Weekly Report

Zhuqi Li

Abstract

This week I still work on tweet-topic matching. In this report, I will include several tweet-topic matching methods I have come up with and the evaluation to them.

Introduction

If we want to determine the attitude of people in a neighborhood to a certain living metric (e.g. transportation, noise), we must know which tweet is related to this metric (the topics of tweets). So this work serves as the basis of our mining work in the later. But this work is not limited to urban neighborhood mining, it can be applied to any short text mining task. However the characteristics of tweets have posed several challenges to us.

- The large volume of tweets makes it too laborious to label all the tweet data. So it is hard to apply any supervised algorithm to this problem.
- The short length of tweets makes their feature vectors very sparse. So classical topic model cannot be applied to this problem directly.

Every coin has two sides. Since the tweets are usually short, their topics heavily rely on several keywords. If we can understand these keywords and the co-occurrence pattern among them, we can estimate the topic of the tweet. Based on this observation, we first study the similarity between keywords and the specific topic and then try to use the word level similarity to estimate the document level similarity.

This report will be organized as follow. We will first briefly introduce the background of Word2vec (Mikolov and Dean 2013; Goldberg and Levy 2014) and Doc2vec (Le and Mikolov 2014). Then we present a method that is based on word level similarity to address this problem. We will also give two methods that is based on sentence level similarity. Then we study how to identify the subtypes for a certain topic. At last, we give our evaluation to all the methods above.

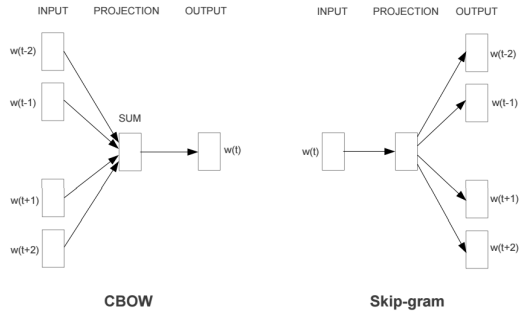


Figure 1: Architecture for CBOW and SG

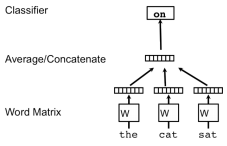


Figure 2: Word2vec architecture.

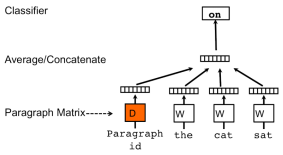


Figure 3: Doc2vec architecture.

Word2vec and Doc2vec

Word2vec

Word2vec is a model that produces word embeddings (Mikolov and Dean 2013; Goldberg and Levy 2014). It can map a word to a vector space based on its context in the document. The model is combined by two sub-models, one is continuous bag-of-word model (CBOW) and the other is skip grams model (SG). CBOW model is used to predict the current word by using a window of surrounding context words. While SG model is used to predict a window of surrounding context words by using the current word. The difference between these two models can be clearly observed from Figure 1. By minimizing the loss function of prediction, these two models generate the vector representation for every word at the same time. Then Word2vec average the two vector representations generated by CBOW and SG and get a new vector for every word.

The vector representation generated by Word2vec can be used to estimate the similarity between words, since words

Iteration	Keywords
0	traffic, transport, bus
1	intersection, shuttle, cab, pedestrian, transit, tunnel, authority, vehicle, passenger, plow
2	pedestrian, port, lyft, passenger, uber, transit, highway, megabus, vehicle, suv
3	pedestrian, passenger, taxi, 71d, suv, transit, tractor, bumper, truckload, motorcycle
4	passenger, pedestrian, truckload, hazmat, tractor, suv, req, bumper, transit, motorcycle

Table 1: A demo for keywords selection.

with similar meaning tending to have similar context. For example, in the following two sentences,

"I love the wine in that bar."

"I love the wine in that cafe.",

The word *bar* and *cafe* have similar context, so there similarity will be high if we only consider these two sentences.

Doc2vec

After getting familiar with Word2vec, we introduce the document level embedding method **Doc2vec** (Le and Mikolov 2014). Doc2vec is based on Word2vec. It first preprocesses the data by adding an extra token for every document. Then It use Word2vec to compute the vector representation for every word (including the extra tokens). Since every document has an exclusive token and the token only appears in the corresponding document, the vector representation of the token is also the vector representation of the document. Figure 2 and Figure 3 have clearly demonstrate the difference between Word2vec and Doc2vec.

