Overview of the NTCIR-13 OpenLiveQ Task

Makoto P. Kato
Kyoto University
mpkato@acm.org

Takehiro Yamamoto
Kyoto University
tyamamot@dl.kuis.kyoto-u.ac.jp

Tomohiro Manabe
Yahoo Japan Corporation
tomanabe@yahoo-corp.jp

Akiomi Nishida
Yahoo Japan Corporation
anishida@yahoo-corp.jp

Sumio Fujita
Yahoo Japan Corporation
sufujita@yahoo-corp.jp

ABSTRACT

This is an overview of the NTCIR-13 OpenLiveQ task. This task aims to provide an open live test environment of Yahoo Japan Corporation's community question-answering service (*Yahoo! Chiebukuro*) for question retrieval systems. The task was simply defined as follows: given a query and a set of questions with their answers, return a ranked list of questions. Submitted runs were evaluated both offline and online. In the online evaluation, we employed *optimized multileaving*, a multileaving method that showed high efficiency over the other methods in our preliminary experiment. We describe the details of the task, data, and evaluation methods, and then report official results at NTCIR-13 OpenLiveQ.

1. INTRODUCTION

Community Question Answering (cQA) services are Internet services in which users can ask a question and obtain answers from other users. Users can obtain relevant information to their search intents not only by asking questions in cQA, but also by searching for questions that are similar to their intents. Finding answers to questions similar to a search intent is an important information seeking strategy especially when the search intent is very specific or complicated. While a lot of work has addressed the question retrieval problem [5, 1, 6], there are still several important problems to be tackled:

Ambiguous/underspecified queries Most of the existing work mainly focused on specific queries. However, many queries used in cQA services are as short as Web search queries, and, accordingly, ambiguous/underspecified. Thus, question retrieval results also need diversification so that users with different intents can be satisfied.

Diverse relevance criteria The notion of relevance used in traditional evaluation frameworks is usually *topical relevance*, which can be measured by the degree of match between topics implied by a query and ones written in a document. Whereas, real question searchers have a wide range of relevance criteria such as freshness, concreteness, trustworthiness, and conciseness. Thus, traditional relevance assessment may not be able to measure real performance of question retrieval systems.

In order to address these problems, we propose a new task called *Open Live Test for Question Retrieval* (*OpenLiveQ*), which provides an open live test environment of Yahoo! Chiebukuro[1] (a Japanese version of Yahoo! Answers) for question retrieval systems. Participants can submit ranked lists of questions for a particular set of queries, and receive evaluation results based on real user feedback. Involving real users in evaluation can solve problems mentioned above: we can consider the diversity of search intents and relevance criteria by utilizing real queries and feedback from users who are engaged in real search tasks.

Our realistic evaluation framework would bring in novel challenges for participants and insights into the gap between evaluation in laboratory settings and that in production environments. More specifically, we expect that (1) participants can propose methods to consider different types of intents behind a query, and to diversify search results so that they can satisfy as many search intents as possible; (2) participants can address the problem of diverse relevance criteria by utilizing several properties of questions; and (3) participants can evaluate their systems with real users in Yahoo! Chiebukuro.

The remainder of the paper is organized as follows. Section 2 describes the OpenLiveQ task in details. Section 3 introduces the data distributed to OpenLiveQ participants. Section 4 explains the evaluation methodology applied to the OpenLiveQ task.

2. TASK

The task of the OpenLiveQ task is simply defined as follows: given a query and a set of questions with their answers, return a ranked list of questions. Our task consists of three phases:

1. **Offline Training Phase** Participants are given *training data* including a list of queries, a set of questions for each query, and clickthrough data (see Section 3 for details). They can develop and tune their question retrieval systems based on the training data.

2. **Offline Test Phase** Participants are given only a list of queries and a set of questions for each query. They are required to submit a ranked list of questions for each query by a deadline. We evaluate submitted results by using graded relevance for each question, and decide which question retrieval systems can be evaluated in the online test phase.

[1] http://chiebukuro.yahoo.co.jp/

3. **Online Test Phase** Selected question retrieval systems are evaluated in a production environment of Yahoo Japan Corporation. *Multileaved comparison* methods [4] are used in the online evaluation.

