Topic Detection and Dissemination Trend Analysis on Social Network*

Jiadong Chen[1], Tianxiang Gao[1], Xiaofeng Gao†[1], Peng He[2], Jinning Li[1], Guihai Chen[1]
[1]Shanghai Jiao Tong University, Shanghai, 200240, China
[2]Tencent, Shenzhen, China
{chenjiadong998, gtx9726, lijinning}@sjtu.edu.cn, {gao-xf, gchen}@cs.sjtu.edu.cn, paulhe@tencent.com

Abstract
Social network has been a hot research field for years and the topic detection and dissemination trend analysis have attracted much attention in last decade. There are lots of researches discussing topic detection and tracking problem using the traditional way, which based on network structures, users' profile, and statistical models. The performances of these algorithms based on traditional ideas seem to touch the ceiling. Recently, machine learning and neural network style algorithms show superior performances in NLP and related research fields. In this paper, We propose a scheme to do the topic detection and tracking tasks. This scheme is based on neural network style NLP techniques and aims to transfer the documents generated from the social network into high-dimension vectors. The first step is to get well trained distributed representations of Chinese words through the classical CBOW algorithm with the special adaptations for Chinese. Then, the second step is to get the paragraph vector for any document using the Chinese word vectors. Tasks of topic detection and tracking can then be solved well by clustering the data points. The algorithm of clustering used in this paper is also designed under the circumstances that large amount of data cannot be loaded into memory. We also implement sufficient experiments using the real data from Tencent, Baidu, and Weibo. The performances of experiments show the accuracy and efficiency of the scheme proposed. Some visualizations and case studies are covered to show the applications of the scheme.

1 Introduction

Topic detection and tracking problem, TDT problem, has been running for decades [1, 2]. In recent years, together with the rapid development of machine learning and natural language processing, the methods for topic detection have pivoted to machine learning based. The most original method is to set up a few keywords and check whether the frequency of certain keywords has a burst and then let people to check if the corresponding keyword related to an uprising topic. Recently, word representations in vector space [3–5] has given this task a new way to do. Once we get the word representations in vector space, then we can get the representations of a sentence, then a paragraph, and finally a document. We should notice that such representations carry semantic meanings and embed the topic into a high dimensional vector. As a result, we can transfer a text-based language problem to a vector-based mathematics problem, which is easy and friendly to computers and the mature machine learning algorithms.

In this paper, we focus on the text generated from the social networks. we regard these pieces of texts as documents no matter what form they have. A twitter, a blog, or even sentences are treated as documents. The reason why we do this is to make sure the compatibility and universality of the method. After having the vector representations of documents, the next step is to discover the topics hidden among the documents and their corresponding vectors. This task is obviously an unsupervised problem because we do not have any labels. So, lots of researchers introduce K-means [6] to solve this problem and regard the stable centroids as the topics. However, K-means has some problems. In this paper, we present a novel topic detection scheme to accelerate the processing procedure for large amount of documents and improve the accuracy and robustness of detection and text classification used in social network analysis. First of all, the proposed method utilizes a neural network model [3] to generate the word representations of each word. This procedure is described for English texts and for Chinese texts, the training method is the same but we have to add another step, segmentation. These low dimensional, compared to one-hot representations, but

*This work is supported by the National Key R&D Program of China (2018YFB1004703), the National Natural Science Foundation of China (61872238, 61672353), the Shanghai Science and Technology Fund (17510740200), the CCF-Tencent Open Research Fund (RAGR20170114), and the Huawei Innovation Research Program (HO2018085286).

†Corresponding author.

informative vector representations of words allow us to discover semantic similarities between words and give us a convenient tool which is the linear combination of word vectors to yield good and accurate results. Lots of previous work [7, 8] have showed that the vector-based representations can accurately calculate semantic relatedness between textual units of uncertain lengths, such as sentences and documents. Although the original continuous bag-of-words, that a document is regarded as a vector of words which the documents contains, has been utilized in various natural language processing tasks, document vectors as a whole takes the order of words and sentences into account and show superior performance compared to the simple addition of word vectors.

2 Related Work

2.1 Topic Detection on Social Networks There are lots of previous good work on topic detection field. [9] aimed to solve the topic search problem and this paper proposed a special data structure called nodes structure and did an improvement of vanilla K-means by approximate random clustering. [10] focused on real-time event detection, which is also topic detection, from social media. Method based on hybrid link prediction and quantum swarm intelligent was proposed in [11]. They also showed a new standard to compare different method and they called it impact value. [12] utilized the ternary classification based method to solve sentiment related topic detection problem and the authors put more weight on the recall number and they also get a good results. Another ACL paper [13] proposed probabilistic model to extract events and topics from the huge corpus. They presented unsupervised latent variable model called the Latent Event Extraction and Visualization (LEEV) as a whole solution for this task. [14] proposed a new way to define topic itself and defined personal wellness events. They did not do any clustering, but fond the hit of keywords wellness.

