Information Retrieval manuscript No.
(will be inserted by the editor)

Online Community Search Using Conversational Structures

Jangwon Seo · W. Bruce Croft · David A. Smith

the date of receipt and acceptance should be inserted later

Abstract Online communities are valuable information sources where knowledge is accumulated by interactions between people. Search services provided by online community sites such as forums are often, however, quite poor. To address this, we investigate retrieval techniques that exploit the hierarchical thread structures in community sites. Since these structures are sometimes not explicit or accurately annotated, we introduce structure discovery techniques that use a variety of features to model relations between posts. We then make use of thread structures in retrieval experiments with two online forums and one email archive. Our results show that using thread structures that have been accurately annotated can lead to significant improvements in retrieval performance compared to strong baselines.

Keywords Online Community · forum search · thread structure

1 Introduction

Many applications now exist on the Internet where people of different ages, in different locations, and with different backgrounds share their ideas and experiences in online spaces. Online communities (e.g., newsgroups, BBS, and forums) appeared with the Internet, and have accounted for a significant portion of online activities. Although a few of them have become somewhat obsolete, most still function as effective tools for establishing social networks and sharing knowledge.

Online communities are good information sources since knowledge shared by communities has accumulated for years. For some types of questions, online community archives can be primary sources for answers. Although community-based question-answer (CQA) services like "Yahoo! Answers"[1] have become popular, this is making use of only a part of community-based knowledge. Further, these services often deal with many broad topics at a shallow level because such services are designed to instantly provide answers to questions rather than to encourage users to participate in

Jangwon Seo · W. Bruce Croft · David A. Smith
Center for Intelligent Information Retrieval, University of Massachusetts, Amherst, 01003
E-mail: {jangwon, croft, dasmith}@cs.umass.edu

[1] http://answers.yahoo.com/

discussion. In contrast, online communities are virtual spaces in which experts from a specific field gather and discuss in depth a variety of topics in the field. Further, there are many communities covering many fields. In particular, communities for sharing information about hobbies such as online games, as well as technical communities, are very active and often provide high-quality information which cannot be obtained from other sources.

Search engines, however, have generally overlooked these online community resources. Online community page search results returned from major search engines are often low quality. Internal search services that are provided by the forums are sometimes even worse. One reason for this is that online forum pages are not the same as general web pages. Our goal is to design effective retrieval models that incorporate online forum properties so that the effectiveness of forum search can be improved.

Online community pages have many unique textual or structural features that distinguish them from general web pages. Generally, a forum has several sub-forums covering high-level topic categories. Each sub-forum has many threads. A thread is a more focused topic-centric discussion unit and is composed of posts created by community members. A thread can also be viewed in terms of dialogue structure. A general web page is a monologue where the utterance is a one-way communication by the page's creator. A CQA "document", which consists of a question and the replies, is a special case of a dialogue where the number of utterances per participant is typically limited to one. In contrast, many-to-many conversations occur frequently in forum threads. This is an advantageous feature that encourages in-depth discussion, compared to general web pages or CQA services.

In this paper, we set two goals based on the dialogue aspects of threads for online community search. The first goal is to discover and annotate thread structures which are based on interactions between community members. In some community sites, thread structure is explicitly annotated. In many others, however, the annotation is missing or inaccurate. We introduce and evaluate techniques that learn to annotate thread structure based on various features that reflect aspects of interactions between posts.

The second goal is to improve retrieval performance for online community search by exploiting the thread structure. We introduce retrieval models that incorporate thread structures and investigate the effects of threads on retrieval performance. The new retrieval techniques are evaluated using test collections created from two online forums and an email archive.

In the next section, we describe related work. Section 3 describes the problem of thread discovery, important features of community sites, and techniques for automatic annotation of thread structure based on those features. This section also reports the results of experiments measuring the accuracy of the discovery techniques. Section 4 describes approaches to thread search and post search in community sites, and the results of retrieval experiments.

2 Related Work

This paper is unique in combining two strands of work, namely thread structure discovery and retrieval models using the structure. Each subtask has been explored separately in different research areas.

The discovery of conversation structures in online communities has been addressed by many researchers. Particularly, most studies have focused discovering thread structures in email corpora. Lewis and Knowles [13] are among the first who have focused on threading email conversations. Smith et al. [23] proposed a new program design to address threaded chats. Yeh and Harnly [28] and Erera and Carmel [10] discussed similarity matching techniques for email thread detection. Shrestha and McKeown [22] introduced techniques for finding question-answer (QA) pairs in an email conversation for email summarization. Carvalho and Cohen [4] focused on more general acts in emails such as request, propose, data, and so on.

There are similar attempts in domains other than emails. Cong et al. [5] also investigated finding QA pairs in online forums. One of the purposes of finding these pairs in online forums is to augment CQA archives. While the amount of data for CQA is limited, there are plenty of forums. If we systemically extract QA pairs from forums, then we can significantly expand the coverage of CQA. On the other hand, in our work we focus on finding thread structure to improve retrieval performance. Therefore, we need to look at all relations between posts beyond just QA pairs.

The discussion of thread structure recovery in newsgroup style conversations by Wang et al. [26] is similar to our work, but is limited in that it used a few simple similarity features and did not show the applicability to retrieval tasks. Recently, Lin et al. [14] modeled semantics and structures of threads by minimizing a loss function based on assumptions for sparsity of topics and reply relations. Although their approach can model thread structures as well, it must be re-optimized whenever a new thread, particularly on a new topic, is created. Therefore, the approach is impractical when thread structure discovery is required in dynamic online communities.

In addition, Elsner and Charniak [9] and Wang and Oard [25] studied conversation disentanglement in online chat dialogues. Although their work is similar to our work in that they investigated structures of discourses using various features, they focused only on an online chat corpus.

Retrieval using threads involves combining information from different features. Retrieval models using multiple contexts or structures are frequently discussed in information retrieval literature. For example, Ogilvie and Callan [17] studied hierarchical language models for XML search, and Liu and Croft [15] introduced cluster-based retrieval using language models.

In the literature of social media search, Elasa et al. [8], Arguello et al. [1] and Seo and Croft [20] focused on how posts and an entire blog can be used for blog site search using resource selection techniques. This work can be considered complementary since we address techniques for another type of social media that is based on posts and threads. Xi et al. [27] learned ranking functions for Newsgroup search combining various features through discriminative learning techniques. Although some of their thread structure-based features are similar to ours, our approach is quite different from theirs in terms of retrieval models and evidence combination. Elsas and Carbonell [7] reviewed techniques for thread search. However, they do not consider structural features in contrast to our work. In addition, many research groups that participated in the email discussion search task in the TREC enterprise track [24] have shown that exploiting threads is effective for email search. Our work extends this result by exploiting more fine-grained thread structure.

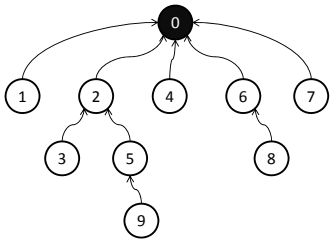

Fig. 1 Example of a thread structure

3 Discovery of Thread Structure

A thread is started on some subject by an initiator and grows as people discuss the subject. Since the first post of the initiator is usually displayed on the top of a thread, we call it the top post. The top post can be any utterance which requires interaction with people, e.g., a question, a suggestion, a claim, or a complaint. If they are interested in the subject of the top post, people post their opinions in reply posts. The reply posts can be any reaction to the top post, e.g., an answer, agreement, disagreement, advice, or sometimes an additional question. Often, a reply post may elicit its own replies. This is a typical phenomenon of a discussion in a thread. Because not all reply posts reply to the top post, many branches (sub-threads) of discussion appear in a thread, and a thread ends up with a tree-shaped structure. We refer to this as a thread structure. Figure 1 shows an example of a manually annotated structure of a thread, where a node represents a post, an arc represents a reply relation between two posts, and each number is a chronological order. That is, the child post with the outgoing arc replies to the parent post with the incoming arc.

