Hierarchical Topic Detection in TDT-2004

Ao Feng, James Allan
Center for Intelligent Information Retrieval
Department of Computer Science
University of Massachusetts
Amherst, MA 01003
{aofeng, allan}@cs.umass.edu

ABSTRACT

Huge volume of news makes it hard for people to keep up with the latest information, and automatic processing of news information becomes necessary. Topic Detection and Tracking is a research program that deals with this problem. From the observations in TDT, news topics can be described in different sizes, making it hard to define the "correct" granularity.

In TDT-2004, the topic detection task was replaced by a new task called hierarchical topic detection, which used a hierarchy to capture more possible granularities. This paper shows the task definition, evaluation schemes, our attempt to generate a proper hierarchy, comparisons of different participants, and detailed analysis of the results.

In the hierarchical structure, not all units satisfy the definition of topics. Our assumption is that the units between topics and stories are events, or sub-topics, and dependency analysis among events can give us better understanding of topics and other concepts. We propose a formal framework of event dependencies, but it still needs data collections, evaluation schemes and actual experiments to test its validity.

Categories and Subject Descriptors

H.3.3 **[Information Search and Retrieval]**: Clustering

General Terms

Algorithms, Experimentation, Measurement, Performance

Key words

Topic Detection and Tracking, Hierarchical Topic Detection, Clustering, Evaluation, Event, Dependency

1. INTRODUCTION

News is among the most important information sources. It comes in different formats - newspaper, television, radio, Web, etc. - and informs people all over the world what happens around them or on the other side of the earth. People like to keep abreast of the latest information but the huge volume of news makes it a big challenge. Automatic organization of news will facilitate people in retrieving information they are interested in. Topic Detection and Tracking (TDT) [3] is a research program that focuses on this problem.

The main idea of TDT is to process news streams and gather information in different news topics. The basic units are stories, where a story is defined as *a topically cohesive segment of news that includes two or more declarative independent clauses about a single event*. However, in TDT's five tasks - topic tracking, link detection, topic detection, first story detection and story segmentation, the main concept is "topic". Four of the five tasks focus on topics. But what is a topic? It is *a seminal event or activity along with all directly related events and activities*. Here "related" is ambiguous since different people may have different judgments on such a relation. Then we have a question to answer - how do we decide whether two stories are within the same topic or not?

To generate the relevance judgments for TDT collections, the Linguistic Data Consortium (LDC) has a detailed annotation guide to select topics and find on-topic stories [5]. However, even with all these rules and quality control, the process is still not obvious. Topics are developed starting from a randomly chosen story. The rules specify the seminal event and the scope of the topic, i.e., which events are considered directly related. Oddly, however, if one of the stories in a topic is selected at random and a new topic is defined from it, the new and original topics may not be the same. The process is open to enough interpretation that topics can be defined at various levels of granularity (a campaign stop, a campaign, a national set of elections) and it is not possible to know the chosen granularity without knowing which story the seed is.

To incorporate the idea of granularity, the task of hierarchical topic detection (HTD) replaced topic detection in TDT-2004. Participants in this task were no longer required to submit flat cluster partitions, but to generate a directed acyclic graph (DAG). Each node represents a topic at a specific granularity, and they can overlap or subsume each other. There were four participants for this task in the evaluation of TDT-2004, each with one or more runs.

From the evaluation results, annotated topics are usually located in the middle of a system's hierarchy. But what do the other units represent? The clusters larger than topics are super-topic subjects if topics are correctly combined. Similarly, units smaller than topics can be called sub-topics. But what are these sub-topics? We believe they correspond to events. The definitions of story and topic both mention event, a very important concept in TDT. An event "identifies something (non-trivial) happening in a certain place at a certain time" [12]. Each event should contain one or more stories that describe it, and one or more directly related events form a topic. Since events are components of topics, analyzing how events are organized in a topic can give us deeper understanding of the topic

content. It may not be too difficult to identify events from a text stream, but the relationship, or dependency, among events, is hard to define.

This paper will introduce some background information of the HTD task, our algorithm in the submission, comparisons and analysis of the results in TDT-2004, and argue the necessity to introduce events in the TDT framework. Section 2 discusses the related previous work in this research area. Section 3 gives the description of the HTD task, and Section 4 introduces the current evaluation method we are using. The descriptions of the algorithm in our submitted run and the evaluation results for all participants are in Section 5. Section 6 analyzes the results and evaluation measures, and reports more experiments based on that. Then we show the event structure within a news topic using some examples and define a formal event framework in section 7. Section 8 concludes our work and suggests the future directions.