2.2 Topic Dissemination Trend Analysis Topic tracking has been an active research area in the field of information retrieval, data mining and social networking since the pilot study of Topic Detection and Tracking research [1]. In the field of information retrieval, many related studies [15, 16] have been completed. They mainly use relevant models, text filtering, text categorization, etc. to track related topics. [17] used the non-homogeneous Poisson process model to track the change trend of microblogging discourse, and the heat factor and trend factor are introduced to realize the real-time detection and trend tracking of the burst topic. Topics Over Time (TOT) [18] used time information as observation variables in the topic modeling process to complete the document generation process and affect the topic distribution. A Post scatter analysis method [19] is proposed to divide the document into different time slices after the topic modeling process.

PET [20] studied the interaction between the text topics and the network structure. Because PET considers text files and network structures, it can better track the evolution of event content and events. However, it ignores the selection phenomenon and cannot track multiple topics at the same time. The selection phenomenon is characterized by past users' interest affecting the current network structure, which is very important to the formation of network links. So, after we get the distributed representation of the documents and the feature vectors of centroids, we can do better in the topic dissemination trend analysis.

3 Method of Topic Detection

In this chapter, we introduce the whole details of the scheme we use to do topic detection. This chapter includes how to get the basic distributed representations of words for Chinese terms, then the sentence vectors, and finally the vector space representations of documents. After getting the distributed representations of documents, we can then cluster the vectors and get the topics hidden underneath these huge amount of text data.

3.1 Word Representation in Vector Space (Word2Vec) Several methods for representing word meanings using mathematical expressions such as vectors and matrices have been proposed. The neural network model has recently gained considerable attention [3, 4, 21, 22]. Neural network models are usually completely parameterise word vectors; in other words, each word w has n parameters in its word vector: $v(w) = (x_{w_1}, x_{w_2}, ..., x_{w_n})$. These parameters are used to estimate the conditional probability that the target word will appear given the contextual word (CBOW model). Each word's parameters are initialized to a random value and then adjusted during the learning process with the goal of maximizing the conditional probability:

$$P(o|c) = P(v_t|w_{t-N}, w_{t-N+1}, \ldots, w_{t+N-1}, w_{t+N}) \tag{3.1}$$

In the equation above, o represent output word, which is the one we want estimate and c means contextual (CBOW model) or center (Skip-gram model) word which is the input to the system. Among them, v_t is the output or target word and $w_{t-N}, \ldots, w_{t+N}$ are the $2N$ context words that occur before and after the v_t. In the learning process, words that may appear in similar context or have similar semantic meaning appear closely to each other in vector space. This word vector has proved to be useful in many NLP tasks.

3.1.1 Model Architecture – Skip-gram model

Prior to the emergence of Word2Vec, neural networks DNNs [23] have been used to process word-to-word relationships with training word vectors. The method used is generally a three-layer neural network structure (of course, it can also be multi-layered) and is divided into an input layer, a hidden layer, and an output layer (softmax layer). How does this model define data input and output? Generally divided into CBOW (Continuous Bag-of-Words) and Skip-gram two models.

When a given center word is given, the skip-gram model tries to maximize the likelihood of contextual words.

$$P(o|c) = \prod_{-N \le j \le N}^{j \ne 0} P(w_{t+j}|v_t) \tag{3.2}$$

In the model, every word has two different vector representations. The first one is for use when the word is the current word or center word. The other representation of a word is used when the word is the output word or contextual word. As a result, we define all parameters in this model in terms of one long vector theta. For the clarity, we use v for the center word vector representations and w for output vector uses.

$$\Theta = \begin{bmatrix} v_{aardvark} \\ \vdots \\ v_{zyzzyva} \\ w_{aardvark} \\ \vdots \\ w_{zyzzyva} \end{bmatrix}_{2V \times d} \tag{3.3}$$

We can see from Eqn. (3.3), all parameters of this model is summarized as a matrix with $2V$ rows and d columns, where V is the number of English vocabulary (around 50K) and the d is the dimension we want to use for the vector representations of words (usually hundreds). Aardvark the first word in the English vocabulary and zyzzyva is the last one in vocabulary. Since this paper focuses on the process of Chinese, we just explain the basic idea of Word2Vec here and the specific adaptation for Chinese text data is discussed in details in this chapter later.

