YJRS at the NTCIR-14 OpenLiveQ-2 Task

Tomohiro Manabe, Sumio Fujita, and Akiomi Nishida

Yahoo Japan Corporation
Chiyoda, Tokyo 102-8282 Japan
{tomanabe, sufujita, anishida}@yahoo-corp.jp

Abstract. We report our work at the NTCIR-14 OpenLiveQ-2 task. From the given data set for question retrieval on a community QA service, we extracted some BM25F-like features and translation-based features in addition to basic features. After that, we constructed multiple ranking models with the data. According to the offline evaluation results, our linear combination model with translated features achieved the best score on Q-measure among our runs. At the first round of online evaluation, our linear models with BM25F-like features and translation-based features obtained the largest credits among 62 runs including other teams' runs. At the final round, our linear combination model with BM25F-like features and neural ranking models with basic features obtained the largest amount of credits among 30 runs which passed the first round. According to the online evaluation results based on real users' feedback, neural ranking is one of the best approaches to improve practical search effectiveness on the service.

Team Name. YJRS

Subtasks. OpenLiveQ-2

Keywords: Probabilistic retrieval model · Linear combination model · Neural ranking model · QA search and retrieval · Document scoring.

1 Introduction

We report our work at the NTCIR-14 OpenLiveQ-2 task. For detailed description of the task and evaluation results, please refer to the overview paper [7].

The task organizers provided us with a data set and a tool[1] for basic feature extraction from the data. The tool's README file includes a short instruction for generating a simple linear combination of the features by using the RankLib implementation[2] of the Coordinate Ascent (CA) method [11].

We tried to add some features to the basic features. They are namely (1) BM25F-like features of the existing fields and (2) features of best answer texts translated with a translation model from answer texts to question texts.

[1] https://github.com/mpkato/openliveq
[2] https://sourceforge.net/p/lemur/wiki/RankLib/

Table 1. List of YJRS runs and offline evaluation results of them.

ID	Description	Q-measure	nDCG@10	ERR@10
91	`baseline 77 features`	0.39124	0.12667	0.08849
92	`YJRS-86 80 features`	0.45609	0.24802	0.15548
93	`baseline + A -> Q translated 94 features`	0.46387	0.25926	0.16244
95	`baseline 77 features (retry)`	0.39559	0.08976	0.06469
100	`ListNet 77 features`	0.37340	0.06296	0.03971
113	`ListNet 77 features 5cv`	0.37240	0.06660	0.04225
136	`YJRS-86 + A -> Q translated 94 features`	0.38514	0.10256	0.07252
144	`GBDT 77 features`	0.37228	0.09128	0.05689
148	`GBDT 77 features (tuned)`	0.37429	0.06710	0.04141

Moreover, we also tried to replace the baseline linear model and the CA method with other sophisticated models and methods, namely (1) a neural ranking model generated with ListNet [2] and (2) Gradient Boosting Decision Trees (GBDT) generated with LightGBM [8].

Our runs obtained the largest amount of credits in both of the first and second online evaluation rounds on Yahoo! *Chiebukuro*[3], a community QA service.

2 Our Approach

Table 1 lists all our runs in temporal order of submission. Their offline evaluation results are also listed. In this section, we explain our runs in this order.

2.1 Baseline Linear Combination Model

First we generated and submitted a baseline run with a linear combination model of 77 basic features generated by following the README file (Run ID: 91).

The 68 among the 77 features are composed of 17 feature types (TF, IDF, ICF[4], TFIDF, TFICF, BM25, language models with three smoothing methods, document length, and their logarithmic and/or normalized variations), most of which are common to the well-known LETOR data set [14], extracted from each of 4 textual fields (question title, question snippets, question text, and best answer text). The other nine features are answer count, view count, their logarithmic variations, rank at the baseline ranking, timestamp, and three 0/1 flags (open to answer, open to vote for best answer, and solved). We actually calculated feature values with our original Solr[5] plug-in instead of the official tool because of its expandability. This is the only difference between the official instruction on the README file and our baseline model generation process.

