Article title: Question Annotation for Community Question-Answer Forums

Authors: Avantika Borikar[1]

Affiliations: Visvesvaraya National Institute of Technology, Nagpur[1]

Orcid ids: 0000-0001-7879-4136[1]

Contact e-mail: avantikasb@gmail.com

License information: This work has been published open access under Creative Commons Attribution License <http://creativecommons.org/licenses/by/4.0/>, which permits unrestricted use, distribution, and reproduction in any medium, provided the original work is properly cited. Conditions, terms of use and publishing policy can be found at <https://www.scienceopen.com/>.

Preprint statement: This article is a preprint and has not been peer-reviewed, under consideration and submitted to ScienceOpen Preprints for open peer review.

DOI: 10.14293/S2199-1006.1.SOR-.PPBHUYD.v1

Preprint first posted online: 29 September 2020

Keywords: Recommender systems, QA forum

Question Annotation for Community Question-Answer Forums

Sarvesh Parab^{*}, Shaikh Mohammad Danish[†], Avantika Borikar[‡] and U.A.Deshpande[§]

Abstract

A decision making process is simplified with the help of recommender systems. Recommender systems process the knowledge sources and the information actively to collect data in order to build useful recommendations. These recommendations suggest suitable items to the user based on the analysis performed on the user's preferences and constraints, both implicit and explicit. Using the content-based filtering approach of recommender systems, this article suggests an innovative idea to annotate the question asked by the user on a QA forum with suitable tags. This article presents a novel scheme to suggest the relevant tags by effectively analyzing questions from a clustered knowledge pool and then ranking the tags according to their relevance. This scheme aims at providing meaningful, trustworthy and persuasive recommendations which will stratify the question in the appropriate domain of a QA forum.

1 Introduction

The community Question-Answer(cQA) websites are large repositories of valuable knowledge. These websites are developing rapidly with the immensely growing source of information in varied areas. StackOverflow, Quora, Yahoo!Answers are the typical examples of cQA services with widespread acceptance. However, rapid growth provides better opportunities as well as puts forward new challenges. Every cQA website aims at providing immediate and satisfactory answers to any question asked by the user. One of the major challenges is directing the new questions to the right community of experts. Question annotation is one of the transformative processes which is basically used to classify and direct the question to the community where it can be answered by the right group of experts. Annotating Questions with appropriate tags proves useful. Firstly, the user is enabled to give multiple tags to a single question. These assigned tags summarize the question at high level. Secondly, tags help in putting the questions to the feeds of the related domain followers and experts which leads to fast and accurate answers. However, the incompleteness of tagging is usually observed which is caused by incomprehensive question understanding or informal tagging behaviours. For example, some user-provided tags in the vocabulary are often biased towards personal perspectives or specific contextual information. This severely hinders the performance of the tag-based systems. As a result, the performance of the question search will be degraded due to the absence of potentially relevant tags which are used to expand the question. Also, it obstructs the efficient question routing to the appropriate domain.

One such similar contribution is [1] which makes use of Adaptive probabilistic hypergraph learning in order to infer semantically close question space. It further uses various similarity scores to rank the tags. In our approach, we've utilized high confidence tags (discussed later in the paper) and number of common tags in order to obtain a reduced relevant question space. We further find semantic closeness (using a hybrid, trained semantic score algorithm discussed later) of each question in this space with the question being asked by the user in order to obtain a final sorted list of relevant questions. The uncommon tags from this sorted list of relevant questions is sorted further based on confidence values to obtain the final list of Top-N tags to be recommended to the user.

This article aims at generating automatic relevant tags which tries to tackle all the difficulties. We have used lexical properties on the similar question space to drill down the knowledge source to find out the relevance score between the question and the tags. We have developed a scheme which will enhance question annotating process using both: content analytics and history of tagging behaviour from the knowledge source.

^{*}S.S.Parab, Department of Computer Science, Visvesvaraya National Institute of Technology, Nagpur, Maharashtra

[†]Shaikh Mohammad Danish, Department of Computer Science, Visvesvaraya National Institute of Technology, Nagpur, Maharashtra

[‡]A.S.Borikar, Department of Computer Science, Visvesvaraya National Institute of Technology, Nagpur, Maharashtra

[§]U.A.Deshpande, Head of the Department, Department of Computer Science, Visvesvaraya National Institute of Technology, Nagpur, Maharashtra

The main contributions of this research are as follows.

