# A Hierarchical Entity-based Approach to Structuralize User Generated Content in Social Media: A Case of Yahoo! Answers

**Baichuan Li**[1,2]*, **Jing Liu**[3]*, **Chin-Yew Lin**[4], **Irwin King**[1,2], **and Michael R. Lyu**[1,2]
[1]Shenzhen Research Institute, The Chinese University of Hong Kong, Shenzhen, China
[2]Department of Computer Science and Engineering
The Chinese University of Hong Kong, Shatin, N.T., Hong Kong
[3]Harbin Institute of Technology, Harbin 150001, P.R. China
[4]Microsoft Research Asia, Beijing 100080, P.R. China
bcli@cse.cuhk.edu.hk jliu@ir.hit.edu.cn cyl@microsoft.com
{king,lyu}@cse.cuhk.edu.hk

## Abstract

Social media like forums and microblogs have accumulated a huge amount of user generated content (UGC) containing human knowledge. Currently, most of UGC is listed as a whole or in pre-defined categories. This "list-based" approach is simple, but hinders users from browsing and learning knowledge of certain topics effectively. To address this problem, we propose a hierarchical entity-based approach for structuralizing UGC in social media. By using a large-scale entity repository, we design a three-step framework to organize UGC in a novel hierarchical structure called "cluster entity tree (CET)". With Yahoo! Answers as a test case, we conduct experiments and the results show the effectiveness of our framework in constructing CET. We further evaluate the performance of CET on UGC organization in both user and system aspects. From a user aspect, our user study demonstrates that, with CET-based structure, users perform significantly better in knowledge learning than using traditional list-based approach. From a system aspect, CET substantially boosts the performance of two information retrieval models (i.e., vector space model and query likelihood language model).

## 1 Introduction

With the development of Web 2.0, social media websites—such as online forums, blogs, microblogs, social networks, and community question answering (CQA) portals—have become the mainstream of web, where users create, share, and exchange information with each other. As a result, more and more UGC is accumulated, with social media websites retaining a huge amount of human knowledge and user experience. At present, most of UGC is organized in a list structure with extra information (e.g., category hierarchies in online forums), or without any other information.

Table 1: Sample questions about *Edinburgh*

| |
|---|
| 1. Where can i buy a **hamburger** in **Edinburgh**? |
| 2. Where can I get a **shawarma** in **Edinburgh**? |
| 3. How long does it take to drive between **Glasgow** and **Edinburgh**? |
| 4. Whats the difference between **Glasgow** and **Edinburgh**? |
| 5. Good **hotels** in **London** and **Edinburgh**? |
| 6. Looking for nice , clean cheap **hotel** in **Edinburgh**? |
| 7. Does anyone know of a reasonably cheap **hotel** in **Edinburgh** that is near to **Niddry Street South** ? |
| 8. Who can recommend a affordable **hotel** in **Edinburgh City Center**? |

This "list-of-content" (list-based approach) is simple and straightforward, but ineffective for browsing and knowledge learning. Consider the following case: a user wants to spend his vacation in Edinburgh. He visits a CQA website to explore which aspects are mostly asked. In this scenario, he may browse some relevant categories like "Travel:United Kingdom:Edinburgh" to get useful information. He may also issue a query like "travel in Edinburgh" to search relevant questions. However, both the browsing and the searching give the user a list of relevant contents (e.g., questions shown in Table 1), not the direct knowledge. Thus, the user has to read these contents, understand them, classify them into various topics, and gain valuable

*This work was done when the first two authors were on internship at MSRA.



*Proceedings of the 2013 Conference on Empirical Methods in Natural Language Processing*, pages 1521–1532,
Seattle, Washington, USA, 18-21 October 2013. 

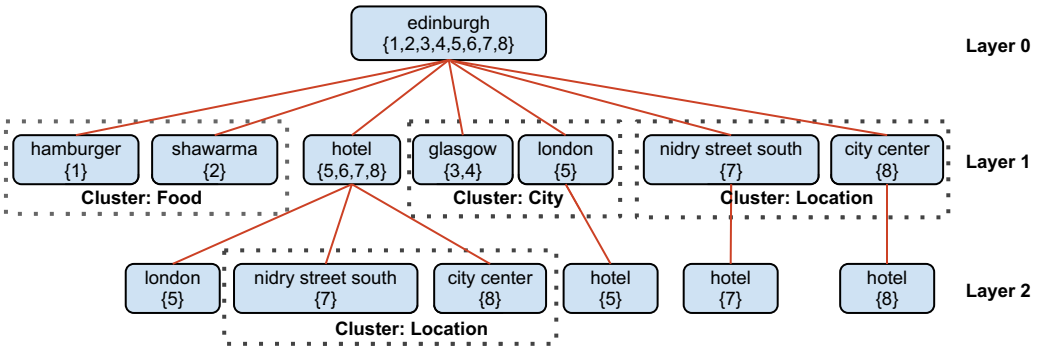

Figure 1: An CET constructed from questions about *Edinburgh*

knowledge himself. Obviously, it is ineffective and time-consuming.

The above problem calls for a new approach to structuralize UGC in social media, which facilitates users to seek knowledge (e.g., travel information about Edinburgh) more effectively. Traditionally, we can utilize topic models (Blei et al., 2003) or social tagging to structuralize UGC. However, for topic models, it is not easy to control the granularity of topics, and it is hard for users to interpret a topic only based on the multinomial distribution (Mei et al., 2007). For social tagging, it is not applicable in many sites and has sparsity problem (Shepitsen et al., 2008). Thus, both topic models and social tagging are not suitable for structuralizing UGC in social media.

In this paper, we propose a novel hierarchical entity-based approach, i.e., "cluster entity tree" or CET, to structuralize UGC in social media by leveraging an existing large-scale entity repository. Figure 1 shows how CET structuralizes UGC in Table 1. In this CET, each node contains one (named) entity and a set of question IDs. With "edinburgh" as the root entity, layer 1 includes all entities that co-occur with "edinburgh". Similarly, entities on layer 2 co-occur with their parent entities on layer 1 and the root entity "edinburgh". For example, "city center" co-occurs with "hotel" and "edinburgh" in Question 8. Deeper layers provide more specific topics about different aspects of Edinburgh (e.g., Edinburgh's hotels). As shown in Fig. 1, the corresponding question IDs are also attached in each node. In addition, entities which share the same parent are clustered to different groups (see dashed rectangles in Fig. 1)[1]. Through this hierarchical structure, we can easily browse corresponding questions and answers, and learn knowledge about Edinburgh more effectively, such as food and location. Moreover, since similar entities (and corresponding questions) are grouped in each layer, it also helps the system manage similar contents.

