between traditional topic model evaluation metrics and the "coherence" of generated topics, and it was found that typical performance measures may not accurately evaluate the understandability of the top words within a topic [18]. Many coherence scores have been proposed – these values aim to emulate the relative score that a human is likely to assign to a topic when considering how much the topic words "make sense". These scores infer the cohesiveness of topics by analyzing the semantic differences/similarities between "top" words within a generated topic. The authors of [19] defined a unifying framework for constructing coherence scores and examined many popular coherence scores, including the coherence score proposed by Mimno et al [20]. The score uses: a topic $z$; it's top $N$ most likely words, $W^z = \{w_1^z, \ldots, w_N^z\}$; the document frequency of word $w$, $D(w)$; and $D(w_1, w_2)$, the number of documents in which both $w_1$ and $w_2$ occurred. It calculates the coherence score as seen in Equation 12 – a higher score indicates a topic that correlates well with what humans determine to be a "coherent" topic. The intuition around this performance

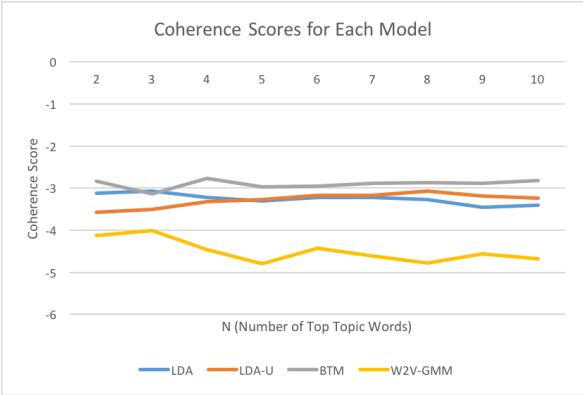


Figure 3: The coherence score of every topic model, plotted as a function of the top $N$ words.

measure is that if the top topic words occur together frequently relative to the number of times that the words appeared separate, then the words are more likely to form a "cohesive" topic.

$$C(z; W^z) = \sum_{n=2}^{N} \sum_{j=1}^{j-1} \log \frac{D(w_n^z, w_j^z) + 1}{D(w_j^z)} \tag{12}$$

This score is used to evaluate models in both the BTM [3] and W2V-GMM [4] papers. In this paper, we also examine the models with this commonly used coherence score in an effort to gauge how "coherent" the topics generated by each model are. One important difference, however, is the authors of [3, 4] calculated counts for $D(w)$ and $D(w_1, w_2)$ from the corpus used to train their models. The authors of [19] completed experiments which relied upon the (English) Wikipedia corpus to calculate their document counts, and realized the resulting scores correlated more closely with human topic ratings. Therefore, when calculating the coherence score of models, we made use of counts from Wikipedia. A database of these counts is provided by the authors of [19], and were freely available to access online at the time that this paper was written [2].

The coherence score for each model was calculated based on the top 10 words taken from all of its topics. The coherence score was then calculated using values of $N$=[2..10]. For each value of $N$, a model's coherence score was averaged. These coherence values were then plotted as a function of $N$, and can be seen in 3.

## 4 Discussion

In Section 3.3, observations were gathered on both topic model clustering quality and topic coherence. This provides insight into how general topic models (such as LDA), and more specialized topic models (such as BTM and W2V-GMM) perform when dealing with short documents, such as tweets.

After examining the $H$ score results gathered in Section 3.3.1, the BTM model seems to predict topics with the minimum intra distance and maximum inter distance. The H score for the W2V-GMM is significantly higher than for the other three models. But that may imply that the topics in the W2V-GMM are mixed with words from tweets gathered with different search queries. Even though lower H score indicates better topic model, that does not have to be true because lot of the search queries are correlated and, therefore, hard to conclude anything only from looking at the H score.

The results of Section 3.4 appeared to indicate that, on average, BTM produced the most coherent topics; however, the generated LDA, LDA-U and BTM models appear to have comparable coherence scores across multiple values for N, so the difference is quite slight.