- This work uses the hybrid similarity measure obtained by combining various measures of semantic similarity, thus postulating a new hybrid score for semantic relatedness. This hybrid score would train itself and adapt according to the diversity of the question-space in the corpus.
- It proposes an idea of high confidence tags. We create a subset of tags from the tag space comprising of tags pre-associated with the ones extracted from question title and that given by the user (if any). Thus, the questions from the knowledge pool containing these tags have a higher level of confidence for tentative semantic relatedness.
- Reducing the time complexity for fetching the top recommendations, this scheme strategically abridges the domain space with high confidence tags followed by ranking with hybrid semantic measure. This not only provides more accurate tag recommendations but also suggests cognate questions.

The remainder of this article is structured as follows. Section 2 introduces and describes the methodology followed for the recommendation of relevant tags. Section 3 gives the evaluation and the analysis of the approach as well as gives the performance highlights followed by our concluding remarks in Section 4.

2 Methodology

2.1 Question Preprocessing

The user question string is processed first to obtain the tags which are directly present in the question. The processing of the question was done with the help of Stanford core NLP library. Question Preprocessing involves converting the given user question into a standard format as described below:

2.1.1 Lemmatizing

While comparing two words, a common problem arises as both words may not be in the same form. Hence, we have applied standard lemmatization algorithms to fetch the base form of the word.

2.1.2 Stop Word Removal

Nearly all question titles have few common recurring words. These words add no value to their semantic relatedness. Hence, it is better to remove such words before computing score. We have maintained a list of all the stop words, and remove the word from the question which is present in the list.

2.1.3 Tokenization

The user may not always separate two words in his question title with a space. He might use many other punctuation marks as delimiters or terminators. Hence we need to identify correct word boundaries. We have used standard tokenizing libraries to do so.

2.1.4 Punctuation Removal

The user question might have punctuation marks like “!”, “?”, “.”, etc. We need to remove them as with them present word identification becomes difficult.

2.1.5 Stemming

The problem is analogous to that in the case of lemmatization. So stemming is the process of reducing words to their word stem, base or root form.

The following hypothetical dataset will be used as a running example throughout the article.

Q	T1	T2	T3	T4	T5	T6	T7	T8	T9	T10
Q1	✓	✓			✓		✓			
Q2							✓	✓	✓	
Q3			✓			✓			✓	✓
Q4					✓				✓	✓
Q5					✓			✓		✓
Q6						✓	✓	✓	✓	
Q7			✓		✓	✓				
Q8				✓				✓	✓	✓
Q9				✓	✓	✓				
Q10	✓			✓	✓					
Q11	✓	✓	✓	✓	✓					
Q12		✓			✓			✓		
Q13					✓	✓		✓	✓	
Q14							✓		✓	✓
Q15	✓		✓					✓	✓	
Q16		✓	✓	✓			✓		✓	
Q17		✓							✓	✓
Q18					✓				✓	✓
Q19				✓			✓	✓		
Q20	✓					✓	✓	✓		

Table 1: Sample Dataset

2.2 Semantic analysis and hybrid similarity measure

2.2.1 Creating vectors and scoring

We need to find a numeric semantic score between the user asked question(Q_u) and a question(Q_c) from our corpus. All the semantic similarity algorithms mentioned below return a relatedness score between two words. Thus we need a way to compare two sentences (Q_u & Q_c).

Now to build vector " V_u ", every term in " Q_n " is compared with " Q_u ". If an exact match is found, then that vector dimension gets the maximum value. Else if not a match, the vector dimension gets the largest of all the semantic scores calculated for that term in " Q_n " against every term in " Q_u ". [4] [5] [6]