Since the Word2Vec is based on neural network and the machine learning guys always want a cost function, we define the cost function as follows.

$$J'(\Theta) = \prod_{t=1}^{T} \prod_{-N \le j \le N}^{j \ne 0} P(w_{t+j}|v_t; \Theta) \tag{3.4}$$

When we do the training, we have a very large corpus and the center word shifts one by one across the corpus. T is the number of words in the corpus and it is usually billions or even trillions. Since the machine learning guys always want to minimize something and transfer the production into summarization, the actual cost function is defined in Eqn. (3.5).

$$J(\Theta) = -\frac{1}{T} \sum_{t=1}^{T} \sum_{-N \le j \le N}^{j \ne 0} \log P(w_{t+j}|v_t; \Theta) \tag{3.5}$$

However, there is something we still do not clarify, which is how to calculate $P(w_{t+j}|v_t)$. According to Mikolov [3, 4], we introduce dot production to do this.

$$P(w_o|v_c) = \frac{e^{w_o^T v_c}}{\sum_{k=1}^{V} e^{w_k^T v_c}} \tag{3.6}$$

This calculation is basically a softmax function and calculates the probability of every word in the vocabulary to be the right one for this blank with index o.

Once we have the cost function, we can do stochastic gradient descent (SGD). Since we have different two types of vector representations for each word, say center word vectors and contextual word vectors. Each step of gradient descent, we update the parameter matrix Θ. As a result, we have to do the partial derivative of cost function $J(\Theta)$ to each contextual word w and center word v.

$$\frac{\partial}{\partial v_c} \log P(w_o|v_c) = w_o - \sum_{x=1}^{V} P(w_x|v_c) w_x \tag{3.7}$$

The left part of the result is just the observation of contextual word w_o. On the right side, it is actually the expectation of all word vectors based on the current training status.

Now, let us find out the derivative of all the output words w_m (including w_o). In this scenario, we have to consider two different circumstances. The first one is the derivative of cost function to w_o and the second one is the normal one, which is the derivative of cost junction to all the other output words.

When $m = o$, $\frac{\partial}{\partial w_m} \log P(w_o|v_c) = v_c - P(w_o|v_c)v_c$.

When $m \ne o$, $\frac{\partial}{\partial w_m} \log P(w_o|v_c) = -P(w_m|v_c)v_c$.

We also can find some very interesting form from the deduction above. The contextual words derivative is so closely related to the center word, which makes so much sense. We update the whole parameter Θ at once.

Actually we train two set of vectors, one for center word and the other for contextual word. However, we want only one set of vectors to represent the word. Mean of the two vectors is quite good for the purpose. Also, we ignore one critical hyperparameter, which is the learning rate η used in the updating step.

$$\Theta^{new} = \Theta^{old} - \eta \frac{\partial}{\partial \Theta^{old}} J(\Theta) \tag{3.8}$$

3.1.2 Model Architecture – CBOW CBOW stands for continuous bag-of-words. The training input of the CBOW model is the word vector corresponding to the context-related words of a certain feature word, and the output is the word vector of this particular one word. Since the CBOW uses the word bag model, these eight words are all equal, that is, regardless of the distance between them and the words we care about, as long as they are within our context. In this case of our CBOW, our input is 8 word vectors, and the output is the softmax probability of all words (the goal of training is to expect the softmax of the specific sample of the training sample to have the highest probability), corresponding to the input layer of the CBOW neural network model. 8 neurons, the output layer has vocabulary size neurons.

The above is the process of how to use CBOW and Skip-Gram to train the model and obtain the word vector in the neural network language model. However, there are many differences between this and Word2Vec in the process of training models with CBOW and Skip-gram to get word vectors.

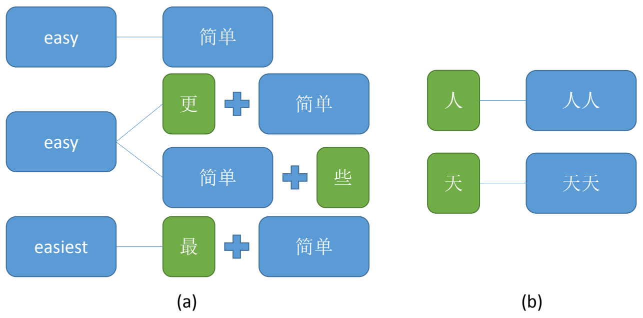

Figure 1: Examples of Chinese Lexical Knowledge

3.1.3 Adaptation for Chinese Text Data As we all know, the laws of language change greatly in different languages. For example, Chinese is a typical analytical language and lacks an inflection point. Fig. 1 shows that function words and reduplication are used to represent different syntax and semantic information. In addition, many semantic relations are closely related to social and cultural factors. Repeated words mean "every". However, basically no one has done similar reasoning in Chinese. The only thing we found in my undergraduate design process was a Chinese database [24] translated from an English database. Although this research and database have been widely used in many studies [25–27], it is no longer a reliable data set, because it contains only 134 Chinese-specific vocabulary, such as city names, provincial names, and so on. And language-related morphology knowledge has not been considered, and this is the most important part of Chinese.

Therefore, we want to go deep into some of the peculiar phenomena in Chinese. By modeling them by analogical reasoning, we can examine more deeply the ability of word vectors to detect Chinese semantics. As far as we know, we found a public training data set [28] and reimplemented it, which resulted in Chinese word vectors.