2. RELATED WORK

The idea of hierarchical topic detection (HTD) was introduced in TDT-2002 to overcome the problems with topic granularity in the "truth" data that were described above. It is not possible for a system to reliably pick the "right" topic for the evaluation, since the randomly selected seed story is not known to the system. However, researchers in the program posited that if topics were specified at multiple levels of granularity simultaneously - i.e., in a hierarchical model - then the "right" topic should be somewhere in that hierarchy, whether it was fine or coarse grained.

A hierarchical structure is able to accommodate topics with different granularities, but there are still some key factors unsolved like a proper evaluation method. [2] pointed out that in the new model the old evaluation scheme could be easily cheated by some degenerate cases and proposed some new methods. Through experiments with some synthetic and real data collections, two algorithms were selected since they showed good performance, and one of them was adopted as the formal evaluation scheme in TDT-2004.

Hierarchical structures are also used in other applications, such as text classification. [4] built a hierarchical classifier based on a concept hierarchy. And [10] provided some evaluation methods in measuring the hierarchical classification performance. Most of the top-down classification algorithms start from categories, or subjects, but stop at the topic level.

When we look deeper into a topic, event organization becomes more important to understand its structure. But not much work has been done in this area. [7] regarded each topic as an interconnected network of events, and had some preliminary experiments. But it oversimplified the event dependency to "related" or "not related", which was not enough to catch the causal relations among events in most cases.

There has not been any formal framework of news event organizations in TDT, but some journalism knowledge can give us good hints. [11] defined a structure of a news schema, which described the necessary components of a complete news article. Although news stories in TDT have a different format, this structure can help us define event dependencies. Some relations in Section 7 come directly from the journalism background.

3. TASK DEFINITION

A new collection, TDT-5, was created for the TDT-2004 evaluation. It has around 400,000 news stories from three different languages - English, Mandarin and Arabic - about 4 times the size of the previous collection, and HTD uses the whole collection. The number of topics is also much larger, totally 250 topics annotated.

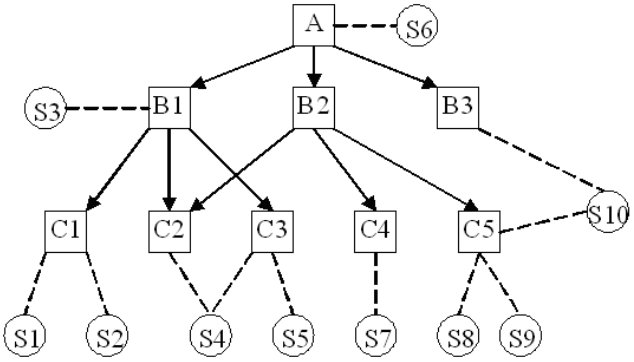

Figure 1: Example of a directed acyclic graph for hierarchical topic detection

In addition to the size, it has some other changes to previous collections. This collection contains only newswire stories, i.e., broadcast news are no longer used. Thus the story segmentation task, which applied only to broadcast news, did not appear in TDT-2004. There were four main tasks for 2004: link detection, tracking (including supervised adaptive tracking), new event detection, and hierarchical topic detection. The last one is a completely new task and no previous experiments are available, so it is regarded as a dry run.

HTD is the replacement of the topic detection task. We used to assume that every topic has a strict definition, so the boundary of each topic is determined once we know what it is about. We also assumed that a story can only belong to one topic. So a typical approach to topic detection is to divide the whole collection into a list of separate clusters, which is very similar to text classification. For HTD, we expect that a story can talk about two or more different news topics, and that the boundary of each topic is obscure. HTD removes the restrictions in topic detection and represents topics in a hierarchical structure.

The expected output of HTD is a DAG, and Figure 1 shows an example for a collection with 10 stories.

Each square is a cluster, and a circle represents a story. When talking about nodes in the DAG, we mean only the clusters. An arrow from cluster A to cluster B means B is a child of A. A dashed line between a story and a cluster means the story belongs directly to the cluster. From this example, we can see some attributes of the DAG:

- Each graph has a root node, which is an ancestor of all other clusters. It is cluster A in this example. We assume that cluster A contains all stories.
- A cluster has one or more parents (except the root node) and one or more children (except the leaf nodes). But no loop is allowed in the hierarchy.
- Each story belongs to one or more clusters, but it cannot be directly attached to a node and its ancestor at the same time.
- A cluster "contains" all stories contained by its children. So cluster B1 includes S1, S2, S3, S4 and S5.