As the open live test will be conducted on a Japanese service, the language scope is limited to Japanese. Meanwhile, we supported participants by providing a tool for feature extraction so that Japanese NLP is not required for participation.

3. DATA

This section explains the data used in the OpenLiveQ task.

3.1 Queries

We sampled 2,000 queries from a Yahoo! Chiebukuro search query log, and used 1,000 queries for training and the rest for testing. Before sampling the queries from the query log, we applied the several filtering rules to remove queries that were not desirable for the OpenLiveQ task.

First, we filtered out *time-sensitive* queries. Participants were given a fixed set of questions for each query, and were requested to submit a ranked list of those questions. Since the *frozen* set of questions were presented to real users during the online evaluation phase, it was not desirable that the relevance of each question changes depending on the time. Thus, we filtered out queries that were highly time-sensitive, and only used the time-insensitive queries in the OpenLiveQ task.

The procedure how we filtered out the time-sensitive queries is as follows. Letting n^q_{recent} be the number of questions for query q posted from July 16th, 2017 to September 16th, 2017, and n^q_{past} be the number of questions of q from January 1st, 2013 to July 15th, 2017, we removed queries such that $n^q_{\text{recent}}/n^q_{\text{past}} > 1.0$ as the time-sensitive queries.

Second, we filtered out porn queries. We judged a query was a porn query if more than 10% questions retrieved by the query belonged to the porn-related category of Yahoo! Chiebukuro.

After removing time-sensitive and porn-related queries, we further manually removed queries that were related to any of the ethic, discrimination, or privacy issues. The organizers checked each of the queries and its questions, and filtered out a query if at least one of the organizers judged it had the issues above.

Finally, we sampled 2,000 queries from the remaining queries and used them in the OpenLiveQ task.

3.2 Questions

We input each query to the current Yahoo! Chiebukuro search system as of December 1-9 in 2016, recorded the top 1,000 questions, and used them as questions to be ranked. Information about all the questions as of December 1-9, 2016 was distributed to the OpenLiveQ participants, and includes

- Query ID (a query by which the question was retrieved),
- Rank of the question in a Yahoo! Chiebukuro search result for the query of Query ID,
- Question ID,
- Title of the question,
- Snippet of the question in a search result,
- Status of the question (accepting answers, accepting votes, or solved),
- Last update time of the question,
- Number of answers for the question,
- Page view of the question,
- Category of the question,
- Body of the question, and
- Body of the best answer for the question.

The total number of questions is 1,967,274. As was mentioned earlier, participants were required to submit a ranked list of those questions for each test query.

3.3 Clickthrough Data

Clickthrough data are available for some of the questions. Based on the clickthrough data, one can estimate the click probability of the questions, and understand what kinds of users click on a certain question. The clickthrough data were collected from August 24, 2016 to November 23, 2016.

The clickthrough data include

- Query ID (a query by which the question was retrieved),
- Question ID,
- Most frequent rank of the question in a Yahoo! Chiebukuro search result for the query of Query ID,
- Clickthrough rate,
- Fraction of male users among those who clicked on the question,
- Fraction of female users among those clicked on the question,
- Fraction of users under 10 years old among those who clicked on the question,
- Fraction of users in their 10s among those who clicked on the question,
- Fraction of users in their 20s among those who clicked on the question,
- Fraction of users in their 30s among those who clicked on the question,
- Fraction of users in their 40s among those who clicked on the question,
- Fraction of users in their 50s among those who clicked on the question, and
- Fraction of users over 60 years old among those who clicked on the question.

The clickthrough data contain click statistics of a question identified by Question ID when a query identified by Query ID was submitted. The rank of the question can change even for the same query. This is why the third value indicates the most frequent rank of the question. The number of query-question pairs in the clickthrough data is 440,163.

Figure 1: Screenshot of the relevance judgment system.

4. EVALUATION

This section introduces the offline evaluation, in which runs were evaluated with relevance judgment data, and online evaluation, in which runs were evaluated with real users by means of multileaving.

4.1 Offline Evaluation

Offline test is carried out before online test explained later, and determines participants whose systems are evaluated in the online test, based on results in the offline test. The offline evaluation was conducted in a similar way to traditional ad-hoc retrieval tasks, in which results are evaluated by relevance judgment results and evaluation metrics such as nDCG (normalized discounted cumulative gain), ERR (expected reciprocal rank), and Q-measure. During the offline test period, participants can submit their results once per day through our Web site[2], and obtain evaluation results right after the submission.

