New Event Detection Based on Indexing-tree and Named Entity

Zhang Kuo
Tsinghua University
Beijing, 100084, China
86-10-62771736
zkuo99@mails.tsinghua.edu.cn

Li Juan Zi
Tsinghua University
Beijing, 100084, China
86-10-62781461
ljz@keg.cs.tsinghua.edu.cn

Wu Gang
Tsinghua University
Beijing, 100084, China
86-10-62789831
wug03@keg.cs.tsinghua.edu.cn

ABSTRACT

New Event Detection (NED) aims at detecting from one or multiple streams of news stories that which one is reported on a new event (i.e. not reported previously). With the overwhelming volume of news available today, there is an increasing need for a NED system which is able to detect new events more efficiently and accurately. In this paper we propose a new NED model to speed up the NED task by using news indexing-tree dynamically. Moreover, based on the observation that terms of different types have different effects for NED task, two term reweighting approaches are proposed to improve NED accuracy. In the first approach, we propose to adjust term weights dynamically based on previous story clusters and in the second approach, we propose to employ statistics on training data to learn the named entity reweighting model for each class of stories. Experimental results on two Linguistic Data Consortium (LDC) datasets TDT2 and TDT3 show that the proposed model can improve both efficiency and accuracy of NED task significantly, compared to the baseline system and other existing systems.

Categories and Subject Descriptors

H.3.3 [**Information Systems**]: Information Search and Retrieval; H.4.2 [**Information Systems Applications**]: Types of Systems – *decision support*.

General Terms

Algorithms, Performance, Experimentation

Keywords

Topic Detection and Tracking, New Event Detection, Named Entity, Real-time Indexing

1. INTRODUCTION

Topic Detection and Tracking (TDT) program aims to develop techniques which can effectively organize, search and structure news text materials from a variety of newswire and broadcast media [1]. New Event Detection (NED) is one of the five tasks in TDT. It is the task of online identification of the earliest report for each topic as soon as that report arrives in the sequence of documents. A Topic is defined as "a seminal event or activity, along with directly related events and activities" [2]. An Event is defined as "something (non-trivial) happening in a certain place at a certain time" [3]. For instance, when a bomb explodes in a building, the exploding is the seminal event that triggers the topic, and other stories on the same topic would be those discussing salvaging efforts, the search for perpetrators, arrests and trial and so on. Useful news information is usually buried in a mass of data generated everyday. Therefore, NED systems are very useful for people who need to detect novel information from real-time news stream. These real-life needs often occur in domains like financial markets, news analysis, and intelligence gathering.

In most of state-of-the-art (currently) NED systems, each news story on hand is compared to all the previous received stories. If all the similarities between them do not exceed a threshold, then the story triggers a new event. They are usually in the form of cosine similarity or Hellinger similarity metric. The core problem of NED is to identify whether two stories are on the same topic. Obviously, these systems cannot take advantage of topic information. Further more, it is not acceptable in real applications because of the large amount of computation required in the NED process. Other systems organize previous stories into clusters (each cluster corresponds to a topic), and new story is compared to the previous clusters instead of stories. This manner can reduce comparing times significantly. Nevertheless, it has been proved that this manner is less accurate [4, 5]. This is because sometimes stories within a topic drift far away from each other, which could lead low similarity between a story and its topic.

On the other hand, some proposed NED systems tried to improve accuracy by making better use of named entities [10, 11, 12, 13]. However, none of the systems have considered that terms of different types (e.g. Noun, Verb or Person name) have different effects for different classes of stories in determining whether two stories are on the same topic. For example, the names of election candidates (Person name) are very important for stories of election class; the locations (Location name) where accidents happened are important for stories of accidents class.

So, in NED, there still exist following three problems to be investigated: (1) How to speed up the detection procedure while do not decrease the detection accuracy? (2) How to make good use of cluster (topic) information to improve accuracy? (3) How to obtain better news story representation by better understanding of named entities.

Driven by these problems, we have proposed three approaches in this paper. (1)To make the detection procedure faster, we propose a new NED procedure based on news indexing-tree created dynamically. Story indexing-tree is created by assembling similar stories together to form news clusters in different hierarchies according to their values of similarity. Comparisons between current story and previous clusters could help find the most similar story in less comparing times. The new procedure can

reduce the amount of comparing times without hurting accuracy. (2)We use the clusters of the first floor in the indexing-tree as news topics, in which term weights are adjusted dynamically according to term distribution in the clusters. In this approach, cluster (topic) information is used properly, so the problem of theme decentralization is avoided. (3)Based on observations on the statistics obtained from training data, we found that terms of different types (e.g. Noun and Verb) have different effects for different classes of stories in determining whether two stories are on the same topic. And we propose to use statistics to optimize the weights of the terms of different types in a story according to the news class that the story belongs to. On TDT3 dataset, the new NED model just uses 14.9% comparing times of the basic model, while its minimum normalized cost is 0.5012, which is 0.0797 better than the basic model, and also better than any other results previously reported for this dataset [8, 13].