Similar process is repeated to get vector " V_c ". Now, the net score between in " V_u " & " V_c " is calculated using Cosine Similarity as follows:

$$\cos \varphi = \frac{V_u \cdot V_c}{\|V_u\| \|V_c\|}$$

2.2.2 Semantic Closeness/Relatedness

Knowledge-Based Similarity is one of semantic similarity measures that bases on identifying the degree of similarity between words using information derived from semantic networks. WordNet is the most popular semantic network in the area of measuring the Knowledge-Based similarity between words; WordNet is a large lexical database of English. Nouns, verbs, adjectives and adverbs are grouped into sets of cognitive synonyms (synsets), each expressing a distinct concept. Synsets are interlinked by means of conceptual-semantic and lexical relations. Knowledge-based similarity measures can be divided roughly into two groups: measures of semantic similarity and measures of semantic relatedness. Semantically similar concepts are deemed to be related on the basis of their likeness. Semantic relatedness, on the other hand, is a more general notion of relatedness, not specifically tied to the shape or form of the concept

There are six measures of semantic similarity; three of them are based on information content: Resnik (res) [7], Lin (lin) [8] and Jiang & Conrath (jcn) [9]. The other three measures are based on path length: Leacock & Chodorow (lch) [10], Wu & Palmer (wup) [11] and Path Length (path).

Furthermore, there are three measures of semantic relatedness: St. Onge (hso) [12], Lesk (lesk) [13] and vector pairs (vector) [14].

We need to get a numeric equivalent of how closely related two questions are. We create vectors for the questions based on the union set of keywords based on both the questions. Then, calculate the cosine similarity score. But this would consider words like 'love' and 'like' as distinct. Hence to have a better understanding of relatedness, we analyse all the extracted keywords and compare them token by token through a synset - which would give a score based on token similarity. We would factor in this along with the cosine similarity score to establish a more realistic comparison of two questions.

2.2.3 Training of Hybrid Similarity Measure

We have seen the different semantic relatedness calculating algorithms above. Each of them has its own merits and demerits. As we are considering a wide range of questions which handle many types English language constructs, there is a need for a hybrid method. For example, word1: love; word2: romance; word3: hate Algorithms like Wup which are based on path lengths give a higher score for word1 and word3 as they are more closely related. But our requirement was to have a higher score for word1 and word2.

Using only one algorithm won't give us the satisfactory results. Hence we need to use combination of different algorithms. Firstly, we need to choose the combination of algorithms, and how much weightage does each algorithm gets [15].

Here is the approach to get the weighted combination of algorithms:

- Create a weight vector W of size six initialized to zero.
- Out of the approximate 1.15 million questions D in our corpus, we have randomly chosen N questions.
- For each question N_i of these N questions:
 - Create a set (T) of tags (t) which were associated with that question.

$$T = \{t_1, t_2, \dots, t_k\}$$

- Create a dataset of all the questions who at least have one tag from T

$$Q = \{q \mid (q \subseteq D) \cap (T \cap T_q \neq \phi)\}$$

Where T_q is the set of tags associated with q

- From this Q set, select at random M questions, and for M_j of them:
 - * Run all the six semantic algorithms to find the score between N_i & M
 - * Store these results in individual CSV files along with their respective index j .
- Thus we will have N CSV files, and for each of them:
 - Sort each of individual algorithmic scores in ascending order and store their respective indices in vectors (V_{1-6}), one vector for every algorithm.
 - Create a set δ_{man} Manhattan distances between the vectors.

$$\delta_{\text{man}} = \delta_{i,j} \mid (\delta_{i,j} = \text{Manhattan}(V_i, V_j))$$

$$(1 \leq i, j \leq 6) \& (i \neq j)$$

2.3 Tag Extraction

2.3.1 Extraction of tags from Question title

This involves searching for presence of multi-word (eg: ruby on rails) and single word (eg: ruby) tags from the Knowledge Base mentioned in the question title. These tags are suggested to the user who selects one or more relevant tags. The user must enter at least one relevant tag explicitly if no tags could be extracted from the question title or if none of the tags suggested to the user from question title were relevant. Let the final list of tags be T_e . Given a question Q in the knowledge base containing tags T_q , Common Tags (CT_q) = $T_e \cap T_q$ and Uncommon Tags (UT_q) = $T_q - CT_q$. Let's say that tags T_1, T_2, T_3 could be extracted from the title of user question Q_u . Further let's assume all these tags are relevant i.e. the user chooses all these tags and asks for further recommendations.