After obtaining the document level vector representation, we can compute the similarity between different documents, which is helpful for us to determine whether a tweet is related to a certain topic.

Word similarity based method

In this section, we introduce a word similarity based method to tweet-topic match problem.

Tweet preprocessing

The preprocessing involves the following three steps.

- Filtering auto-tweet. This step utilize the hashtag of tweet to remove the tweets about automatic check-in and hiring advertisements.
- Removing stop words. This step will prevent some common words with high frequency from overwhelming the important words.
- Lemmatize the rest of words. Instead of stemming the words, we use the lemmatizer to preprocess the words, which can preserve the part-of-speech. After this step, we can get a set of term for every sentence.

Word-level similarity based on Word2vec

After preprocessing the tweets, we can get the a bag of terms for every sentence. Then we use use Word2vec to train the model. We input the bag of terms of all the sentences to the model, then the model can output a vector for every single word. For every pair of words, we can use the cosine similarity between their vector representations to indicae their similarity.

Keywords selection

In this section, we introduce how we select the topic related keywords. The method starts from several topic related words, we try to find the most similar words to the keywords (those with the highest cosine similarity to the keywords) in every iteration. If these words are related to the topic, we add them to the set of keywords. After several iterations, no more topic related words can be found.

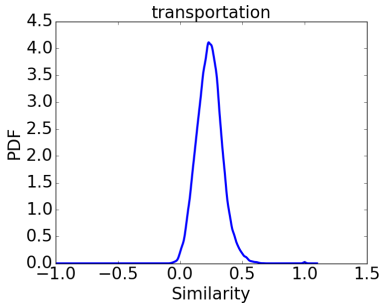
Table 1 has demonstrated this process. We start from a set of keywords *traffic*, *transport* and *bus*, and use this set of keywords to find the most similar words to them. Then we find that *intersection*, *shuttle*, *cab*, *tunnel* and *authority* is also related to the topic of transportation. So we add them to the set of keywords and start the next iteration. After four iterations, no more words should be added to the set of keywords, then we stop the iteration.

Word-level relevance

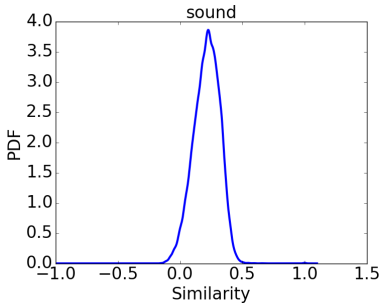
After obtaining the set of keywords of a certain topic, we use the similarity of a word to the set of keywords to denote the its relevance to the topic. we plot the relevance distribution in Figure 4. An interesting fact we can find from the figure is that the relevance distributions are roughly bell-shaped. In order to verify whether the distribution obey the normal distribution, we use Quantile-Quantile Plot (QQ plot) to show the similarity between the normal distribution and the relevance distribution. From Figure 5, we can see that the relevance distribution is congruent with the normal distribution. The R-square values of the relevance distribution for the seven topics are around 0.995, which is a strong indication that the words' relevance to a topic obey the normal distribution. Currently, I have not come up with the mathematical explanation for this fact, so we just use this characteristic to help us determine whether a tweet is related to a certain topic.

Topic related tweet filtering

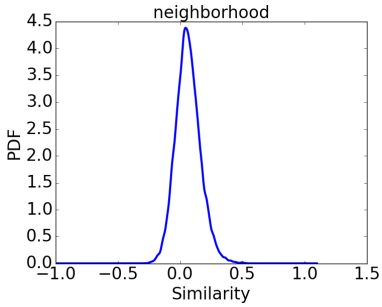
In this section, we introduce how to filter the topic related tweet, which require computing the tweet-level similarity. The intuitive methods to compute the tweet-level relevance are (1) summating the relevance of all the words up, (2) computing the average relevance value of the words in a tweet. But these two methods do not work. For the first method, the long tweet which is not relevant to the topic may also



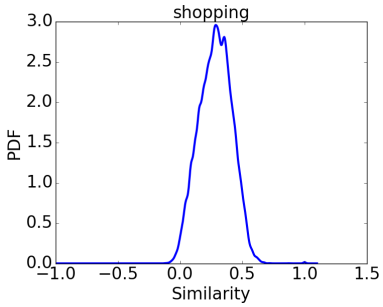
(a) Transportation.