By utilizing a large-scale entity repository, CET avoids the granularity, interpretation, and sparsity problems. Entity repositories like Freebase[2] provide a large number of named entities across various pre-defined topics, which avoid the granularity and sparsity problems. In addition, they usually give descriptions of entities, which prevent the interpretation problem. Therefore, CET is more suitable for structuralizing UGC.

In this paper, we propose a three-step framework to construct CETs.

1. Entity extraction. In this step, we extract entities from documents using an existing entity repository.
2. Tree construction. we build the co-occurrence relationship between any two entities and construct hierarchical "entity trees (ETs)".
3. Hierarchical entity clustering. In an ET, some entities are more similar than other entities which share the same parents. Therefore, on each layer of the ET we cluster entities with the same parents (e.g., "london", "nidry street south", and "city center" on layer 2 of Fig. 1) and finally construct a CET.

We select Yahoo! Answers as a test case to evaluate 1) the performance of our framework for con-

[1]Single-node clusters are not surrounded by rectangles, like "hotel{5}" on layer 2.

[2]http://www.freebase.com/

structing CET and 2) the effectiveness of CET in UGC structuralization. Yahoo! Answers is a popular CQA portal, where users post questions and provide answers in different categories. Experimental results demonstrate the good performance of our framework for constructing CET. We further evaluate the effectiveness of CET in structuralizing UGC from both user and system aspects. From a user aspect, our user study show that, with CET-based organization, users perform significantly better in knowledge learning than using list-based approach. From a system aspect, CET boosts systems' information retrieval substantially: the mean reciprocal rank (MRR) of vector space model (VSM) and query likelihood language model (QLLM) are improved by 9.3% and 8.2%, respectively.

To summarize, our contributions are three-fold:

1. We propose a novel hierarchical entity-based approach to structuralize UGC in social media. To our knowledge, we are the first to utilize entities to structuralize UGC for content browsing and knowledge learning at a large scale;
2. We present a three-step framework to construct CETs and show its effectiveness from empirical results;
3. We demonstrate the significant advantages of our approach for both users and systems in knowledge learning and retrieval.

The paper proceeds as follows. We present related work in Section 2. We detail our framework to construct CETs and show empirical results in Section 3. Section 4 and Section 5 evaluate the effectiveness of CET on knowledge organization from user and system aspects respectively. We conclude the paper in Section 6.

## 2 Related Work

**UGC organization in social media.** Most UGC in social media is unstructured, or organized in a predefined category hierarchy. These categories give shallow semantics of UGC, and boosts the performance of information retrieval (Cao et al., 2008; Duan et al., 2008; Cao et al., 2012) and recommendation (Guo et al., 2008; Li et al., 2011). With new content kept adding into a hierarchy, we need to maintain category hierarchy (Yuan et al., 2012) to make content within the same category more topically cohesive.

Apart from category hierarchy, UGC can also be organized by topic models and tags. Topic models, such as latent Dirichlet allocation (LDA) (Blei et al., 2003), are widely applied in document clustering and classification, However, it is not trivial to control the granularity of topics (Chen et al., 2011). What's more, it is generally difficult to understand a topic only from the multinomial distribution (Mei et al., 2007). Social tagging provides an alternative approach in document organization (Gupta et al., 2010). In some UGC websites like stackoverflow[3], users usually add tags to describe their contents. However, tags are not widely applicable and tags are usually sparse (Shepitsen et al., 2008). Our hierarchical entity-based approach prevents these problems by employing a large-scale entity repository. As the entity repository provides a unified set of entities across various of pre-defined topics, and gives descriptions of entities, CET avoids the granularity, interpretation and sparsity issues.

The work in (Zhu et al., 2013), which automatically generates and updates topic terms to organize UGC, is mostly related to our work. In this paper, given a root topic, subtopics and lower-level topics are extracted from UGC, which form a hierarchical structure to organize corresponding UGC. However, in (Zhu et al., 2013) more external sources are utilized to identify subtopics. In addition, relationships among subtopics which under the same parent are not investigated. The metro maps proposed in (Shahaf et al., 2013) are also related to our work. Different from (Shahaf et al., 2013), we employ a large-scale entity repository to extract more meaningful and interpretable key terms (entities), which make each subtopic much easier to understand.

**Entity extraction.** In our framework, we leverage an entity repository to extract named entities from UGC. A common approach is to utilize a Named Entity Recognition (NER) system like Stanford NER (Finkel et al., 2005), which recognizes the names of things (e.g., person and product names) from texts. For cross-domain NER, (Rüd et al., 2011) employed search engines. For short-text NER, (Liu et al., 2012) proposed a graphical model. However, most of above systems are restricted from

[3]http://stackoverflow.com/

producing labels for a few entity classes. To address the problem, (Ling and Weld, 2012) defined a fine-grained set of 112 tags based on Freebase for entity extraction. However, this approach still faces the "low-recall" problem in the real world. Our approach, which leverages a large-scale entity repository, addresses this issue.

**Entity-based document classification and retrieval.** Entity repository has been employed in other research areas, like document classification and retrieval. (Schonhofen, 2006) utilized Wikipedia to classify documents. (Yerva et al., 2012a) proposed an entity-based classification for tweets. In addition, entity-based retrieval models were proposed and applied in both QA archives (Singh, 2012a) and tweets (Yerva et al., 2012b). Besides, (Singh, 2012b) proposed an entity-based translation language model and demonstrated that it outperformed classical translation language model in question retrieval. However, to the best of our knowledge, no previous study leverages entities to organize UGC in social media.

## 3 CET Construction

In this section, we formulate the framework to construct CET and show the empirical results. Firstly, we provide the definitions of the entity repository and CET.

