Leveraging Procedural Knowledge for Task Understanding: OAQA at TREC 2015 Tasks Track

Zi Yang and Eric Nyberg

Language Technologies Institute, School of Computer Science, Carnegie Mellon University
5000 Forbes Avenue, Pittsburgh, PA 15213, USA
{ziy,ehn}@cs.cmu.edu

1 Introduction

CMU OAQA team participated in the Task Understanding subtask in the Tasks track, and submitted three runs `rsf`, `lsf`, and `lsfs` for official evaluation. In our earlier work [1], we proposed to leverage procedural knowledge (e.g. wikiHow or eHow) for task-oriented search problems including query suggestion, which is similar to the goal of this Task Understanding task. Therefore, we reused the same system[1], the wikiHow corpus[2], and the pretrained model to generate key phrases for the 50 sample queries. Since our approach entirely relies on a procedural knowledge base, the `freebaseID` fields in the given input file were not used in our system, although ambiguity might become an issue in some cases.

We intend to take the advantage of the TREC evaluation to test if our hypotheses that were stated in our earlier work still hold:

- The coverage of the current open procedural knowledge bases such as wikiHow can satisfy our daily task-oriented information needs.
- The subtasks that are suggested by the procedural knowledge base are comprehensive for the task accomplishment.
- The proposed approach can accurately identify the relevant task from the procedural knowledge base, and subsequently generate and rank the key phrases, according to the specific task-oriented task requirement.

The working note first describes the system and necessary changes from the original system in Section 2. Next, we analyzes the result in Section 3. We conclude and make possible improvement suggestion in Section 4.

2 Approach

We used the method proposed for the search task suggestion problem from [1]. The online prediction algorithm which consists of three major steps: (a) identify task from procedural knowledge base, (2) retrieve and rank subtasks, (3) generate search tasks.

[1] Code and model are available at https://github.com/ziy/pkb
[2] Corpus is available at https://archive.org/details/wikihowcom

We briefly describe the system and the settings that were used to produce the three runs in this section. Interested readers can refer to the original paper for details.

In the first step, given a task-oriented search task represented by query, we retrieved a list of candidate tasks from the procedural knowledge base. We reused the same English wikiHow snapshot, which contains 198,163 non-redirect articles within namespace "0" (Main). We made two changes from the original approach in order to improve recall, merely for the purpose of evaluation. First, we kept all the 48,188 articles that are marked as stub (incomplete and need more information) and have no "Introduction" or "Steps", which were originally excluded for quality and training purpose. Second, rather than search for the exact matching text in either the summary or the explanation, we used Lucene's `EnglishAnalyzer` and default retrieval model to rank the tasks according to their overall relevance to the original query and allow partial matching. We kept the top 5 articles with their Lucene scores for further processing in the following steps.

Once we identified a list of most relevant tasks, we collected all the first-level subtasks from all the tasks. We also reused the pretrained query construction model $M^{\mathbb{Q}}$, which is a CRF-based word sequence labeling model, pretrained using various syntactic features and structural features on the training set described in our earlier work. We applied $M^{\mathbb{Q}}$, for each subtask's summary and explanation to identify each word sequence w_{ij} labeled as $\mathbf{B}^{\mathbb{Q}}\mathbf{I}^{\mathbb{Q}}\ldots\mathbf{I}^{\mathbb{Q}}$. Since the training corpus was constructed by finding all the search queries that match the task summaries, the extracted key phrases by our system tend to have similar characteristics to the wikiHow task representation, which are usually verb phrases. We noticed the example outputs for the sample queries also include noun phrases as well, e.g. unemployment benefits calculator, vatican city on sunday, etc., but due to time constraints, we reused the same model from our previous work, although our system also allows to plug in a query construction model that favors both noun and verb phrases.

In the final step, we need to rank the extracted key phrases. The original approach suggested to identify one key phrase from each subtask of the relevant task, and rank them globally by combining the weight obtained from subtask retrieval with the likelihood estimation from the query construction model $M^{\mathbb{Q}}$, which can lead to the most coverage of all necessary subtasks. Since relevance, rather than diversity, is the only consideration factor in the Task Understanding task, we slightly adjusted the approach by allowing multiple key phrases to be extracted and included in the final list. Moreover, even if some relevant tasks might include similar subtasks, key phrases extracted from any top-5 relevant task could be included in the final list. Three different runs differ in the corpus scope and the score combination method:

- `rsf`: exclude articles marked as `stub` as in the original paper, and a much higher weight is assigned to the Lucene score when combining with the query construction score.
- `lsf`: exclude articles marked as `stub` as in the original paper, and a much higher weight is assigned to the query construction score when combining with the Lucene score.
- `lsfs`: include articles marked as `stub`, and a much higher weight is assigned to the query construction score when combining with the Lucene score.

3 Result & Analysis

In this section, we analyze the system performance by demonstrating system outputs at four different stages in the system: task identification, task description comprehensiveness, key phrase extraction result, and ranking result. We conduct error analysis at each stage.