To simulate the online test in the offline test, we conducted relevance judgment with an instruction shown below: "Suppose you input <query> and received a set of questions as shown below. Please select all the questions on which you want to click". Assessors were not present with the full content of each question, and requested to evaluate questions in a similar page to the real SERP in Yahoo! Chiebukuro. This type of relevance judgment is different from traditional ones, and expected to result in being similar to results of the online test. Five assessors were assigned to each query, and the relevance grade of each question was estimated as the number of assessors who select the question in the relevance judgment. For example, the relevance grade was 2 if two out of five assessors marked a question. We used Lancers[3], a Japanese crowd-sourcing service, for the relevance judgment. Figure 1 shows a screenshot of a system used for the relevance judgment, where assessors can click on either the title or the blue square for voting.

Only a score of nDCG@10 for each submitted run was displayed at our website. The top 10 teams in terms of nDCG@10 were invited to the online evaluation.

[2]http://www.openliveq.net/
[3]http://www.lancers.jp/

Algorithm 1: Optimized Multileaving (OM)

Require: Input rankings $\mathcal{I}$, number of output rankings m, and number of items in each output ranking l

$\mathcal{O} \leftarrow \{\}$;
for $k = 1, \ldots, m$ **do**
  for $i = 1, \ldots, l$ **do**
    Select j randomly;
    $r = 1$;
    while $I_{j,r} \in O_k$ **do**
      $r \leftarrow r + 1$
    end
    if $r \leq |I_j|$ **then**
      $O_{k,i} = I_{j,r}$;
    end
  end
  $\mathcal{O} \leftarrow \mathcal{O} \cup \{O_k\}$;
end
return $\mathcal{O}$;

4.2 Online Evaluation

Submitted results were evaluated by multileaving [4]. Ranked lists of questions were combined into a single SERP, presented to real users during the online test period, and evaluated on the basis of clicks observed. Based on our preliminary experiment [2], we opt to use *optimized multileaving* (OM) in the multileaved comparison. Results submitted in the offline test period were used in as-is in the online test. Note that some questions could be excluded in the online test if they were deleted for some reasons before or during the online test.

A multileaving method takes a set of rankings $\mathcal{I} = \{I_1, I_2, \ldots, I_n\}$ and returns a set of combined rankings $\mathcal{O} = \{O_1, O_2, \ldots, O_m\}$, where each combined ranking O_k consists of l items. The i-th items of an input ranking I_j and an output ranking O_k are denoted by $I_{j,i}$ and $O_{k,i}$, respectively. When a user issues a query, we return an output ranking O_k with probability p_k and observe user clicks on O_k. If $O_{k,i}$ is clicked by the user, we give a credit $\delta(O_{k,i}, I_j)$ to each input ranking I_j. Each multileaving method consists of a way to construct $\mathcal{O}$ from $\mathcal{I}$, probability p_k for each output ranking O_k, and a credit function δ. The original multileaving methods decide which input ranking is better for each input ranking pair every time it presents an output ranking, whereas we opt to accumulate the credits through all the presentations and to measure the effectiveness of each input ranking on the basis of the sum of the credits, mainly because this approach must provide more informative evaluation results.

OM [4] is a multileaving method that generates output rankings by Algorithm 1, and computes the presentation probability p_k that maximizes the *sensitivity* of the output rankings while ensuring no *bias*. The sensitivity of an output ranking is the power to discriminate effectiveness differences between input rankings when the output ranking is being presented to users. Intuitively, the sensitivity is high if random clicks on an output ranking give a similar amount of credits to each input ranking. High sensitivity is desirable as it leads to fast convergence of evaluation results. Bias of output rankings measures the difference between the ex-

pected credits of input rankings for random clicks. If the bias is high, a certain input ranking can be considered better than the others even if only random clicks are given. Thus, multileaving methods should reduce the bias as much as possible.

