A SENTIMENT-TOPIC MODEL FOR TWITTER

YUNFEI LU AND NIKITA NANGIA

ABSTRACT. Topic modeling has been well developed in last decade and can be efficiently used for the large data set to do clustering, classification or, simply, topic trend modeling. However, using LDA for short texts, like tweets, is still a work in progress. The informality of the text and word sparsity make it hard to use the well-developed topic modeling techniques like LDA to get meaningful results. There is work being done in the field and in this paper we wish to explore some existing models. The main contribution of this paper is the Sentiment-Topic model which is method that adapts the Author-Topic model of Rosen-Zvi et al. (2014) to using sentiment data. We found mixed results in end but the findings for the neutral sentiment class are encouraging.

1. INTRODUCTION

With the advent of social media platforms like Twitter and Facebook, short texts in the form of updates and tweets have become an instrumental source of information. Even though each tweet is no more than a 140 characters, the cumulative sum of information available on Twitter is astonishing and complex. To be able to efficiently interpret such short texts then is vital. Understanding public sentiment is crucial to conducting polls and making predictions in the political sphere. With the recent failure of most experts to predict the outcome of Brexit and the United Stated 2016 election, developing better tools and techniques to understand public sentiment has become more important. Ideally, what we would like, it to build a tool to clearly understand the main topics under discussion and the general sentiment towards with regards to them.

In this vein, considerable research has been devoted to sentiment and opinion analysis of the Twitter corpus (Go et al., 2009; Pak et al., 2010; Agarwal et al. 2011). To tease out the issues being discussed, the field of topic modeling is ideally primed to be applied to the problem.

2. LATENT DIRICHLET ALLOCATION

Topic modeling has been researched for years and is still gaining increasing attention. Latent Dirichlet Allocation (LDA) is one of the standard

methods used for topic modeling. LDA is an unsupervised machine learning technique and it traditionally uses a bag-of-words approach. In LDA, each document in the corpus is considered to be a mixture of topics and is represented as a probability distribution over all topics. Each topic in turn, is represented as a probability distribution over words in the vocabulary of the corpus. Finally then, learning the distributions of the topics and word is a Bayesian inference problem. Gibbs sampling can be used to approximate the posterior distribution,

$$P(Z_{(m,n)}|Z_{-(m,n)}, \boldsymbol{W}; \alpha, \beta) = \frac{P(Z_{(m,n)}, Z_{-(m,n)}, \boldsymbol{W}; \alpha, \beta)}{P(Z_{-(m,n)}, \boldsymbol{W}; \alpha, \beta)}$$

2.1. **Term Frequency-Inverse Document Frequency.** Oftentimes, a Term Frequency-Inverse Document Frequency (TF-IDF) matrix is used instead of a bag-of-words style representation. In TF-IDF, term frequency is the number of times a word appears in a given document., i.e. tweet. While inverse document frequency is the logarithmically scaled inverse fraction of the documents that contain the word. Term frequency helps weigh important words within a single tweet highly and IDF reduces the weight of common words in a corpus. Then a TF-IDF count is calculated as follows

$$tf - idf_{w,t} = tf_{w,t} \times idf_w$$

where $tf_{w,t}$ is the term-frequency of a word in a tweet and idf_w is the IDF of the same word in the full corpus.

The TF-IDF matrix of a corpus gives us the TF-IDF scores of each word in the corpus. Since using TF-IDF has been proved useful in other applications of LDA, we chose to try using it in our experiments.

2.2. **Beyond Vanilla LDA.** Many variations and extensions of LDA have been implemented. For example, Wang et al. (2009) proposed a topic model to analyze image corpora in order to solve computer vision problems. Blei et al. (2012) introduced a supervised LDA model for better discriminant analysis. Rosen-Zvi et al. (2004) offered an author-topic model, which can model authors and their corresponding topic distributions. In their paper, they found that the model shows better performance than standard LDA when only a small number of words can be obtained from the documents. Applying topic models for short or few documents for text clustering is more challenging because of data sparsity and the limited contexts within such texts. One approach is to use only one topic per document (Nigam et al., 2000). This method is called Dirichlet Multinomial Mixture (DMM) model. In this model, each document is assumed to have only one topic. The process of generating a single document, d, from the collection of

documents, D, is to first select a topic for the document. Then, all the words in the documents are generated based on the topic-to-word Dirichlet-multinomial component.

Another model is the Biterm-Topic model (BTM) which is a word co-occurrence based model that learns the topic by modeling word-word co-occurring in the same context. For example, in the same short text window. Unlike LDA that models the word occurrences, a BTM models the biterm occurrences in a corpus. In generation procedure, a biterm is generated by drawing two words independently from the same topic.