The rest of the paper is organized as follows. We start off this paper by summarizing the previous work in NED in section 2. Section 3 presents the basic model for NED that most current systems use. Section 4 describes our new detection procedure based on news indexing-tree. In section 5, two term reweighting methods are proposed to improve NED accuracy. Section 6 gives our experimental data and evaluation metrics. We finally wrap up with the experimental results in Section 7, and the conclusions and future work in Section 8.

2. RELATED WORK

Papka et al. proposed Single-Pass clustering on NED [6]. When a new story was encountered, it was processed immediately to extract term features and a query representation of the story's content is built up. Then it was compared with all the previous queries. If the document did not trigger any queries by exceeding a threshold, it was marked as a new event. Lam et al build up previous query representations of story clusters, each of which corresponds to a topic [7]. In this manner comparisons happen between stories and clusters.

Recent years, most work focus on proposing better methods on comparison of stories and document representation. Brants et al. [8] extended a basic incremental TF-IDF model to include source-specific models, similarity score normalization based on document-specific averages, similarity score normalization based on source-pair specific averages, term reweighting based on inverse event frequencies, and segmentation of documents. Good improvements on TDT bench-marks were shown. Stokes et al. [9] utilized a combination of evidence from two distinct representations of a document's content. One of the representations was the usual free text vector, the other made use of lexical chains (created using WordNet) to build another term vector. Then the two representations are combined in a linear fashion. A marginal increase in effectiveness was achieved when the combined representation was used.

Some efforts have been done on how to utilize named entities to improve NED. Yang et al. gave location named entities four times weight than other terms and named entities [10]. DOREMI research group combined semantic similarities of person names, location names and time together with textual similarity [11][12]. UMass [13] research group split document representation into two parts: named entities and non-named entities. And it was found that some classes of news could achieve better performance using named entity representation, while some other classes of news could achieve better performance using non-named entity representation. Both [10] and [13] used text categorization technique to classify news stories in advance. In [13] news stories are classified automatically at first, and then test sensitivities of names and non-name terms for NED for each class. In [10] frequent terms for each class are removed from document representation. For example, word "election" does not help identify different elections. In their work, effectiveness of different kinds of names (or terms with different POS) for NED in different news classes are not investigated. We use statistical analysis to reveal the fact and use it to improve NED performance.

3. BASIC MODEL

In this section, we present the basic New Event Detection model which is similar to what most current systems apply. Then, we propose our new model by extending the basic model.

New Event Detection systems use news story stream as input, in which stories are strictly time-ordered. Only previously received stories are available when dealing with current story. The output is a decision for whether the current story is on a new event or not and the confidence of the decision. Usually, a NED model consists of three parts: story representation, similarity calculation and detection procedure.

3.1 Story Representation

Preprocessing is needed before generating story representation. For preprocessing, we tokenize words, recognize abbreviations, normalize abbreviations, add part-of-speech tags, remove stopwords included in the stop list used in InQuery [14], replace words with their stems using K-stem algorithm[15], and then generate word vector for each news story.

We use incremental TF-IDF model for term weight calculation [4]. In a TF-IDF model, term frequency in a news document is weighted by the inverse document frequency, which is generated from training corpus. When a new term occurs in testing process, there are two solutions: simply ignore the new term or set *df* of the term as a small const (e.g. *df* = 1). The new term receives too low weight in the first solution (0) and too high weight in the second solution. In incremental TF-IDF model, document frequencies are updated dynamically in each time step *t*:

$$df_t(w) = df_{t-1}(w) + df_{D_t}(w) \quad (1)$$

where D_t represents news story set received in time *t*, and $df_{Dt}(w)$ means the number of documents that term *w* occurs in, and $df_t(w)$ means the total number of documents that term *w* occurs in before time *t*. In this work, each time window includes 50 news stories.

Thus, each story *d* received in *t* is represented as follows:

$$d \rightarrow \{weight(d,t,w_1), weight(d,t,w_2), ..., weight(d,t,w_n)\}$$

where *n* means the number of distinct terms in story *d*, and $weight(d,t,w)$ means the weight of term *w* in story *d* at time *t*:

$$weight(d,t,w) = \frac{\log(tf(d,w)+1)\log((N_t+1)/(df_t(w)+0.5))}{\sum_{w' \in d} \log(tf(d,w')+1)\log((N_t+1)/(df_t(w')+0.5))} \quad (2)$$

where N_t means the total number of news stories before time *t*, and $tf(d,w)$ means how many times term *w* occurs in news story *d*.

3.2 Similarity Calculation

We use Hellinger distance for the calculation of similarity between two stories, for two stories d and d' at time t, their similarity is defined as follows:

$$sim(d, d', t) = \sum_{w \in d, d'} \sqrt{weight(d, t, w) * weight(d', t, w)} \quad (3)$$

3.3 Detection Procedure

For each story d received in time step t, the value

$$n(d) = \max_{time(d') < time(d)} (sim(d, d', t)) \quad (4)$$

is a score used to determine whether d is a story about a new topic and at the same time is an indication of the confidence in our decision [8]. $time(d)$ means the publication time of story d. If the score exceeds the threshold θ_{new}, then there exists a sufficiently similar document, thus d is a old story, otherwise, there is no sufficiently similar previous document, thus d is an new story.

