FAQ Retrieval using Noisy Queries : English Monolingual Sub-task

Shahbaaz Mhaisale[1], Sangameshwar Patil[2], Kiran Mahamuni[2], Kiranjot Dhillon[3], Karan Parashar[3]

[1]Delhi Technological University, Delhi, India
[2]Tata Consultancy Services, Pune, India
[3]Bharati Vidyapeeth College of Engineering, Pune, India

Abstract:

This working note describes our first participation in the FIRE 2013 shared task on "FAQ Retrieval using Noisy Queries". Short messaging service (SMS) and voice based interfaces such as Siri have become quite popular for quick information retrieval these days. The problem at hand is – "Given an SMS query or speech-to-text transcript of a voice based query, find the best matching frequently asked question (FAQ) from a database". We first normalize the query, then compute the similarity between the query and each question using weighted edit distance. We also make use of a nearest neighbour classifier to check similarity of an incoming query with the historical query database. This classifier helped to improve the ranking among the set of predicted FAQs which are deemed relevant for the input query. We determine whether a query is out of domain (i.e. cannot be answered using current set of questions in the FAQ database) using a simple classifier using FAQ database itself. Our method identified 105 out of 148 out-of-domain queries correctly and 185 out of 392 in-domain queries correctly with mean reciprocal rank (MRR) of 0.8836.

1 Introduction

Mobile phones have evolved into personal digital assistants and have started taking over from the traditional, more dominant personal computers as the primary device for information access. The humble short messaging service (SMS) is being used in many innovative ways for enhancing the ubiquity of information access. Recently, an additional medium of voice based interface is also getting popular, for instance, Apple iOS' Siri[3], Samsung's S-voice[4] etc.

The shared task "FAQ retrieval using noisy queries" at FIRE 2013 [1] poses the challenge of information access from a database of frequently asked questions (FAQ) using the SMS and speech transcription from voice based systems as the interfaces. A system intended to tackle the above challenge needs to address the following sub-problems. As posed by the organizers [1], the main task is to identify the best matching question $\boldsymbol{Q^*}$ from the provided database $\boldsymbol{D}$ of frequently asked questions for a given input (noisy) query $\boldsymbol{S}$. A system is expected to produce a ranked list of questions in the decreasing order of relevance with respect to the query $\boldsymbol{S}$. An additional task is to identify the out-of-domain queries. If the FAQ database **D** does not contain a question relevant for the query $\boldsymbol{S}$, then the system should clearly indicate that query $\boldsymbol{S}$ is out of scope (i.e. out-of-domain, as required by the organizers).

2 Our Approach

Our method combines a variation of the algorithm developed by Kothari et al. [2] with query normalization techniques [5,7] and nearest neighbour classification. It differs from the algorithm in [2] with respect to the similarity computation between the normalized query string and the question text in

FAQ database. Instead of the consonant skeleton distance as used in [2], we use weighted edit distance (a variation of the standard Levenshtein distance [6]).

Algorithm: getMatchingFAQs
Input:

- Database of frequently asked questions (***D***)
- Training data (***T***)
- Input query (***S***)

Output:

A list ***L*** of at most ***n*** relevant questions from ***D*** along with the relevance score (i.e. ***L***= $[(f_1, r_1), (f_2, r_2), \ldots, (f_n, r_n)]$ where each $f_i \in D$ and r_i is the relevance score for f_i). If there are no questions relevant to input query ***S***, then an empty list ***L*** is returned.