It is not convenient to represent or process such a graph, so it is required to be converted to XML format in the task definition. The XML representation of figure 1 is shown in figure 2.

```
<htd rootVertex="A" system="toyv1">
<vertexSet>
  <vertex name="A">
    <story docID="S6">
  </vertex>
  <vertex name="B1">
    <story docID="S3">
  </vertex>
  <vertex name="B2">
  </vertex>
  <vertex name="B3">
    <story docID="S10">
  </vertex>
  <vertex name="C1">
    <story docID="S1">
    <story docID="S2">
  </vertex>
  <vertex name="C2">
    <story docID="S4">
  </vertex>
  <vertex name="C3">
    <story docID="S4">
    <story docID="S5">
  </vertex>
  <vertex name="C4">
    <story docID="S7">
  </vertex>
  <vertex name="C5">
    <story docID="S8">
    <story docID="S9">
    <story docID="S10">
  </vertex>
</vertexSet>
<edgeSet>
  <edge srcVertex="A" destVertex="B1"/>
  <edge srcVertex="A" destVertex="B2"/>
  <edge srcVertex="A" destVertex="B3"/>
  <edge srcVertex="B1" destVertex="C1"/>
  <edge srcVertex="B1" destVertex="C2"/>
  <edge srcVertex="B1" destVertex="C3"/>
  <edge srcVertex="B2" destVertex="C2"/>
  <edge srcVertex="B2" destVertex="C4"/>
  <edge srcVertex="B2" destVertex="C5"/>
</edgeSet>
<htd>
```

Figure 2: XML representation of HTD output

system output	relevance judgment	
	relevant	non-relevant
in cluster	R_+	N_+
not in cluster	R_-	N_-
total	r	n-r

Table 1: Distribution of stories for different judgments

4. EVALUATION

Since the evaluation method used in the topic detection task did not work in HTD, [2] introduced more evaluation schemes and two of them were proved to work for HTD. The minimal cost evaluation was selected as the official evaluation measure in TDT-2004. The expected travel cost method was not used because of its assumed computational cost.

The main evaluation measure in the topic detection task was detection cost. We describe it here because it is a key component of the HTD task's evaluation, too. Suppose that we want to compare a truth topic (cluster) from the annotation with some system-generated cluster to see how well they match up. Obviously a perfect system cluster would contain all and precisely the stories in the truth topic. We can count the number of relevant (on topic) stories that are put into the cluster, the number of relevant stories that are left out of the cluster, and so on. Table 1 depicts the possibilities.

Here R_+, N_+, R_-, N_- are the numbers of stories in each category, r is the number of on-topic stories, and n is the collection size. Two error rates - P_{miss} (missed detection rate) and P_{fa} (false alarm rate) - are defined based on this table.

$$P_{\text{miss}} = \frac{R_-}{r} \quad (1)$$

$$P_{\text{fa}} = \frac{N_+}{n-r} \quad (2)$$

The detection cost is a linear combination of these error rates.

$$C_{\text{det}} = C_{\text{miss}} P_{\text{miss}} P(\text{target}) + C_{\text{fa}} P_{\text{fa}} (1 - P(\text{target})) \quad (3)$$

where C_{miss} and C_{fa} are the costs of a missed detection and a false alarm respectively, and $P(\text{target})$ is the prior probability to find a relevant story. Each topic is mapped to the cluster with the minimal detection cost.

In the new HTD framework, this evaluation method would be deceived by a power set of all possible story combinations. Then every topic can find a cluster that contains exactly the on-topic stories, and the detection cost becomes zero. To avoid such a degenerate case, travel cost was introduced to compensate for it. The idea of the minimal cost evaluation is to find an optimal node in the hierarchy with a good trade-off between detection cost and hierarchy structure to get the smallest total cost. The formula to calculate the total cost is:

$$C_{\text{total}} = W_{\text{det}} C_{\text{det}} + (1 - W_{\text{det}}) C_{\text{travel}} \quad (4)$$

where W_{det} is used to assign weights between C_{det} (detection cost) and C_{travel} (travel cost). The travel cost is defined as the effort a user spends to travel from the root node to the current one. It is composed of two parts. $CTITLE$, the cost to examine a node in the hierarchy, is used to penalize "deep" hierarchies. And $CBRANCH$, the cost to generate each child of the current node, is to avoid large fan-out, i.e., "wide" nodes. Details of the travel cost can be found in [2].