We first find that the system could correctly identify the task in the procedural knowledge base as long as there exist any, and it could also assign a higher score to the most relevant ones. For example, for topic 1 (getting organized at work), our system identified the following related wikiHow articles, with their Lucene scores in the parentheses:

```
- Get Organized and Concentrate on Your Work (4.082339)
- Be Organized at Work (3.4436269)
- Organize School Work (2.709037)
- Organize Work Space (2.6684413)
- Organize Work for School (2.5773914)
```

and for topic 6 (elliptical trainer).

```
- Work out on an Elliptical Machine (2.6765485)
- Use an Elliptical Machine (2.5518372)
- Use Elliptical Exercise Equipment Effectively (2.4160774)
- Draw an Elliptical Arch (2.2027557)
- Make Elliptical Garden Beds (2.0584767)
```

But we also found some topics are not mentioned in the current wikiHow procedural knowledge base. For example, for topic 2 (choose bathroom), our system could also identify the following related articles.

```
- Choose Lights for a Bathroom (4.8557177)
- Choose the Right Bathroom Vanities (4.751217)
- Choose Bathroom Towel Colors (4.6867957)
- Choose the Right Shower Curtain for Your Bathroom (3.5411198)
- Redecorate a Bathroom (2.2694883)
```

Although most of them are actually subtasks for the requested topic, subtasks under them might not be directly necessary to accomplish the original task.

Another observation is that using both summary and explanation fields of the task description in the retrieval step produced the best result, esp. the explanation field. We use topic 6 (elliptical trainer) as an example. The system would identify the following articles if only the summary field was used:

```
- Draw an Elliptical Arch (2.2111845)
- Make Elliptical Garden Beds (2.2111845)
- Use an Elliptical Machine (2.2111845)
- Work out on an Elliptical Machine (2.2111845)
- Use Elliptical Exercise Equipment Effectively (1.9347864)
```

We see that the first four articles have the same score and thus are unranked.

We found that the search intent of some topics is unambiguous, which include most verb phrases (including morphological variations) such as topic 4 (lower heart rate) as we discussed

in the earlier work, and also include some noun phrases topic 25 (barbados holidays), which very likely means "plan barbados holiday". But some topics are ambiguous, which are mostly noun phrases, such as topic 6 (elliptical trainer) can mean "how to buy an elliptical trainer" or "how to use an elliptical trainer". We manually judged the coverage of the current wikiHow procedural knowledge base by categorizing the results into four types: (1) the same task is identified, some (2) subtask or (3) super-task is identified, and (4) no relevant task is identified, where the second case is sometimes due to ambiguity and the third case usually happens when the topic has specific constraints such as topic 49 (raise venture capital ca), but wikiHow has only answers given to the general version "raise venture capital" without the location constraint. We discovered that there are 19, 19, 10, and 2 topics that belong to the four types respectively. We concluded that wikiHow could provide some useful advice to most (48/50) of the tasks, however there is still room to improve the coverage for super-tasks and subtasks. We also examined the content richness of the retrieved articles, and found that most of them contain enough information to support users' decision making. Actually, the retrieval model guarantees the qualify the returned articles by selecting only those that contain multiple occurrences of the query words.

We then check the performance of key phrase extraction. One of the main categories of procedural knowledge is recipe. Our model that was pretrained using wikiHow corpus could identify the key phrases at a fairly good accurate. For example, the top 5 key phrases extracted for topic 28 (how to cook pork tenderloin) are the following:

```
- buy fresh meat
- kick grilled tenderloin
- cook it
- chill meat
- slice your tenderloin
```

However, we found the most errors came from the query extraction and ranking step. The first type of error is that the phrase boundary was not corrected identified, or the key phrase does not mean a concrete task even if they grammatically follow the task summary description format. For example, the following key phrases were extracted and ranked in the top 5 positions for some topics:

```
- buy one
- comfortable
- buy
- sell
- expert '' elevation
- look it up online
```

We see that many extracted phrases tend to be short and thus semantically incomplete, which implies the query construction model is not robust enough to boundary detection. Besides consider retrain the model, we could also test some unsupervised approaches or rule-based approaches to increase the stability of the overall system.

Based on the reported results, our system achieves around 80% of the medium scores in most metrics, with 9 out of 34 topics over the medium (topics 4, 5, 12, 16, 18, 22, 28, 30, 37). The reported results also suggest that `rsf` run achieved a higher performance than the other two, i.e. the `stub` articles should be excluded, and key phrases extracted from the first articles should be ranked higher even if they are less confidence in terms of the query construction model score, rather than the most confident key phrases extracted from the second or lower ranked articles.

4 Conclusion

In this working note, we describe the system and necessary changes from the original system, analyze the result and identify the errors. As suggested in the earlier sections, we should consider to improve the robustness of the query construction model, and we may also employ unsupervised approaches or rule-based approaches to extract key phrases from each task description. We can also utilize the automatic procedural knowledge discovery approach proposed in our early work to extract subtasks directly from the Web documents to compliment the coverage limitation of the current approach.

References

1. Yang, Z., Nyberg, E.: Leveraging procedure knowledge for task-oriented search. In: Proceedings of the 38th international ACM SIGIR Conference on Research & Development in Information Retrieval (SIGIR'2015). ACM (2015)