Steps:

1. Preprocess the FAQ database ***D*** and
    a. Create a dictionary W_Q of terms appearing in the text of questions in ***D***. Each term is weighed with its TF-IDF with respect to ***D***.
    b. Create dictionary W_{QA} of terms in questions as well as answers for each domain in the FAQ database ***D***. Each term is weighed with its TF-IDF with respect to ***D***.
2. Let ***S'*** = normalized version of the query ***S*** using the typical SMS normalization patterns as identified in [5, 7, 8].
   Let *len_S* = number of words in query ***S'*** excluding the stopwords.
3. For each token t_i in the query ***S'***,
   For each question $f_j \in D$
    a. Compute similarity of t_i with terms in each question text using the weighted edit distance and LCSRatio as described in Section 2.2 .
    b. Choose the maximum similarity score as the score s_{ij} for the term t_i with respect to question f_j.
    c. Let *len_j* = number of words in question f_j excluding the stopwords
    d. Relevance score r_j of question f_j for query ***S'*** is computed as $r_j = \left(\prod_{i=1 \text{ to } len\_S} s_{ij}\right) / (len_j)^2$
4. Add at most ***n*** questions from ***D*** to the output list ***L*** according to decreasing relevance score.
5. Check nearest neighbour for query ***S*** in training data ***T***:
    a. Compute cosine similarity of query ***S*** with queries from training data ***T***. Let s* be the query from ***T*** with maximum similarity.
    b. If similarity(***S***, s*) > θ, then include the best matching question f* for s* as available from ***T*** in the output relevance list ***L*** with the similarity as the relevance score. (Note: θ is a configuration parameter with a value typically being 0.9 or more). If list ***L*** already contains ***n*** questions, then replace the question having least score with question f*.
6. Check for out-of-scope queries:
    a. Compute the in-scope score for the query ***S*** using the dictionary W_{QA} for each domain (from step 1).

       Query-in-scope score = $\sum_{\text{term "t" present both in query and W\_QA}}$ TF-IDF (t)

    b. If the query-in-scope score is below a threshold (ρ , a configurable parameter) for each domain, then we term that the input query ***S*** does not have good enough relevant questions in the FAQ database and its output list ***L*** is set to empty list.

We make use of query normalization as a preprocessing step followed by weighted edit distance based similarity computation between the incoming query ***S*** and the questions in FAQ database ***D***. We use a nearest neighbour classifier built using the training data to augment the set of relevant questions for query ***S*** as well as to enhance the ranking among the set of relevant questions. We then use a simple classifier to determine whether the query ***S*** is out-of-scope with respect the given FAQ database. The detailed steps are described in the algorithm *getMatchingFAQs*.

2.1 Query Normalization and FAQ Preprocessing

While dealing with noisy queries, query normalization or cleansing is naturally the beginning phase for query processing. In the current version of our system, we focus on normalization of SMS queries and not on cleansing of transcription of speech queries. Normalization of noisy text such as SMS queries, tweets, chat messages etc. has received significant attention in the literature for instance, [5,7]. We make use of some of the typical abbreviations used in SMS text [8] as well as patterns [5, 7] for creating a normalized version ***S'*** of the input noisy query ***S.*** This normalized version is used for further processing of similarity computation with questions in the FAQ database and relevance scoring.

We also preprocess the FAQ database and similar to [2] create a dictionary W_Q which consists of all the terms in the questions of FAQ corpus. Along with each term is stored it's IDF and a list of synonyms (for nouns only). Additionally, we also create a dictionary W_{QA} for each domain (e.g. health, agriculture, tourism) in the FAQ database. These per-domain dictionaries W_{QA} are used for identifying whether an input query is within the scope of currently available FAQ or not. They contain the terms from both question as well as answers from the FAQ database.

2.2 Similarity Computation and Relevance Scoring

The normalized input query ***S'*** is tokenized and a similarity score is calculated for each token in the resulting sequence and terms in the question text of each FAQ. Our similarity computation is in the spirit of algorithm in [2], but contains important variations from them. We use a weighted edit distance is used instead of the consonant skeleton distance as used in [2].