**Definition 3.1 (Entity Repository)** *Let $ER = \{R, g\}$ be an entity repository, where $R$ is a set of named entities and $g : R \times R$ is a mapping function that defines the similarity of any two entities.*

Note that we do not require a hierarchical structure in an ER (like Freebase); only a similarity function is needed.

**Definition 3.2 (Cluster Entity Tree)** *Let $D$ be a set of documents, $ER = \{R, g\}$ be an entity repository, $e$ be an entity, a cluster entity tree $CET_e = (v_e, V, E, C)$ is defined as a tree structure, with the root node $v_e$, node set $V$, edge set $E$, and cluster set $C$. Each node $v_s \in V$ on $CET_e$ includes an entity extracted from the set of documents $D_e \in D$ containing $e$, and a list $L(s)$ which stores the indexes of documents containing entity $s$ and its superior entities. If $v_s$ is $v_t$'s parent node, entity $t$ must co-occur with $s$ and $s$'s all superior entities at least once in the same document. Each element of $C$ (one cluster) includes a set of nodes which share the same parent node, and the entities within a cluster are more similar to each other than the entities in other clusters.*

### 3.1 Framework

This section shows our three-step framework for constructing CET: entity extraction (Section 3.2.1), tree construction (Section 3.2.2), and hierarchical entity clustering (Section 3.2.3).

#### 3.1.1 Entity Extraction

We adopt a simple entity repository based approach to extract entities, which address the "low-recall" problem for traditional NER methods (details are given in Section 3.3.2). This approach involves two phases: candidate entity extraction and entropy-based filtering.

**Candidate entity extraction.** We employ the Stanford Parser[4] to parse each document to a parse tree. Then, we extract all noun phrases, preprocess them (including stemming), and extract the noun phrases which are included in our entity repository. In our experiments, we adopt a large-scale enterprise entity repository (anonymized for blind reviews).

**Entropy-based filtering.** The candidate entities generated from the last step may contain many false examples, which are not relevant to the main semantics of documents, like "we", "how do i", etc. To filter them, we propose an entropy-based method. Given a document with a category label (or tags, which are available in most UGC sites), we get the distributions of each candidate entity over all top categories. The entropy of a candidate entity $e_i$ is calculated as follows:

$$Entropy(e_i) = -\sum_{c=1}^{|C|} P_c(e_i) log P_c(e_i), \quad (1)$$

where $|C|$ is the number of top categories and $P_c(e_i)$ is the number of $e_i$ in category $c$ divided by all number of candidate entities in that category.

Top-ranked entities are general terms among categories. We set a threshold $\alpha$ and remove all candidate entities with entropy larger than $\alpha$. The setting of $\alpha$ is a tradeoff: higher values will introduce more noise, while smaller values will lead to decreased

[4] http://nlp.stanford.edu/software/lex-parser.shtml

recall. In our experiments, we empirically set $\alpha$ as 1.5 since it provides the most satisfying results.

#### 3.1.2 Tree Construction

Given an entity (e.g., "edinburgh"), we first search documents containing this entity and make the entity together with document ids as the root node. Then, from searched documents we find all entities that co-occur with the root entity. These entities and corresponding document ids form layer-1 nodes of the entity tree (see the example in Fig. 1). Afterwards, for each entity in layer-1 nodes, we search entities that co-occur with it and its superiors, combine them and corresponding document ids as new nodes, and put these new nodes under current node, which form layer-2 nodes. Iteratively, we construct the entity tree with the given entity as the root.

#### 3.1.3 Hierarchical Entity Clustering

Under the same parent, some entities[5] may share similar topics. Therefore, the final step is to hierarchically cluster entities with the same parents at different layers of entity trees. This step not only facilitates knowledge learning but also reduces the width of a tree. In this paper, we follow the work in (Hu et al., 2012) and employ an agglomerative clustering algorithm for two reasons: 1) it is easy to implement and its time complexity is $O(N^2)$; 2) there is no need to set the number of clusters. Although the clustering results may be influenced by instance order, our empirical results demonstrate its effectiveness. Any other advanced algorithms like spectral clustering (Ng et al., 2002) can also be applied, but that is not the emphasis of this paper.

**Algorithm.** For a set of entities with the same parent, the agglomerative clustering algorithm works as follows:

1. Select one entity and create a new cluster which contains the entity;
2. Select the next entity $e_i$, create an empty candidate list, calculate the similarity between the entity and all existing clusters. Three strategies are employed[6]:
   - AC-MAX: If the similarity between entity $e_i$ and entity $e_j$ in one of the clusters (the first one) is larger than threshold $\theta_{max}$, we put the cluster index and corresponding similarity in the candidate list.
   - AC-MIN: If the similarity between entity $e_i$ and any entity $e_j$ in one of the clusters is larger than threshold $\theta_{min}$, we put the cluster index and corresponding similarity in the candidate list.
   - AC-AVG: If the mean similarity between entity $e_i$ and any entity $e_j$ in one of the clusters is larger than threshold $\theta_{avg}$, we put the cluster index and corresponding similarity in the candidate list.
3. If the candidate list is not empty, put $e_i$ in the cluster with highest similarly.
4. If the candidate list is empty, a new cluster with $e_i$ as the element will be created.
5. Stop when all entities are clustered.

**Similarity Function.** In our entity repository, the similarity between two entities is computed using the approach in (Shi et al., 2010), which estimates the similarity of two terms according to their *first-order* and *second-order* co-occurrences. For example, "such as NP, NP" is a good pattern for detecting similar entities using *first-order* co-occurrences. In addition, if two entities usually co-occur with a third entity (*second-order* co-occurrence), these two entities are likely to be similar. To construct similarity functions, pattern-based approaches (Ohshima et al., 2006; Zhang et al., 2009) utilize *first-order* co-occurrences while distributional similarity approaches (Pasca et al., 2006; Pennacchiotti and Pantel, 2009) employ *second-order* co-occurrences. In the following, we briefly introduce the pattern-based approach (PB) and the distributional similarity approach (DS) in (Shi et al., 2010).