Not all online communities, however, handle threads in the same manner. There are generally two ways that online communities maintain or display threads: flat-view and threaded-view. Flat-view systems, as their name implies, flatten structures of threads and show users all posts in a thread in chronological order. On the other hand, threaded-view systems allow a user to choose a preceding post to reply to, and display posts in structured views. Figure 2 shows a user-view example of a threaded-view system.

The flat-view looks natural because it resembles aspects of real conversations. Further, the flat view is sometimes more readable than the threaded-view. In particular, if a thread is very long, then it may be difficult for people to grasp all contents of threads in complicated structured views.

On the other hand, if we want to know how discussions flow or how posts interact, the threaded view is more helpful. In particular, if a thread is large, the thread may address many topics, each slightly different to each other. Then we can split the thread into smaller topical units according to the branches of the thread structure. An atomic topical unit such as a passage is known to be useful for information retrieval. Additionally, the threaded view appears to be suitable for social media analysis tasks such as expert finding.

Currently, flat-view online community pages are still much more prevalent although some online communities have emerged that use the threaded view, such as Slashdot[2]

[2] http://slashdot.org/

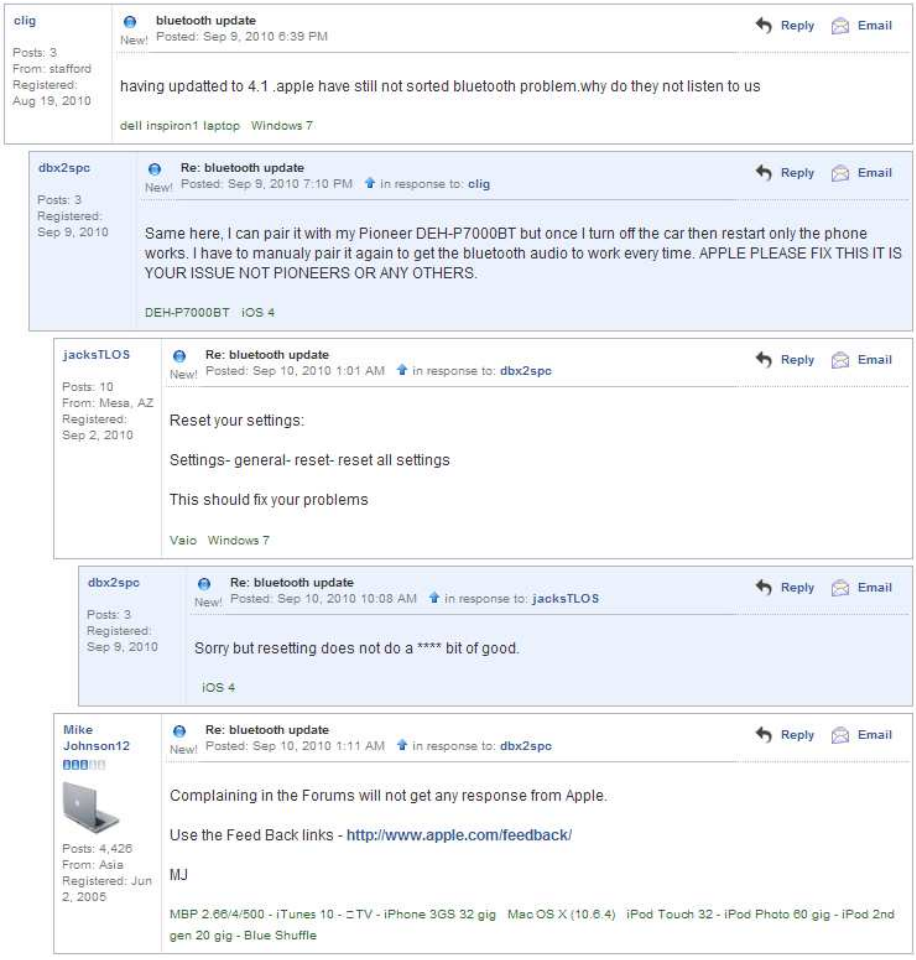

Fig. 2 Example of the threaded-view. An indentation indicates a reply relation.

and Apple Discussion[3]. One reason for this is that many online forums use popular publishing software such as phpBB[4] and vBulletin[5]. Most of these tools either don't support a threaded view or don't provide it as a default. Considering the small number of online communities which support threaded views, we believe that techniques for converting flat-view threads to threaded-view threads are needed for online community search, data mining, and social media analysis. We refer to this conversion as discovery of thread structures.

For simplicity and clarity, we make a number of assumptions about the thread structure discovery task. First, we assume that a thread structure is shaped like a rooted tree in which the top post is a root, each child post has only one parent post, and no node is isolated. Although there may be some cases which violate this assumption, such

[3] http://discussions.apple.com/
[4] http://www.phpbb.com/
[5] http://www.vbulletin.com/

```
function find_reply_relations(post, N) return parent
  for i ← 1 to N-1 do      # for each child
    for j ← 0 to i-1 do      # for each candidate parent
      l[j] ← compute_reply_likelihood(post[i], post[j])
    parent[i] ← argmax_j l[j]
```

Fig. 3 Algorithm for finding all reply relations in a thread

as answering questions from two posts, these cases are not frequent and, furthermore, most threaded-view systems make the same assumption. The second assumption is that we can find a parent-child (reply) relation considering only pairs of posts. In other words, a reply relation between two posts is independent of their grandparents and grandchildren. Lastly, we assume that a chronological order of posts in a thread is known so that we can consider only the preceding posts of a child post as candidate parent posts.

These assumptions significantly reduce the complexity of thread structure discovery. Under the first assumption, there are only $N - 1$ reply relations, where N is the number of posts. Further, when we are given a child post, we can find a reply relation by picking a most likely parent post from among all preceding posts. Under the second assumption, a greedy approach is the optimal approach to find a thread structure. That is, if we can find a correct parent post for each post, then we can build a correct thread structure. Finally, the third assumption simplifies the problem because we know which posts precede others.

Constructing a thread structure with reply relations is trivial; thus, finally, our problem is reduced to finding reply relations. Our algorithm for finding all reply relations in a thread is described as shown in Figure 3. This requires only $O(N^2)$ pairwise comparisons.

In the next section, we introduce the features used for reply relation detection and a process for learning the
`compute_reply_likelihood()` function in Figure 3. Finally, we evaluate the performance of our algorithm using experimental results.

3.1 Intrinsic Features

A straightforward method that we can use to determine a reply relation between two posts is to directly look at the contents of the posts. If two posts address a similar topic, then the posts are likely to have a reply relation. Further, we can frequently observe that a child post quotes or reuses text from the parent post. That is, word or phrase overlap can be evidence of a reply relation between posts.

We use text similarity as a feature in order to address both topical similarity and text overlap. There are numerous measures of text similarity. Among them, we use the *idf*-weighted cosine similarity. Cosine similarity is not only simple but also theoretically grounded by the vector-space model. Further, since a post is usually short and *tf* does not often function as more than an indicator of a term occurrence, it is necessary to use *idf* to weight topical terms. The following variation of the *idf*-weighted cosine similarity

[3] is used.

$$sim(\mathbf{p_1}|\mathbf{p_2}) = \frac{\sum_{k=1}^{m} d_k \cdot q_k}{\sqrt{\sum_{k=1}^{m} {d_k}^2}\sqrt{\sum_{k=1}^{m} {q_k}^2}}$$

$$d_k = 1 + \log tf_{1k}, \quad q_k = (1 + \log tf_{2k}) \log \frac{D+1}{df_k}$$

where $\mathbf{p_1}$ and $\mathbf{p_2}$ are word vectors of a parent candidate and a child post respectively, m is the size of vocabulary, *tf* is a term frequency, *df* is a document (post) frequency, and D is the total number of posts in the collection. A drawback of the *idf*-weighted cosine similarity is that it is non-symmetrical. However, our task is to find the most likely parent post among the preceding posts of a post, similarly to traditional information retrieval tasks. In this setting, we do not need to consider reverse relations of the parent post and the reply post; thus, symmetry is not necessary. Moreover, this non-symmetrical similarity measure is known to work well for similar retrieval tasks.