Parameter	Value
P(target)	0.02
C_{miss}	1.0
C_{fa}	0.1
W_{det}	0.66
$CBRANCH$	2.0
$CTITLE$	1.0
$OPTBR$	3
$AVESPT$	88
$MAXVTS$	3

Table 2: Parameter settings for HTD evaluation in TDT-2004

To make the costs comparable, both the detection cost and the travel cost are normalized before they are linearly combined to calculate the total cost. C_{det} is normalized over the smaller cost of two extreme cases: no misses, 100% false alarm and 100% misses, no false alarm.

$$C_{det-norm} = C_{\text{det}}/min(C_{\text{miss}}P(\text{target}), C_{\text{fa}}(1 - P(\text{target}))) \quad (5)$$

The range of the travel cost depends highly on the collection size and is not comparable to the detection cost, so normalization is also necessary to this component. There are two normalization schemes for the travel cost. The first is to normalize C_{travel} over the travel cost of the optimal hierarchy, which has singleton leaf nodes, optimal branching factor and non-overlapping clusters.

$$C_{travel-norm1} = C_{\text{travel}}/((CBRANCH * OPTBR + CTITLE) * \log_{OPTBR} numStories) \quad (6)$$

here $OPTBR$ is the optimal branching factor, decided by $CBRANCH$ and $CTITLE$. $numStories$ is the number of stories in the whole collection. Since all leaf nodes are singletons, it is also the number of leaf nodes.

The second method assumes a one-level DAG. All stories are divided into clusters of equal sizes (the average number of stories in each topic) and each story is allowed in multiple clusters, then all clusters are directly attached to the root node.

$$C_{travel-norm2} = C_{\text{travel}}/(CBRANCH * MAXVTS * numStories/AVESPT + CTITLE) \quad (7)$$

$MAXVTS$ is the number of clusters each story is assigned to, and $AVESPT$ is the observed average number of stories per topic.

The parameter settings in the TDT-2004 evaluation are in Table 2. Equation 6 was the original normalization scheme in the evaluation plan but it was replaced by Equation 7 after some preliminary experiments. In section 6 we will have some analysis of them.

5. EXPERIMENTS

The TDT-5 collection contains 407,503 stories, about four times the size of TDT-4. For such a large collection, most traditional algorithms with $\Omega(n^2)$ time complexity are not applicable. The easiest algorithm to generate a hierarchy is agglomerative clustering, but it will generate a 407,503*407,503 similarity matrix, which is expensive in both time and space. So our aim for the HTD dry run is to build a system that is fast enough to process the whole collection in a reasonable amount of time, and the performance should be acceptable.

Experiments in [7] show that stories in the same event tend to be close in time, which is useful to solve the complexity problem. Since we only need the on-topic stories in a cluster, and each topic is composed of one or more events, grouping stories into events can reduce the number of units. And time locality of event organization tells us we only need to compare an incoming story to a limited number of previous stories.

The first step of our submitted run is event threading that uses a bounded 1-NN algorithm. 75% topics in TDT-5 are in one language, and we had the observation that stories from the same source are more likely to be in the same event. So we divide the whole collection into 15 source-language classes:

English AFE, APE, NYT, XIE, LAT, UME, CNE

Mandarin XIN, AFC, CNA, ZBN

Arabic AFA, XIA, ANN, UMM

We use the machine translated English version for Mandarin and Arabic stories.

Stories in each class are sorted by time order, and we run a single pass clustering algorithm to build events. Each incoming story is compared to a certain number of previous stories, and this number is approximately the number of stories in a token file[1]. If the similarity (cosine similarity of tf-idf term vectors) between the current story and the most similar previous stories is over the given threshold, the story is assigned to the corresponding event. Otherwise, a new event is created for this story.

We will get a list of events after the first step, but the number of units is still too large for agglomerative clustering. We have an assumption that topics also have the time locality attribute, then we can agglomerate events in small subsets.

All events in the same source-language class are sorted by time order, using the time stamp of the first story. Then a certain number of events are taken out for agglomeration. At each cycle, the closest event pair is taken out and merged. We observed in our preliminary experiments that larger clusters always had more chance to be merged, so we modified the similarity calculation to favor smaller clusters.

$$sim(cluster_1, cluster_2) = \frac{cos(vector_1, vector_2)}{|cluster_1|} \quad (8)$$

where $vector_1$ and $vector_2$ are the centroid vectors of the clusters, and $|cluster_1|$ is the size of the first cluster, i.e., the cluster with an earlier first story. We always combine the later event to the earlier one to preserve the time order.

We know how to get the optimal branching factor in [2], and the value for the parameter settings in Table 2 is 3. The agglomeration process continues until the number of clusters is 1/3 of the original size. In order not to miss those large clusters, only the first half of clusters are finalized and removed from the agglomeration window, and new events are brought in to continue the agglomerative clustering. When all events are used up, the number of clusters should be one third of the events.

Now the list of clusters can be clustered again used the same agglomeration algorithm, and the process continues until the number of clusters is smaller than a given value, which is proportional to the square root of the source-language class size. Then different

[1]The collection is available in multiple SGML token files, each containing the stories from a specific source in a certain period of time.

sources in the same language are merged and one round of agglomeration is carried out. Finally clusters in all three languages are put together and clustered until there is only one cluster left.