3.2 Distributed Representations of Sentences and Documents

We have got Chinese word vectors. Considering that our task is topic detection and tracking, it is an unsupervised problem and we have to deal with paragraph or documents which are not fixed length text data. Therefore, we need to transform the textual content that we have already got and the textual content are going to enter into the system into fixed length vectors. In this paper, we introduce the paragraph vector [5], which is an unsupervised framework that learns the continuous distribution vector representation of text fragments. Text can be variable, ranging from sentences to documents. The name Paragraph Vector is to emphasize the fact that this method can be applied to variable length piece of text, anything from a phrase or sentence to a large documents.

3.2.1 Method Let us recall the word to vector method discussed in the last part, especially CBOW model. Just like in the Fig. 2, context of three words (the, cat, and sat) is used to predict the fourth word (on). The input words are mapped to columns of the matrix W mentioned before to predict the output word.

More formally, given a sequence of training words $w_1, w_2, w_3, \ldots, w_N$, the objective of the word vector model is to maximize the conditional probability.

$$P(w_t | w_{t-N}, w_{t-N+1}, \ldots, w_{t+N}) \tag{3.9}$$

After the training converges, words with similar meanings are mapped to adjacent positions in the vector space. Many studies which employed this Word2Vec method have achieved very good results. But how can we get vector representations of paragraphs or even documents from word vectors?

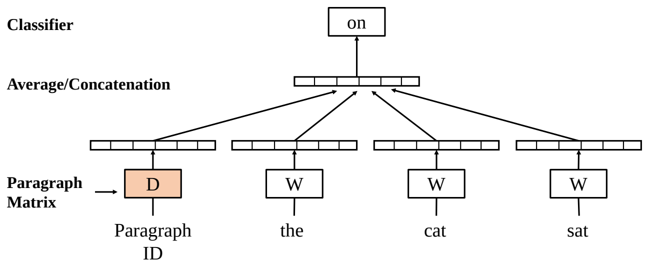

Figure 2: Framework for Learning Paragraph Vector

In this model, the concatenation or average of the vector and the context of the three words are used to predict the fourth word. Paragraph vectors represent missing information in the current context and can serve as memory for the topic of the paragraph.

In this model, the vector parameters are adjusted at the same time, and the target words are predicted using context words and documents similar to the previously described word vector learning method [3, 4]. The probability that a target word appears in a given context is not only constrained by context words, but is also limited by documentation.

$$P(w_t|w_{t-N}, w_{t-N+1}, \ldots, w_{t+N}, d) \tag{3.10}$$

In the above equation, d represents the document, including the order of the words. By using words and paragraph vectors for modeling and maximizing probabilities, paragraph vectors can be adjusted to capture co-occurrence statistics for words within a document. Just as the word vector captures the similarity between words, paragraph vectors capture the similarities between documents. Therefore, paragraph vectors that represent documents containing the same topics are likely to be close to each other in the vector space.

3.2.2 Prediction Time We have talked about how to get the paragraph vectors for variable length text data. However, in the original implementation, the paragraph vectors are trained simultaneously with word vectors. We believe, the paragraph vectors capture some more deeply hidden character of the text data, which is the general topic of this piece of date stream. What we do in this paper is to detect topic and tract them, as a result, we cannot afford to train them with word vectors. On the contrary, we have well trained word vectors in advance and want to get paragraph vectors then. In the machine learning context, this step is called prediction. At prediction time, we need to do an inference step to calculate the paragraph vector for the document or paragraph we want. We get this also by gradient descent. In this procedure, we just need to freeze all the other parameters of the model, the word vectors W, and run the SGD. Since there are less parameter to train, this step costs little time and reach convergence very quickly.

3.3 Topic Detection by Clustering Paragraph Vectors To detect the hidden topics inherent in huge amount of documents, we first the cluster the learned paragraph vectors using CURE algorithm and obtain k cluster centers of the paragraph vectors. As for the metric of distance, we use the cosine similarity between paragraph vectors. There is another choice, using Euclidean distance since we are dealing with vectors. However, cosine similarity metric has showed the more stable and robust results [3, 29].

4 Topic Dissemination Trend Analysis

4.1 Two Problems and Solutions Until now, there are still gaps between the theory and the real applications. We already talked about how to do the whole scheme in theory and do not consider some physical limitations. For example, we are dealing with social networks, the huge amount of text data is very slow to load to main memory and cannot be loaded into memory as a whole. But, in the implementation, we have to check whether a centroid is still active. If centroids are active, we have to keep them in the memory and put those not active ones into disk. In addition, we have to decide to assign a new entering paragraph vector to a current centroid or let it alone to be a new cluster. Furthermore, for those active centroids, we have to keep and track their status and we do not want the data points in those clusters to stay in the memory any more. The question is how to maintain enough status for a centroid. We will answer these questions in this section.

4.1.1 Summarizing Sets of Points Just like us mentioned above, we do want the status of clusters but do not want them data points to stay in the memory and the truth is we cannot have those data in memory. So, we propose a method to track and maintain the status of clusters and their corresponding centroids.