Note that our thread structure discovery technique did not empirically show large variance over different similarity measures. Therefore, the other measures can be used if required. Nevertheless, the variation of *idf*-weighted cosine similarity worked best in our experiments; we reported only the results using the measure in this paper.

Now we consider which part of a post the similarity measure is applied to. There is also the issue of how term vectors are constructed.

Quotation vs. Original Content

Many online community systems support an option to quote text from the preceding post when a post is uploaded. Such systems provide split views of the quotation and the original content. For example, some systems split views using special tags whereas others use some special characters such as '⟩' in the beginning of the quoted line. In such systems, we can easily determine which text is quoted.

Once we obtain the quotation and the original content separately, we can consider various combinations for similarity measurements. First, we can measure the similarity between the original content of a parent candidate and the original content of a child post. This similarity is to measure topical similarity between the posts. Second, similarity between the original content of a parent candidate and a quotation of a child post can be considered. This similarity shows how text is reused between the posts. Last, we can measure similarity between the full texts of posts without separating the quotation from the original content.

Unigram vs. n-gram

We can construct a term vector of a post with unigrams or n-grams. The fact that two posts share the same phrases or compound words rather than single words can be strong evidence for both text reuse and topical similarity. Therefore, if term vectors are composed of n-grams, we may expect more accurate discovery results. However, most n-gram terms are scarce and the vector space would be sparse. Accordingly, using n-grams can be unreliable in some cases. We will empirically investigate how different constructions of term vectors have effects on discovery results.

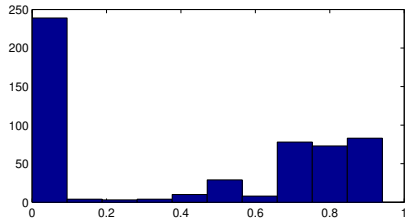

Fig. 4 Histogram of normalized location indices

3.2 Extrinsic Features

A post is an utterance in an informal dialogue rather than a speech or formal writing. While a few online communities such as technical email archives or political discussion forum are formal, many online communities such as game forums, social forums, or travel forums are generally informal. That is, in many cases, a post tends to be short and "instant". Therefore, similarity features are not often enough to capture relations between posts because of sparse word distributions. For example, a post asks a question, "What is the best authentic Mexican food?", and the next post says "Taco!" Although the two posts clearly have a question-answer relation through the context, the relation cannot be discovered with similarity features only. Thus, we need to use features which can describe context as well as content. Here we introduce several of these extrinsic features.

Location Prior

Most online community systems provide a view of posts in a thread in chronological order. We can assume that a relation between posts is inferred from the locations of the posts in the chronological time frame. For example, the top post in a thread has 0 as its location index, and the nth post in chronological order has $n-1$ as its location index. If the thread actually has a chronological structure like a dialogue by two individuals, then each post replies to the immediate preceding post. In other words, a post with location index i replies to a post with location index $i-1$. On the other hand, if a thread has a structure in which the top post asks a question and the others answer the question, then the parent post of every post is the top post with location index 0.

We want to predict where a parent post is located when the location of a child post is given. Formally, we want to estimate $P(i_1|i_2)$, that is, the likelihood that a post with location index i_1 is a parent post of a child post with location index i_2. We can directly extract an empirical distribution of the likelihood from annotated thread structures. However, because the amount of annotated data is not enough, each conditional distribution given the location index of each child post may be inaccurately estimated by sparse data. As a solution, we normalize location indices by the location index of a child post, i.e. i_1/i_2 and i_2/i_2. We refer to the normalized value as a normalized location index. We then estimate the likelihood using normalized location indices instead of real location indices. For example, if the original location indices i_1 and i_2 are 3 and 7, then the normalized location indices are $3/7 = 0.43$ and $7/7 = 1$ respectively. Therefore, all normalized location indices fall into $[0, 1]$.

Figure 4 shows a histogram of normalized location indices of related post pairs in the Cancun dataset (See Section 3.4 for a detailed description of the dataset). As we

see, there are two peaks in the histogram. A higher peak is located around 0 and a lower peak is located around 0.8. The former shows how many relations are biased toward the top post and the latter shows how many relations are biased toward the immediate preceding post. Relations with the immediate preceding post can be interpreted as chronological ordering. These two peaks commonly appear in all collections that we used.

We consider the distribution as a Gaussian Mixture which consists of two Gaussian distributions and estimate the mixture by the Expectation-Minimization [2]. Given the estimated distribution and location indices of two posts, we can compute the likelihood of a relation between the posts as follows:

$$P(i_1|i_2) = F_L\left(\frac{i_1+1}{i_2}\right) - F_L\left(\frac{i_1}{i_2}\right)$$

where F_L is a cumulative distribution function (cdf) of the estimated distribution. We refer to this likelihood as a location prior and use it as an extrinsic feature.

Note that this estimated prior worked better in preliminary experiments although a location itself can be considered as a feature. In fact, as shown in Figure 4, a location cannot be considered as a monotonic feature.

Time Gap

A difference between posting times of two posts can be evidence of a relation between the posts. If a post is created 10 months after the other post was posted, then the chance that the posts have any relation is probably small. Conversely, if two posts are sequentially posted with a small time gap, then the chance of a relation increases.

Since the posting time difference has a wide value range, we need to normalize the difference as follows:

$$gap(t_1|t_2) = \frac{t_2 - t_1}{t_2 - t_0}$$

where t_0, t_1, and t_2 are the posting times of the top post, a parent candidate post, and a child post. We refer to this normalized value as a time gap.

Same Author

Assuming that turn-taking between speakers happens in a thread, the fact that two posts are written by the same author usually can be used as negative evidence of a relation. We use an indicator of the same author relationship as a feature, that is, 1 if the authors of two posts are the same, 0 otherwise.

Author Reference

In flat-view systems, it is not easy to tell which post a post is replying to. Accordingly, users often refer to the author of the specific post by writing the name or ID of the author in order to express an intention to reply to a specific post. We call this behavior an author reference. Existence of an author reference between two posts can be explicit evidence of a relation. We use an indicator of an author reference as a feature, that is, 1 if there is an author reference, 0 otherwise.

Inferred Turn-taking

This feature is derived from a same author relation and an author reference relation. Let post A, B and C be posted in this order in a thread. If post A and B have an author reference and post A and C have a same author relation, then we can infer that post C replies to post B when assuming turn-taking with $A \rightarrow B \rightarrow C$. We call the inferred relation between post B and C an inferred turn-take and express it as an indicator, that is, 1 if there is an inferred turn-take, 0 otherwise. Note that this does not break our second assumption about independence of grandparents because we do not use a relation but a feature extracted from preceding posts.

3.3 Learning

We consider the thread structure discovery task as a ranking task. That is, if each child post is considered as a query, parent candidate posts are considered as documents to be retrieved. Since a post has only one parent post, we have only one relevant document for each query. Although our task can be seen as a classification task or regression task, the strength of a relation between two objects is relative to other relations. Therefore, it sounds feasible to model relative preferences rather than an absolute decision boundary. Indeed, we conducted preliminary experiments using a linear regression algorithm, but ranking algorithms consistently showed better performance.

Since we have several heterogeneous features, it seems inappropriate to use traditional information retrieval techniques. Instead, we use the ranking SVM algorithm [11] because it is known to address such settings well. The ranking SVM learns a ranking function based on pairwise labels by solving an optimization problem as follows:

$$\min_{\mathbf{w}} M(\mathbf{w}) = \frac{1}{2}||\mathbf{w}||^2 + C\sum_{i,j}\xi_{ij}$$
$$\text{subject to } \left\langle \mathbf{w}, \mathbf{x_{iR}} - \mathbf{x_{ij}} \right\rangle \geq 1 - \xi_{ij}$$
$$\forall_i \forall_j \xi_{i,j} \geq 0$$

where x_{iR} is a feature vector extracted from a relation between child post p_i and its parent post, x_{ij} is a feature vector extracted from a relation from p_i and non-parent post p_j, w is a weight vector of a ranking function. We use a linear kernel for the ranking SVM. Finally, the learned ranking function is the `compute_reply_likelihood()` in Figure 3.