3. SENTIMENT-TOPIC MODEL

The new model that we wanted to build and test is the Sentiment-Topic Model (STM). This is based on the Author-Topic model previously discussed. The author-topic model has been proved to be better than a vanilla LDA when dealing with the short text data (Hong and Davison, 2010). Since twitter has too many authors to reasonably use in such a model, we forgo the original intent of the work done by Rosen-Zvi et al. (2004) and instead use sentiment in place of authors.

We have three possible sentiments: positive, negative, and neutral. For each sentiment, the sentiment model determines a probability distribution over words. So we build a generative model that represents each tweet as a mixture of topics and allows the mixture weights of different topics to be informed by the sentiments of the tweet. As a result, we learn which topics appear in the corpus, the relevance of topics to tweets, and which topics fall under each sentiment class.

3.1. **Sentiment analysis.** The sentiment analysis itself is done with a N“aïve Bayes Classifier which is built around a bag-of-words model. For each word in a given document, or tweet, the probability of it being in the positive, negative, or neutral class is calculated. This probability is calculated by counting the word's appearance in the positive, negative, and neutral corpus that the NB model was trained on. We use Laplacian smoothing to avoid probabilites being brought to zero in the case that a word doesn't appear in the training set. The following equation defines the probability fo a tweet being in a particualr sentiment class,

$$P(tweet|C) = P(C)\prod_i P(w_i|C) = P(C)\prod_i \frac{count(w_i, C)}{\sum_i count(w_i, C)}$$

where $P(C)$ is the probability of the sentiment class and $count(w_i, C)$ is the number of times word w_i appears in sentiment class C.

4. DATA SELECTION

In this experiment, twitter streaming APIs is used to get the data. The streaming APIs give developers low latency access to Twitter's global stream of Tweet data. The filter is used to only get the data in the Great New York area and only English is accepted at this time. The timestamp of the data is December 15th. The raw data is JSON format and a simple parser is written in order to get the valid text data.

Since some users would retweet other's tweet and simply comment few words. In order to make this kind of data meaningful, the current tweet will be combined with the tweet which is retweeted from together as a valid data. All tweets will be combined with their hashtag to become valid data. NLTK package is used to remove the stop words and Stemmers is used to remove morphological affixes from words, leaving only the word stem. All punctuations are removed. Emojis and URLs are also removed. Since all algorithms are used in this paper is based on the bag-of-words, all sentence are parsed as a number of tokens and save in the file.

5. IMPLEMENTATION

We implemented and tested DMM, a vanilla LDA model, and the STM. DMM is has a very similar to LDA in logistics and implementation. In LDA, there are two Dirichlet distributions, whereas in DMM there is only one Dirichlet distribution. Gibbs Sampling is used in both models.

Sentiment-Topic model is built based on the LDA. First, all data was classified by the trained Sentiment Analysis algorithm which is based on the Bayesian Classifier. Then, the LDA was combined with Author model simply change the probabilistic model from the standard LDA model to : .

Since Hong et al. (2010) claimed that the DMM model has better performance on short text topic modeling than LDA does, our first comparison was between the DMM and vanilla LDA models. And the result shows very similar behavior in the topic descriptions with many overlapping words. However, in the topic allocation, since the text in the LDA model could have several topics allocated, it is more flexible than DMM which has a single topic in each text. Therefore, if one topic is wrong, there are another backups in the LDA model.

The second comparison was made between the an LDA using a TF-IDF matrix and an LDA using a traditional bag-of-words structure. The results show that LDA with TF-IDF tends to pick some common words as the topic words, which makes the topic modeling ultimately less meaningful. However, LDA with the traditional bag-of-words was better at picking distinct words to form topics.

The third comparison was between an LDA and the Sentiment-Topic model. It turns out that when using LDA, some topic words are adjectives that don't offer much information. In the STM, the neutral sentiment topics tends to have a fewer uninformative adjectives. On the other hand, the positive and negative sentiment topics have an abundance of adjectives, offering no context or insight. Note that the neutral class is much larger than the positive and negative classes.

A biterm model and Latent Features with LDA were also tested. The biterm model did not give us good results. As for the latent feature model, we had to use pre-trained vectors due to time and computational constraints. The model with pre-trained vectors yielded poor results as well. We won't be presenting results from these two models.

All code can be found here on github.

6. Experimental Examples

We present some examples of topic results from different models that were tested. We see that the topics for the positive and negative sentiment classes are more adjective rich than the neutral sentiment class. The topics are qualitatively analyzed based on the similarity and relatedness of the words in each topic.