- The number of points in clusters, N.
- The vector **SUM**, whose i^{th} component is the sum of the coordinates of the points in the i^{th} dimension.
- The vector **SUMSQ**: i^{th} component is equal to

sum of squares of coordinates in i^{th} dimension.

This method contains one scalar and two vectors. It seems simple, however, in fact, it almost include everything we need to maintain status. We only need $2d+1$ values to represent any size cluster, where d is the number of dimensions. Average in each dimension can calculated as $\frac{\mathbf{SUM}}{N}$, where $\mathbf{SUM}_i$ is the i^{th} component of **SUM**. And average in each dimension if also the coordinates of the centroids of each cluster. There is an another very import statistic can be calculated, variance, which is very important in the solution for the problem that whether assigning a new entering paragraph vector in current centroid or put it aside as a new cluster. Variance of a cluster's discard set in dimension i is

$$\frac{\mathbf{SUMSQ}_i}{N} - \left(\frac{\mathbf{SUM}_i}{N}\right)^2. \tag{4.11}$$

And the standard deviation is the square root of the variance result.

4.1.2 Sufficient Close This subsection is to answer two questions. The first one is how we decide if a data point (the entering paragraph vector) is close enough to a cluster that we will add the point to that cluster. The second one is how we decide whether two cluster deserve to be combined into one big cluster.

For the first question, we need a way to decide whether to put a new pint into a cluster. We choose Mahalanobis distance discussed in last chapter to do this job. The Mahalanobis distance is defined in Eqn. (4.12).

$$dist(\mathbf{x}, \mathbf{c}) = \sqrt{\sum_{i=1}^{d} \left(\frac{x_i - c_i}{\sigma_i}\right)^2} \tag{4.12}$$

We ask the Mahalanobis distance is less than a threshold so that the point with high likelihood of belonging to currently nearest centroid. We usually choose 2σ as the threshold and accept a point for a cluster if its Mahalanobis distance is less than 2 standard deviations. Here, we also have an assumption that data points normally distributed along each axis. Although this is a very strong assumption and we cannot guarantee, we just do it and the performance is quite good.

For the second question, we need to decide whether two relative small clusters should be combined into one bigger cluster. Recall that we can calculate the variance of clusters, since N, **SUM**, and **SUMSQ** allow us to make this calculation very quickly while using approximate no memory space. What we just do is to computer the variance of the combined cluster. We combine these two clusters if the combined variance is below some threshold. This threshold is a hyper parameter and we can use it to control the sensitivity of the scheme. If we choose a relative small threshold, the topic seems more specific. On the contrary, if we choose a large number, then the topics we get are general.

5 Experiment

5.1 Setup

5.1.1 Datasets Our experiments are conducted on real dataset provided by Tencent and eighteen real-world event datasets from Weibo and Baidu, covering nine most popular events that occurred from 2015 to 2016. The data are extracted via internal APIs. The detailed information of our datasets is listed in Table 1.

5.2 Verification of Topic Detection Since the data we got from Tencent is not enough to do real clustering and the data from Weibo and Baidu are topic related, we use the data from Tencent as noise for the topic detection. we ran my topic detection algorithm and do the topic descriptor automatic generation.

We can see from the table that under the noise of data from Tencent, our model works quite well. Most of the descriptors for these nine topics are good. To my surprise, the automatic generated descriptors even has a general summarization of the topic. However, there are still some results which are too general, such as Wedding for Huo and Lin went public with romance and game for Pokemon go. In addition, some results descriptors are not the most wanted words, but present the very mean of the topic. Tuning the threshold discussed in the last chapter can eliminate this problem. In general, it is a little tricky to check the topic detection alone. When combined with the topic tracking, our method can show the true strength in the detection and tracking.

5.3 Verification of Topic Dissemination Trend Analysis To evaluate the effectiveness and accuracy of our topic detection and tracking method, we compare the paragraph vector model generated by our model with the dissemination trend generated by other two baselines. For the visualization and comparison, we propose a new concept called topic popularity and it is very easy to calculate. For our model in this paper, the popularity for a topic is the N in the cluster. The calculation for the two baselines is listed below.

Naive Frequency: The popularity of a topic is generated simply by word count frequency within the time interval t_i.