2.3.2 Finding High Confidence Tags

- Let $T_q = (T_{q1}, T_{q2}, \dots, T_{q10})$ be the set of tags obtained from the question title and the tags entered by the user (if any).
- Let $T = (T_1, T_2, \dots, T_n)$ be the list of the total tags in the knowledge base. The basic motivation behind finding high confidence tags is to find a subset of T wherein each tag T_i in this subset occurs many times with one or more subsets of T in the knowledge base, i.e. there is a high confidence that T_i occurs with T in the user question
- We define the Confidence Level for a tag T_t calculated on the basis of a set $S = (T_i, T_j, T_k, \dots, T_o)$ as $\text{Confidence}(T_t, S) = (\text{Number of questions in which } T_t \text{ occurs along with each of element in the powerset of the tags in } S) / (\text{Number of questions in which all the tags in } S \text{ occur})$
- We define the list Top- M High-Confidence(HC) tags for a set $S = (T_i, T_j, T_k, \dots, T_o)$ as the set $HC(S, M) = (HCT_1, HCT_2, \dots, HCT_m)$ where $\text{Confidence}(HCT_1, S) \geq \text{Confidence}(HCT_2, S) \geq \dots \geq \text{Confidence}(HCT_{m-1}, S) \geq \text{Confidence}(HCT_m, S)$
- Let T be a tag in the knowledge base. Let S_t be the non-empty subsets of T_e (refer 2.3.1) wherein T (belongs to) $HC(S, M)$
The final confidence value of T is then computed as $HCV_t = (\text{for all } S \text{ in } S_t) (\max \text{Confidence}(T, S))$

Keeping the sample dataset in reference, we compute Top- M high confidence tags for each non-empty subset of $\{T_1, T_2, T_3\}$ (refer Table 3). The confidence values of the tags thus obtained is computed as the Max confidence value obtained for the corresponding tag. (refer Table 2).

	T4	T5	T6	T7	T8	T9	T10	High Confidence Tags
T1	0.4	0.6	0.2	0.4	0.4	0.2	0	T5, T4, T7
T2	0.4	0.6	0	0.4	0.2	0.4	0.2	T5, T4, T7
T3	0.4	0.4	0.4	0.2	0.2	0.6	0.2	T9, T4, T6
$T1 \cup T2$	0.5	1	0	0.5	0	0	0	T5, T7, T4
$T2 \cup T3$	1	0.5	0	0.5	0	0.5	0	T4, T5, T7
$T1 \cup T3$	1	0.5	0	0	0.5	0.5	0	T4, T8, T9
$T1 \cup T2 \cup T3$	1	1	0	0	0	0	0	T4, T5, T6

Table 2: Top- M High Confidence tags in each subset ($M=3$)

Thus, we find the high confidence tags for every combination of the tags $T1$, $T2$ and $T3$ from Table 2

2.4 Ranking for tags

2.4.1 Extraction of relevant questions from the knowledge base

The total confidence value (CV_q) and number of common tags (refer 2.3) are calculated for questions in the Knowledge base containing at least one tag common with the list of tags T_e (refer 2.3). These questions are then sorted based on the order (total confidence value, Number of common tags) resulting in sorted list of relevant questions.

Keeping the sample dataset in reference, the above Venn diagram summarizes the set of relevant questions and number of common tags with Q_u . The total confidence values of each relevant question is then computed as shown in Table 4 (substituting confidence values from Table 2) and the questions are sorted based on (Total Confidence Value, Number of Common tags).

Keeping the sample dataset in reference and considering $N=6$, Top-7 questions corresponding to 6 uncommon tags (refer 2.3.1). The semantic score is computed for each question in this list as shown in Table 4.

2.4.2 Reduction of the extracted set of relevant questions

The list of relevant questions from 2.4.1 needs to be further reduced in order to apply hybrid similarity measure (refer 2.2.3) in real time environment (an alternative to this is to make use of distributed processing). This is achieved by selecting the Top-M questions Q_m that contribute to N uncommon tags (refer 2.3).

Hybrid Semantic score is found for each question in the list Q_m with the user question, the list Q_m is further sorted based on the semantic score to obtain sorted reduced list of relevant questions SQ_m .