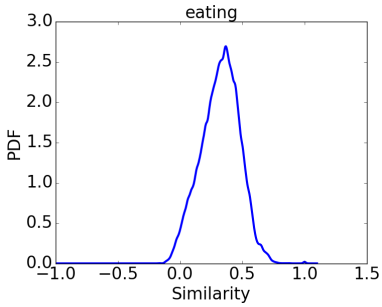
(b) Sound.



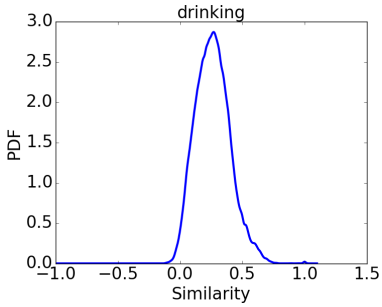
(c) Neighborhood.



(d) Shopping.

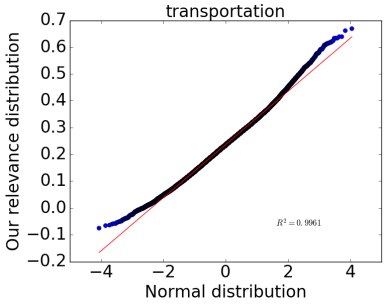


(e) Eating.

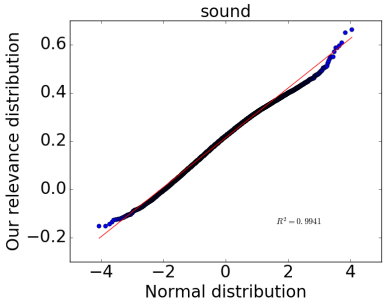


(f) Drinking

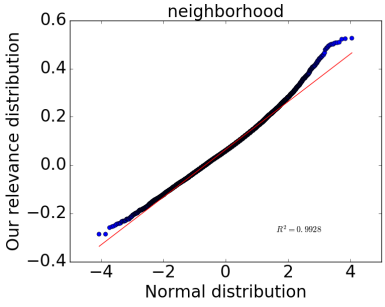
Figure 4: The relevance distribution to different topics.



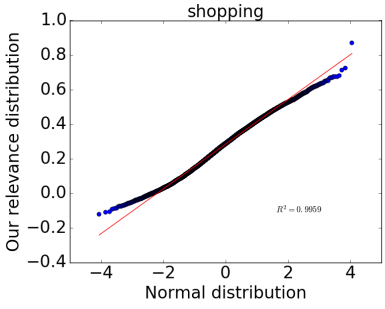
(a) Transportation.



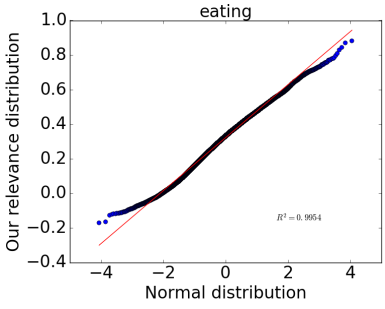
(b) Sound.



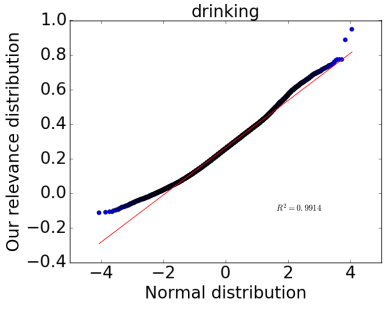
(c) Neighborhood.



(d) Shopping.



(e) Eating.



(f) Drinking

Figure 5: The QQ-plot for relevance distribution to different topics.

achieve high score because of their length. For the second method, the relevance of long tweet may be underestimated because some irrelevant words may harm the whole tweet.

Another way to make up is normalization. We can subtract the average relevance from the relevance of a word before summating them up. But this method will also incur two kinds of false positive cases. We use two examples to demonstrate the two kinds of false positive cases. In the two examples, the terms have been labeled in red and the number below the term denotes its relevance to the topic.

Then we look at the first kind of false positive. The following tweet is related to hiring not to transportation. However, the relevance of words in this tweets is above the the average (0.236), because of their co-occurrence with topic related words in the other tweets. But this kind of tweets often lack the high relevance word (the most relevant word of this sentence only has a relevance of 0.525). So we can set a threshold for the highest relevance of word in a tweet. If the highest relevance is below the threshold, it indicates that the tweet may be not actually relevant to the topic. By this way, we can remove this false positive case.