TABLE 1. Dirichlet Multinomial Mixture Model

Topic 6	
trump	0.01444
it	0.007589
u	0.007131
amp	0.006825
realdonaldtrump	0.004813
like	0.004712
people	0.00461
get	0.004151
dont	0.003846
one	0.003464
election	0.003234
time	0.003158
news	0.003158
think	0.003133
know	0.003031
russia	0.002878
say	0.002852
russian	0.002725
right	0.002623
new	0.002572

Topic 14	
wind	0.009808
humidity	0.008328
museum	0.005553
pressure	0.004998
amp	0.004813
art	0.004813
today	0.004627
in	0.004627
gt	0.004627
brooklyn	0.003887
nyc	0.003702
one	0.003332
great	0.003332
53	0.003332
moma	0.003332
night	0.003147
last	0.003147
im	0.002962
open	0.002962
13mph	0.002962

TABLE 2. Vanilla LDA Model using Bag-of-Words

Topic 7	
trump	0.034834
realdonaldtrump	0.011803
u	0.010714
election	0.008193
it	0.007162
russia	0.006818
vote	0.006474
russian	0.006417
news	0.006303
putin	0.006188
white	0.005672
via	0.005615
amp	0.004985
people	0.004527
country	0.004527
hillary	0.004527
state	0.00424
donald	0.00424
medium	0.004183
foxnews	0.004183

Topic 14	
job	0.102181
hiring	0.067385
ny	0.066834
careerarc	0.038766
newyork	0.033197
latest	0.027572
work	0.027076
were	0.025091
click	0.021617
great	0.020734
opening	0.019301
want	0.017757
fit	0.017646
anyone	0.016543
here	0.015496
see	0.01522
apply	0.014944
recommend	0.014669
might	0.010478
hospitality	0.010092

TABLE 3. Negative Sentiment-Topic Model

Topic 3	
trump	0.085232
realdonaldtrump	0.049748
cold	0.028458
weird	0.028458
1	0.028458
attacked	0.028458
vanity	0.028458
fair	0.028458
weather	0.021361
bad	0.021361
starving	0.021361
guess	0.021361
shame	0.021361
seeing	0.014264
make	0.014264
insecure	0.014264
gave	0.014264
restaurant	0.014264
review	0.014264
mad	0.007168

Topic 5	
mad	0.107042
no	0.062992
oh	0.056699
bullshit	0.050406
alone	0.03782
w	0.025234
tragedy	0.018942
leave	0.018942
hard	0.012649
wait	0.012649
washingtonpost	0.012649
honestly	0.012649
shes	0.012649
bap	0.012649
moody	0.012649
blocked	0.012649
still	0.012649
video	0.012649
yeezys	0.012649
ig	0.012649

TABLE 4. Neutral Sentiment-Topic Model

Topic 7	
trump	0.039644
realdonaldtrump	0.0151
u	0.011136
election	0.008337
russia	0.006763
russian	0.00653
putin	0.006239
vote	0.005947
vanityfair	0.005889
way	0.005656
news	0.005422
it	0.004956
america	0.004839
president	0.004839
donald	0.004665
hillary	0.004606
house	0.00449
white	0.004431
country	0.004373
people	0.004256

Topic 2	
it	0.050397
cold	0.032743
today	0.016702
like	0.01487
outside	0.014065
snow	0.013991
winter	0.012233
weather	0.011208
im	0.009743
day	0.009743
nyc	0.009523
degree	0.00923
snowing	0.008278
got	0.007985
feel	0.007179
morning	0.007179
wind	0.007106
right	0.006886
warm	0.006813
amp	0.0063

TABLE 5. Positive Sentiment-Topic Model

Topic 8	
thanks	0.167775
love	0.150999
much	0.050347
haha	0.027281
appreciate	0.023087
sharing	0.018893
glad	0.016796
thats	0.012602
dude	0.012602
amazing	0.010506
awesome	0.010506
sunshine	0.010506
fucking	0.008409
liked	0.008409
yall	0.008409
boo	0.006312
adventure	0.006312
school	0.006312
ya	0.004215
support	0.004215

Topic 13	
lmao	0.071635
god	0.062395
wow	0.048534
cute	0.048534
oh	0.043914
super	0.043914
omg	0.032364
bless	0.025434
worth	0.013883
never	0.013883
people	0.013883
best	0.013883
dress	0.011573
mario	0.011573
face	0.009263
boy	0.009263
call	0.006953
tho	0.006953
smart	0.006953
hilarious	0.006953

7. Discussion and Assessment

We tested the following models with our dataset: Biterm, Latent Feature with LDA, LDA, DMM, and Sentiment-Topic model. Among them, Biterm focuses more on bigram words which make the topic not meaningful for the users. LDA with TF-IDF has been proved to work well in the context of large documents. But for our dataset of short text, it doesn't prove useful. Since tweets are only 140 characters long, the term-frequency is rarely more than 1 for any word in a tweet. Therefore, the usefulness of TF-IDF in emphasizing rare words and diminishing the weight of frequent words is lost for our use-case.