There are a lot of singleton clusters in the hierarchy. They are replaced by the corresponding story to reduce travel cost.

We submitted three runs with different parameter settings in the official evaluation, and the parameters used in our best run (UMass1) are:

K-NN bound The number of previous stories each incoming story is compared to. 100

threshold If the maximal similarity is below this value, a new event is created. 0.30

window size The maximal number of clusters in the agglomerative clustering. 240

stop The number of clusters to stop clustering in a source-language class. $5\sqrt{numStories}$

All previous TDT collections, including TDT-2, TDT-3 and TDT-4, can be used as the training set. We choose TDT-4 since it is the most recent corpus, and its news content is more similar to TDT-5 than earlier collections. TDT-4 contains both newswire and broadcast news, but TDT-5 has only newswire stories. So we train the parameters on the newswire subset of TDT-4, which is the most similar to the test set.

There are four participants of the HTD task in TDT-2004.

TNO Netherlands Organization for Applied Scientific Research TNO, Netherlands

ICT Institute of Computing Technology, Chinese Academy of Sciences, China

UMass University of Massachusetts Amherst, USA

CUHK The Chinese University of Hong Kong, China

The ICT system does agglomerative clustering to build micro-clusters, then many rounds of single pass clustering are carried out to form the hierarchy. At each level, the threshold decreases by 0.10 from the previous value. It uses tf-idf term weighting, and time decay is also considered in addition to cosine similarity.

CUHK uses K-means to cluster stories into general topics, then the general topics are divided into specific ones with divisive centroid clustering.

TNO outperformed others in the evaluation, and let us see what TNO did[2]. First, a subsample of 20,000 stories is taken from the TDT-5 collection, and the similarity matrix is built using symmetrical document likelihood [9]. Then a binary hierarchy is created via agglomerative clustering, and the cluster similarity is calculated by average pair wise linking. The hierarchy is simplified to reduce travel cost, and every remaining story is compared to all stories in the subsample and assigned to the 10 most similar leaf nodes (each leaf node in the hierarchy contains exactly one story in the subsample).

The evaluation results of the multilingual run for all participants of HTD are shown in Table 3.

The running time of our system[3] is about one day, which is not bad considering the collection size. We do not have much information of others' time complexity, but our simulation of TNO's algorithm takes about a week under the same running condition.

[2]The details of the algorithm were acquired via email exchange.

[3]On a Sun server with 16GB RAM and 16 64-bit Sparc CPUs at 950Mhz. The program uses only one CPU and takes about 2GB memory.

Participant	TNO	ICT	UMass	CUHK
Best system	TNO3	ICT1e	UMass1	CUHK1
P_{miss}	0.0196	0.1062	0.2761	0.3683
P_{fa}	0.0042	0.0031	0.0030	0.0058
C_{det}	0.0008	0.0024	0.0058	0.0079
$C_{det-norm}$	0.0403	0.1212	0.2910	0.3969
C_{travel}	75.7	2595.0	176.2	1336.1
$C_{travel-norm1}$	0.9201	31.5279	2.1416	16.2331
$C_{travel-norm2}$	0.0027	0.0934	0.0063	0.0481
C_{total}	0.0275	0.1118	0.1942	0.2783

Table 3: Evaluation results of HTD in TDT-2004, participants are sorted by the total cost in increasing order

6. ANALYSIS

From Table 3, the systems in the HTD official evaluation differ a lot in performance. If we only compare the total cost, ICT, UMass and CUHK are within the same range, but TNO is much better than others. When looking at the detailed items, we can see:

- C_{total} is mainly decided by the detection cost. TNO and ICT have small detection costs, especially TNO that gets only 1/10 the cost of CUHK.
- The normalization scheme in Equation 7 (norm2) gets very small normalized travel cost. It can almost be neglected for TNO and UMass.
- C_{travel} of the ICT and CUHK systems are over ten times larger than the other two. If we use the normalization scheme in Equation 6 (norm1), their normalized travel costs will be outrageous.
- TNO has a very small detection cost mainly because of the small P_{miss}, and its P_{fa} is not better than others.

TNO is the only participant that allows one story to be assigned to multiple clusters. We believe that contributes heavily to the small P_{miss} since more clusters can contain the same story, thus increasing the chance for a cluster to include most on-topic stories. Of course it will increase false alarms, but the number of off-topic stories is much larger, and P_{fa} will not change much. Table 3 shows that P_{fa} of the TNO system is just a little over ICT and UMass. It explains the "tiny" detection cost in comparison to others.

Since the TNO hierarchy is binary, the branching factor is always 2. It also benefits the detection cost since it can keep most of the possible granularities in the hierarchy, and the chance to find a cluster that is close to the "truth" topic is much larger.