2.4.3 Finding the sorted list of N relevant tags

The uncommon tags in SQ_m are finally ranked as follows:

1. Tag T_1 belongs to relevant question RQ_1 and tag T_2 belongs to relevant question RQ_2 . If RQ_1 is higher in SQ_m , T_1 gets a higher rank than T_2 and vice versa.
2. Tag T_1 and Tag T_2 both belong to the same relevant question RQ . In this case, we use the confidence level of the tags T_1 and T_2 to decide the ranking between them i.e if $HCV_{t1} > HCV_{t2}$ then T_1 gets a higher rank than T_2 and vice versa.

The reduced sorted list of relevant questions obtained from Table 4 is utilized to recommend Top-6 tags as shown in Table 5.

Common Tags	Count	Uncommon Tags	Total Confidence	Question Number	Ranking
T1, T2	2	T5, T7	$1 + 0.5 = 1.5$	Q1	4
T3	1	T6, T9, T10	$0.4 + 0.6 + 0 = 1$	Q3	9
T3	1	T5, T6	$1 + 0.4 = 1.4$	Q7	6
T1	1	T4, T5	$1 + 1 = 2$	Q10	3
T1, T2, T3	3	T4, T5	$1 + 1 = 2$	Q11	2
T2	1	T5, T8	$1 + 0.5 = 1.5$	Q12	5
T1, T3	2	T8, T9	$0.5 + 0.6 = 1.1$	Q15	8
T2, T3	2	T4, T7, T9	$1 + 0.5 + 0.6 = 2.1$	Q16	1
T2	1	T9, T10	$0.6 + 0 = 0.6$	Q17	10
T1	1	T6, T7, T8	$0.4 + 0.5 + 0.5 = 1.4$	Q20	7

Table 3: Total confidence value and ranking of each relevant question

Question Number	High Confidence	No. of Common Tags	Semantic Closeness
Q16	2.1	2	0.8
Q11	2	3	0.7
Q10	2	1	0.75
Q1	1.5	2	0.5
Q12	1.5	1	0.6
Q20	1.4	1	0.8
Q7	1.4	1	0.6

Table 4: Further sorting of Top-M questions based on hybrid semantic score

Question Number	Unsorted Tags	Sorted Tags
Q16	T9	T4
	T4	T9
	T7	T7
Q11	T5	T4
	T4	T5
Q10	T4	T4
	T5	T5
Q1	T5	T5
	T7	T7
Q12	T5	T5
	T8	T8
Q20	T8	T6
	T6	T7
	T7	T8
Q7	T5	T5
	T6	T6

T4

T9

T7

T5

T8

T6

Table 5: Top-N tags to be recommended

3 Evaluation and analysis

The proposed implementation was thoroughly tested with over 50,000 questions already asked on StackOverflow website. The tags suggested by our system was compared with the tags already present with the question on StackOverflow.

3.1 Accuracy Measure

We define the accuracy as:

$$A = \frac{[\sum_{t_i \in UST} 1 \times 100] + [\sum_{t_i \in TFRT} 1 \times p_i] + [\sum_{t_i \in TNF} f_i \times 0]}{[\sum_{t_i \in UST} 1] + [\sum_{t_i \in TFRT} 1] + [\sum_{t_i \in TNF} f_i]}$$

where,

- A is the accuracy measure
- UST (User Selected Tags) has the tags selected by the user from the suggested tags
- $TRFT$ (Tags Found in Ranked Tags) are tags for the question found in the recommendation
- TNF (Tags Not Found) are the tags in the question title or recommended by our system but are present in the question
- $p_i = \frac{N_T - m_i}{N_T}$ where m_i is the rank of the tag t_i by our system and N_T is the number of tags in our system
- $f_i = \frac{M_i}{N_Q}$ where M_i is the number of question having tag t_i and N_Q is the total number questions in our corpus. It has a weight of 0 as the absence of the tag should not contribute to the accuracy.

3.2 Performance

Assuming that the average question has 5 tags on StackOverflow, of which two were derived from the title and 2 were recommended by our system and 1 tag was left out

We can understand from the graph/image that even if our system fails to give one of the recommended tag, the drop in the accuracy is depended on how prevalent was that tag in the corpus.

Figure 1: **A)** A random question taken from stackoverflow.com helps us demonstrate how this system arrives at the proper annotations. **B)** It picks out the plausible tags from the question title. **C)** Then compares all questions in the corpus and recommends a list of tags and ranks them as it deems them most relevant.