Speedway Foodservice Specialist #Whitehall, PA

0.451 0.511 0.442 0.525 0.276

Then we look at the second kind of false positive. From the following tweet we can see that there is a topic related keyword (uber) in this tweet, but the tweet is not actually related to the topic. However, if we use the normalized summation to denote the relevance of the tweet, we can get the relevance is 0.922, which is higher than a single-word tweet "traffic" (the relevance is 1-0.236=0.764).

My uber driver is playing creepy flute

1.0 0.527 0.069 0.173 0.223

solos and has a doll in the backseat

0.259 0.032 0.527

The case above shows the fact that even we subtract the average relevance from the relevance of the word, the total relevance of the tweet is still affected by the length of the tweet. In order to get rid of the effect of the length, we should choose a more proper subtraction factor D than average relevance. Choosing the subtraction factor is a tradeoff. If the subtraction factor is too large, some long tweets related to the topic will have very low relevance, which lead to a higher false negative rate. If the subtraction factor is too small, some long tweets that is no related to the topic may have very high relevance, which lead to a higher false positive rate. In the following part, we introduce how to determine the proper subtraction factor.

We consider two categories of words. The first category is the keywords to the topic, the second category is the words other than keywords. The relevance of the words in the first category is a constant 1, while the relevance of the words in the second category obey the normal distribution $x_2 \sim N(\mu, \sigma^2)$. Suppose a tweet contain n_1 words in the first category and n_2 words in the second category. Suppose the relevance to the topic of words in a tweet is a random variable of normal distribution. The summation of the relevance of all the words in the tweet also obeys the normal distribution, because

- the result of summating random variables of normal distribution is also random variable of another normal distribution,
- the summation of random variables of normal distribution and the constant random variable is also random variable of normal distribution.

The expectation of the summation of words' relevance is $\mu' = n_1 + n_2\mu$. The standard variance of the summation of words' relevance is $\sigma' = \sqrt{n_2}\sigma$. We set one σ' to be the threshold to determine whether the tweet is related to the topic. Then the subtraction factor D could be written as

$$D = \frac{n_1 + n_2\mu + \sqrt{n_2}\sigma}{(n_1 + n_2)}.$$

After obtaining a proper subtraction factor D, we compute the relevance of a tweet by $\sum_{i=1}^{n}(r_i - D)$, where r_i is the relevance of the i'th term word of the tweet. Then we set 0 as the threshold for the tweet-level relevance to determine whether a tweet is related to a certain topic.

Sentence similarity based methods

After introducing the word-level similarity based method, we continue to present two tweet-level similarity based method.

Keywords reference

In the previous section, we have introduced the Doc2vec to generate the document level embedding. The **keywords reference** is an intuitive method that utilize the document level embedding to determine whether a tweet is related to a certain topic.

The tweet preprocessing and keywords selection method is the same as the word similarity based method. After selecting a set of keywords, we construct a virtual tweet that only contains the topic related keywords. Then we use the Doc2vec to compute the vector embedding for every tweet. The relevance of a tweet to a topic can be computed by the cosine similarity between it and the virtual tweet. Last we should set a threshold for the similarity to decide whether a tweet is related to the topic.

Sampling reference

The simple keywords reference method has some clear drawbacks. (1) By concatenating all the topic related tweet into a virtual tweet, it actually undermines the co-occurrence pattern between some certain keywords (e.g. port and authority). (2) Most topic related tweets only contain several keywords, while the virtual tweet contains all the keywords. So the similarity between most of the tweets and the virtual tweet are pretty low, which makes it harder to differentiate the positive case from the negative case.

Based on the observation above, we propose the **sampling reference** method. In this method, we construct multiple virtual tweets which serve as reference. Each tweet is composed of a few keywords and several other words with high

relevance. The keywords are sampled from the set of keywords based on their co-occurrence frequency. Several other words are sampled from the set of high relevance words based on their frequency. After getting the set of reference tweets by sampling, we train the Doc2vec model with the original data as well as our virtual tweets. After training, we can get the vector representation for both the original tweets and virtual tweets. Then we use the polling method to compute relevance of a tweet to the topic. For each virtual tweet, it can vote for several original tweets that have high similarity to it. Then we rank the original tweets based on the polling result. Last we should set a threshold for the similarity to decide whether a tweet is related to the topic.