[2] https://github.com/AKSW/Palmetto

Examining the difference in coherence score between LDA and LDA-U is somewhat difficult, as the difference appears to be quite minimal. According to Figure 3, it seems like LDA may provide more coherent topics when examining a smaller subset of the top topic words, and LDA-U may be slightly better when examining a larger number of the top topic words, but more data would have to be gathered in order to reinforce such a statement. In the future, more rigorous investigation into the relation between the coherence scores of LDA and LDA-U could be completed for larger ranges of $N$ to explore this potential trend.

It is also interesting to note that, according to Figure 3, LDA, a more traditional approaches to topic modelling, produced more coherent topics in comparison to the W2V-GMM model, which was designed with short documents in mind. Furthermore, it even appears as though the W2V-GMM becomes progressively less coherent as more topic words are considered in the calculation of the coherence score. This is contrary to what was discussed in the original paper [4], which claimed that the model could outperform both [1] and [3] when working with short documents. This apparent poor performance of the W2V-GMM model could potentially be due to the assumptions that the authors make when describing the model. Specifically, the author of [4] describes an implementation that assumes components within the GMM have diagonal covariance matrices. This assumption is quite limiting, as it assumes that each $D$-dimensional Gaussian distribution within the GMM can be decomposed into $D$ independent univariate Gaussian distributions, which ignores a significant amount of potential dependence within the data that is being modelled.

## 5 Conclusion

After implementing models that were specifically designed to work well with short text (BTM [3] and W2V-GMM [4]), and modifying LDA to work with short text (through the creation of the LDA-U), some overall observations were made.

After examining both the $H$ and coherence scores, it appeared as though BTM was superior to all other models when working with short documents. Although LDA has been proved to be effective in modelling topics within documents, BTM seems to be an improvement over LDA when modelling the topics of tweets. This implies that the design of the BTM helps it to generate topics which are both coherent, and consistent with the expected pre-defined groups that were defined in the tweet data.

As was mentioned in 4, it seemed as though LDA-U performed similarly to LDA and BTM, and even appeared to outperform the W2V-GMM. This implies that aggregating data may not have any effect on the effectiveness of the topic model when working with short text. Furthermore, the metadata required to complete such aggregation of data may not always be available, potentially making models such as LDA-U impractical[4].

Experiments showed that W2V-GMM did not perform well in comparison to the other models discussed in this paper. However, the original implementation described by the author makes assumption the GMM components within the model have diagonal covariance matrices [4]. A model that makes use of full covariance matrices may perform significantly better.

## 6 Future Work

Future work could involve analyzing research that has been completed to improve upon topic models that were discussed in this paper. For example, "Twitter-BTM" [21] is a model that attempts to address some of the downfalls of the standard BTM [3] model; specifically the discarding of potentially useful user information and the assumption that two words which co-occur together are drawn from the same topic. Furthermore, as mentioned in Section2.3, the W2V-GMM model implemented assumes that the mixture component covariances are diagonal [4]. An implementation using fully computed covariance matrices should be experimented with to examine the full potential of the model.

Unfortunately, due to time and financial constraints, it wasn't possible to gain data from humans on the coherence of the topics generated by the topic models discussed in this paper. Therefore, if possible, it would be beneficial to complete a crowdsourced analysis of the relation between human feedback and the coherence of the topics generated by the models that were generated for

this paper. This would help to further the understanding of the effects that short documents have on the coherence of generated topics.

## References


[1] David M Blei, Andrew Y Ng, and Michael I Jordan. Latent dirichlet allocation. *the Journal of machine Learning research*, 3:993–1022, 2003.

[2] Liangjie Hong and Brian D Davison. Empirical study of topic modeling in twitter. In *Proceedings of the first workshop on social media analytics*, pages 80–88. ACM, 2010.

[3] Xiaohui Yan, Jiafeng Guo, Yanyan Lan, and Xueqi Cheng. A biterm topic model for short texts. In *Proceedings of the 22nd international conference on World Wide Web*, pages 1445–1456. International World Wide Web Conferences Steering Committee, 2013.