Weighted Edit Distance (term, token) =
0.3 * (no. of vowel insertions) + 0.3 * (no. of insertions at end of token)
+ 0.5 * (no. of insertions) + 0.5 * (no. of vowel substitutions)
+ 2.0 * (no. of deletions)

We use the LCSRatio as defined in [2] using the longest common sequence between query token and a question term. The similarity score between a term in the question text of a FAQ and a query token is computed as:

$$Similarity\ score(FAQterm, Query_{\_token}) = \frac{IDF\ \ FAQterm\ \ * LCSRatio(FAQterm, Query_{\_token})}{Weighted\ Edit\ Distance(FAQterm, Query_{\_token})}$$

For each query token, the FAQterm with maximum similarity score is chosen. Finally, as mentioned in the step 3.d of the algorithm *getMatchingFAQs*, the relevance score for a normalized query ***S'*** is computed using the product of maximum similarity score of each token in ***S'*** . The product of similarity scores is divided by the square of the length of the FAQ (number of words, excluding the stopwords)**.** A very long FAQ automatically receives a larger score since it has more words to match with the SMS as compared to shorter FAQs. For this reason, we penalize the scores of long FAQs in proportion to the square of their length. We experimentally found that the power 2 performs better than 1, 1.5, 2.5, 3 or 4.

2.3 Post-processing

As described in step 4 of the algorithm *getMatchingFAQs*, we also make use of a nearest neighbour classifier to check similarity of an incoming query with the historical query database. This classifier helped to improve the ranking among the set of predicted FAQs which are deemed relevant for the input query.

The out-of-domain classification of query is done in the step 5 of the algorithm *getMatchingFAQs*. We determine whether a query is out of domain (i.e. cannot be answered using current set of questions in the FAQ database) using the weighted overlap between the query text and the set of per-domain dictionaries $\boldsymbol{W_{QA}}$ created while preprocessing the FAQ database. Finally, a list ***L*** consisting of at most ***n*** relevant questions from ***D*** along with the relevance score is returned as output. If there are no questions relevant to input query ***S***, then an empty list ***L*** is returned.

3 Experimental Results

The FIRE 2013 test dataset has a total of 540 queries. Out of these 540 input queries, number of in-domain queries are 392 and number of out-of-domain queries are 148.

As per the results provided by the organizers, our method identified 105 out of 148 out-of-domain queries correctly and 185 out of 392 in-domain queries correctly with mean reciprocal rank (MRR) of 0.8836 .

4 Conclusion

We used the techniques of query normalization, weighted edit distance similarity and nearest neighbour classification to build a system for participating in the FIRE 2013 task of FAQ retrieval using noisy queries. Current system gives the better accuracy for SMS text related queries than for speech queries. Processing speech related queries needs to be improved. We found that preprocessing of noisy SMS queries speeds up the system and also helps in improving the accuracy.

References

[1] FIRE 2013 Shared Task detailed description: FAQ retrieval using noisy queries, http://www.isical.ac.in/~fire/faq-retrieval/2013/faq-retrieval.html (URL verified 19 Nov. 2013)

[2] Govind Kothari, Sumit Negi, Tanveer A. Faruquie, Venkatesan T. Chakaravarthy, and L. Venkata Subramaniam. 2009. SMS based interface for FAQ retrieval. In Proceedings of the 47th Annual Meeting of the ACL and the 4th IJCNLP of the AFNLP, pages 852–860, 2009.

[3] Samsung S-voice, http://www.samsung.com/global/galaxys3/svoice.html (URL verified 19 Nov. 2013)

[4] Apple iOS Siri, http://www.apple.com/ios/siri/ (URL verified 19 Nov. 2013)

[5] W. Wong, W. Liu, and M. Bennamoun. 2007. Enhanced Integrated Scoring for Cleaning Dirty Texts. In Proceedings of IJCAI-2007 Workshop on Analytics for Noisy Unstructured Text Data.
[6] D. Jurafsky and J. H. Martin. An Introduction to Natural Language Processing, Computational Linguistics, and Speech Recognition. Prentice Hall, 2000.
[7] Eleanor Clark and Kenji Araki, 2011. Text normalization in social media: progress, problems and applications for a pre-processing system of casual English. In Procedia – Social and Behavioural Sciences 27 (2011) 2 – 11.
[8] Web resources for abbreviations and acronyms typically used in SMS, http://www.abbreviations.com/acronyms/SMS (URL verified 19 Nov. 2013)