3.4 Collections

We use three online community collections in order to evaluate techniques for thread structure discovery. Two of them are online forums. The other is an email archive. The detailed statistics of each collection is presented in Table 1.

World of Warcraft (WOW) forum

We crawled the general discussion forum[6] of the World of Warcraft (WOW), a popular online game. The collection contains threads created from August 1, 2006 to April

[6] http://forums.worldofwarcraft.com/board.html?forumId=10001

	WOW	Cancun	W3C
#threads	16,274	58,150	72,214
avg.# posts per thread	84.4	9.1	2.1
avg. post length (in words)	57.3	67.0	249.6
size (in Gigabytes)	14.0	7.0	3.4

Table 1 Statistics of collections

1, 2008. Among our three collections, the WOW collection is the most casual online community. Most users of online game forums are in the youth demographic. Many posts are not only short but also poorly composed. We can frequently observe broken English, typos and abbreviations. Furthermore, people tend to write posts without serious thought, which often results in long threads as shown in Table 1.

We randomly picked 60 threads which contain at least 5 posts. We split them into 2 sets of 40 threads with overlap of 20 threads, and assigned the sets to two annotators. An annotator tagged all reply relations between posts in each thread in the given set, i.e. 1 if a pair of posts is a reply relation, 0 otherwise. To merge the annotations for the overlap threads, we took 10 threads from each annotator, e.g., odd numbered threads from annotator 1 and even numbered threads from annotator 2. Cohen's kappa, the inter-annotator agreement computed with the annotations of the overlap threads, was 0.88.

We can extract all the features that we introduced earlier from the WOW collection. In particular, the WOW forum displays the quotation and the original content differently using HTML tags. Therefore, we could extract quotations and original contents separately using simple rules.

Cancun forum

We crawled the Cancun forum[7] of tripadvisor.com, a popular travel guide site. The Cancun collection contains threads accumulated for about 4 years from September 7, 2004 to November 23, 2008. The Cancun forum is somewhat more formal than the WOW forum. Posts are relatively well written, and the average length of posts is longer than the WOW forum.

We annotated structures of 60 threads through the same process as the WOW forum. Cohen's kappa of the Cancun forum annotations was 0.86.

A major difference of the Cancun forum to the WOW forum is that the Cancun form does not systemically support quotation. Therefore, we cannot extract quotations and original contents separately.

W3C email archive

We also used the 'lists' sub-collection of the W3C collection from the email discussion search task of the TREC enterprise track [24]. The collection was crawled from the mailing list[8] of the World Wide Web Consortium (W3C). Email archives or newsgroups are old-style online communities but are still active in technical areas. The W3C collection is the most formal of our collections. Most participants are scholars or experts in the field and most posts are written in a polite tone. As you see in Table 1, the average length of a post is much longer than the other collections.

[7] http://www.tripadvisor.in/ShowForum-g150807-i8

[8] http://lists.w3c.org/

The W3C collection provides thread structures in the '`thread.html`' file in each group archive. However, many of these thread structures are wrong. We frequently find cases where an earlier email replies to a later email. This is because the 'msg-id' and 'inreply-to' tags in email headers are often lost. A thread of emails is usually constructed by matching tags. If they are missing, then email archive tools infer threads using heuristics such as title matching. Such inferences are often inaccurate.

To build an annotation set for thread structure discovery, we refined the thread structures by picking threads only composed of emails whose 'inreply-to' tag matches a 'msg-id' tag of any other post in the same thread. Finally, in this set, we obtained 1635 threads which contain at least 3 emails.

All features that we introduced earlier are available in the W3C collection. Since quoted text begins with some special characters such as '⟩', we can easily divide each message into the quotation and the original content. We removed all lines which start with multiple special characters because they are a part of replies to replies which we do not consider in our thread structure discovery task.

Note that we refer to an email as a post in other sections of this paper for consistency.

3.5 Experiments

We conducted experiments for thread structure discovery on each collection. To investigate the effectiveness of features, we tested various combinations.

We compute accuracy to evaluate the performance of each combination of features as follows.

$$accuracy = \frac{|\{\text{reply relations}\} \cap \{\text{detected relations}\}|}{|\{\text{reply relations}\}|}$$

Accuracy is computed for each thread, and the final evaluation measure is the average of accuracy scores. Note that, in this setting, this metric is the same as recall or precision because they have the same denominator (i.e., the number of posts in a thread - 1). Also, we can employ other information retrieval evaluation metrics such as mean reciprocal rank (MRR) because our task is considered as a ranking task. However, the fact that a true reply relation is highly ranked by our algorithm as long as the relation is not located at rank 1, does not affect the discovered thread structure. This is a difference from other retrieval tasks such as ad hoc retrieval where a ranked list is generally provided to users. Accordingly, we do not consider such metrics for evaluation.

For the WOW and Cancun collections, because the annotated data is small, we performed 10-fold cross validation for evaluation, that is, we used 54 threads per partition as training data. On the other hand, since the W3C collection has enough data for training, i.e. 1,635 threads, we used 1,535 threads as training data and 100 threads as test data.

For intrinsic feature extraction, only the title and body text in each post were used. The text was pre-processed by the Porter stemmer [19] and stopword removal.

	None	LP	TG	AR	SA	IT	All
None		0.5770	0.5867	0.2959	0.2996	0.2890	0.5302
F+U	0.5858	0.7629	0.7223	0.5901	0.6140	0.5858	0.8025
F+N	0.5856	0.7908	0.7249	0.5880	0.6129	0.5856	0.8125
O+U	0.4364	0.5704	0.5103	0.4421	0.4469	0.4374	0.5745
O+N	0.4346	0.5738	0.5131	0.4403	0.4469	0.4356	0.5770
Q+U	0.5791	**0.8814**	**0.8824**	0.5824	0.5613	0.5791	0.8698
Q+N	0.5779	**0.8809**	**0.8873**	0.5812	0.5672	0.5779	**0.8842**
O+Q+U	0.6570	**0.8922**	0.8228	0.6604	0.6534	0.6570	0.8726
O+Q+N	0.6531	**0.8851**	0.8234	0.6564	0.6502	0.6531	**0.8798**

Table 2 Thread structure discovery results on the WOW collection. Values are accuracy scores. Each row corresponds to an intrinsic feature: full text (F), original contents (O), quotations (Q), unigram (U) and n-gram (N). Each column corresponds to an extrinsic feature: location prior (LP), time gap (TG), author reference (AR), same author (SA), inferred turn-taking (IT) and all extrinsic features (ALL). Bold values indicate the best score group, i.e. the score is not statistically significantly different from the best score (by the paired randomization test with p-value ¡ 0.05).

	None	LP	TG	AR	SA	IT	All
None		0.4839	0.4861	0.5104	0.4034	0.4139	0.5630
O+U	0.4697	0.5057	0.5563	0.4922	0.5159	0.4840	0.6165
O+N	0.4656	0.5083	0.5509	0.4862	0.5025	0.4818	**0.6279**

Table 3 Thread structure discovery results on the Cancun collection

	None	LP	TG	AR	SA	IT	All
None		0.7149	0.7284	0.7156	0.6520	0.6726	0.7811
F+U	0.8988	0.8785	0.9017	0.8954	0.9210	0.9104	0.9162
F+N	0.9065	0.8996	0.9137	0.9200	0.9336	0.9114	0.9343
O+U	0.6317	0.6973	0.7658	0.7134	0.7397	0.7138	0.8053
O+N	0.6309	0.6966	0.7621	0.7152	0.7380	0.7130	0.8061
Q+U	0.8907	0.9130	0.9078	0.9058	0.8986	0.9066	0.9351
Q+N	0.8907	0.9130	0.9078	0.9058	0.8986	0.9066	0.9343
O+Q+U	0.9067	0.9133	0.9282	0.9354	0.9366	0.9273	**0.9533**
O+Q+N	0.9170	0.9183	0.9393	0.9295	**0.9457**	0.9222	**0.9617**

Table 4 Thread structure discovery results on the W3C collection

3.6 Results and Discussion

Table 2, 3 and 4 show the experimental results for the three collections. In the tables, each row corresponds to an intrinsic feature and each column corresponds to an extrinsic feature.