4. New NED Procedure

Traditional NED systems can be classified into two main types on the aspect of detection procedure: (1) S-S type, in which the story on hand is compared to each story received previously, and use the highest similarity to determine whether current story is about a new event; (2) S-C type, in which the story on hand is compared to all previous clusters each of which representing a topic, and the highest similarity is used for final decision for current story. If the highest similarity exceeds threshold θ_{new}, then it is an old story, and put it into the most similar cluster; otherwise it is a new story and create a new cluster. Previous work show that the first manner is more accurate than the second one [4][5]. Since sometimes stories within a topic drift far away from each other, a story may have very low similarity with its topic. So using similarities between stories for determining new story is better than using similarities between story and clusters. Nevertheless, the first manner needs much more comparing times which means the first manner is low efficient. We propose a new detection procedure which uses comparisons with previous clusters to help find the most similar story in less comparing times, and the final new event decision is made according to the most similar story. Therefore, we can get both the accuracy of S-S type methods and the efficiency of S-C type methods.

The new procedure creates a news indexing-tree dynamically, in which similar stories are put together to form a hierarchy of clusters. We index similar stories together by their common ancestor (a cluster node). Dissimilar stories are indexed in different clusters. When a story is coming, we use comparisons between the current story and previous hierarchical clusters to help find the most similar story which is useful for new event decision. After the new event decision is made, the current story is inserted to the indexing-tree for the following detection.

The news indexing-tree is defined formally as follows:

$$S\text{-}Tree = \{r, N^C, N^S, E\}$$

where r is the root of *S-Tree*, N^C is the set of all cluster nodes, N^S is the set of all story nodes, and E is the set of all edges in *S-Tree*. We define a set of constraints for a *S-Tree*:

i . $\forall i, i \in N^C \rightarrow i$ is an non-terminal node in the tree
ii . $\forall i, i \in N^S \rightarrow i$ is a terminal node in the tree
iii . $\forall i, i \in N^C \rightarrow$ out degree of i is at least 2
iv . $\forall i, i \in N^C \rightarrow i$ is represented as the centroid of its desendants

For a news story d_i, the comparison procedure and inserting procedure based on indexing-tree are defined as follows. An example is shown by Figure 1 and Figure 2.

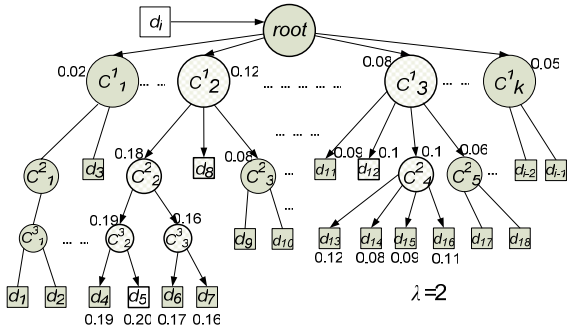

Figure 1. Comparison procedure

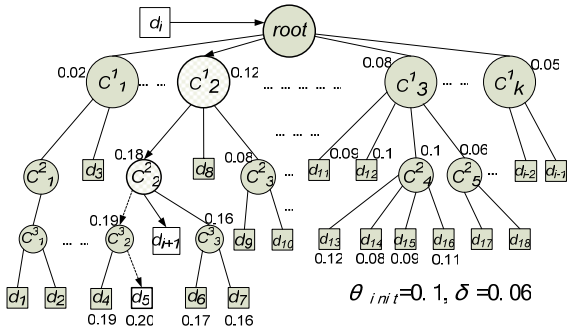

Figure 2. Inserting procedure

Comparison procedure:

Step 1: compare d_i to all the direct child nodes of r and select λ nodes with highest similarities, e.g., C^1_2 and C^1_3 in Figure 1.

Step 2: for each selected node in the last step, e.g. C^1_2, compare d_i to all its direct child nodes, and select λ nodes with highest similarities, e.g. C^2_2 and d_8. Repeat step 2 for all non-terminal nodes.

Step 3: record the terminal node with the highest similarty to d_i, e.g. s_5, and the similarity value (0.20).

Inserting d_i to the *S-tree* with r as root:

Find the node n which is direct child of r in the path from r to the terminal node with highest similarity s, e.g. C^1_2. If s is smaller than $\theta_{init}+(h\text{-}1)\delta$, then add d_i to the tree as a direct child of r. Otherwise, if n is a terminal node, then create a cluster node instead of n, and add both n and d_i as its direct children; if n is an non-terminal node, then repeat this procedure and insert d_i to the sub-tree with n as root recursively. Here h is the length between n and the root of *S-tree*.

The more the stories in a cluster similar to each other, the better the cluster represents the stories in it. Hence we add no constraints on the maximum of tree's height and degree of a node. Therefore, we cannot give the complexity of this indexing-tree based procedure. But we will give the number of comparing times needed by the new procedure in our experiments in section7.

5. Term Reweighting Methods

In this section, two term reweighting methods are proposed to improve NED accuracy. In the first method, a new way is explored for better using of cluster (topic) information. The second one finds a better way to make use of named entities based on news classification.