We used the RankLib implementation of CA [11] for learning a linear combination model from the training data composed of 1,000 queries and 986,125

[3] https://chiebukuro.yahoo.co.jp/

[4] |documents in collection|/|keyword occurrences in collection|

[5] https://lucene.apache.org/solr/

questions in total. It optimizes each parameter one-by-one. To optimize a parameter, it examines some smaller and larger points from the current value and greedily adopts a new value that improves the objective function. After optimizing the last parameter, it shuffles and iterates over all the parameters again. The optimization finishes if modification of no parameter can improve the value of the objective function. As its objective function, we used default ERR@10 [3]. As relevance judgment between a query and questions, we naively normalized given CTRs. For the normalization, we divided the CTRs by the max for the query, multiplied them by 4 (max relevance grade), then truncated them to integers.

2.2 Extended BM25F Features

In the last NTCIR-13 OpenLiveQ task [6], we proposed an extension of the baseline method [10]. Because the method achieved the largest amount of credits in the last task, we also generated and submitted a run for this task with the method (Run ID: 92). The differences from the baseline are as follows:

- In addition to the 77 basic features, we use three BM25F [15] features extended for handling numeric document fields as well as textual fields. They are based on three different field weighting strategies.
- We use nDCG@10 as the objective function of CA.
- We perform 5-fold cross validation on the training data.

In the last task, we assigned negative BM25F weights to some fields, however, we instead used zero in this task because of harmful effect of negative weights.

2.3 Translation Features

Another approach which achieved one of the largest credits in the last task is based on a translation model [4]. Its key idea is to adopt different language models behind questions (or queries) from answers. Based on this idea, we translated best answer texts into model-generated question texts and extracted 17 numeric features explained in Section 2.1 from the resulting text. The translation was based on a translation model from answer texts to question texts constructed with the GIZA++ toolkit [12] and publicly available Yahoo! *Chiebukuro* corpus[6]. The translation model is a set of correspondences between one answer term and multiple question terms with their translation probabilities, e.g., fruit $\rightarrow$ apple (50%), banana (30%), orange (20%). To translate a answer text into a single (not probabilistic) question text, we iterated over the answer text cumulating the probabilities for each question term as the term's score, sorted the terms by score, then extracted top-l terms where l is the number of term occurrences in the answer text. We generated a run by linearly combining the 94 features in total by the baseline method and submitted it (Run ID: 93).

Because the optimization in the baseline method is a probabilistic process and the nDCG@10 score of the baseline run (Run ID: 91) was relatively worse than the last task [10], we attempted to generate the baseline run again (Run ID: 95), however, that did not improve nDCG@10 score.

[6] https://www.nii.ac.jp/dsc/idr/yahoo/chiebkr2/Y_chiebukuro.html

2.4 Neural Ranking Model

Neural ranking is one of the state-of-the-art approaches to document scoring for retrieval. Among a wide variety of neural ranking methods, we used ListNet [2] for combining the baseline 77 features (Run ID: 100). The key concept of ListNet is its list-wise loss function. As the ranking model, we used a simple three-layer fully-connected feed-forward neural network whose size of hidden layer is 200. Before inputting feature values, we normalized them into [0, 1] with simple min-max normalization. We used Chainer[7] and its implementation of Adam optimizer [9] (initial learning rate is 0.0007 and learning rate decay factor is 0.995) as our neural ranking framework. We set 512 to batch size, 1000 to number of iterations, and 0.0005 to the weight decay factor. Further tuning of hyper-parameters with grid search did not improve offline evaluation results significantly.

We also tried to apply 5-fold cross validation to this approach (Run ID: 113).

2.5 Combining Extended BM25F and Translation Features

We can independently apply the modifications explained in Section 2.2 and 2.3. We also generated a run which incorporates both modifications (Run ID: 136).

2.6 Ensemble Tree Model

The GBDT is another state-of-the-art approach to document scoring for retrieval. We used the LightGBM implementation [8] of GBDT and the nDCG-based LambdaRank objective function [1] for generating a run by combining the baseline 77 features (Run ID: 144). The model of GBDT is ensemble trees, i.e., linear combination of regression trees. The GBDT generates trees one-by-one for minimizing errors on already generated trees with gradient descent. We learned an ensemble of 100 trees including 15 leaves each with a learning rate of 0.1. The LightGBM implementation supports other techniques, e.g., feature binning, bagging, pruning, and so on. We set 255 to max number of bins, 0.9 to bagging fraction, 50 to minimum number of data points per leaf, and 5.0 to minimum summation of hessians per leaf.

We also tried to improve its effectiveness by a simple grid search over its hyper-parameter space, however, that did not improved the score (Run ID: 148).

3 Offline Test Results

In Table 1, we listed the offline evaluation results of all our runs on three evaluation measures.