TF-IDF: TF-IDF [30] is a well developed method to evaluate relative importance of a word in a corpus

Table 1: Overall Information of Datasets from Baidu and Weibo

No.	Event Name	Category	Date	# of Peaks	# of Records (k)		Size (MB)	
					Weibo	Baidu	Weibo	Baidu
①	Sinking of a Cruise Ship[#]	Disaster	06/01-06/30 (2015)	Multiple	308.45	1560.4	320.59	401.48
②	Chinese Stock Market Crash	Disaster	06/16-07/13 (2015)	Multiple	701.71	420.40	578.77	74.14
③	AlphaGo	High-Tech	02/20-03/29 (2016)	Multiple	838.12	2337.3	654.89	406.83
④	Leonardo DiCaprio, Oscar Best Actor	Entertainment	02/20-03/09 (2016)	Single	2569.5	730.82	1788.9	139.52
⑤	Kobe Bryant's Retirement	Sports	04/10-04/19 (2016)	Single	3655.3	2300.9	2274.8	403.69
⑥	Huo and Lin Went Public with Romance[†]	Entertainment	05/10-05/29 (2016)	Hybrid	1535.2	1615.2	1027.1	289.98
⑦	Brexit Referendum	Politics	06/20-06/30 (2016)	Single	957.16	2160.4	715.51	392.32
⑧	Pokémon Go	High-Tech	07/02-07/20 (2016)	Hybrid	936.38	3652.2	695.90	625.87
⑨	The South China Sea Arbitration	Politics	07/10-07/19 (2016)	Single	7671.0	7815.3	5918.2	1451.9

[#] A Chinese cruise ship called *Dongfang Zhi Xing* sank into Yangtze River.
[†] Actor Wallace Huo and Actress Ruby Lin went public with Romance on May 20th, 2016.

Table 2: Topic Detection and Descriptor Generation

Topic	Automatic Generated Topic Descriptor
①	Dongfang Star sinking disaster
②	Stock falling down risk
③	AlphaGo Go competition human
④	Leonardo best actor Oscar
⑤	Kobe retire
⑥	Wedding
⑦	English Europe broke
⑧	Game
⑨	South sea no

and is always used as a baseline.

All the topic popularity including them generated by Naive Frequency and TF-IDF are normalized in the same way as Eqn. (5.13).

$$\overline{pop}(E_i) = \frac{pop(E_i)}{\sum\limits_{1\le k\le n} pop(E_k)}. \tag{5.13}$$

Based on the three generated topic popularity, we present a thorough discussion and comparison to validate our paragraph vector model.

5.3.1 Accuracy We pick up the peaks in figures and backtrack what exactly happened in reality. An event is always pushed forward by series of "little" events and we call them sub-events, which are reflected as peaks in topic trend figures and the highest popularity within those time intervals. We regard these records as the detected subevents.In the Event *Capsizing of a Cruise Ship*, the real-world event evolution involves four key sub-events. On the night of June 1, 2015, the cruise

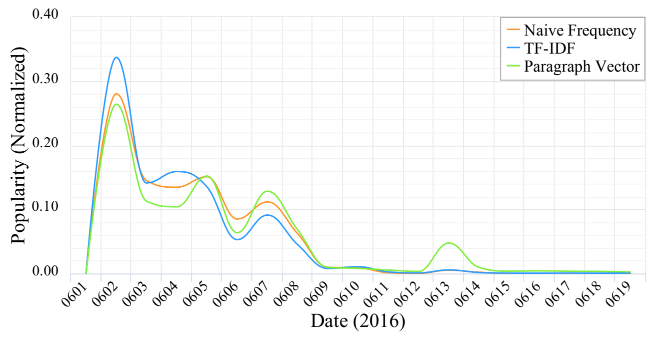

Figure 3: Topic Trend Dissemination

ship sank in a severe thunderstorm. Such a shocking disaster raised tremendous public attention on June 2. On June 5, the ship was hoisted and set upright. A mourning ceremony was held on June 7, and on June 13, total 442 deaths and only 12 survivors were officially confirmed, which marked the end of the rescue work. All these four key sub-events are highly consistent with detected sub-events, and corresponded well with four peaks generated by our model. The topic trend dissemination figures generated by our model shows four peaks, which is illustrated in Fig. 3. All these peaks are highly consistent with the four key sub-events in real world, while the end of rescue work on June 13 is missed by approaches based on Naive Frequency and TF-IDF. In conclusion, our model based on paragraph vector shows the ability to track the development of events precisely.

5.3.2 Sensitivity In our model, a high-dimension vector space distance threshold *th* is introduced to control the size of cluster and another hyper parameter d is introduced to control how accuracy and sensitivity of word vector and paragraph vector. To demonstrate the function of our model on highlighting contributive topics, we conduct experiments on our model with distinct d on Event *Pokémon Go* and compare the results with the other two baselines. Compared with the baselines, our model are more sensitive to the burst phases of an topic, as is shown in Fig. 4, especially on data points 07/06, 07/08, and 07/11. The event popularity on these days are larger than those obtained by Naive Frequency and TF-IDF.