In the WOW collection, the similarity of quotations is more helpful than topical similarity of original contents. However, we can see a performance gain from using both of them. Unigram and n-gram do not show significant differences in performance. Among the extrinsic features, the location prior and the time gap are the most helpful features. When using either of them, we see improvements of at least 20%. The best combinations require at least similarity of quotations as an intrinsic feature and either or both of the location prior and the time gap as an extrinsic feature. The best scores have almost 90% accuracy.

In the Cancun collection, the scores are much worse than those of the WOW collection. This is mainly because the Cancun collection does not have any quotations. On

	WOW	CANCUN	W3C
Top-based	0.5773	0.5202	0.4676
Chronological	0.2713	0.4839	0.7161
Graph-based Propagation	0.3132	0.5315	0.6526

Table 5 Thread structure discovery accuracy on baselines. Two baselines (the first and second rows) consider specific thread structures, i.e., the top-based structure and the chronological structure. Another baseline (the third row) uses the graph-based propagation algorithm [5].

the basis solely of the non-quotation features, the performance in the Cancun collection is similar to or better than the WOW collection. Another difference from the WOW results is that author reference is more effective. We hypothesize that users refer to other posts more frequently in the Cancun collection because they cannot use quotations supported by the forum system. In addition, the location prior and the time gap are also helpful. The best performance is achieved when all features are used.

In the W3C collection, we see very good results even using only the intrinsic features. Quotations, in particular, are very helpful. In emails, not only is text usually long enough, but also the whole text of each mail is almost always quoted by a reply. The high accuracy obtained by the intrinsic features can be explained by these characteristics of email. However, we still observe performance gains from using extrinsic features in addition to intrinsic features.

For baselines for comparison, we can assume specific thread structures. Specifically, two simple structures can be considered. The first is that all posts reply to the top post. We call this a top-based structure. The second structure is that all posts reply to the immediate preceding posts. We call this a chronological structure.

Another baseline to consider is a graph-based propagation algorithm introduced by Cong et al. [5]. Although the algorithm is used for detecting relevant answer posts for a question post in a forum thread, their task is similar to ours in that they also seek relations between posts in a thread. The graph-based propagation algorithm performs a random walk on a directed graph which encodes inter-post relations with edge weights computed by:

$$w(p_1 \rightarrow p_2) = \frac{1}{1 + KL(p_1||p_2)} + \lambda_1 \frac{1}{dist(q, p_2)} + \lambda_2 \, authority(p_2)$$

where q is are a query post, p_1 and p_2 any two candidate posts in the same thread, $KL(p_1||p_2)$ is the Kullback-Leibler divergence of language models of p_1 and p_2, and $dist(q, p_2)$ is the locational distance between q and p_2. $authority$ of a post is computed by normalizing $(\#reply^2/\#start)$ where $\#reply$ is the number of replies by the author of p_2 and $\#start$ is the number of threads initiated by the author. λ_1 and λ_2 are linear combination parameters which were set to the same values as reported in [5]. From this formula, we can know that this algorithm tries to incorporate similarity, locational information and authorship information of posts into a graph. Posts are ranked by the stationary distribution obtained by a random walk on this graph; then, the relation between the first ranked post and the question post is predicted as a reply relation.

Table 5 shows the results of thread structure discovery using the baselines. Interestingly, each collection shows a different aspect. The WOW forum is biased toward the top-based structure. This shows that people tend to read only the top post and reply to it because a thread in the WOW forum is often very long as shown in Table 1. Conversely, the W3C collection is biased toward the chronological structure. Although the W3C archive is a public community based on a mailing list, the characteristic of

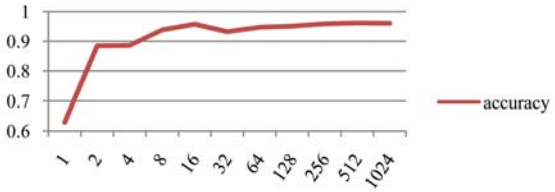

Fig. 5 Learning curve on the W3C collection. The change of accuracy on test sets (y-axis) depending on the number of threads in the training set (x-axis) is plotted.

the discussions is more private compared to online forums. That is, a discussion is often similar to one-to-one conversation rather than a group discussion even though everyone can listen to it. Since each participant knows all issues in the preceding mails, a new mail naturally tends to be a reply to the immediate preceding mail. In the Cancun forum, the two specific structures are almost equally likely. This shows that the different aspects of the other two online communities are mixed in the Cancun forum.

Comparing the performances of the baselines to ours, our algorithm significantly outperforms discovery based on the specific structures regardless of types of online communities. This presents that threads cannot be assumed to have a simple structure. Also, the graph-based propagation algorithm shows significantly worse performance than the best performance of our algorithm. This is because the graph-based propagation algorithm tries to identify a relevant post (which is often created by an authoritative author or informative) to a query post rather than a real parent post of a child post in a thread structure. For example, a highly relevant post may appear after a long discussion involving a number of posts following a query post. The graph-based propagation algorithm picks up the post even when it is not a direct reply to the query post, whereas we would like to reconstruct all contexts via direct reply relations.

One question is what features should be used in practice. The answer is simple: If all features are available, use them all. For the Cancun and the W3C collection, the best accuracy is gained when using all features. For the WOW collection, although using all features is not the best, the difference from the best performance is not statistically significant. The most effective intrinsic feature is the similarity of quotations, and there is no notable difference between unigram and n-gram. Therefore, if resources are limited and quotations exist, the best approach for intrinsic features is to compute the similarity of only quotations using unigram. For extrinsic features, the location prior and the time gap are almost always effective. The authorship-based features, i.e. the same author, the author reference, and the inferred turn-taking, are shown to be effective only in the formal community such as the W3C where authors' real names are known. In many informal communities such as the WOW and the Cancun, only user IDs are public. Because user IDs are often combinations of alphabets and numbers that the others except the owner cannot understand, in such communities, references do not frequently occur, and we cannot easily recognize the reference even when there is. Accordingly, the effect of the authorship-based features is limited.

There is also the question of how much training data is required to achieve good accuracy. Since the W3C collection has sufficient training data, we plot a learning curve according to the amount of training data as shown in Figure 5. We can see that the curve becomes stable from around 50∼60 threads. Although this may vary between

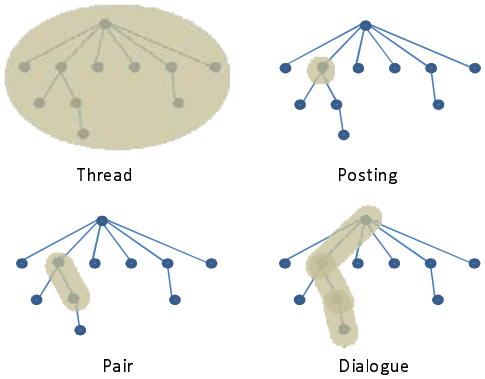

Fig. 6 Contexts in a thread structure

collections, it provides some support for the size of training data used on the other collections (i.e., 54 threads).

4 Multiple Context-based Retrieval

In this section, we introduce approaches to improve retrieval performance using thread structures discovered by the algorithms introduced in Section 3.

4.1 Context Extraction based on Thread Structure

A document is composed of self-contained text units in various levels, e.g., sentences, paragraphs or sections. Similarly, a thread is composed of different self-contained sub-structures. We call a sub-structure a context.