**PB.** Some well-designed patterns are leveraged to extract similar entities from a huge repository of webpages. The set of terms extracted by applying a pattern one time is called a raw semantic class (RASC). Given two entities $t_a$ and $t_b$, PB calculates their similarity based on the number of RASCs containing both of them (Zhang et al., 2009):

$$Sim(t_a, t_b) = \log(1 + \sum_{i=1}^{r_{ab}} P_{ab_i})) \cdot \sqrt{idf(t_a) \cdot idf(t_b)}, \quad (2)$$

where $idf(t_a) = \log(1 + \frac{N}{C(t_a)})$, $P_{ab_i}$ is a pattern which can generate RASC(s) containing both term

[5]Here we use the entity to represent the node.

[6]We modify the clustering algorithm in (Hu et al., 2012) slightly to assign a unique cluster for each entity.

$t_a$ and term $t_b$, $r_{ab}$ is the total number of such patterns, $N$ is the total number of RASCs, and $C(t_a)$ is the number of RASCs containing $t_a$. The above similarity is normalized using the following function:

$$Sim_{PB}(t_a, t_b) == \frac{\log Sim(t_a, t_b)}{2 \log Sim(t_a, t_a)} + \frac{\log Sim(t_a, t_b)}{2 \log Sim(t_b, t_b)}. \quad (3)$$

**DS.** DS approach assumes that terms appearing in similar contexts tend to be similar. In this approach, a term is represented by a feature vector, with each feature corresponding to a context in which the term appears. The similarity between two terms is computed as the similarity between their corresponding feature vectors. Jaccard similarity is employed to estimate the similarity between two terms. Suppose the feature vector of $t_a$ and $t_b$ are **x** and **y** respectively,

$$Sim_{DS}(t_a, t_b) = \frac{\sum_i \min(x_i, y_i)}{\sum_i (x_i) + \sum_i (y_i) - \sum_i \min(x_i, y_i)}. \quad (4)$$

Shi et al. (Shi et al., 2010) found that PB performed better when dealing with proper nouns; while DS was relatively good at estimating similarity of other types of entities. The similarity function in our ER follows the suggestion of (Shi et al., 2010): if at least one entity is proper noun, PB is employed; otherwise DS is used.

### 3.2 Experiments

#### 3.2.1 Setup

We evaluate the performance of our framework employing questions from Yahoo! Answers. 54.7 million questions are crawled from all 26 top categories in Yahoo! Answers, which consist of question titles and corresponding categories. From these questions, we construct the following two test sets for evaluating entity extraction and entity clustering:

**Set EE.** This set is employed to evaluate the performance of entity extraction. It contains 520 randomly sampled questions, 20 from each top category. One author is asked to label entities for each question.

**Set EC.** This set is constructed to automatically evaluate hierarchical entity clustering and select the best clustering strategy. The construction process is as follows. First, we map the four top categories of Yahoo! Answers to some categories of Freebase manually, as shown in Table 2. Second, from questions at each top category of Yahoo! Answers, we extract entities which appear exactly once in the corresponding Freebase categories. For instance, if an entity is extracted from questions in the category *Computers & Internet*, and it appears two or more times in *Computer* and *Internet* categories in Freebase, it will be filtered. Therefore, each entity is attached with a unique Freebase category label (i.e., the ground truth for clustering). Questions containing at least two entities are selected for Set EC. Table 3 reports the statistics. Intuitively, entities with a same Freebase category label should be in one cluster.

Table 2: Category mapping between Yahoo! Answers and FreeBase

| Yahoo! Answers | FreeBase |
|---|---|
| Cars & Transportation | Aviation, Transportation, Boats Spaceflight, Automotive, Bicycles, Rail |
| Computers & Internet | Computer, Internet |
| Sports | Soccer, Olympics,Sports, American football, Baseball,Basketball,Ice Hockey,Martial Arts, Cricket,Tennis,Boxing,Skiing |
| Travel | Travel, Location, Transportation |

Table 3: Number of questions and entities in Set EC

| Category | Number of Questions | Number of Entities |
|---|---|---|
| Cars & Transportation | 1,220,427 | 3,267,596 |
| Computers & Internet | 2,912,280 | 7,324,655 |
| Sports | 2,363,758 | 6,230,868 |
| Travel | 1,347,801 | 3,728,286 |

Note that Set EC only covers a small set of real entities and clustering on Set EC is partial clustering. However, it leverages Freebase labels and avoids manual labeling, which is time-consuming. Furthermore, partial clustering results are enough for evaluating different strategies' performance and choosing the best strategy.

Following the common practice, we evaluate entity extraction using precision, recall, and $F1$ score. For evaluating entity clustering results, we adopt B-cubed metrics. As reported in (Amigó et al., 2009), B-cubed metrics are more suitable than traditional metrics, such as NMI and purity.

Table 4: Entity extraction for various methods

| Method | Precision | Recall | F1 |
|---|---|---|---|
| Standord NER | 0.750 | 0.155 | 0.257 |
| FIGER | 0.763 | 0.154 | 0.256 |
| Freebase | 0.644 | 0.595 | 0.619 |
| Our | 0.647 | 0.809 | 0.719 |

Table 5: Clustering results using AC-MAX ($\theta_{max}$=0.1)

| Level | *Travel* | | | | *Cars & Transportation* | | | | *Computer & Internet* | | | | *Sports* | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | Count | P | R | F1 | Count | P | R | F1 | Count | P | R | F1 | Count | P | R | F1 |
| 1 | 748 | 0.972 | 0.653 | 0.743 | 1281 | 0.948 | 0.868 | 0.897 | 3064 | 0.913 | 0.664 | 0.743 | 890 | 0.941 | 0.883 | 0.901 |
| 2 | 200 | 0.974 | 0.730 | 0.798 | 1202 | 0.989 | 0.956 | 0.965 | 11344 | 0.961 | 0.842 | 0.879 | 636 | 0.978 | 0.964 | 0.963 |
| 3 | 120 | 1.000 | 0.833 | 0.890 | 858 | 1.000 | 0.981 | 0.988 | 8184 | 0.978 | 0.899 | 0.920 | 492 | 0.965 | 0.882 | 0.899 |
| 4 | NA | NA | NA | NA | 1776 | 1.000 | 0.980 | 0.986 | 3648 | 0.990 | 0.908 | 0.934 | 1080 | 0.978 | 0.844 | 0.881 |
| 5 | NA | NA | NA | NA | NA | NA | NA | NA | 2520 | 1.000 | 0.952 | 0.968 | NA | NA | NA | NA |
| Total | 1068 | 0.976 | 0.688 | 0.770 | 5117 | 0.984 | 0.946 | 0.959 | 28760 | 0.968 | 0.857 | 0.891 | 3098 | 0.965 | 0.886 | 0.907 |