The sensitivity can be maximized by minimizing *insensitivity*, defined as the variance of credits given by an output ranking O_k through rank-dependent random clicks [4]:

$$\sigma_k^2 = \sum_{j=1}^{n} \left(\left(\sum_{i=1}^{l} f(i)\delta(O_{k,i}, I_j) \right) - \mu_k \right)^2, \qquad (1)$$

where $f(i)$ is the probability with which a user clicks on the i-th item. We follow the original work [4] and use $f(i) = 1/i$ and $\delta(O_{k,i}, I_j) = 1/i$ if $O_{k,i} \in I_j$; otherwise, $1/(|I_j|+1)$. The mean credit μ_k of the output ranking O_k is computed as: $\mu_k = (1/n) \sum_{j=1}^{n} \sum_{i=1}^{l} f(i)\delta(O_{k,i}, I_j)$. Since each output ranking O_k is presented to users with probability p_k, OM should minimize the expected insensitivity: $\mathbb{E}[\sigma_k^2] = \sum_{k=1}^{m} p_k \sigma_k^2$.

The bias of output rankings $\mathcal{O}$ is defined as the difference between expected credits of input rankings. The expected credit of an input ranking I_j given rank-independent random clicks on top-1, top-2, $\cdots$, and top-r items is defined as follows:

$$\mathbb{E}[g(I_j, r)] = \sum_{k=1}^{m} p_k \sum_{i=1}^{r} \delta(O_{k,i}, I_j). \qquad (2)$$

If the expected credits of input rankings are different, evaluation results obtained by multileaving is biased. Thus, the original version of OM imposes a constraint that the expected credits of all the input rankings must be the same, *i.e.* $(\forall r, \exists c_r, \forall j)\ \mathbb{E}[g(I_j, r)] = c_r$.

According to the paper by Schuth et al. [4] and their publicly available implementation[4], their version of OM tries first to satisfy the constraint for letting the bias be zero, and then to maximize the sensitivity given that the bias constraint is satisfied. However, we found that the bias constraint cannot be satisfied for more than 90% of the cases in our experiment, *i.e.* we could not find any solution that satisfied the bias constraint. Hence, we propose using a more practical implementation of OM that minimizes a linear combination of the sum of biases and the insensitivity as follows:

$$\min_{p_k} \alpha \sum_{r=1}^{l} \lambda_r + \sum_{k=1}^{m} p_k \sigma_k^2 \qquad (3)$$

$$\text{subject to } \sum_{k=1}^{m} p_k = 1,\ 0 \le p_k \le 1\ (k = 1, \ldots, m), \text{ and}$$

$$\forall r, \forall j, j', -\lambda_r \le \mathbb{E}[g(I_j, r)] - \mathbb{E}[g(I_{j'}, r)] \le \lambda_r\ ,$$

where α is a hyper-parameter that controls the balance between the bias and insensitivity, and λ_r is the maximum difference of the expected credits in any input rankings pairs. If λ_r is close to zero, the expected credits of input rankings are close, and accordingly, the bias becomes small. Our implementation is publicly available[5].

The online evaluation at Yahoo! Chiebukuro was conducted between May 9, 2017 and August 8, 2017. The total number of impressions used for the online evaluation is 410,812.

[4]https://github.com/djoerd/mirex
[5]https://github.com/mpkato/interleaving

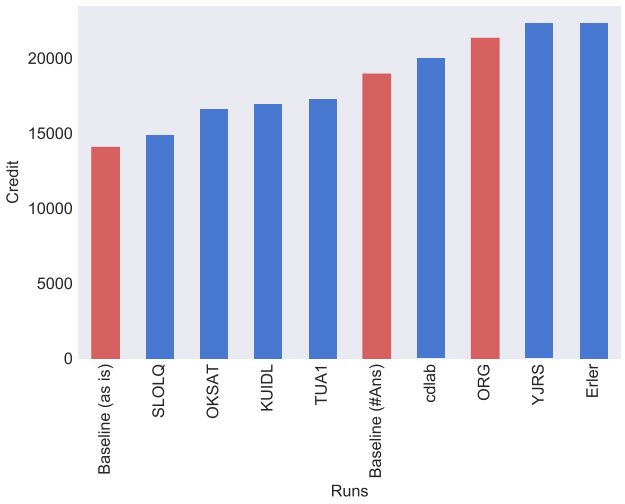


Figure 3: Cumulated credits in the online evaluation.