Figure 6 presents four contexts. The first context is the coarsest-grained context, i.e. the thread itself. The second context is the finest-grained context, i.e. a post. While we can use thread contexts to get a general picture about the topic addressed by a thread, we can use post contexts to get detailed information. The third context is a pair defined by a reply relation. This context is directly extracted from a relation discovered by thread structure discovery algorithms. A pair context contains an interaction between two users. For example, the context may be a question-answer pair. If what we want is an answer to a question, a pair context can be suitable. The fourth context contains all posts in a path from the root node (top post) to a leaf node. We refer to this context as a dialogue because by looking at the context we can follow a conversation flow, e.g., how the discussion was started, what issue was discussed, and what the conclusion was.

Note that we can extract thread contexts and post contexts without regard to the structure of a thread. However, pair contexts and dialogue contexts must be extracted from a thread structure.

4.2 Multi-context-based Retrieval

We address two retrieval tasks using multiple contexts: thread search and post search. Since posts in casual online forums such as WOW or Cancun are usually too short to

provide information on their own, people are likely to want to find relevant threads rather than posts. On the other hand, emails (posts) in a technical email archive like the W3C archive are often long enough to deliver information. In that case, a more suitable task is to find relevant emails (posts).

For these two tasks, we introduce retrieval techniques based on a language modeling approach to retrieval [6]. In our work, the query likelihood $P(Q|D)$ is estimated under the term independence assumption as follows:

$$P(Q|D) = \prod_{q \in Q} ((1-\lambda)P_{ML}(q|D) + \lambda P_{ML}(q|C)) \quad (1)$$

where q is a query term in query Q, D is a document, C is the collection, λ is a smoothing parameter, and $P_{ML}(\cdot)$ is the maximum likelihood estimate, i.e. $P_{ML}(w|D) = tf_{w,D}/|D|$. If we use the Dirichlet smoothing [29], then $\lambda = \mu/(\mu + |D|)$ where μ is a Dirichlet smoothing parameter.

Note that we here employ log-linear mixture models for evidence combination. For example, in the following subsections, Equation (2) and (7) use the geometric means instead of the arithmetic mean to combined multiple language models or multinomial distributions. Also, in Equation (3), the weighted product is used instead of the weighted sum. We refer to these mixture models as log-linear mixture models because the geometric mean-based mixture models are equivalent to linear mixtures of logarithms of the language models in terms of ranking. Although we also did experiments with linear mixture models (the arithmetic mean and weighted sum instead of the geometric mean and weighted product), the models were not successful as the log-linear mixture models. Therefore, we reported only the results using the log-linear mixture models. For comparison of various representation techniques for document representation in Information Retrieval, see Liu and Croft [16]. Also, Seo and Croft [21] provides a theoretical evidence explaining why log-linear mixture models or geometric mean-based representations work for information retrieval tasks. They demonstrated that the geometric mean of multiple language model representations appears closer to the center of mass in a certain geometry.

4.2.1 Thread Search

The simplest approach to thread search is to consider a thread as a document [8, 20]. That is, all posts are concatenated ignoring any existing structure, and a language model for a thread is built. We refer to this as global representation (GR).

$$\Phi_{GR}(Q, T_i) = P(Q|T_i)$$

where Φ is a ranking function and $P(Q|T_i)$ is a query likelihood score of query Q for thread T_i.

A drawback of global representation is that relevant local contexts can be dominated by non-relevant contexts. A thread often addresses a broad topic or a mixture of sub-topics, but user queries may be specific. For example, in a game forum, while a thread addresses "the best weapons", a user query may be "the best sword for warriors". A global representation may not locate the thread even when highly relevant local contexts for the query are contained in it. For threads as long as those in the WOW collection, this problem can be serious.

To tackle this drawback, we employ more advanced techniques using discovered structures. Resource selection techniques can be used for this task because a thread can be considered as a collection of local contexts, i.e. posts, pairs or dialogues. In particular, we consider the pseudo-cluster selection technique (PCS) that has been used for blog site search [20]. Pseudo-cluster selection retrieves the top N local contexts and aggregates local contexts in the ranked list according to which thread the local context comes from. We call the local context group a pseudo-cluster. Finally, relevant threads are located according to a geometric mean of scores of the top K local contexts in a pseudo-cluster as follows:

$$\Phi_{PCS}(Q,T_i) = \left(\prod_{j=1}^{K} P(Q|L_{ij}) \right)^{1/K} \tag{2}$$

where $P(Q|L_{ij})$ is a query likelihood score based on the language model of local context L_{ij} in thread T_i.

If a pseudo-cluster contains fewer than K local contexts, then the upper bound of the pseudo-cluster is used as follows:

$$L_{\min} = \mathrm{argmin}_{L_{ij}} p(Q|L_{ij})$$

$$\Phi_{PCS}(Q,T_i) = \left(P(Q|L_{\min})^{K-m} \prod_{j=1}^{m} P(Q|L_{ij}) \right)^{1/K}$$

where m is the number of local contexts in a pseudo-cluster. PCS has proved effective for thread search based on post contexts [7].

Pseudo-cluster selection reflects how much relevant information exists locally in a thread whereas global representation reflects the cohesiveness of the thread. Therefore, we consider a weighted-product of the ranking function of global representation and the ranking function of pseudo-cluster selection to improve retrieval performance as follows:

$$\Phi_{Product}(Q,T_i) = \Phi_{PCS}(Q,T_i)^{(1-\pi)} \cdot \Phi_{GR}(Q,T_i)^{\pi} \tag{3}$$

where π is a weight parameter.

4.2.2 Post Search

We retrieve relevant posts using estimated language models for posts. If we have post contexts only, language models are estimated using smoothing as follows:

$$P(w|D) = (1-\lambda_1)P_{ML}(w|D) + \lambda_1 P_{ML}(w|C) \tag{4}$$

where D is a post, C is the collection, and λ_1 is a smoothing parameter.

If we know that the post belongs to thread T, then we can do two-stage smoothing similarly to cluster-based retrieval [15]. This is also similar to an effective approach for the email discussion search task of the TREC 2006 Enterprise track [18].

$$P(w|D) = (1-\lambda_1)P_{ML}(w|D) + \lambda_1((1-\lambda_2)P_{ML}(w|T) + \lambda_2 P_{ML}(w|C)) \tag{5}$$

WOW	the best solo PvP class how to beat warlock recommended quest chains for level 70s
CANCUN	winter weather in Cancun couple only all inclusive hotel Isla Mujeres tour

Table 6 Example queries for the WOW collection and the Cancun collection

Further, if we have another context X_z, i.e. a pair context or a dialogue context, then we can add one more smoothing stage. However, in contrast to thread contexts, a post can belong to multiple pair contexts or dialogue contexts. We compute a geometric mean to combine language models of the contexts as follows:

$$P_z(w|D) = (1-\lambda_1)P_{ML}(w|D) + \lambda_1((1-\lambda_2)P_{ML}(w|X_z) + \lambda_2((1-\lambda_3)P_{ML}(w|T) + \lambda_3 P_{ML}(w|C))) \quad (6)$$

$$P(w|D) = \left(\prod_{z=1}^{Z} P_z(w|D)\right)^{1/Z} \quad (7)$$

where Z is the number of contexts which contain D.

4.3 Test Collections

For retrieval experiments, we used the three collections used for thread structure discovery. While two online forums were used for the thread search task, the W3C collection was used for the post (email) search.

Since the W3C collection has been used for the email discussion search task of the TREC enterprise track, there is a relevance judgment set provided by TREC, which contains 110 queries and 58,436 relevance judgments. Since our post search task is almost the same as the email discussion search task, we used these relevance judgments to evaluate post search in the W3C collection. Note that although the judgments were made in multi-grades, the grade reflects whether an email contains pro/con statement rather than the degree of relevance. Therefore, we used the judgments as binary relevance judgments.