#### 3.2.2 Results: Entity Extraction

Two ERs (i.e., ours and Freebase) are employed in entity extraction for comparison. In addition, we compare our approach with Stanford NER (Finkel et al., 2005) and fine-grained entity recognition (FIGER) (Ling and Weld, 2012). Table 4 reports the results of different methods. We can find that Stanford NER and FIGER get a relatively high precision in extracting entities. However, their recalls are very low and only about 15% of entities are recognized. With the help of entity repositories, recall is significantly improved with a small decrease of precision. Therefore, the $F1$s of entity-based approaches are much higher. This observation shows the great advantage of utilizing an entity repositories in entity extraction and the effectiveness of our approach. As our ER performs better than Freebase, we adopt it as our entity repository in the following evaluations.

#### 3.2.3 Results: Hierarchical Entity Clustering

Table 5 reports the count of clusters, B-Cubed Precision, Recall, and F1 for different layers of clustering across four categories using **AC-MAX**. In our experiments, **AC-MAX** performed better than **AC-MIN** and **AC-AVG**. Due to space limitations, we only report the results of **AC-MAX** here. For **AC-MAX**, we changed the settings of $\theta$ from 0.01 to 0.9, and the best performance was achieved when $\theta_{max}$ was set at around 0.1. From Table 5 we can find that, although **AC-MAX**'s accuracy varies across categories (e.g., the $F1$ of *Transportation* is much higher than that of *Travel*), it performs well in general. Thus, we adopt **AC-MAX** with $\theta = 0.1$ for hierarchical entity clustering.

## 4 User Study

In this section, we investigate the influence of CET on users' content browsing and knowledge learning from a user study. In the study, we design 24 tasks in four popular Yahoo! Answers categories (see Table 2). For each category, we design three ***knowledge-learning*** tasks and three ***question-search*** tasks, as shown in Table A.1 in the supplementary material. A ***knowledge-learning*** task asks for some knowledge about a main entity from question texts. For instance, "find the games running on macbook pro" requires game names as the answer, where the main entity is "macbook pro". A ***question-search*** task, however, asks users to find similar questions to the question in the task. For example, "questions about who will win the MVP in NBA this year" asks for finding similar questions, and filling their question IDs as the answer. For each task, we give some suggested keywords (entities) to facilitate information gathering.

To evaluate user experience, we ask participants to fill out a questionnaire after each task. Following the work in (Kato et al., 2012), we collect information from 5 aspects: familiarity, easiness, satisfaction, adequate time, and helpfulness. A 5-point Likert scale is designed for each questionnaire. "5" means the participant totally agrees while "1" means the participant totally disagrees.

### 4.1 Setup

**Programs.** We develop two programs in our user study. One is CET-based, and the other is traditional list-based[7]. The list-based program searches questions by utilizing Apache Lucene[8]. The standard analyzer and the default search algorithm are adopted. For each query, top 200 most relevant questions are retrieved.

**Data.** We extract 70,195 questions which contain at least one of the 24 main entities (see Table A.1)

[7]The interface of CET-based program is provided in the supplementary material (see Fig. A.1). The interface of list-based program is similar, but the CET display area is replaced by a flat-ranked list.

[8]http://lucene.apache.org/core/

Table 6: User study results

| | *Knowledge-learning* Tasks | | *Question-search* Tasks | |
|---|---|---|---|---|
| | CET-based | List-based | CET-based | List-based |
| **# Queries** | 2.99 | 4.47 | 2.56 | 3.38 |
| **# Answers** | 8.32 | 6.06 | 10.60 | 10.92 |
| **Precision** | 0.38 | 0.19 | 0.40 | 0.44 |
| **Time** | 136.44 | 121.87 | 103.71 | 87.75 |

Table 7: Questionnaire results

| | *Knowledge-learning* Tasks | | *Question-search* Tasks | |
|---|---|---|---|---|
| | CET-based | List-based | CET-based | List-based |
| **Familiarity** | 3.18 | 3.22 | 3.07 | 3.28 |
| **Easiness** | 3.64 | 3.66 | 4.10 | 4.06 |
| **Satisfaction** | 3.70 | 2.94 | 3.86 | 3.44 |
| **Enough Time** | 3.87 | 3.83 | 4.44 | 4.54 |
| **Helpfulness** | 4.16 | 3.03 | 4.31 | 3.71 |

in the four categories. For each question, we extract the entities with the help of our entity repository. For each main entity, we build the corresponding CET from all extracted questions.

**Participants.** Sixteen volunteers are invited in the user study. They are first briefly informed of the research design and taught how to use two programs. To familiarize the participants with our programs promptly, we provide demonstrations using sample entities. Each volunteer is asked to finish 12 tasks (6 ***knowledge-learning*** tasks and 6 ***question-search*** tasks) using the CET-based program and 12 other tasks using the list-based program in random order. Thus, each task is finished by exactly 8 different participants using each program.

### 4.2 Results and Discussions

Table 6 reports the user study results, where we give the statistics for users' performance with the two programs. We evaluate from the number of queries issued, number of answers found, the precision of answers, and query time for each task. As our 24 tasks contain both knowledge-learning tasks and question-search tasks, we report their results separately. Z-tests are employed for significance tests.

From Table 6, we observe that more queries are issued in the knowledge-learning tasks than in the question-search tasks using both programs. However, the CET-based program reduces the number of queries substantially in both tasks. Because the CET-based program provides a series of clustered entities, it helpes users further refine queries through clicking on entities rather than reconstructing a new query. However, the list-based program only lists relevant questions, and users have to issue new queries according to returned questions.