Sub-topic identification

Sometime people are not content with knowing only the general topic, they also want to know some fine-grained subtopics related to the general topic. For example, for the topic *transportation*, some people who always drive their cars care more about parking, while the other people who prefer public transportation care more about bus routes. So in this section, we introduce a sub-topic identification method to address this problem.

In the previous section, we have introduced how to get the vector representation for every tweet and filter the topic related tweets. The sub-topic identification method leverages some the previous results. The method first get the vector representation for all the tweets related to a certain topic. Then we use a **spherical k-means clustering** method to cluster the tweets into several different clusters. Then, we compute the frequency of a word in different clusters. If a word tends to appear only in a certain cluster, then it will be selected as the feature words for this cluster. Otherwise the word will not get considered. At last, the feature words in a certain cluster will be considered as the sub-topic.

(Notes: the sub-topic identification origins from a failed clustering experiment. Previously, I wanted to use the clustering method to filter the tweets related to a general topic (e.g. transportation). I have run the clustering algorithm for all the tweets, hoping to find a cluster that contains most of transportation related tweets. But the experiments failed. Then I figured out that the topic *transportation* is actually the combination of several small clusters, but the small clusters are actually not very close to each other. This fact motivates me to come up with this sub-topic identification problem.)

Evaluation

In this section, we evaluate the performance of our method by several experiments.

Tweet-topic matching

In this section, we compare the performance of the three tweet-topic matching method, (i.e. the Word-level similarity based method, Keywords reference method and sampling reference method). For topics *Transportation*, *Sound*, *Neighbor*, *Shopping*, *Eating* and *Drinking*, we get about 2700, 1000, 600, 3700, 2100, and 7200 topic related tweets respectively. In order to estimate the false positive rate, we manually check 100 tweets for every method and every topics. The false positive rate of different algorithms to different topics was plotted in Figure 6. From the Figure, we can clearly see that the false positive rate of Word-level similarity based method and sampling reference method is about 15%, which is better than the paper in ICWSM 16 (Guha et al. 2016) (from 21% to around 50%).

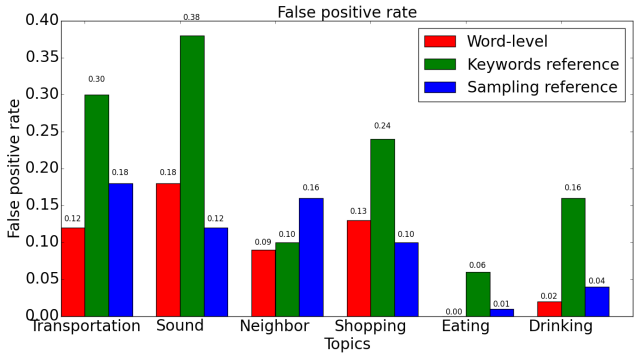


Figure 6: False positive rate of the three methods

For the false negative, we also random selected 100 tweets that not in the set of topic-related tweets and checked the positive case in this 100 tweets. However, we only found a tweet related to the topic $Drinking$. So it's hard to measure the actual false negative rate. Because we have about 1.8 million tweets, but only thousands of them are related to a certain topic, which make the false negative case too sparse to measure. The paper in ICWSM 16 (Guha et al. 2016) also does not measure the false negative result probably because of this reason.

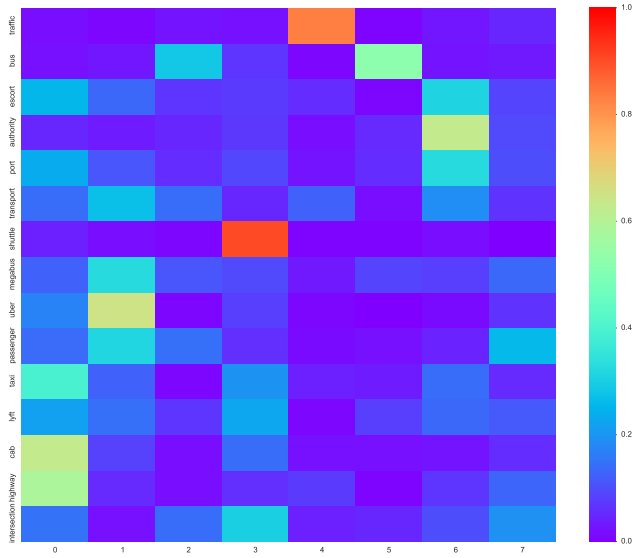


Figure 7: Correlation between keywords and clusters

Sub-topic identification

Then we show the experiment result for our sub-topic identification method. The sub-topics are denoted by a bag of keywords, which mainly appear in a certain cluster. We present the result from two aspects. The first is the correlation between keywords and clusters. The second is the correlation between keywords and keywords.