On the other hand, we had to make our own relevance judgments for the other two collections. For each collection, we chose 30 popular titles among titles of threads which were created after our crawl and asked two people to manually generate keyword queries from the titles. Table 6 shows a few examples of queries for the WOW and the Cancun collection. We created relevance judgment pools using retrieval techniques introduced in Section 4.2 and linear mixture models. We made ternary relevance judgments, i.e. 0 for irrelevant threads, 1 for relevant threads, and 2 for highly relevant threads. In total, we made relevance judgments for 2,591 threads for the WOW collection and 2,401 threads for the Cancun collection. A summary of the relevance judgment sets are presented in Table 7.

	#topics	#judged threads	#relevant threads	#highly relevant threads
WOW	30	86.4	5.7	3.6
CANCUN	30	80.0	14.0	22.1

Table 7 Summary of relevance judgments of two forum collections (WOW and CANCUN). The numbers of judged threads and relevant threads are averaged per topic.

	NDCG@10	MAP
Thread	0.4200	0.3705
Post	0.2966	0.2565
Post+Thread	0.4519	0.3875
Pair	0.3763^{β}	0.2998^{β}
Pair+Thread	$0.4447^{\alpha\beta}$	$0.3885^{\alpha\beta}$
Dialogue	0.4374^{β}	0.3599^{β}
Dialogue+Thread	$0.4823^{\alpha\beta\gamma}$	$0.4073^{\alpha\beta\gamma}$

Table 8 Retrieval Performance on the WOW collection (Thread Search). The superscripts α, β and γ indicate statistically significant improvements on each baseline, i.e. 'Thread', 'Post', 'Post + Thread', respectively (by the paired randomization test with p-value < 0.05).

4.4 Experiments

We discovered structures for all threads in each collection using the SVM classifier trained with the best feature combinations in Section 3 and the algorithm in Figure 3. Then, we applied multi-context-based retrieval techniques to contexts extracted from the structures. Text was stemmed by the Krovetz stemmer [12], and no stopwords were removed for retrieval experiments. Note that although we used different stemmers for thread structure discovery and retrieval experiments for convenience in implementing each system, this does not mean that a specific stemmer is preferred for each task.

As evaluation metrics, we used normalized discounted cumulative gain at 10 (NDCG@10) and mean average precision (MAP) for thread search with the WOW collection and the Cancun collection. MAP and precision at 10 (P@10) were used for post search with the W3C collection. In all cases, MAP and P@10 are computed considering a judged document whose grade is equal to or greater than 1 as relevant.

Dirichlet smoothing was used to estimate language models for all experiments. Accordingly, smoothing parameters (λ, λ_1, λ_2 and λ_3) in Equation 1, 4, 5 and 6 are determined by $\mu/(|D|+\mu)$ where μ is a Dirichlet smoothing parameter for each context or smoothing stage. To evaluate performance, we performed 10-fold cross validation. For thread search, the parameters to be tuned are the Dirichlet smoothing parameters for context language models, the number of posts in a pseudo cluster, and the weight parameter for the combination of GR and PCS. For post search, the Dirichlet smoothing parameters for each smoothing stage were tuned. The parameters were exhaustively searched to maximize NDCG@10 for thread search and MAP for post search.

4.5 Results

Table 8 and 9 show results of thread search on the WOW collection and the Cancun collection. 'Thread' means global representation based on a thread context. 'Post', 'Pair' and 'Dialogue' mean pseudo-cluster selection based on each context. '+ Thread'

	NDCG@10	MAP
Thread	0.4612	0.2630
Post	0.4763	0.2887
Post+Thread	0.4942	0.2896
Pair	0.4478	0.2413
Pair+Thread	0.4897^{α}	0.2857^{α}
Dialogue	0.4938^{α}	0.2618
Dialogue+Thread	$0.5141^{\alpha\beta}$	0.2973^{α}

Table 9 Retrieval Performance on the Cancun collection (Thread Search)

	NDCG@10	MAP
Dialogue+Thread	$0.4651^{\alpha\beta}$	0.3869^{β}

Table 10 Retrieval performance of the WOW collection (based on inaccurate thread structure discovery)

	MAP	P@10
Post	0.2405	0.4404
Post+Thread	0.2931	0.4945
Post+Dialogue+Thread	$0.3036^{\alpha\beta}$	$0.5101^{\alpha\beta}$
Post+Pair+Thread	$0.3101^{\alpha\beta}$	$0.5147^{\alpha\beta}$

Table 11 Retrieval performance on the W3C collection (Post Search). The superscripts α and β indicate statistically significant improvements on the baselines, i.e. 'Post' and 'Post + Thread', respectively (by the paired randomization test with p-value < 0.05)

means that a weighted-product of GR and PCS is used. The top three rows in the tables are considered as baselines because they do not need to use structures of threads.

In the WOW collection, techniques based on dialogue contexts show better or at least comparable performance to techniques based on the other contexts. Particularly, when using dialogue contexts and thread contexts together, the best performance is achieved, and the improvements over all baselines are statistically significant. This demonstrates that a performance improvement in thread search can be achieved using thread structures, particularly, dialogue contexts. A weighted-product of GR and PCS shows better performance than solely GR or PCS. The combination of GR and PCS proves to be an effective approach for thread search as well as for blog site search.

In the Cancun collection, similar trends are shown, that is, dialogue context-based search and the combination of GR and PCS consistently present better performance than the others. However, the improvements are not always statistically significant, in contrast to in the WOW collection. This is presumed to be due to the relative inaccuracy of thread structure discovery in the Cancun collection. To justify this assumption, we investigated retrieval performance based on inaccurate thread structures in the WOW collection. To simulate inaccurate discovery, we used unigram similarity in the full text only as a feature ('F+U' row, 'None' column in Table 2) and applied the best retrieval technique, i.e. 'Dialogue + Thread' to contexts extracted from the inaccurate structure. The results are shown in Table 10. This performance is not only worse than the performance based on accurate structure discovery but also fails to show significant differences over the baseline 'Post+Thread'. This shows that the accuracy of thread structure discovery can be critical in our retrieval framework.

Table 11 shows the results of post search on the W3C collection. Each row represents which contexts are used for smoothing. The one-stage and two-stage smoothing at the

	MAP	P@10
Cluster-based LM	0.2422	0.4541

Table 12 Retrieval performance of cluster-based language models on the W3C collection (Post Search). These results do not show statistically significant differences from the baseline 'Post' in Table 11 (by the paired randomization test with p-value < 0.05).

top two rows, which use post contexts and threads contexts only, do not require thread structures. Therefore, we consider them as baselines. For both the pair context and the dialogue context, addition of the thread context for smoothing achieved statistically significant improvements. This shows that contexts based on thread structure are also helpful for post search.

4.6 Comparison with cluster-based language model

A question which raises from the post search results is whether the improvements really come from thread structures or from other structures implied in the thread structures. For example, since we used similarity among posts as a feature for thread structure discovery, we can guess that similarity structures rather than the thread structures may lead to the improvements. To examine this assumption, we apply a cluster-based language model approach [15], which performs document smoothing with clusters built with similar documents, to the post search task. In particular, we follow the best performing practice among various techniques introduced in [15]. That is, we made clusters in a query-independent way using the cosine measure for document similarity. To assign documents into a cluster, the k-mean algorithm implemented in the Lemur toolkit [9] was used. This resulted in 14,346 clusters for the W3C collections. Using these clusters, we estimate a document language model as follows:

$$P(w|D) = (1-\lambda_1)P_{ML}(w|D) + \lambda_1((1-\lambda_2)P_{ML}(w|cl) + \lambda_2 P_{ML}(w|C))$$

where cl is a cluster which D belongs to. To estimate the cluster language model, a big document is created by concatenating all documents in the cluster. Parameters λ_1 and λ_2 are determined by 10-fold cross validation, as done in the previous experiments.

Table 12 shows the post search results by this model. The results fail to show any significant improvement even on the simplest baseline ('Post' in Table 11) which does not use thread structures. This demonstrates that simple similarity structures without considering thread structures are not helpful for post search.