Overall, the offline evaluation results were not stable. For example, the nDCG@10 and ERR@10 scores of runs 91 and 95 were quite different although they are generated with exactly the same (but probabilistic) method.

[7] https://chainer.org/

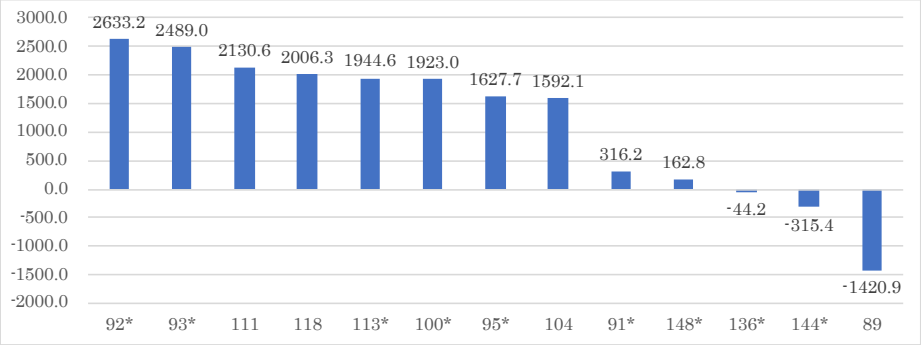

Fig. 1. First-round online test results of YJRS runs (marked with *) and other teams' best runs. X-axis is run IDs and y-axis is credit. Runs are in descending order of credit.

In contrast, the Q-measure produced relatively stable scores through the offline evaluation period. This fact supports the validity of selection of Q-measure as the primary evaluation measure of this offline test. According to this measure, BM25F-based (Run ID: 92) and translation-based (Run ID: 93) modifications to the baseline linear model were significantly effective among our runs. There was no significant difference among the other runs.

Because none of our runs were duplicated, all our runs were also evaluated on the next online test round. In the next section, we discuss on its results.

4 Online Test Results

In this task, the online test period consists of two rounds of multileaved comparisons [5] with Pairwise Preference Multileaving [13] on Yahoo! *Chiebukuro*. The key idea of multileaved comparison is as follows: When a user submits a test query to the search system, it looks up corresponding rankings from runs under the comparison. After that, it interleaves the rankings into a mixed ranking then returns it to the user. If the user clicks an item on the mixed ranking, the run from which the item is extracted obtains some credit. We treat the two rounds as independent experiments, i.e., we do not sum up the credits because it is difficult to assign reasonable weights to each of the rounds.

4.1 First Round

Figure 1 illustrates the credits of YJRS runs (marked with *) and the other teams' best runs in the first round of the online evaluation. In total 61 runs were compared to one another in this round however we omitted other 48 runs due to space limitations.

Table 2 counts page views in the first round of the online evaluation when the run arranged in the row obtained a larger credit than the run arranged in the column. YJRS runs are marked with *.

Table 2. First-round online test results of YJRS runs (marked with *) and other teams' best runs. Each element is count of page views where run arranged in row obtained larger credit than run arranged in column.

	92*	93*	111	118	113*	100*	95*	104	91*	148*	136*	144*	89
92*	-	2873	6092	4612	5792	5771	4178	6166	6966	6973	7306	7038	7509
93*	3021	-	5856	5084	5706	5693	4226	5912	6809	6849	7176	6933	7413
111	5716	5307	-	4440	5750	5733	5532	4757	6857	6902	7246	6934	7443
118	4385	4716	4676	-	5753	5695	5162	5570	6956	6980	7266	7072	7425
113*	5523	5319	5881	5744	-	489	4726	6303	4554	4432	5102	5044	6116
100*	5503	5322	5851	5701	492	-	4730	6311	4631	4494	5173	5068	6128
95*	3926	3813	5650	5087	4701	4714	-	5919	5827	6027	6492	6201	6839
104	5856	5520	4827	5371	6312	6289	5959	-	7126	7136	7436	7147	7435
91*	5790	5552	6066	6034	3406	3482	4876	6220	-	2820	2291	3101	3975
148*	5838	5647	6175	6075	3319	3382	5071	6261	2935	-	2891	1418	4000
136*	5898	5663	6167	6100	3716	3778	5258	6261	2001	2461	-	2578	2793
144*	5809	5566	6109	6038	3876	3912	5118	6143	3076	1315	2885	-	3875
89	5774	5585	6066	5938	4431	4475	5265	5964	3301	3222	2430	3235	-

According to Figure 1 and Table 2, our BM25F-based (Run ID: 92) and translation-based (Run ID: 93) variations of the baseline method consistently obtained the largest credits among all the runs. This observation is also consistent with the offline evaluation results. Between these two runs, the BM25F-based run obtained a slightly larger amount of credits in total while the translation-based run obtained larger credits than the BM25F-based in more page views.