There is another advantage of the small branching factor. Although the optimal branching factor for the current parameter setting is 3, the travel cost will not increase a lot if we change it to 2. But if it is over 10, it will definitely hurt C_{travel}. The huge travel costs of ICT and CUHK are probably caused by that.

Our system does well in travel cost, but the detection cost is not satisfactory. The assumption of temporal locality is valid for events, but it may not be true for topics. Another problem is that clusters in different source-language classes may be combined too late.

We had more experiments after the official submission, and the performance is listed in Table 4.

UMass25, UMass29, UMass30 are modified versions of our submitted run, and UMass32 is a simple implementation of TNO's system.

System	UMass25	UMass29	UMass30	UMass32
P_{miss}	0.2435	0.1831	0.1416	0.0365
P_{fa}	0.0011	0.0076	0.0074	0.0057
C_{det}	0.0050	0.0044	0.0036	0.0013
$C_{det-norm}$	0.2490	0.2205	0.1777	0.0646
C_{travel}	468.9	404.6	400.5	389.7
$C_{travel-norm1}$	5.6970	4.9162	4.8660	4.7356
$C_{travel-norm2}$	0.0169	0.0146	0.0144	0.0140
C_{total}	0.1701	0.1505	0.1222	0.0474

Table 4: Evaluation results for more systems after the official submission, sorted by system name

UMass25 Window size is increased to 480, and clusters in the same language are merged after one round of clustering.

UMass29 It has all the changes in UMass25. The maximal branching factor of the hierarchy is limited to 6, and each cluster is merged to at most 2 other clusters in the agglomerative clustering process.

UMass30 Similar to UMass29 but the window size is 960.

UMass32 A simulation of the TNO system. Cosine similarity matrix is calculated for a subsample of 20,000 stories. And we do agglomerative clustering in the subsample to form a binary hierarchy. The simplification step in the TNO system is skipped. And each story not in the subsample is also assigned to the top 10 leaf nodes.

Comparing the results in Table 4 to our official run, we can see that a larger window size and earlier merge can help improve the performance. Overlapping clusters and limited branching factor also work, just as shown by the TNO result. Our simulation of the TNO system gets excellent performance, but there is still some distance to the TNO run. It may be evidence that the similarity function has some effect on the performance.

The normalization method in Equation 6 (norm1) is very strict since systems not paying much attention to the branching factor will be penalized a lot, just like ICT ant CUHK in Table 3. That is the reason why it was replaced after the preliminary experiments. The official normalization scheme for the travel cost is in Equation 7 (norm2). However, it does not work well because the normalized travel cost is too small to have much influence on the total cost. And it can be easily cheated by some degenerate case as follows.

In the 250 topics of TDT-5, only 20 (8%) have more than 100 stories. We can generate all possible combinations of 1 to 100 stories.

$$\sum_{i=1}^{100} \left(\begin{array}{c} i \\ 407,503 \end{array} \right) < 100 \left(\begin{array}{c} 100 \\ 407,503 \end{array} \right) = 1.09 \times 10^{405} \quad (9)$$

Then we will use them as leaf nodes and build a balanced binary tree. For 92% topics with at most 100 stories, a "perfect" cluster can be found in the hierarchy, so the detection cost is 0!!! For the other 8%, we can prepare an empty cluster and a cluster that contains the whole collection, then the detection cost after normalization will be at most 1. So, the average normalized detection cost is at most 0.08. The travel cost after normalization will be

$$\frac{(2 \times 2 + 1) \log_2 1.09 \times 10^{405}}{2 \times 3 \times 407,503/88} = 0.2421 \quad (10)$$

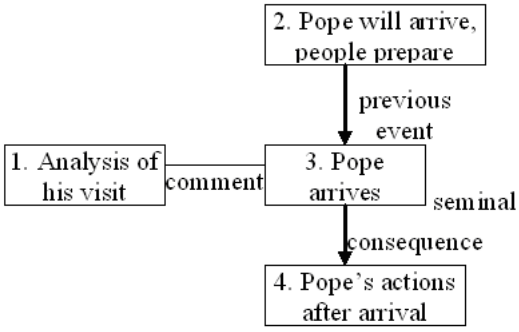

Figure 3: Event structure of topic 20012: Pope visits Cuba

Such a system output is so huge that it can not even be carried by any hard drive that the current technology is able to manufacture, but its total cost is at most

$$0.66 \times 0.08 + (1 - 0.66) \times 0.2421 = 0.1351 \quad (11)$$

which is comparable to most of the submitted runs in Table 3.