4 Conclusion

This article describes the problems associated with tagging questions in a cQA manually thereby suggesting a novel scheme in order to recommend Top-N relevant tags to the user taking help of predefined knowledge base of questions. The procedure involves preprocessing the user question in order to convert it into a standard format (refer 2.1), extracting tags from question title (refer 2.3.1) and utilizing these to find high confidence tags (refer 2.3.2). This is followed by obtaining relevant question space and sorting it based on two criteria (refer 2.4.1), reducing the relevant question space further (refer 2.4.2) and applying trained hybrid semantic score algorithm (refer 2.2) between the user question and each question in the reduced relevant question space, post which the resultant questions are sorted based on the semantic score computed. The uncommon tags (refer 2.3.1) obtained from the final list of questions is further sorted based on two criteria (refer 2.4.3) to give the Top-N relevant tags. These tags are recommended to the user. Additionally, the recommender also suggests relevant questions to the user as the final sorted reduced list of relevant questions.

Our proposed scheme can benefit cQA's by reducing the biased tagging of questions and incomplete tagging of questions tremendously. Furthermore, it will also help the users by increasing reach of the question to relevant users thereby ensuring quick response. It also helps the user in that the user would just have to think of one tag (in the worst case), the rest of the tags can be selected from the recommendations. Distributed computing can be leveraged in huge knowledge bases to ensure real-time usage of the recommender. Furthermore, the recommender currently doesn't infer any knowledge from the question content. In-depth research regarding such an enhancement in the recommender remains to be investigated.

References

- [1] L. Nie, Y.-L. Zhao, X. Wang, J. Shen and T.-S. Chua, "Learning to Recommend Descriptive Tags for Questions in Social Forums," ACM Transactions on Information Systems, vol. 32, no. 1, p. 5, January 2014.
- [2] "StackOverflow Data Exchange," March 2016. [Online]. <https://data.stackexchange.com/stackoverflow/query/new>.

- [3] "StackOverflow," [Online]. <https://blog.stackoverflow.com/2014/01/stack-exchange-cc-data-now-hosted-by-the-internet-arch> [Accessed April 2016].
- [4] H. Liu and P. Wang, "Assessing Sentence Similarity Using WordNet," *Journal of Software*, vol. 8, no. 6, p. 1455, June, 2013.
- [5] Y. Li, D. McLean, Z. A. Bandar, J. D. O'Shea and K. Crockett, "Sentence Similarity Based on Semantic Nets and Corpus Statistics," *IEEE Transactions on knowledge and data engineering*, vol. 18, no. 8, August 2006.
- [6] Y. Li, Z. Bandar, D. McLean and J. O'Shea, "A Method for Measuring Sentence Similarity and its Application to Conversational Agents," *Intelligent Systems Group, Department of Computing and Mathematics, Manchester Metropolitan University, Manchester*.
- [7] R. Resnik, "Using information content to evaluate semantic similarity.," *14th International Joint Conference on Artificial Intelligence*, 1995.
- [8] Lin, "Extracting Collocations from Text Corpora," *Workshop on Computational Terminology*, 1998.
- [9] J. Jiang and D. Conrath, "Semantic similarity based on corpus statistics and lexical taxonomy," *International Conference on Research in Computational Linguistics*, 1997.
- [10] C. Leacock and Chodorow, "Combining local context and WordNet sense similarity for word sense identification.," 1998.
- [11] Wu and Z. Palmer, "Verb semantics and lexical selection," *Association for Computational Linguistics*, 1994.
- [12] G. Hirst and St-Onge, "Lexical chains as representations of context for the detection and correction of malapropisms.," pp. 305-332, 1998.
- [13] B. and S. Pedersen, "An adapted Lesk algorithm for word sense disambiguation using WordNet.," *Intelligent Text Processing and Computational Linguistics*, pp. 136-145, 2002.
- [14] Patwardhan, "Incorporating dictionary and corpus information into a context vector measure of semantic relatedness.," 2003.
- [15] W. Gomaa and A. A. Fahmy, "A Survey of Text Similarity Approaches," *International Journal of Computer Applications*, vol. 68, no. 13, p. 16, April 2013.