According to Figure 1, among our runs, the ListNet with 5-fold cross validation (Run ID: 113) and simple ListNet (Run ID: 100) runs followed the top two runs. Between these runs, there was no significant difference. This indicates that an optimization of neural ranking models by a cross validation was not needed in our experiments. Our learning process generates models with stable performance, or Run 100 was as well-trained as Run 113 by chance.

Our baseline runs (Run IDs: 95 and 91) occupied the fifth and sixth positions among our runs, however, the credits they obtained were significantly different. We consider that this is because of unstableness of CA we used for generating the linear combination models. This hypothesis also explains a broad range of credits our runs with linear combination models (Run IDs: 92, 93, 95, 91, and 136) obtained.

Our GBDT runs (Run IDs: 148 and 144) did not perform as good as other best runs explained above.

4.2 Second Round

As same as the first round, Figure 2 illustrates the credits in total and Table 3 counts page views.

In this round, 30 runs were compared to one another. The 30 runs were selected mainly based on credit obtained in the first round. As indicated in this

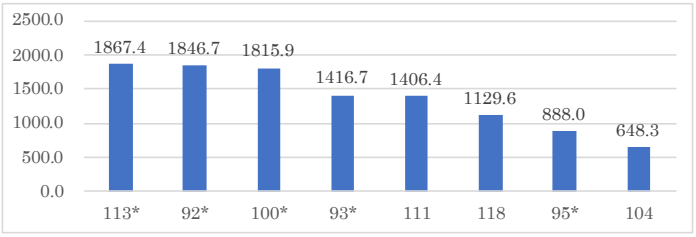

Fig. 2. Second-round online test results of YJRS runs (marked with *) and other teams' best runs. X-axis is run IDs and y-axis is credit. Runs are in descending order of credit.

Table 3. Second-round online test results of YJRS runs (marked with *) and other teams' best runs. Each element is count of page views where run arranged in row obtained larger credit than run arranged in column.

	113*	92*	100*	93*	111	118	95*	104
113*	-	6433	440	6231	6662	6700	5475	7114
92*	6131	-	6093	3449	6864	5502	4703	6963
100*	451	6426	-	6235	6674	6694	5478	7127
93*	5919	3612	5923	-	6506	5917	4732	6650
111	6036	6630	6026	6194	-	5386	6167	5313
118	6100	5242	6061	5547	5381	-	5867	6247
95*	5012	4658	5023	4561	6362	6006	-	6693
104	6656	6705	6653	6404	5277	6223	6550	-

figure and table, among our runs, ListNet ones (Run IDs: 113 and 100), the last task's best one (Run ID: 92), translation-based one (Run ID: 93), and baseline one (Run ID: 95) passed the first round. Because each run have more chance to obtain credit in this round than in the first round, we consider the results of this round is more reliable than the first round.

Comparing the evaluation results of the second round with those of the first round, our ListNet runs (Run IDs: 100 and 113) occupied better positions. We consider that this is because of more accurate comparison in this round. Between these two runs, there was no significant difference of credits or win-lose page view counts. This fact again indicates the stable performance of our learning process of neural ranking models.

Other tendencies were almost the same as the first round. This fact supports the idea of two-round multileaved comparison. More precisely, the order among Run IDs 92, 93, 111, 118, 95, and 104 were same as the first round. However, we observed a quite different scale of credit amounts between the first and second rounds. For example, Run 93 obtained only 77% of the credit which Run 92 obtained in the second round after 95% in the first round. Run 104 obtained only 73% of the credit of Run 95 in the second round after 98% in the first round.