5. EVALUATION RESULTS

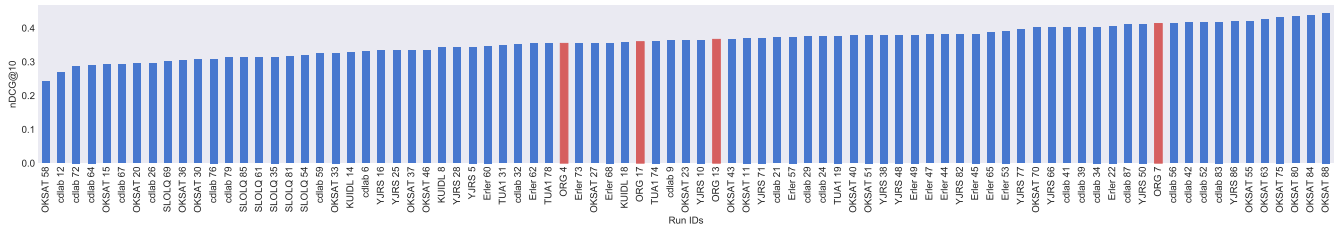
Figures 2(a), 2(b), and 2(c) show results of the offline evaluation in terms of nDCG@10, ERR@10, and Q-measure. Baseline runs are indicated in red. The first baseline (ORG 4) produced exactly the same ranked list as that used in the production. The other baselines used a linear feature-based model optimized by coordinate ascent [3], with different parameter settings. One of the linear feature-based model (ORG 7) was the best baseline method in terms of all the metrics. The best baseline was outperformed by some teams: OKSAT, YJRS, and cdlab in terms of nDCG@10 and ERR, and YJRS and Erler in terms of Q-measure.

Figure 5 shows cumulated credits in the online evaluation. In the online evaluation, the best run from each team was selected: KUIDL 18, TUA1 19, Erler 22, SLOLQ 54, cdlab 83, YJRS 86, and OKSAT 88. Moreover, we included the best baseline method (ORG 7), current production system (ORG 4), and a simple baseline that ranks questions by the number of answers obtained (not used in the offline evaluation). YJRS and Erler outperformed the best baseline (ORG), while there is not a statistically significant difference between them as explained later. The online evaluation showed a different result from those obtained in the offline evaluation. In particular, OKSAT performed well at the online evaluation, while it showed relatively low performance among the submitted runs.

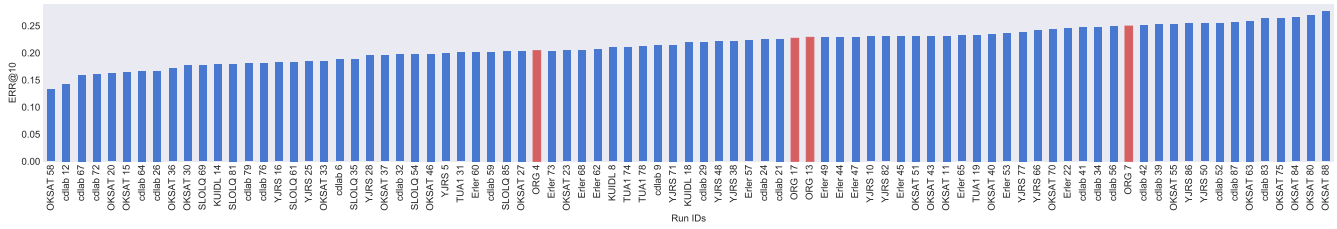
Figure 5 shows the number of run pairs between which t-tests did not find significant differences, where the x-axis indicates the number of days passed. As this is multiple comparison, p-values were corrected by Bonferroni's method. The multileaving evaluation could find statistically significant differences for most of the run pairs ($37/45 = 82.2\%$) within 10 days. After 20 days passed, significant differences were found for $41/45 = 91.1\%$ of run pairs. After 64 days passed, only three run pairs remained insignificant: KUIDL-TUA1, KUIDL-OKSAT, and YJRS-Erler.