4.7 Result Presentation Using Local Contexts

Leveraging thread structures can be effective for presentation of search results as well as for retrieval performance improvements. A search engine usually displays summaries of retrieved documents, i.e. snippets. However, a retrieved thread of some forums such as WOW can be not only very long but also a mixture of multiple topics. Summaries based on a small part of the thread may be irrelevant. Conversely, because a retrieved email from email achieves may be short, people may not get sufficient information even when the whole email is displayed.

[9] http://www.lemurproject.org/lemur.php

	WOW	CANCUN	W3C
avg.# posts per dialogue	4.1	4.5	4.8

Table 13 Average number of posts in a dialogue context

To tackle both of these problems, we suggest using dialogue contexts for result presentation. We believe that through a self-contained dialogue context, people can find relevant information in a thread without seeing the whole thread. In addition, a summary of a dialogue context can be more helpful than a summary of an entire thread. Since it is difficult to empirically show that a dialogue context is satisfying for result presentation, we provide indirect evidence. Table 13 shows the average number of posts in a dialogue context of each collection. We can see that a dialogue context is consistently composed of 4 or 5 posts regardless of thread sizes of the collections. This seems reasonable for people to review, compared to the average number of posts in a thread in the WOW collection, i.e. 84.4 posts. Of course, to prove the effectiveness of dialogue contexts as result representation units, more thorough studies should follow. For example, we can design a user study for a readability comparison between snippets, dialogue contexts, and full threads. Further, we can consider a study for how many relevant posts a dialogue contains. We leave these as future work.

5 Conclusion and Future Work

In this paper, we investigated whether search for community sites such as forums could be improved using thread structure. We defined the thread structure discovery task and introduced various intrinsic and extrinsic features, and algorithms for this task. Our results show that threads can often be accurately identified using our approach. We then introduced retrieval methods based on contexts extracted from the thread structures. We showed that combinations of multiple thread contexts can achieve significant retrieval effectiveness improvements over strong baselines.

There are three obvious directions for future work. First, we can find experts in online communities using thread structures. The information about experts can be used for authority scores to further improve retrieval. The second challenge is to identify relevant online communities using thread search. The last direction is to exploit more advanced linguistic features for the thread structure discovery task because online communities do not always have all of the features that we introduced.

Acknowledgements This work was supported in part by the Center for Intelligent Information Retrieval (CIIR) and in part by NSF grant #IIS-0711348. Any opinions, findings and conclusions or recommendations expressed in this material are the authors' and do not necessarily reflect those of the sponsor.

References

1. Arguello J, Elsas J, Callan J, Carbonell J (2008) Document representation and query expansion models for blog recommendation. In: Proceedings of the Second International Conference on Weblogs and Social Media (ICWSM 2008)
2. Bishop CM (2006) Mixture models and EM. In: Pattern Recognition and Machine Learning, Springer, pp 423–459

3. Buckley C, Allan J, Salton G (1994) Automatic routing and ad-hoc retrieval using SMART. In: The second Text REtrieval Conference (TREC-2) Proceedings
4. Carvalho VR, Cohen WW (2005) On the collective classification of email "speech acts". In: SIGIR '05: Proceedings of the 28th annual international ACM SIGIR conference on Research and development in information retrieval, pp 345–352
5. Cong G, Wang L, Lin CY, Song YI, Sun Y (2008) Finding question-answer pairs from online forums. In: SIGIR '08: Proceedings of the 31th annual international ACM SIGIR conference on Research and development in information retrieval, pp 467–474
6. Croft WB, Lafferty J (2003) Language Modeling for Information Retrieval. Kluwer Academic Publishers
7. Elsas JL, Carbonell JG (2009) It pays to be picky: an evaluation of thread retrieval in online forums. In: SIGIR '09: Proceeding of the 32rd international ACM SIGIR conference on Research and development in information retrieval, pp 714–715
8. Elsas JL, Arguello J, Callan J, Carbonell JG (2008) Retrieval and feedback models for blog feed search. In: SIGIR '08: Proceedings of the 31st annual international ACM SIGIR conference on Research and development in information retrieval, pp 347–354
9. Elsner M, Charniak E (2008) You talking to me? a corpus and algorithm for conversation disentanglement. In: the 46th Annual Meeting of the Association for Computational Linguistics: Human Language Technology Conference (ACL-08: HLT), pp 834–842
10. Erera S, Carmel D (2008) Conversation detection in email systems. Lecture Notes in Computer Science 4956:498–505
11. Joachims T (2002) Optimizing search engines using clickthrough data. In: The eighth ACM SIGKDD Conference on Knowledge Discovery and Data Mining (KDD '02), pp 133–142
12. Krovetz R (1993) Viewing morphology as an inference process. In: SIGIR '93: Proceedings of the sixteenth annual international ACM SIGIR conference on Research and development in information retrieval, pp 191–202
13. Lewis DD, Knowles KA (1997) Threading electronic mail - a preliminary study. Inf Process Manage 33(2):209–217
14. Lin C, Yang JM, Cai R, Wang XJ, Wang W, Zhang L (2009) Modeling semantics and structure of discussion threads. In: The 18th International World Wide Web Conference (WWW '09), pp 1103–1104
15. Liu X, Croft WB (2004) Cluster-based retrieval using language models. In: SIGIR '04: Proceedings of the 27th annual international ACM SIGIR conference on Research and development in information retrieval, pp 186–193
16. Liu X, Croft WB (2008) Evaluating text representations for retrieval of the best group of documents. In: Proceedings of 30th European Conference on IR Research, (ECIR 2008), pp 454–462
17. Ogilvie P, Callan J (2004) Hierarchical language models for retrieval of XML components. In: INitiative for the Evaluation of XML Retrieval (INEX) 2004
18. Petkova D, Croft WB (2007) UMass at TREC 2006: Enterprise track. In: The fifteenth Text REtrieval Conference (TREC 2006) Proceedings
19. Porter M (1980) An algorithm for suffix stripping. Program 14(3):130–137
20. Seo J, Croft WB (2008) Blog site search using resource selection. In: CIKM '08: Proceedings of the seventeenth ACM international conference on Information and knowledge management, pp 1053–1062

21. Seo J, Croft WB (2010) Geometric representations for multiple documents. In: SIGIR '10: Proceeding of the 33rd international ACM SIGIR conference on Research and development in information retrieval, pp 251–258
22. Shrestha L, McKeown K (2004) Detection of question-answer pairs in email conversations. In: COLING '04: The 20th International Conference on Computational Linguistics
23. Smith M, Cadiz JJ, Burkhalter B (2000) Conversation trees and threaded chats. In: CSCW '00: Proceedings of the 2000 ACM conference on Computer supported cooperative work, pp 97–105
24. Soboroff I, de Vries AP, Craswell N (2007) Overview of the TREC 2006 enterprise track. In: Text REtrieval Conference (TREC) 2006
25. Wang L, Oard DW (2009) Context-based message expansion for disentanglement of interleaved text conversations. In: Proceedings of Human Language Technologies: The 2009 Annual Conference of the North American Chapter of the Association for Computational Linguistics, pp 200–208
26. Wang YC, Joshi M, Cohen WW, Rose C (2008) Recovering implicit thread structure in newsgroup style conversations. In: Proceedings of the Second International Conference on Weblogs and Social Media (ICWSM 2008)
27. Xi W, Lind J, Brill E (2004) Learning effective ranking functions for newsgroup search. In: SIGIR '04: Proceedings of the 27th annual international ACM SIGIR conference on Research and development in information retrieval, pp 394–401
28. Yeh JY, Harnly A (2006) Email thread reassembly using similarity matching. In: CEAS 2006 - Third Conference on Email and Anti-Spam
29. Zhai C, Lafferty J (2001) A study of smoothing methods for language models applied to ad hoc information retrieval. In: SIGIR '01: Proceedings of the 24th annual international ACM SIGIR conference on Research and development in information retrieval, pp 334–342