For the HTD evaluation, we prefer the normalization algorithm in Equation 6 with some changes in the parameter setting. Since the normalized travel cost is large in comparison to the detection cost, we can set a smaller weight for this part of the total cost. TNO won the evaluation this year with a binary tree, but the depth is over 40. Such a deep hierarchy is not realistic in most actual systems, so we suggest that CTITLE be increased and CBRANCH decreased so that we can have a larger optimal branching factor.

Another choice for the evaluation method is the expected travel cost in [2]. It was not adopted because of the time complexity. Now we have a mark-up version of it that reduces the computational cost. The running time for the mark-up algorithm is short enough to be used in the HTD evaluation.

7. EVENT FRAMEWORK

TDT started with a pilot study [1] in 1997 and continued with open annual evaluations for 7 years until TDT-2004 [6, 8]. In all these years, the key concept was always "topic". In the HTD experiments, we notice that topics are in the middle of the hierarchy. For example, in the UMass32 run, the hierarchy depth is 168, and the depth of annotated topics ranges from 37 to 152. The super-topics are subjects, which have been thoroughly studied in text classification. But the sub-topics, or events, need more research. From the definition of a topic, it relies directly on the seminal event and "related" events. To understand the concept better, we need deeper insight into the event structure.

[7] had an attempt to reveal the intrinsic structure of a news topic. It defined causal dependency as *occurrence of event B is related to and influenced by the occurrence of event A*. It also had some preliminary experiments to catch the dependencies among events. But the definition of causal dependency is still obscure. How can we tell if two events are related, and what kind of influence is it?

We can start from a simple topic. Topic 20012 in collection TDT-2 describes the Pope's historic visit to Cuba in the winter of 1998, and the event structure is in Figure 3.

This topic is composed of 4 events. Event 3 is the seminal event, which is the key event in the structure. If event 3 is removed, we can see that the structure will become three separate parts. If Pope

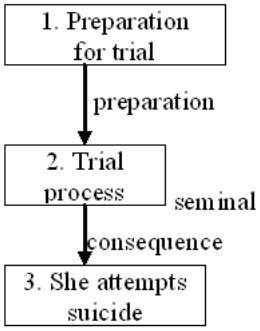

Figure 4: Event structure of topic 20022: Diane Zamora

does not intend to arrive, the visit will not happen at all, so there is no need to prepare, nothing to analyze, and nothing can happen after his "arrival". And we can see that event 1, 2 and 4 are all connected to the seminal event by a line, with or without arrows. Each line represents some kind of dependency, and the arrow shows the time order of events.

Similarly to topics, events can also have different granularities. There are smaller events within each event shown in Figure 3.

Event 1 The visit's impact on Catholicism in Cuba.

Event 2 Castro welcomes Pope, Pope will arrive soon, Catholics waiting for Pope, people's preparation.

Event 3 Pope arrives at Cuba.

Event 4 First day, Pope urges Cuba and the world to open to each other, has a mass in Santa Clara, and meets Castro. Second day, he has a mass in Camagury, objects embargo, and visits University of Havana. Third day, another mass is held in Santiago, and exiles try to get back to Cuba.

The relations between these small events are hard to define, so we can treat them as parallel events.

We can have more ideas with another example. Topic 20022, also in TDT-2, is about the trial of a naval midshipman. Diane Zamora is convicted of helping to murder her lover's girlfriend, and she attempts suicide after being sentenced to prison for life. The event structure is in Figure 4.

The event structure is relatively simple. Event 2 is the seminal event, event 1 is its preparation, and event 3 is the consequence. Note that the relation between event 1 and 2 is "preparation" instead of "previous event" because event 1 is a necessary precondition of event 2. There are also smaller events in the topic. The preparation includes the criminal charge and jury selection. And the trial is composed of 5 consecutive days and the final sentence, with a series of events happening on each day.

From our observation of the existing TDT collections, we are trying to define an event framework. Each topic is composed of one or more top level events, and they form an interconnected graph. An edge in the graph is a binary relation, representing some causal dependency between two events. Some definitions are listed below:

Seminal event The event that contains the main idea of the topic. Most of other events should be directly related to it. Usually there is only one seminal event in each topic, but in some cases there can be two or more.

Previous event An event that precedes the seminal event it time. Usually it is not a logical premise of the seminal event.

Preparation An event that precedes the seminal event in time, and it is also a precondition of the seminal event.

Consequence An event that is later than the seminal event. The result of the seminal event leads it to happen directly. In another word, the seminal event is the logical premise of it.

Background Usually far from the seminal event in time or does not have a clear time stamp. It contains some information about a proper name in the seminal event, like a person, an organization, a location, etc.

Comment Evaluation or response to some event in written form. The ordinary format is newspaper.