5.1 Term Reweighting Based on Distribution Distance

TF-IDF is the most prevalent model used in information retrieval systems. The basic idea is that the fewer documents a term appears in, the more important the term is in discrimination of documents (relevant or not relevant to a query containing the term). Nevertheless, in TDT domain, we need to discriminate documents with regard to topics rather than queries. Intuitively, using cluster (topic) vectors to compare with subsequent news stories should outperform using story vectors. Unfortunately, the experimental results do not support this intuition [4][5]. Based on observation on data, we find the reason is that a news topic usually contains many directly or indirectly related events, while they all have their own sub-subjects which are usually different with each other. Take the topic described in section 1 as an example, events like the explosion and salvage have very low similarities with events about criminal trial, therefore stories about trial would have low similarity with the topic vector built on its previous events. This section focuses on how to effectively make use of topic information and at the same time avoid the problem of content decentralization.

At first, we classify terms into 5 classes to help analysis the needs of the modified model:

Term class A: terms that occur frequently in the whole corpus, e.g., "year" and "people". Terms of this class should be given low weights because they do not help much for topic discrimination.

Term class B: terms that occur frequently within a news category, e.g., "election", "storm". They are useful to distinguish two stories in different news categories. However, they cannot provide information to determine whether two stories are on the same or different topics. In another words, term "election" and term "storm" are not helpful in differentiate two election campaigns and two storm disasters. Therefore, terms of this class should be assigned lower weights.

Term class C: terms that occur frequently in a topic, and infrequently in other topics, e.g., the name of a crash plane, the name of a specific hurricane. News stories that belong to different topics rarely have overlap terms in this class. The more frequently a term appears in a topic, the more important the term is for a story belonging to the topic, therefore the term should be set higher weight.

Term class D: terms that appear in a topic exclusively, but not frequently. For example, the name of a fireman who did very well in a salvage action, which may appears in only two or three stories but never appeared in other topics. Terms of this type should receive more weights than in TF-IDF model. However, since they are not popular in the topic, it is not appropriate to give them too high weights.

Term class E: terms with low document frequency, and appear in different topics. Terms of this class should receive lower weights.

Now we analyze whether TF-IDF model can give proper weights to the five classes of terms. Obviously, terms of *class A* are lowly weighted in TF-IDF model, which is conformable with the requirement described above. In TF-IDF model, terms of *class B* are highly dependant with the number of stories in a news class. TF-IDF model cannot provide low weights if the story containing the term belongs to a relative small news class. For a term of *class C*, the more frequently it appears in a topic, the less weight TF-IDF model gives to it. This strongly conflicts with the requirement of terms in *class C*. For terms of *class D*, TF-IDF model gives them high weights correctly. But for terms of *class E*, TF-IDF model gives high weights to them which are not conformable with the requirement of low weights. To sum up, terms of *class B, C, E* cannot be properly weighted in TF-IDF model. So, we propose a modified model to resolve this problem.

When θ_{init} and θ_{new} are set closely, we assume that most of the stories in a first-level cluster (a direct child node of *root* node) are on the same topic. Therefore, we make use of a first-level cluster to capture term distribution (*df* for all the terms within the cluster) within the topic dynamically. *KL* divergence of term distribution in a first-level cluster and the whole story set is used to adjust term weights:

$$weight_D(d,t,w) = \frac{weight(d,t,w) * (1 + \gamma * KL(P_{cw} \parallel P_{tw}))}{\sum_{w' \in d} weight(d,t,w') * (1 + \gamma * KL(P_{cw'} \parallel P_{tw'}))} \quad (5)$$

where
$$p_{cw}(y) = \frac{df_c(w)}{N_c}, \; \bar{p}_{cw}(y) = 1 - \frac{df_c(w)}{N_c} \quad (6)$$

$$p_{tw}(y) = \frac{df_t(w)}{N_t}, \; \bar{p}_{tw}(y) = 1 - \frac{df_t(w)}{N_t} \quad (7)$$

where $df_c(w)$ is the number of documents containing term w within cluster C, and N_c is the number of documents in cluster C, and N_t is the total number of documents that arrive before time step t. γ is a const parameter, now is manually set 3.

KL divergence is defined as follows [17]:

$$KL(P \parallel Q) = \sum_x p(x) \log \frac{p(x)}{q(x)} \quad (8)$$

The basic idea is: for a story in a topic, the more a term occurs within the topic, and the less it occurs in other topics, it should be assigned higher weights. Obviously, modified model can meet all the requirements of the five term classes listed above.

5.2 Term Reweighting Based on Term Type and Story Class

Previous work found that some classes of news stories could achieve good improvements by giving extra weight to named entities. But we find that terms of different types should be given different amount of extra weight for different classes of news stories.

We use open-NLP[1] to recognize named entity types and part-of-speech tags for terms that appear in news stories. Named entity types include person name, organization name, location name, date, time, money and percentage, and five POSs are selected: none (NN), verb (VB), adjective (JJ), adverb (RB) and cardinal number (CD). Statistical analysis shows topic-level discriminative terms types for different classes of stories. For the sake of convenience, named entity type and part-of-speech tags are uniformly called term type in subsequent sections.