6. CONCLUSIONS

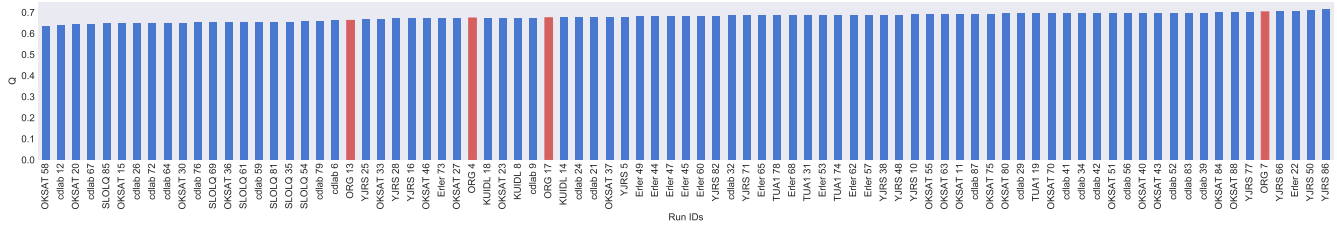
In this paper, we described the details of the NTCIR-13



(a) nDCG@10



(b) ERR@10



(c) Q-measure

Figure 2: Offline evaluation.

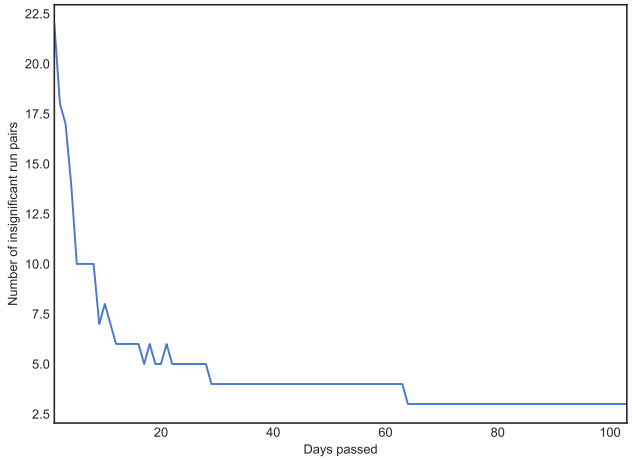


Figure 4: Number of insignificant run pairs as a function of days passed.

OpenLiveQ task, and reported official results of the submitted runs. Out findings are summarized as follows:

- There was a big difference between the offline evaluation where assessors voted for relevant questions and online evaluation where real users voted for relevant questions via clicks.
- The current online evaluation method, *multileaving*, could seemingly handle a dozen of runs, while it could not evaluate a hundred of runs within a few months.
- Systems developed by participants could achieve big improvements from the current question retrieval system in terms of the online evaluation, while there is room for improvement over a strong baseline using learning to rank.

Those findings motivated us to organize the second round of OpenLiveQ (OpenLiveQ-2) in NTCIR-14, in which we will bring the following changes:

- We will employ a log-based offline evaluation method that turned out to be line up with online evaluation results according to our recent study [2].
- We will improve multileaving methods to evaluate a hundred of runs within a few months.

7. ACKNOWLEDGMENTS

We thank the NTCIR-13 OpenLiveQ participants for their effort in submitting runs. We appreciate significant efforts made by Yahoo Japan Corporation for providing valuable search data and an open live test environment.

8. REFERENCES

[1] X. Cao, G. Cong, B. Cui, and C. S. Jensen. A generalized framework of exploring category

information for question retrieval in community question answer archives. In *WWW*, pages 201–210, 2010.

[2] T. Manabe, A. Nishida, M. P. Kato, T. Yamamoto, and S. Fujita. A comparative live evaluation of multileaving methods on a commercial cqa search. In *SIGIR 2017*, pages 949–952, 2017.

[3] D. Metzler and W. B. Croft. Linear feature-based models for information retrieval. *Information Retrieval*, 10(3):257–274, 2007.

[4] A. Schuth, F. Sietsma, S. Whiteson, D. Lefortier, and M. de Rijke. Multileaved comparisons for fast online evaluation. In *CIKM*, pages 71–80, 2014.

[5] K. Wang, Z. Ming, and T.-S. Chua. A syntactic tree matching approach to finding similar questions in community-based qa services. In *SIGIR*, pages 187–194, 2009.

[6] G. Zhou, Y. Liu, F. Liu, D. Zeng, and J. Zhao. Improving question retrieval in community question answering using world knowledge. In *IJCAI*, pages 2239–2245, 2013.