5 Conclusions

We explained the outline of our work at the NTCIR-14 OpenLiveQ-2 task. Three of our runs obtained the largest amounts of credit at the second round of online evaluation. For generating one of them, we extracted three BM25F-like features in addition to 77 basic features from the corpus then constructed linear combination models on the data with CA [11]. In this construction, we used nDCG@10 as the objective function and performed 5-fold cross validation of the models. For generating the other two, we extracted 77 basic features from the corpus then constructed neural ranking models on the data with ListNet [2]. The evaluation results also suggested that our CA-based learning process of linear combination models is less stable than our ListNet-based learning process of neural ranking models. Considering this concern, using our neural ranking models must be a good idea for improving the effectiveness of Yahoo! *Chiebukuro* Search.

References

1. Burges, C.J., Ragno, R., Le, Q.V.: Learning to rank with nonsmooth cost functions. In: Schölkopf, B., Platt, J.C., Hoffman, T. (eds.) Advances in Neural Information Processing Systems 19, pp. 193–200. MIT Press (2007)
2. Cao, Z., Qin, T., Liu, T.Y., Tsai, M.F., Li, H.: Learning to rank: From pairwise approach to listwise approach. In: Proceedings of the 24th International Conference on Machine Learning. pp. 129–136. ICML '07, ACM, New York, NY, USA (2007)
3. Chapelle, O., Metlzer, D., Zhang, Y., Grinspan, P.: Expected reciprocal rank for graded relevance. In: Proceedings of the 18th ACM Conference on Information and Knowledge Management. pp. 621–630. CIKM '09, ACM, New York, NY, USA (2009)
4. Chen, M., Li, L., Sun, Y., Zhang, J.: Erler at the NTCIR-13 OpenLiveQ task. In: Proceedings of the 13th NTCIR Conference on Evaluation of Information Access Technologies (2017)
5. Kato, M.P., Manabe, T., Fujita, S., Nishida, A., Yamamoto, T.: Challenges of multileaved comparison in practice: Lessons from NTCIR-13 OpenLiveQ task. In: Proceedings of the 27th ACM International Conference on Information and Knowledge Management. pp. 1515–1518. CIKM '18, ACM, New York, NY, USA (2018)
6. Kato, M.P., Yamamoto, T., Manabe, T., Nishida, A., Fujita, S.: Overview of the NTCIR-13 OpenLiveQ task. In: Proceedings of the 13th NTCIR Conference on Evaluation of Information Access Technologies (2017)
7. Kato, M.P., Yamamoto, T., Manabe, T., Nishida, A., Fujita, S.: Overview of the NTCIR-14 OpenLiveQ-2 task. In: Proceedings of the 14th NTCIR Conference on Evaluation of Information Access Technologies (2019)
8. Ke, G., Meng, Q., Finley, T., Wang, T., Chen, W., Ma, W., Ye, Q., Liu, T.Y.: LightGBM: A highly efficient gradient boosting decision tree. In: Guyon, I., Luxburg, U.V., Bengio, S., Wallach, H., Fergus, R., Vishwanathan, S., Garnett, R. (eds.) Advances in Neural Information Processing Systems 30, pp. 3146–3154. Curran Associates, Inc. (2017)
9. Kingma, D.P., Ba, J.: Adam: A method for stochastic optimization. CoRR **abs/1412.6980** (2014)

10. Manabe, T., Nishida, A., Fujita, S.: YJRS at the NTCIR-13 OpenLiveQ task. In: Proceedings of the 13th NTCIR Conference on Evaluation of Information Access Technologies (2017)
11. Metzler, D., Bruce Croft, W.: Linear feature-based models for information retrieval. Inf. Retr. **10**(3), 257–274 (Jun 2007)
12. Och, F.J., Ney, H.: Improved statistical alignment models. In: Proceedings of the 38th Annual Meeting on Association for Computational Linguistics. pp. 440–447. ACL '00, Association for Computational Linguistics, Stroudsburg, PA, USA (2000)
13. Oosterhuis, H., de Rijke, M.: Sensitive and scalable online evaluation with theoretical guarantees. In: Proceedings of the 2017 ACM on Conference on Information and Knowledge Management. pp. 77–86. CIKM '17, ACM, New York, NY, USA (2017)
14. Qin, T., Liu, T.Y., Xu, J., Li, H.: LETOR: A benchmark collection for research on learning to rank for information retrieval. Inf. Retr. **13**(4), 346–374 (Aug 2010)
15. Robertson, S., Zaragoza, H., Taylor, M.: Simple BM25 extension to multiple weighted fields. In: Proceedings of the Thirteenth ACM International Conference on Information and Knowledge Management. pp. 42–49. CIKM '04, ACM, New York, NY, USA (2004)