Determining whether two stories are about the same topic is a basic component for NED task. So at first we use χ^2 statistic to compute correlations between terms and topics. For a term t and a topic T, a contingence table is derived:

Table 1. A 2×2 Contingence Table

Doc Number	belong to topic T	not belong to topic T
include t	A	B
not include t	C	D

The χ^2 statistic for a specific term t with respect to topic T is defined to be [16]:

$$\chi^2(w,T) = \frac{(A+B+C+D)*(AD-CB)^2}{(A+C)*(B+D)*(A+B)*(C+D)} \quad (9)$$

News topics for the TDT task are further classified into 11 "rules of interpretations" (ROIs)[2]. The ROI can be seen as a higher level class of stories. The average correlation between a term type and a topic ROI is computed as:

$$\chi^2_{avg}(P_k, R_m) = \frac{1}{|R_m|}\sum_{T\in Rm}\left(\frac{1}{|P_k|}\sum_{w\in Pk} p(w,T)\chi^2(w,T)\right)$$
$$k=1...K,\ m=1...M \quad (10)$$

where K is the number of term types (set 12 constantly in the paper). M is the number news classes (ROIs, set 11 in the paper). P_k represents the set of all terms of type k, and R_m represents the set of all topics of class m, $p(t,T)$ means the probability that t occurs in topic T. Because of limitation of space, only parts of the term types (9 term types) and parts of news classes (8 classes) are listed in table 2 with the average correlation values between them. The statistics is derived from labeled data in TDT2 corpus. (Results in table 2 are already normalized for convenience in comparison.)

The statistics in table 2 indicates the usefulness of different term types in topic discrimination with respect to different news classes. We can see that, location name is the most useful term type for three news classes: Natural Disasters, Violence or War, Finances. And for three other categories Elections, Legal/Criminal Cases, Science and Discovery, person name is the most discriminative term type. For Scandals/Hearings, date is the most important information for topic discrimination. In addition, Legal/Criminal Cases and Finance topics have higher correlation with money terms, while Science and Discovery have higher correlation with percentage terms. Non-name terms are more stable for different classes.

From the analysis of table 2, it is reasonable to adjust term weight according to their term type and the news class the story belongs to. New term weights are reweighted as follows:

$$weight_T(d,t,w) = \frac{weight_D(d,t,w)*\alpha^{class(d)}_{type(w)}}{\sum_{w'\in d} weight_D(d,t,w')*\alpha^{class(d)}_{type(w')}} \quad (11)$$

where $type(w)$ represents the type of term w, and $class(d)$ represents the class of story d, $\alpha^c{}_k$ is reweighting parameter for news class c and term type k. In the work, we just simply use statistics in table 2 as the reweighting parameters. Even thought using the statistics directly may not the best choice, we do not discuss how to automatically obtain the best parameters. We will try to use machine learning techniques to obtain the best parameters in the future work.

In the work, we use BoosTexter [20] to classify all stories into one of the 11 ROIs. BoosTexter is a boosting based machine learning program, which creates a series of simple rules for building a classifier for text or attribute-value data. We use term weight generated using TF-IDF model as feature for story classification. We trained the model on the 12000 judged English stories in TDT2, and classify the rest of the stories in TDT2 and all stories in TDT3. Classification results are used for term reweighting in formula (11). Since the class labels of topic-off stories are not given in TDT datasets, we cannot give the classification accuracy here. Thus we do not discuss the effects of classification accuracy to NED performance in the paper.

6. EXPERIMENTAL SETUP

6.1 Datasets

We used two LDC [18] datasets TDT2 and TDT3 for our experiments. TDT2 contains news stories from January to June 1998. It contains around 54,000 stories from sources like ABC, Associated Press, CNN, New York Times, Public Radio International, Voice of America etc. Only English stories in the collection were considered. TDT3 contains approximately 31,000 English stories collected from October to December 1998. In addition to the sources used in TDT2, it also contains stories from NBC and MSNBC TV broadcasts. We used transcribed versions of the TV and radio broadcasts besides textual news.

TDT2 dataset is labeled with about 100 topics, and approximately 12,000 English stories belong to at least one of these topics. TDT3 dataset is labeled with about 120 topics, and approximately 8000 English stories belong to at least one of these topics. All the topics are classified into 11 "Rules of Interpretation": (1)Elections, (2)Scandals/Hearings, (3)Legal/Criminal Cases, (4)Natural Disasters, (5)Accidents, (6)Ongoing Violence or War, (7)Science and Discovery News, (8)Finance, (9)New Law, (10)Sports News, (11)MISC. News.

6.2 Evaluation Metric

TDT uses a cost function C_{Det} that combines the probabilities of missing a new story and a false alarm [19]:

$$C_{Det} = C_{Miss} * P_{Miss} * P_{Target} + C_{FA} * P_{FA} * P_{Nontarget} \quad (12)$$