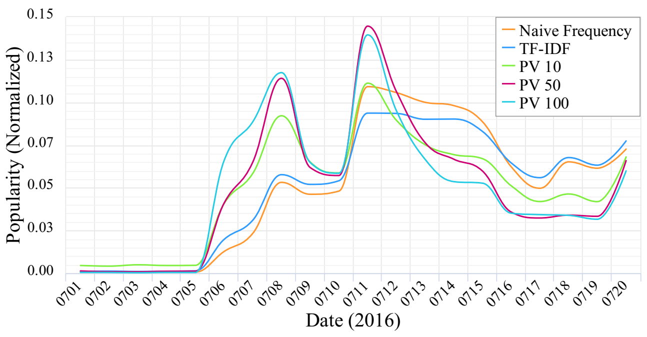

Figure 4: Topic Dissemination Trend, Different Dimension of Paragraph Vector

An event whose trend figure rises fast at some data points possesses the potential to draw wider attention. It is reasonable for a popularity model not only to depict the current state of event popularity, but also take the potential future trends into consideration. In this way, a quick response to the burst phases of an event is more valuable for real-world applications. This advantage of our model can lead to a powerful technique for first story detection on ongoing events.

5.3.3 Superior Robustness to Noise To verify whether our model can effectively filter out noisy words, we further implement an experiment on a simulated corpus. We first extract 50K Baidu queries with the highest frequency in the corpus of Event *Kobe's Retirement* and make them as the base data for a 6-day simulated corpus. Then we randomly pick noisy queries from Internet that are not relevant to Event *Kobe's Retirement* at all. The amount of noisy queries is listed in Table 3.

Table 3: Number of Noisy Records Added

Day	1	2	3	4	5	6
# (k)	0.000	1.063	2.235	3.507	4.689	6.026

Since each day's base data are identical, say that 50K queries, a good model is supposed to filter noisy queries out and generate a topic dissemination trend series with all identical data points, which form a horizontal line in X-Y plane. Series generated by our model, Naive Frequency and TF-IDF are shown in Fig. 5. It is shown that our model successfully filters out the noise and generates the series which is a horizontal line and captures the real topic and get the real popularity of the topic, while the other two methods Naive Frequency and TF-IDF are obviously effected by the noisy queries and generate series that cannot accurately reflect the event popularity.

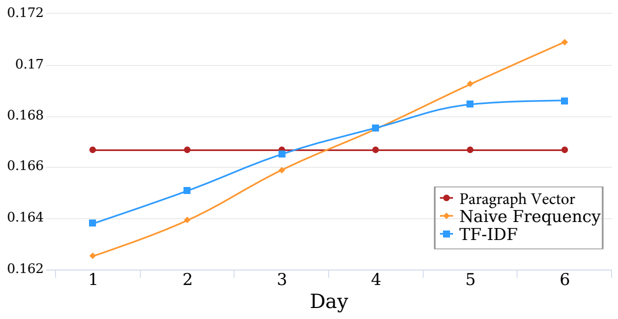

Figure 5: Topic Dissemination Trend, Simulated Corpus

6 Conclusion

In this paper, we have studied how to do topic detection and the following topic dissemination trend analysis. To address the topic detection issue, we utilize machine learning and neural network style method, Word2Vec and Paragraph Vector to transform the documents into a high-dimension vector space, which are covered in our topic detection model. To furthermore analyze the topic dissemination trend, we propose the relative topic tracking method based on paragraph vectors controlled by two hyper parameters, dimension of the paragraph vector and the distance threshold between cluster. Besides that, visualization and case studies are covered in the paper. Experimental results on data from Tencent and eighteen real-world event datasets from Weibo and Baidu validate the effectiveness and applicability of our scheme.

References

[1] J. Allan, J. G. Carbonell, G. Doddington, J. Yamron, and Y. Yang, "Topic detection and tracking pilot study final report," 1998.

[2] J. Allan, "Introduction to topic detection and tracking," in *Topic detection and tracking*. Springer, 2002, pp. 1–16.

[3] T. Mikolov, I. Sutskever, K. Chen, G. S. Corrado, and J. Dean, "Distributed representations of words and phrases and their compositionality," in *NIPS*, 2013, pp. 3111–3119.

[4] T. Mikolov, K. Chen, G. Corrado, and J. Dean, "Efficient estimation of word representations in vector space," *CoRR*, vol. abs/1301.3781, 2013.

[5] Q. V. Le and T. Mikolov, "Distributed representations of sentences and documents," in *ICML*, 2014, pp. 1188–1196.

[6] T. Kanungo, D. M. Mount, N. S. Netanyahu, C. D. Piatko, R. Silverman, and A. Y. Wu, "An efficient k-means clustering algorithm: Analysis and implementation," *TPAMI*, vol. 24, no. 7, pp. 881–892, 2002.

[7] J. Pennington, R. Socher, and C. D. Manning, "Glove: Global vectors for word representation," in *EMNLP*, 2014, pp. 1532–1543.

[8] K. Cho, B. van Merrienboer, Ç. Gülçehre, D. Bahdanau, F. Bougares, H. Schwenk, and Y. Bengio, "Learning phrase representations using RNN encoder-decoder for statistical machine translation," in *EMNLP*, 2014, pp. 1724–1734.