Verbal reaction Oral feedback to something, usually via broadcast or TV.

Reaction The response to some event that happens in the real world. It is different from comment or verbal reaction since some concrete action must be taken.

The names of some dependency types above are from [11], and others come from our annotation experiments. These relation types should be able to cover most dependencies within the event structure. However, there are more detailed relations in specific domains. Suppose we are talking about a natural disaster topic. The seminal event is the occurrence of the disaster, and there should be damages, rescue attempts, external aids and recovery from disaster. These relations are specific to this domain but very useful to form the event framework for such a topic.

In fact, the top level events do not agree well with the definition of events. Each of them usually contains several time-specific events, and sometimes intermediate events serve as bridges between the top level event and the "real" events. So we expect a hierarchy for each topic, and the clustering algorithms in Section 5 can be good candidates to generate it.

8. CONCLUSIONS AND FUTURE WORK

Starting from the pilot study, "topic" has been the main unit in TDT. Obscurity in the definition makes it hard to set the border of a topic, and different granularities are all acceptable in some sense. To incorporate this idea, HTD was introduced to replace the old topic detection task in TDT-2004. The evaluation results for the participants in HTD differ a lot, and TNO obviously outperforms others. We analyze the reason why TNO is successful and design more experiments to explore those reasons. Results of the experiments help us understand the hierarchical structure.

In the hierarchies, topics are intermediate units, and smaller clusters under each topic should be events. To have a better understanding of topics, we need the event structure for each topic. Some examples show the event framework and we suggest how to move beyond earlier work [7] by giving detailed definitions of dependency types.

There is an opinion in the TDT research community that the status of topics should be replaced by events. Our work provides a good background for this research direction, but we still need a lot of work to implement it. Currently there is no collection annotated based on events, and the evaluation scheme for the event structure is not available yet. Further experiments are also necessary to test the idea.

Acknowledgments

This work was supported in part by the Center for Intelligent Information Retrieval and in part by SPAWARSYSCEN-SD grant number N66001-02-1-8903. 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] J. Allan, J. Carbonell, G. Doddington, J. Yamron, and Y. Yang. Topic detection and tracking pilot study: Final report. In *Proceedings of the DARPA Broadcast News Transcription and Understanding Workshop*, pages 194–218, 1998.

[2] J. Allan, A. Feng, and A. Bolivar. Flexible intrinsic evaluation of hierarchical clustering for TDT. In *Proceedings of the ACM Twelfth International Conference on Information and Knowledge Management*, pages 263–270, Nov 2003.

[3] James Allan, editor. *Topic Detection and Tracking:Event based Information Organization*. Kluwer Academic Publishers, 2000.

[4] Wesley T. Chuang, Asok Tiyyagura, Jihoon Yang, and Giovanni Giuffrida. A fast algorithm for hierarchical text classification. In Yahiko Kambayashi, Mukesh Mohania, and A. Min Tjoa, editors, *Proceedings of DaWaK-00, 2nd International Conference on Data Warehousing and Knowledge Discovery*, pages 409–418, London, UK, 2000. Springer Verlag, Heidelberg, DE.

[5] Christopher Cieri, Stephanie Strassel, David Graff, Nii Martey, Kara Rennert, and Mark Liberman. Corpora for topic detection and tracking. In James Allan, editor, *Topic Detection and Tracking: Event-based Information Organization*, pages 33–66. Kluwer Academic Publishers, Boston, 2002.

[6] DARPA, editor. *Proceedings of the DARPA Broadcast news Workshop*, Herndon, Virginia, February 1999.

[7] R. Nallapati, A. Feng, F. Peng, and J. Allan. Event threading within news topics. In *Proceedings of the ACM Thirteenth International Conference on Information and Knowledge Management*, pages 446–453, Nov 2004.

[8] NIST. Proceedings of the TDT 2004 workshop. Notebook publication for participants only, December 2004.

[9] Martijn Spitters and Wessel Kraaij. TNO at TDT2001: Language model-based topic detection. In *Topic Detection and Tracking (TDT) Workshop 2001*. NIST, November 2001.

[10] Aixin Sun and Ee-Peng Lim. Hierarchical text classification and evaluation. In *ICDM*, pages 521–528, 2001.

[11] Teun A. van Dijk, editor. *News as Discourse*. Lawrence Erlbaum Associates, Publishers, 1988.

[12] Yiming Yang, Jaime Carbonell, Ralf Brown, Thomas Pierce, Brian T. Archibald, and Xin Liu. Learning approaches for detecting and tracking news events. In *IEEE Intelligent Systems Special Issue on Applications of Intelligent Information Retrieval*, volume 14 (4), pages 32–43, 1999.