[1]. http://opennlp.sourceforge.net/

[2]. http://projects.ldc.upenn.edu/TDT3/Guide/label.html

Table 2. Average correlation between term types and news classes

	Location	Person	Date	Organization	Money	Percentage	NN	JJ	CD
Elections	0.37	1	0.04	0.58	0.08	0.03	0.32	0.13	0.1
Scandals/Hearings	0.66	0.62	0.28	1	0.11	0.02	0.27	0.13	0.05
Legal/Criminal Cases	0.48	1	0.02	0.62	0.15	0	0.22	0.24	0.09
Natural Disasters	1	0.27	0	0.04	0.04	0	0.25	0.04	0.02
Violence or War	1	0.36	0.02	0.14	0.02	0.04	0.21	0.11	0.02
Science and Discovery	0.11	1	0.01	0.22	0.08	0.12	0.19	0.08	0.03
Finances	1	0.45	0.04	0.98	0.13	0.02	0.29	0.06	0.05
Sports	0.16	0.27	0.01	1	0.02	0	0.11	0.03	0.01

where C_{Miss} means the cost of missing a new story, P_{Miss} means the probability of missing a new story, and P_{Target} means the probability of seeing a new story in the data; C_{FA} means the cost of a false alarm, P_{FA} means the probability of a false alarm, and $P_{Nontarget}$ means the probability of seeing an old story. The cost C_{Det} is normalized such that a perfect system scores 0 and a trivial system, which is the better one of mark all stories as new or old, scores 1:

$$Norm(C_{Det}) = \frac{C_{Det}}{min(C_{Miss} * P_{Target}, C_{FA} * P_{Nontarget})} \quad (13)$$

New event detection system gives two outputs for each story. The first part is "yes" or "no" indicating whether the story triggers a new event or not. The second part is a score indicating confidence of the first decision. Confidence scores can be used to plot DET curve, i.e., curves that plot false alarm vs. miss probabilities. Minimum normalized cost can be determined if optimal threshold on the score were chosen.

7. EXPERIMENTAL RESULTS

7.1 Main Results

To test the approaches proposed in the model, we implemented and tested five systems:

System-1: this system is used as baseline. It is implemented based on the basic model described in section 3, i.e., using incremental TF-IDF model to generate term weights, and using Hellinger distance to compute document similarity. Similarity score normalization is also employed [8]. S-S detection procedure is used.

System-2: this system is the same as system-1 except that S-C detection procedure is used.

System-3: this system is the same as system-1 except that it uses the new detection procedure which is based on indexing-tree.

System-4: implemented based on the approach presented in section 5.1, i.e., terms are reweighted according to the distance between term distributions in a cluster and all stories. The new detection procedure is used.

System-5: implemented based on the approach presented in section 5.2, i.e., terms of different types are reweighted according to news class using trained parameters. The new detection procedure is used.

The following are some other NED systems:

System-6: [21] for each pair of stories, it computes three similarity values for named entity, non-named entity and all terms respectively. And employ Support Vector Machine to predict "new" or "old" using the similarity values as features.

System-7: [8] it extended a basic incremental TF-IDF model to include source-specific models, similarity score normalization based on document-specific averages, similarity score normalization based on source-pair specific averages, etc.

System-8: [13] it split document representation into two parts: named entities and non-named entities, and choose one effective part for each news class.

Table 3 and table 4 show topic-weighted normalized costs and comparing times on TDT2 and TDT3 datasets respectively. Since no heldout data set for fine-tuning the threshold θ_{new} was available for experiments on TDT2, we only report minimum normalized costs for our systems in table 3. System-5 outperforms all other systems including system-6, and it performs only 2.78e+8 comparing times in detection procedure which is only 13.4% of system-1.

Table 3. NED results on TDT2

Systems	*Min Norm*(C_{Det})	*Cmp times*
System-1	0.5749	2.08e+9
System-2①	0.6673	3.77e+8
System-3②	0.5765	2.81e+8
System-4②	0.5431	2.99e+8
System-5②	0.5089	2.78e+8
System-6	0.5300	--

① θ_{new}=0.13

② θ_{init}=0.13, λ =3,δ =0.15

When evaluating on the normalized costs on TDT3, we use the optimal thresholds obtained from TDT2 data set for all systems. System-2 reduces comparing times to 1.29e+9 which is just 18.3% of system-1, but at the same time it also gets a deteriorated minimum normalized cost which is 0.0499 higher than system-1. System-3 uses the new detection procedure based on news indexing-tree. It requires even less comparing times than system-2. This is because story-story comparisons usually yield greater similarities than story-cluster ones, so stories tend to be combined

together in system-3. And system-3 is basically equivalent to system-1 in accuracy results. System-4 adjusts term weights based on the distance of term distributions between the whole corpus and cluster story set, yielding a good improvement by 0.0468 compared to system-1. The best system (system-5) has a minimum normalized cost 0.5012, which is 0.0797 better than system-1, and also better than any other results previously reported for this dataset [8, 13]. Further more, system-5 only needs 1.05e+8 comparing times which is 14.9% of system-1.

Table 4. NED results on TDT3

Systems	*Norm*(C_{Det})	*Min Norm*(C_{Det})	*Cmp times*
System-1	0.6159	0.5809	7.04e+8
System-2①	0.6493	0.6308	1.29e+8
System-3②	0.6197	0.5868	1.03e+8
System-4②	0.5601	0.5341	1.03e+8
System-5②	0.5413	0.5012	1.05e+8
System-7	--	0.5783	--
System-8	--	0.5229	--

① θ_{new}=0.13

② θ_{init}=0.13, λ =3,δ =0.15

Figure5 shows the five DET curves for our systems on data set TDT3. System-5 achieves the minimum cost at a false alarm rate of 0.0157 and a miss rate of 0.4310. We can observe that System-4 and System-5 obtain lower miss probability at regions of low false alarm probabilities. The hypothesis is that, more weight value is transferred to key terms of topics from non-key terms. Similarity score between two stories belonging to different topics are lower than before, because their overlapping terms are usually not key terms of their topics.