As previously mentioned, we used pre-trained word embeddings for the LDA model with latent features. Ideally, LDA with latent features should perform better (Nguyen et al., 2015). If the word embeddings are trained on the relevant corpus, these embeddings will contain more useful information about things like similarity than a sparse one-hot vector. However, the training of word embeddings is a time consuming process and needs to be redone anytime the corpus changes. The alternative is to use pre-trained embeddings. This is what we did. The disadvantage here is clear however. Using precomputed information forces model to ignore unseen words from the test data. Additionally, pre-trained information tends to have hidden bias, which will lead to higher error. This is precisely what happened in our model. Using the pre-trained embedding in this model yielded poor results.

DMM works equally as well as LDA in our experiment. One strict advantage it has over LDA is the ease of its implementation. Additionally, this model is faster than LDA which is a great advantage when dealing with such large amounts of data.

The sentiment-topic model works better than LDA on such social media text, even if the most significant component of the model is the LDA itself. The experimental result show that some of the topics are still a little vague. There are two reasons that we speculate. One is that the data used in the experiment is not separated fairly, which means neutral text takes a huge part of the whole dataset. Thus, the topics hidden behind the positive and negative classes would be easily ignored by the model. Another reason is that the data set is still small. As we are keenly aware, the more data is used, the better a model will perform and the more expressive the topic model will be. Meanwhile, the result shows that for the positive part and negative sentiment classes, the topic modeling is a failure. Almost all topic words are adjectives. But the favourable repercussions are that the Sentiment-Topic model removes lots of uninformative adjectives from the topics for the neutral sentiment class. Thus, the topics in the neutral sentiment class a more meaningful and specific. Since the neutral sentiment data constitutes

a large part of all the data (over 93%), it's possible that we're sacrificing a relatively small part of data for improved results.

8. Conclusion

Having investigated a few variations of LDA models for topic modeling on Twitter data, we find that our Sentiment-Topic model performs well in some regards, and poorly in others. The STM yields better results than a vanilla-LDA and the DMM model when only considering the neutral sentiment class. Since the neutral sentiment data constitutes the bulk of the data, we believe that some data loss is worth the improves results we see in topic modeling. For future work, it could be interesting to elaborate this idea and apply the author-topic modeling framework to features such as locational information, and combine such models with the Sentiment-Topic model.

References

[1] Rosen-Zvi, M., Griffiths, T., Steyvers, M., and Smith, P. 2004. Theauthor-topic model for authors and documents. In *Proceedings of the 20th Conference on Uncertainty in Artificial Intelligence*, pages 487-494. AUAI Press.

[2] C. Wang, D. Blei, and L. Fei-Fei. 2009. Simultaneous image classification and annotation. *Proc. CVPR*.

[3] K. Nigam, A. McCallum, S. Thrun, and T. M. Mitchell. 2000. Text classification from labeled and unlabeled documents using em. *Machine Learning*, pages 103-134.

[4] Blei, David. 2012. Probabilistic topic models. In *Communications of the ACM 55(4)*, pages 77-84.

[5] Pak, A., and Paroubek, P. 2010. Twitter as a corpus for sentiment analysis and opinion mining. In *Proc. of LREC*.

[6] Alec Go, Richa Bhayani, and Lei Huang. 2009. Twitter sentiment classification using distant supervision. *Technical report*, Stanford.

[7] Apoorv Agarwal, Boyi Xie, Ilia Vovsha, Owen Rambow, and Rebecca Passonneau. 2011. Sentiment analysis of Twitter data. In *Proceedings of the Workshop on Language in Social Media*, pages 30-38.

[8] Hong, L., and Davison, B. D. 2010. Empirical study of topic modeling in twitter. In *Proc. First Workshop on Social Media Analytics*, SOMA 10, pages 80-88. NYC, NY, USA: ACM.

[9] D. Q. Nguyen, R. Billingsley, L. Du, and M. Johnson. Improving topic models with latent feature word representations. *TACL.*