[9] J. Li, C. Liu, J. X. Yu, Y. Chen, T. Sellis, and J. S. Culpepper, "Personalized influential topic search via social network summarization," in *ICDE*, 2017, pp. 17–18.

[10] Q. Li, A. Nourbakhsh, S. Shah, and X. Liu, "Real-time novel event detection from social media," in *ICDE*, 2017, pp. 1129–1139.

[11] W. Hu, H. Wang, Z. Qiu, C. Nie, L. Yan, and B. Du, "An event detection method for social networks based on hybrid link prediction and quantum swarm intelligent," *World Wide Web (WWW)*, vol. 20, no. 4, pp. 775–795, 2017.

[12] G. Paltoglou, "Sentiment-based event detection in twitter," *JASIST*, vol. 67, no. 7, pp. 1576–1587, 2016.

[13] D. Zhou, T. Gao, and Y. He, "Jointly event extraction and visualization on twitter via probabilistic modelling," in *ACL*, 2016.

[14] M. Akbari, X. Hu, L. Nie, and T. Chua, "From tweets to wellness: Wellness event detection from twitter streams," in *AAAI*, 2016, pp. 87–93.

[15] R. K. Pon, A. F. Cardenas, D. Buttler, and T. Critchlow, "Tracking multiple topics for finding interesting articles," in *KDD*, 2017, pp. 560–569.

[16] B. Li, W. Li, and Q. Lu, "Topic tracking with time granularity reasoning," *ACM Transactions on Asian Language Information Processing (TALLIP)*, vol. 5, no. 4, pp. 388–412, 2006.

[17] S. Huang, Y. Liu, and D. Dang, "Burst topic discovery and trend tracing based on storm," *Physica A: Statistical Mechanics and its Applications*, vol. 416, pp. 331–339, 2014.

[18] X. Wang and A. McCallum, "Topics over time: a non-markov continuous-time model of topical trends," in *KDD*, 2006, pp. 424–433.

[19] T. L. Griffiths and M. Steyvers, "Finding scientific topics," *Proceedings of the National academy of Sciences*, vol. 101, no. 1, pp. 5228–5235, 2004.

[20] C. X. Lin, B. Zhao, Q. Mei, and J. Han, "PET: a statistical model for popular events tracking in social communities," in *ACM International Conference on Knowledge Discovery and Data Mining (KDD), Washington, DC, USA, July 25-28, 2010*, pp. 929–938.

[21] R. Collobert, J. Weston, L. Bottou, M. Karlen, K. Kavukcuoglu, and P. P. Kuksa, "Natural language processing (almost) from scratch," *Journal of Machine Learning Research (JMLR)*, vol. 12, pp. 2493–2537, 2011.

[22] J. P. Turian, L. Ratinov, and Y. Bengio, "Word representations: A simple and general method for semi-supervised learning," in *Annual Meeting of the Association for Computational Linguistics (ACL), Uppsala, Sweden, July 11-16, 2010*, pp. 384–394.

[23] Y. Bengio, R. Ducharme, and P. Vincent, "A neural probabilistic language model," in *Annual Conference on Neural Information Processing Systems (NIPS), Denver, CO, USA, November 27-30, 2000*, pp. 932–938.

[24] X. Chen, L. Xu, Z. Liu, M. Sun, and H. Luan, "Joint learning of character and word embeddings," in *Joint Conference on Artificial Intelligence (IJCAI), Buenos Aires, Argentina, July 25-31, 2015*, pp. 1236–1242.

[25] L. Yang and M. Sun, "Improved learning of chinese word embeddings with semantic knowledge," in *Chinese Computational Linguistics and Natural Language Processing Based on Naturally Annotated Big Data - 14th China National Conference (CCL), Guangzhou, China, November 13-14, 2015*, pp. 15–25.

[26] R. Yin, Q. Wang, P. Li, R. Li, and B. Wang, "Multi-granularity chinese word embedding," in *Conference on Empirical Methods in Natural Language Processing (EMNLP), Austin, TX, USA, November 1-4, 2016*, pp. 981–986.

[27] T. Su and H. Lee, "Learning chinese word representations from glyphs of characters," in *Conference on Empirical Methods in Natural Language Processing (EMNLP) 2017, Copenhagen, Denmark, September 9-11, 2017*, pp. 264–273.

[28] L. Shen, Z. Zhe, H. Renfen, L. Wensi, L. Tao, and D. Xiaoyong, "Analogical reasoning on chinese morphological and semantic relations," *arXiv preprint arXiv:1805.06504*, 2018.

[29] A. M. Dai, C. Olah, and Q. V. Le, "Document embedding with paragraph vectors," *CoRR*, vol. abs/1507.07998, 2015.

[30] Y. Tang, P. Ma, B. Kong, W. Ji, X. Gao, and X. Peng, "ESAP: A novel approach for cross-platform event dissemination trend analysis between social network and search engine," in *WISE*, 2016, pp. 489–504.