7.2 Parameter selection for indexing-tree detection

Figure 3 shows the minimum normalized costs obtained by system-3 on TDT3 using different parameters. The θ_{init} parameter is tested on six values spanning from 0.03 to 0.18. And the λ parameter is tested on four values 1, 2, 3 and 4. We can see that, when θ_{init} is set to 0.12, which is the closest one to θ_{new}, the costs are lower than others. This is easy to explain, because when stories belonging to the same topic are put in a cluster, it is more reasonable for the cluster to represent the stories in it. When parameter λ is set to 3 or 4, the costs are better than other cases, but there is no much difference between 3 and 4.

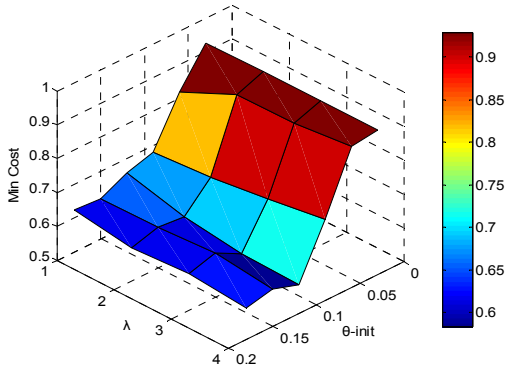

Figure 3. Min Cost on TDT3 (δ =0.15)

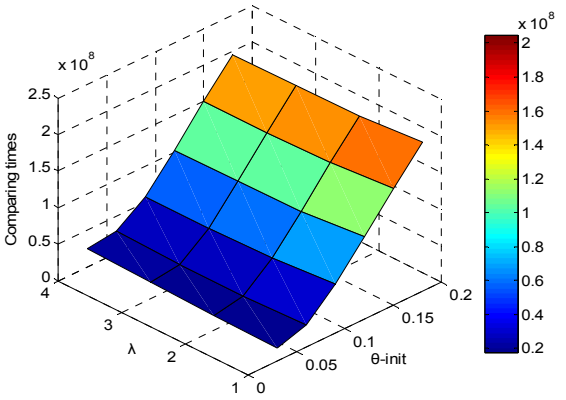

Figure 4. Comparing times on TDT3 (δ =0.15)

Figure 4 gives the comparing times used by system-3 on TDT3 with the same parameters as figure 3. The comparing times are strongly dependent on θ_{init}. Because the greater θ_{init} is, the less stories combined together, the more comparing times are needed for new event decision.

So we use θ_{init} =0.13,λ =3,δ =0.15 for system-3, 4, and 5. In this parameter setting, we can get both low minimum normalized costs and less comparing times.

8. CONCLUSION

We have proposed a news indexing-tree based detection procedure in our model. It reduces comparing times to about one seventh of traditional method without hurting NED accuracy. We also have presented two extensions to the basic TF-IDF model. The first extension is made by adjust term weights based on term distributions between the whole corpus and a cluster story set. And the second extension to basic TF-IDF model is better use of term types (named entities types and part-of-speed) according to news categories. Our experimental results on TDT2 and TDT3 datasets show that both of the two extensions contribute significantly to improvement in accuracy.

We did not consider news time information as a clue for NED task, since most of the topics last for a long time and TDT data sets only span for a relative short period (no more than 6 months). For the future work, we want to collect news set which span for a longer period from internet, and integrate time information in NED task. Since topic is a relative coarse-grained news cluster, we also want to refine cluster granularity to event-level, and identify different events and their relations within a topic.

Acknowledgments

This work is supported by the National Natural Science Foundation of China under Grant No. 90604025. Any opinions, findings and conclusions or recommendations expressed in this material are the author(s) and do not necessarily reflect those of the sponsor.

9. REFERENCES

[1] http://www.nist.gov/speech/tests/tdt/index.htm

[2] In *Topic Detection and Tracking. Event-based Information Organization*. Kluwer Academic Publishers, 2002.