We run the algorithm for tweets related to topic *transportation* on eight clusters. The correlation between keywords and clusters are show in Figure 7. The horizontal axis represents the eight clusters and the vertical axis represents the selected keywords. The color of a grid stands for the portion of a keyword appear in a certain cluster. If the portion approaches 1, the color tends to be red. If the portion approaches 0, the color tends to be blue. From the figure we can see some interesting facts about the keywords' distribution on clusters. The tweets about the taxi and uber mainly lie in the cluster 0 and cluster 1. The cluster 3 is mainly about the shuttle, the cluster 4 is filled with tweets complaining the traffic and the cluster 5 and 6 is related to public transportation. Based on this observation, we can divide the topic *transportation* in to 4 sub-topics.

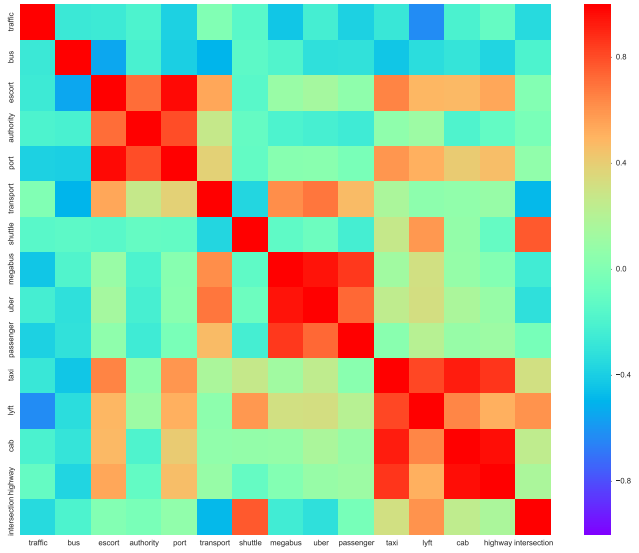


Figure 8: Correlation between keywords and keywords

We also study the correlation between keywords. We compute the distribution of keywords on different clusters. For a pair of keywords, we compute the correlation coefficient between them and plot and result in Figure 8. The color of a grid stands for correlation coefficient between two keywords, if the correlation coefficient tends to be 1, the color will approach red. If correlation coefficient tends to be -1, the color will approach blue. From the figure, we can clearly see that the keywords have formed several groups based on their possibility co-occurrence in different clusters. From the result, we can know that cluster result indeed reflect the word frequency and co-occurrence pattern between different words.

Future work

There are two possible directions to further this study. The first is to refine the topic-tweet matching. The second is to utilize current matching result to analyze the people's opinion to metrics in urban neighbourhood.

For the first direction. I should work on the following things:

- try to find the possible explanation to the normal distribution of relevance of words to a topic,
- try to improve the performance of the algorithms (maybe apply some tricks to it).
- try to run our algorithm in the dataset of the ICWSM 16 (Guha et al. 2016) paper to see the performance of our algorithm in their dataset.
- try to systematize our method with some mathematical formulas.

For the second direction, the major work lies on the visualization. I should work on the following things:

- find more interesting topics about the city (ask the people around me or use online questionnaire)
- visualize different metrics to the city. (use the visualization techniques of the former ICWSM papers)
- study the correlation of tweets from topic domain, time domain, location domain and sentiment domain. (Inter-domain mining)
- try some interesting applications based on our current result (e.g. reconstruct the bus routes based on the online geo-tagged tweets)

References

Goldberg, Y., and Levy, O. 2014. word2vec explained: deriving mikolov et al.'s negative-sampling word-embedding method. *arXiv preprint arXiv:1402.3722*.

Guha, S.; Chakraborty, T.; Datta, S.; Kumar, M.; and Varma, V. 2016. Tweetgrep: Weakly supervised joint retrieval and sentiment analysis of topical tweets. In *Tenth International AAAI Conference on Web and Social Media*.

Le, Q. V., and Mikolov, T. 2014. Distributed representations of sentences and documents. In *ICML*, volume 14, 1188–1196.

Mikolov, T., and Dean, J. 2013. Distributed representations of words and phrases and their compositionality. *Advances in neural information processing systems*.