[4] Vivek Kumar Rangarajan Sridhar. Unsupervised topic modeling for short texts using distributed representations of words. In *Proceedings of NAACL-HLT*, pages 192–200, 2015.

[5] Xing Yi and James Allan. A comparative study of utilizing topic models for information retrieval. In *Advances in Information Retrieval*, pages 29–41. Springer, 2009.

[6] Jianshu Weng, Ee-Peng Lim, Jing Jiang, and Qi He. Twitterrank: finding topic-sensitive influential twitterers. In *Proceedings of the third ACM international conference on Web search and data mining*, pages 261–270. ACM, 2010.

[7] Philip Resnik, William Armstrong, Leonardo Claudino, Thang Nguyen, Viet-An Nguyen, and Jordan Boyd-Graber. Beyond lda: exploring supervised topic modeling for depression-related language in twitter. *NAACL HLT 2015*, page 99, 2015.

[8] Thomas Hofmann. Probabilistic latent semantic indexing. In *Proceedings of the 22nd annual international ACM SIGIR conference on Research and development in information retrieval*, pages 50–57. ACM, 1999.

[9] Radim Řehůřek and Petr Sojka. Software Framework for Topic Modelling with Large Corpora. In *Proceedings of the LREC 2010 Workshop on New Challenges for NLP Frameworks*, pages 45–50, Valletta, Malta, May 2010. ELRA. `http://is.muni.cz/publication/884893/en`.

[10] Tomas Mikolov, Kai Chen, Greg Corrado, and Jeffrey Dean. Efficient estimation of word representations in vector space. *arXiv preprint arXiv:1301.3781*, 2013.

[11] Tomas Mikolov, Ilya Sutskever, Kai Chen, Greg S Corrado, and Jeff Dean. Distributed representations of words and phrases and their compositionality. In *Advances in neural information processing systems*, pages 3111–3119, 2013.

[12] Jeffrey Pennington, Richard Socher, and Christopher D Manning. Glove: Global vectors for word representation. In *EMNLP*, volume 14, pages 1532–1543, 2014.

[13] Li-Qiang Niu and Xin-Yu Dai. Topic2vec: Learning distributed representations of topics. *arXiv preprint arXiv:1506.08422*, 2015.

[14] F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot, and E. Duchesnay. Scikit-learn: Machine learning in Python. *Journal of Machine Learning Research*, 12:2825–2830, 2011.

[15] Tomáš Mikolov, Ilya Sutskever, Anoop Deoras, Hai-Son Le, Stefan Kombrink, and J Cernocky. Subword language modeling with neural networks. *preprint (http://www. fit. vutbr. cz/imikolov/rnnlm/char. pdf)*, 2012.

[16] Tom Kenter and Maarten de Rijke. Short text similarity with word embeddings. In *Proceedings of the 24th ACM International on Conference on Information and Knowledge Management*, pages 1411–1420. ACM, 2015.

[17] Steven Bird, Ewan Klein, and Edward Loper. *Natural language processing with Python*. " O'Reilly Media, Inc.", 2009.

[18] Jonathan Chang, Sean Gerrish, Chong Wang, Jordan L Boyd-Graber, and David M Blei. Reading tea leaves: How humans interpret topic models. In *Advances in neural information processing systems*, pages 288–296, 2009.

[19] Michael Röder, Andreas Both, and Alexander Hinneburg. Exploring the space of topic coherence measures. In *Proceedings of the eighth ACM international conference on Web search and data mining*, pages 399–408. ACM, 2015.

[20] David Mimno, Hanna M Wallach, Edmund Talley, Miriam Leenders, and Andrew McCallum. Optimizing semantic coherence in topic models. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing*, pages 262–272. Association for Computational Linguistics, 2011.

[21] Weizheng Chen, Jinpeng Wang, Yan Zhang, Hongfei Yan, and Xiaoming Li. User based aggregation for biterm topic model. *Volume 2: Short Papers*, page 489.