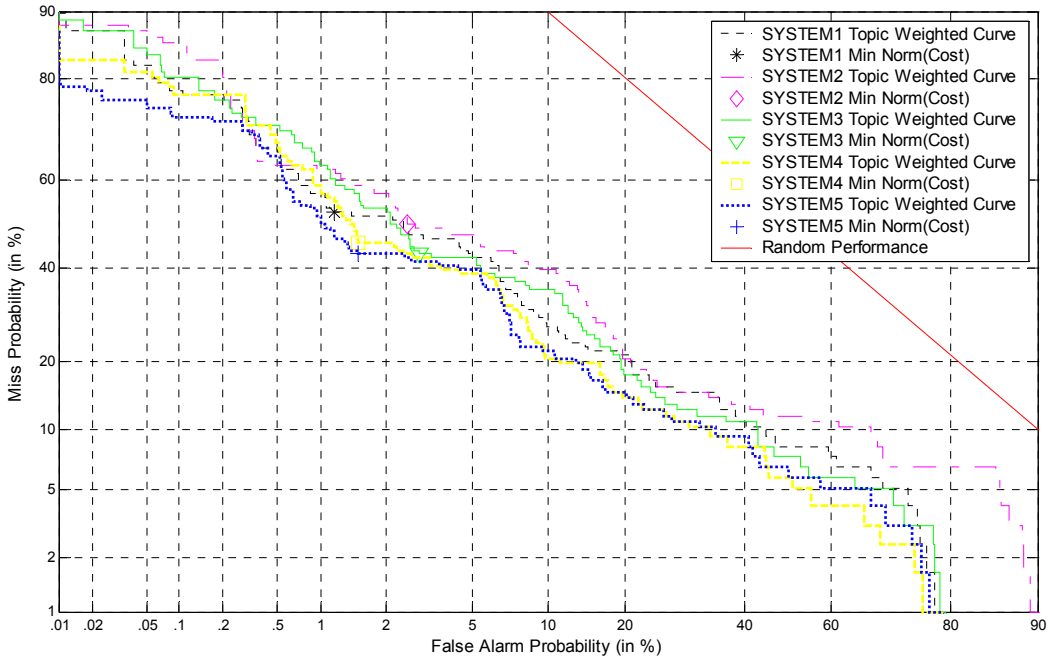

Figure 5. DET curves on TDT3

[3] Y. Yang, J. Carbonell, R. Brown, T. Pierce, B.T. Archibald, and X. Liu. Learning Approaches for Detecting and Tracking News Events. In *IEEE Intelligent Systems Special Issue on Applications of Intelligent Information Retrieval, volume 14 (4)*, 1999, 32–43.

[4] Y. Yang, T. Pierce, and J. Carbonell. A Study on Retrospective and On-line Event Detection. In *Proceedings of SIGIR-98, Melbourne, Australia*, 1998, 28-36.

[5] J. Allan, V. Lavrenko, D. Malin, and R. Swan. Detections, Bounds, and Timelines: Umass and tdt-3. In *Proceedings of Topic Detection and Tracking Workshop (TDT-3)*, Vienna, VA, 2000, 167-174.

[6] R. Papka and J. Allan. On-line New Event Detection Using Single Pass Clustering TITLE2:. Technical Report UM-CS-1998-021, 1998.

[7] W. Lam, H. Meng, K. Wong, and J. Yen. Using Contextual Analysis for News Event Detection. *International Journal on Intelligent Systems*, 2001, 525-546.

[8] B. Thorsten, C. Francine, and F. Ayman. A System for New Event Detection. In *Proceedings of the 26th Annual International ACM SIGIR Conference*, New York, NY, USA. ACM Press. 2003, 330–337.

[9] S. Nicola and C. Joe. Combining Semantic and Syntactic Document Classifiers to Improve First Story Detection. In *Proceedings of the 24th Annual International ACM SIGIR Conference*, New York, NY, USA. ACM Press. 2001, 424–425.

[10] Y. Yang, J. Zhang, J. Carbonell, and C. Jin. Topic-conditioned Novelty Detection. In *Proceedings of the 8th ACM SIGKDD International Conference*, ACM Press. 2002, 688–693.

[11] M. Juha, A.M. Helena, and S. Marko. Applying Semantic Classes in Event Detection and Tracking. In *Proceedings of International Conference on Natural Language Processing (ICON 2002)*, 2002, pages 175–183.

[12] M. Juha, A.M. Helena, and S. Marko. Simple Semantics in Topic Detection and Tracking. *Information Retrieval*, 7(3–4): 2004, 347–368.

[13] K. Giridhar and J. Allan. Text Classification and Named Entities for New Event Detection. In *Proceedings of the 27th Annual International ACM SIGIR Conference*, New York, NY, USA. ACM Press. 2004, 297–304.

[14] J. P. Callan, W. B. Croft, and S. M. Harding. The INQUERY Retrieval System. In *Proceedings of DEXA-92, 3rd International Conference on Database and Expert Systems Applications*, 1992, 78-83.

[15] R. Krovetz. Viewing Morphology as An Inference Process. In *Proceedings of ACM SIGIR93*, 1993, 61-81.

[16] Y. Yang and J. Pedersen. A Comparative Study on Feature Selection in Text Categorization. In J. D. H. Fisher, editor, *The Fourteenth International Conference on Machine Learning (ICML'97)*, Morgan Kaufmann, 1997, 412-420.

[17] T. M. Cover, and J.A. Thomas. *Elements of Information Theory*. Wiley. 1991.

[18] The linguistic data consortium, http://www.ldc,upenn.edu/.

[19] The 2001 TDT task definition and evaluation plan, http://www.nist.gov/speech/tests/tdt/tdt2001/evalplan.htm.

[20] R. E. Schapire and Y. Singer. Boostexter: A Boosting-based System for Text Categorization. In *Machine Learning 39(2/3):1*, Kluwer Academic Publishers, 2000, 35-168.

[21] K. Giridhar and J. Allan. 2005. Using Names and Topics for New Event Detection. In *Proceedings of Human Technology Conference and Conference on Empirical Methods in Natural Language*, Vancouver, 2005, 121–128