By using the CET-based program, volunteers find more answers in knowledge-learning tasks ($z = 1.69, p < 0.05$). The reason is that the CET-based program clusters similar results in the same group, and if the user finds one answer she can easily get more answers. On the contrary, the list-based program returns a list of questions, and users need to find answers question-by-question. For question-search tasks, users of the list-based program find more answers, but the difference is not significant ($z = 0.19$). As the list-based program returns similar questions as top-ranked results, users are able to fill in answers easily. For CET-based program users, they have to find corresponding key entities in the CETs first. Therefore, they spend more time (the fourth row in Table 6) finding entities and less time filling answers. It is worth noting that our GUI prototype for CET is non-optimal, and users' searching time on CET-based program can be further reduced with better user interface.

The precision of answers from CET-based program users is twice of that from list-based program users ($z = 4.15, p < 0.0001$) in knowledge-learning tasks, which demonstrates the advantage of CET in helping knowledge-learning. For question-search tasks, CET-based program users perform slightly worse than list-based program users, but the difference is not significant ($z = 0.48$). Since users of the CET-based program spend more time finding entities, they have limited time to check the answers.

In both tasks, users spend more time on the CET-based program. According to users' post-user-study feedbacks, a few volunteers reported that they sometimes spent a considerable amount of time on finding entities from CETs; however, one positive observation is that most users find "the entity-based interface" very interesting, which stimulates them to spend more time on exploring answers.

The questionnaires reveal more about user experience on these two programs (see Table 7). Users' responses to task familiarity and easiness are similar. However, users of entity-based interface are more satisfied in both knowledge-learning tasks ($z = 3.98, p < 0.0001$) and question-search tasks ($z =$

1.38), and they feel that entity-based interface is more helpful in finding answers for both knowledge-learning tasks ($z = 6.47, p < 0.0001$) and question-search tasks ($z = 2.55, p < 0.01$). These promising observations show that CET helps knowledge learning greatly through structuralizing content.

## 5 CET-based Question Re-Ranking

In this section, we show that CET also helps systems to better retrieve information through re-ranking. In the following, we continue to use Yahoo! Answers as a test case.

**Algorithm 1** CET-based search results re-ranking

**Input:** query $q$, question collection $Q$, a ranked list of $k$ relevant questions $Q_q = \{q_1, q_2, ..., q_k\}$ to $q$, an entity repository ER, an empty list $\Theta$.

**Output:** A new ranked list of questions.

1: Extract entities from each question of $Q$ and construct a entity co-occurrence graph;
2: Get the PageRank score of each entity;
3: **if** There is no entity $e$ in $q$ **then**
4: return $Q_q$;
5: **else**
6: Identify the key entity $e$ from $q$ which has the highest PageRank score;
7: Construct the CET $cet_e$ from $Q$ based on ER;
8: **for** each question $q_i$ **do**
9: For all entities in $q_i$, build a entity chain $C$ in descending order of PageRank scores;
10: From $C$ extract the first entity $\hat{e}$ that is not similar to $e$;
11: **if** $\hat{e}$ exists **then**
12: Put $q_i$ in the corresponding cluster of nodes;
13: **else**
14: Put $q_i$ in $\Theta$;
15: **end if**
16: **end for**
17: **end if**
18: Rank all clusters on $cet_e$ according to their first elements' original ranking;
19: Output the final ranking cluster by cluster and append the questions in $\Theta$ at the last.

Intuitively, questions sharing similar topics should be ranked similarly. However, traditional question retrieval models (Cao et al., 2010) such as QLLM and VSM do not capture key semantics and give more weights for entity terms. CET provides a feasible way to address this issue. By utilizing CET, entities are given more weight while trivial words are not. In addition, through clustering questions with similar topics, those questions which are ranked lower will be brought higher by their top-ranked neighbors.

Table 8: Re-ranking results for VSM and QLLM (* means that $p < 0.05$ in students' t-test)

| | VSM | Re-ranking | QLLM | Re-ranking |
|---|---|---|---|---|
| **MRR** | 0.3838 | **0.4195*** (9.30%) | 0.3593 | **0.3889*** (8.24%) |
| **MAP** | 0.3376 | **0.3558*** (5.39%) | 0.3326 | **0.3479*** (4.60%) |
| **Prec@1** | 0.2500 | **0.3125*** (25.00%) | 0.2438 | **0.2688*** (10.25%) |

Algorithm 1 illustrates the re-ranking algorithm in detail. We first extract entities from each question of the whole question collection, and construct a entity co-occurrence graph (Line 1). Then, we calculate the PageRank score of each entity (Line 2). Line 3-5 check whether $q$ contains at least one entity. If the answer is no, we return the original ranking. Otherwise, we identify the key entity in $q$ (Line 6) and construct the CET $cet_e$ whose root entity is $e$ (Line 7). Line 8-16 iteratively put questions in corresponding clusters of $cet_e$. In Line 8, we first build an entity chain for question $q_i$, in which entities of $q_i$ are ranked according to their PageRank scores. Afterwards, the first entity $\hat{e}$, which is not similar to $e$ (the similarity is calculated in Section 4.2.1 and the threshold of similarity is set to 0.1), is picked up as the main aspect of $e$, and $q_i$ is grouped into the corresponding cluster on $cet_e$ (Line 7-8). If $\hat{e}$ does not exist, we put $q_i$ in a new cluster (Line 13-14). Then, we rank all clusters according to their first elements' original rankings (Line 18) and output the final re-ranked list (Line 19).

We perform our re-ranking on 160 randomly selected questions from *Computers & Internet* and *Travel* categories of our data set[9]. Each category contains 80 questions. All other questions in these two categories constitute the question collection $Q$. For each question, we utilize the VSM and QLLM[10] respectively to get the top 15 most relevant questions (excluding itself). The correct ranking is manually labeled and checked by two annotators. We firstly employed the VSM and QLLM respectively to retrieve the top 15 results and then obtained manual judgments. Given a retrieved question by VSM

[9]These 160 questions are not used for constructing the entity co-occurrence graph.

[10]Following (Zhai and Lafferty, 2004), we set $\lambda$ to 0.2.

or QLLM, two assessors are asked to label it with "relevant" or "irrelevant". If their annotations are opposite, the third assessor is involved to determine the final label.

We re-rank these questions using Algorithm 1. Table 8 shows the results of MRR, mean average precision (MAP), and Precision@1. We can see that CET-based re-ranking improves the performance of standard retrieval models substantially. For VSM, our re-ranking boosts the MRR and MAP by 9.3% and 5.4%, respectively. It is worth noting that our re-ranking improves Prec@1 significantly: from 0.25 to 0.31. The reason is that traditional methods may give relatively low weights to the key terms (entities), while CET-based re-ranking addresses the problem. QLLM and re-ranking report similar results. Figures 2 and 3 illustrate the performance of various approaches across categories. We find that our re-ranking is neither category-biased nor algorithm-biased, yet it performs better than original models on both categories. The above results demonstrate that, by utilizing the hierarchical entity-based approach, CET greatly improves the retrieval performance of these two standard models.

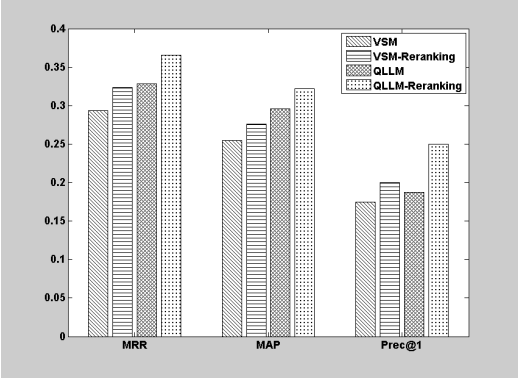


Figure 2: Re-ranking results of *Computer & Internet*

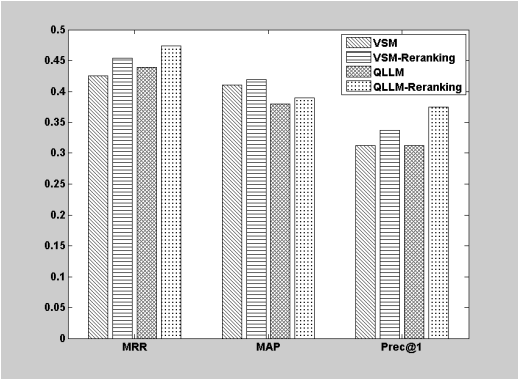


Figure 3: Re-ranking results of *Travel*

## 6 Conclusion and Future Work

Traditional list-based organization of UGC in social media is not effective for content browsing and knowledge learning due to large volume of documents. To address this problem, we propose a novel hierarchical entity-based approach to structuralize UGC in social media. By using a large-scale entity repository, we construct a three-step framework to organize knowledge in "cluster entity trees". Experimental results show the effectiveness of the framework in constructing CET. We further evaluate the performance of CET on knowledge organization from both user and system aspects. Our user study demonstrates that, with CET-based organization, users perform significantly better in knowledge learning than using list-based approach. In addition, CET boosts systems' content search performance substantially through re-ranking.

To our best knowledge, this work is the first attempt to utilize entities to structuralize UGC in social media, and there are some limitations to be improved in our future work. First, we employ Yahoo! Answers as our test data, in which questions (documents) are usually short. We observe that nearly 92% of all 54.7 million questions contain 1-4 entities, which means the depth of CETs are usually not so deep. However, long documents, such as Blog posts, will lead to deep CETs and hinder users' knowledge learning. Second, our current entity extraction focuses on named entities instead of canonical entities. In the future, we plan to employ document summarization techniques to shorten the depth of CETs. We also aim to incorporate semantic analysis and normalize named entities to canonical entities, which make CET more suitable for practical use.

## Acknowledgments

The work described in this paper was fully supported by the Shenzhen Major Basic Research Program (Project No. JC201104220300A) and the Research Grants Council of the Hong Kong Special Administrative Region, China (Project Nos. CUHK413212, CUHK415212).

## References


Enrique Amigó, Julio Gonzalo, Javier Artiles, and Felisa Verdejo. 2009. A comparison of extrinsic clustering evaluation metrics based on formal constraints. *Inf. Retr.*, 12(4):461–486, August.

David M. Blei, Andrew Y. Ng, and Michael I. Jordan. 2003. Latent dirichlet allocation. *J. Mach. Learn. Res.*, 3:993–1022, March.

Yunbo Cao, Huizhong Duan, Chin-Yew Lin, Yong Yu, and Hsiao-Wuen Hon. 2008. Recommending questions using the mdl-based tree cut model. In *Proc. of WWW*, WWW '08, pages 81–90.

Xin Cao, Gao Cong, Bin Cui, and Christian S. Jensen. 2010. A generalized framework of exploring category information for question retrieval in community question answer archives. In *Proceedings of the 19th international conference on World wide web*, WWW '10, pages 201–210, New York, NY, USA. ACM.

Xin Cao, Gao Cong, Bin Cui, Christian S. Jensen, and Quan Yuan. 2012. Approaches to exploring category information for question retrieval in community question-answer archives. *ACM Trans. Inf. Syst.*, 30(2):7:1–7:38, May.

Mengen Chen, Xiaoming Jin, and Dou Shen. 2011. Short text classification improved by learning multi-granularity topics. In *Proceedings of the Twenty-Second international joint conference on Artificial Intelligence - Volume Volume Three*, IJCAI'11, pages 1776–1781. AAAI Press.

Huizhong Duan, Yunbo Cao, Chin yew Lin, and Yong Yu. 2008. Searching questions by identifying question topic and question focus. In *In Proceedings of 46th Annual Meeting of the Association for Computational Linguistics: Human Language Tchnologies (ACL:HLT*.

Jenny Rose Finkel, Trond Grenager, and Christopher Manning. 2005. Incorporating non-local information into information extraction systems by gibbs sampling. In *Proceedings of the 43rd Annual Meeting on Association for Computational Linguistics*, ACL '05, pages 363–370, Stroudsburg, PA, USA. Association for Computational Linguistics.

Jinwen Guo, Shengliang Xu, Shenghua Bao, and Yong Yu. 2008. Tapping on the potential of QA community by recommending answer providers. In *Proc. of CIKM*, CIKM '08, pages 921–930.

Manish Gupta, Rui Li, Zhijun Yin, and Jiawei Han. 2010. Survey on social tagging techniques. *SIGKDD Explor. Newsl.*, 12(1):58–72, November.

Yunhua Hu, Yanan Qian, Hang Li, Daxin Jiang, Jian Pei, and Qinghua Zheng. 2012. Mining query subtopics from search log data. In *Proceedings of the 35th international ACM SIGIR conference on Research and development in information retrieval*, SIGIR '12, pages 305–314, New York, NY, USA. ACM.

Makoto P. Kato, Tetsuya Sakai, and Katsumi Tanaka. 2012. Structured query suggestion for specialization and parallel movement: effect on search behaviors. In *Proc. of WWW*, WWW '12, pages 389–398.

Baichuan Li, Irwin King, and Michael R. Lyu. 2011. Question routing in community question answering: putting category in its place. In *Proc. of CIKM*, CIKM '11, pages 2041–2044.

Xiao Ling and Daniel S. Weld. 2012. Fine-grained entity recognition. In *AAAI*.

Xiaohua Liu, Ming Zhou, Furu Wei, Zhongyang Fu, and Xiangyang Zhou. 2012. Joint inference of named entity recognition and normalization for tweets. In *Proceedings of the 50th Annual Meeting of the Association for Computational Linguistics: Long Papers - Volume 1*, ACL '12, pages 526–535, Stroudsburg, PA, USA. Association for Computational Linguistics.

Qiaozhu Mei, Xuehua Shen, and ChengXiang Zhai. 2007. Automatic labeling of multinomial topic models. In *Proc. of KDD*, KDD '07, pages 490–499.

Andrew Y Ng, Michael I Jordan, Yair Weiss, et al. 2002. On spectral clustering: Analysis and an algorithm. *Advances in neural information processing systems*, 2:849–856.

Hiroaki Ohshima, Satoshi Oyama, and Katsumi Tanaka. 2006. Searching coordinate terms with their context from the web. In *Proceedings of the 7th international conference on Web Information Systems*, WISE'06, pages 40–47, Berlin, Heidelberg. Springer-Verlag.

Marius Pasca, Dekang Lin, Jeffrey Bigham, Andrei Lifchits, and Alpa Jain. 2006. Organizing and searching the world wide web of facts - step one: the one-million fact extraction challenge. In *proceedings of the 21st national conference on Artificial intelligence - Volume 2*, AAAI'06, pages 1400–1405. AAAI Press.

Marco Pennacchiotti and Patrick Pantel. 2009. Entity extraction via ensemble semantics. In *Proceedings of the 2009 Conference on Empirical Methods in Natural Language Processing: Volume 1 - Volume 1*, EMNLP '09, pages 238–247, Stroudsburg, PA, USA. Association for Computational Linguistics.

Stefan Rüd, Massimiliano Ciaramita, Jens Müller, and Hinrich Schütze. 2011. Piggyback: using search engines for robust cross-domain named entity recognition. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies - Volume 1*, ACL-HLT '11, pages 965–975, Stroudsburg, PA, USA. Association for Computational Linguistics.

Peter Schonhofen. 2006. Identifying document topics using the wikipedia category network. In *Proceedings*

*of the 2006 IEEE/WIC/ACM International Conference on Web Intelligence*, WI '06, pages 456–462, Washington, DC, USA. IEEE Computer Society.

Dafna Shahaf, Jaewon Yang, Caroline Suen, Jeff Jacobs, Heidi Wang, and Jure Leskovec. 2013. Information cartography: creating zoomable, large-scale maps of information. In *Proceedings of the 19th ACM SIGKDD international conference on Knowledge discovery and data mining*, KDD '13, pages 1097–1105, New York, NY, USA. ACM.

Andriy Shepitsen, Jonathan Gemmell, Bamshad Mobasher, and Robin Burke. 2008. Personalized recommendation in social tagging systems using hierarchical clustering. In *Proceedings of the 2008 ACM conference on Recommender systems*, RecSys '08, pages 259–266, New York, NY, USA. ACM.

Shuming Shi, Huibin Zhang, Xiaojie Yuan, and Ji-Rong Wen. 2010. Corpus-based semantic class mining: distributional vs. pattern-based approaches. In *Proceedings of the 23rd International Conference on Computational Linguistics*, COLING '10, pages 993–1001, Stroudsburg, PA, USA. Association for Computational Linguistics.

Amit Singh. 2012a. Entity based QA retrieval. In *Proceedings of the 2012 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning*, EMNLP-CoNLL '12, pages 1266–1277, Stroudsburg, PA, USA. Association for Computational Linguistics.

Amit Singh. 2012b. Entity based translation language model. In *Proc. of WWW*, WWW '12 Companion, pages 599–600.

Surender Reddy Yerva, Zoltán Miklós, and Karl Aberer. 2012a. Entity-based classification of twitter messages. *IJCSA*, 9(1):88–115.

Surender Reddy Yerva, Zoltan Miklos, Flavia Grosan, Alexandru Tandrau, and Karl Aberer. 2012b. Tweetspector: entity-based retrieval of tweets. In *Proc. of SIGIR*, SIGIR '12, pages 1016–1016.

Quan Yuan, Gao Cong, Aixin Sun, Chin-Yew Lin, and Nadia Magnenat Thalmann. 2012. Category hierarchy maintenance: a data-driven approach. In *Proceedings of the 35th international ACM SIGIR conference on Research and development in information retrieval*, SIGIR '12, pages 791–800, New York, NY, USA. ACM.

Chengxiang Zhai and John Lafferty. 2004. A study of smoothing methods for language models applied to information retrieval. *ACM Trans. Inf. Syst.*, 22(2):179–214, April.

Huibin Zhang, Mingjie Zhu, Shuming Shi, and Ji-Rong Wen. 2009. Employing topic models for pattern-based semantic class discovery. In *Proceedings of the Joint Conference of the 47th Annual Meeting of the ACL and the 4th International Joint Conference on Natural Language Processing of the AFNLP: Volume 1 - Volume 1*, ACL '09, pages 459–467, Stroudsburg, PA, USA. Association for Computational Linguistics.

Xingwei Zhu, Zhao-Yan Ming, Xiaoyan Zhu, and Tat-Seng Chua. 2013. Topic hierarchy construction for the organization of multi-source user generated contents. In *Proceedings of the 36th international ACM SIGIR conference on Research and development in information retrieval*, SIGIR '13, pages 233–242, New York, NY, USA. ACM.