Multiple Choice Question (MCQ) Answering System for Entrance Examination

Somnath Banerjee
Università degli Studi di Milano-Bicocca
37 PUBLICATIONS **225** CITATIONS
SEE PROFILE

Pinaki Bhaskar
IIT - CNR, PIsa, Italy
28 PUBLICATIONS **148** CITATIONS
SEE PROFILE

Dr. Partha Pakray
National Institute of Technology, Silchar
116 PUBLICATIONS **409** CITATIONS
SEE PROFILE

Sivaji Bandyopadhyay
Jadavpur University
304 PUBLICATIONS **3,110** CITATIONS
SEE PROFILE

Multiple Choice Question (MCQ) Answering System for Entrance Examination

Somnath Banerjee[1], Pinaki Bhaskar[1], Partha Pakray[1], Sivaji Bandyopadhyay[1], and Alexander Gelbukh[2]

[1] Department of Computer Science and Engineering,
Jadavpur University, Kolkata – 700032, India
[2] Center for Computing Research,
National Polytechnic Institute, Mexico City, Mexico
{ s.banerjee1980, pinaki.bhaskar, parthapakray}@gmail.com,
sivaji_cse_ju@yahoo.com, gelbukh@gelbukh.com

Abstract. The article presents the experiments carried out as part of the participation in the pilot task of QA4MRE@CLEF 2013. In the developed system, we have first generated answer pattern by combining the question and each answer option to form the Hypothesis (H). Stop words and interrogative word are removed from each H and query words are identified to retrieve the most relevant sentences from the associated document using Lucene. Relevant sentences are retrieved from the associated document based on the TF-IDF of the matching query words along with n-gram overlap of the sentence with the H. Each retrieved sentence defines the Text T. Each T-H pair is assigned a ranking score that works on textual entailment principle. A matching score is automatically assigned to each answer options based on the matching. A parallel procedure also generates the possible answer patterns from given questions and answer options. Each sentence in the associated document is assigned an inference score with respect to each answer pattern. Evaluated inference score for each answer option is added with the matching score. The answer option that receives the highest selection score is identified as the most relevant option and selected as the answer to the given question.

Keywords: Question Answering technique, QA4MRE Data Sets, Named Entity, Textual Entailment.

1 Introduction

The main objective of QA4MRE [3] is to develop a methodology for evaluating Machine Reading systems through Question Answering and Reading Comprehension Tests. Machine Reading task obtains an in-depth understanding of just one or a small number of texts. The task focuses on the reading of single documents and identification of the correct answer to a question from a set of possible answer options. The identification of the correct answer requires various kinds of inference and the consideration of previously acquired background knowledge. Ad-hoc collections of background knowledge have been provided for each of the topics in all the languages involved in the exercise so that all participating systems work on the same background knowledge. Texts have been included from a diverse range of sources, e.g. newspapers, newswire, web, blogs, Wikipedia entries.

Answer Validation (AV) is the task of deciding for given a question and an answer from a QA system, whether the answer is correct or not and it was defined as a problem of RTE in order to promote a deeper analysis in Question Answering [3]. Answer Validation Exercise (AVE) is a task introduced in the QA@CLEF competition. AVE task is aimed at developing systems that decide whether the answer of a Question Answering system is correct or not. There were three AVE competitions AVE 2006 [4], AVE 2007 [5] and AVE 2008 [6]. AVE systems receive a set of triplets (Question, Answer and Supporting Text) and return a judgment of "SELECTED", "VALIDATED" or "REJECTED" for each triplet.

Section 2 describes the corpus statistics. Section 3 describes the system architecture. The experiments carried out on test data sets are discussed in Section 4 along with the results. The conclusions are drawn in Section 5.

2 Corpus Statistics

Like main task, this pilot task focuses on the reading of single documents and the identification of the answers to a set of questions about information that is stated or implied in the text. Questions are in the form of multiple choices, each having five options, and only one correct answer. The detection of correct answers is specifically designed to require various kinds of inference.

The Entrance Exams 2013 test set will be composed of reading comprehension tests taken from the Japanese Center Test, which is a nation-wide achievement test for Japanese university admissions: Each reading test will consist of one single document, with 5 questions and a set of five choices per question. So, there will be in total:

- - 9 reading test documents
- - 46 questions (5 questions for each document except document 2 with 6 questions)
- - 184 choices/options (4 for each question)

Test documents, questions, and options were made available in English. Participating systems will be required to answer these 45 questions by choosing in each case one answer from the five alternatives. There will always be one and only one correct option.

Systems will also have the chance to leave some questions unanswered if they are not confident about the correctness of their response.

3 Machine Reading System Architecture

The architecture of machine reading system is described in Figure 1 and the proposed architecture is made up of two main modules. Each of these modules is now being described in subsequent subsections.

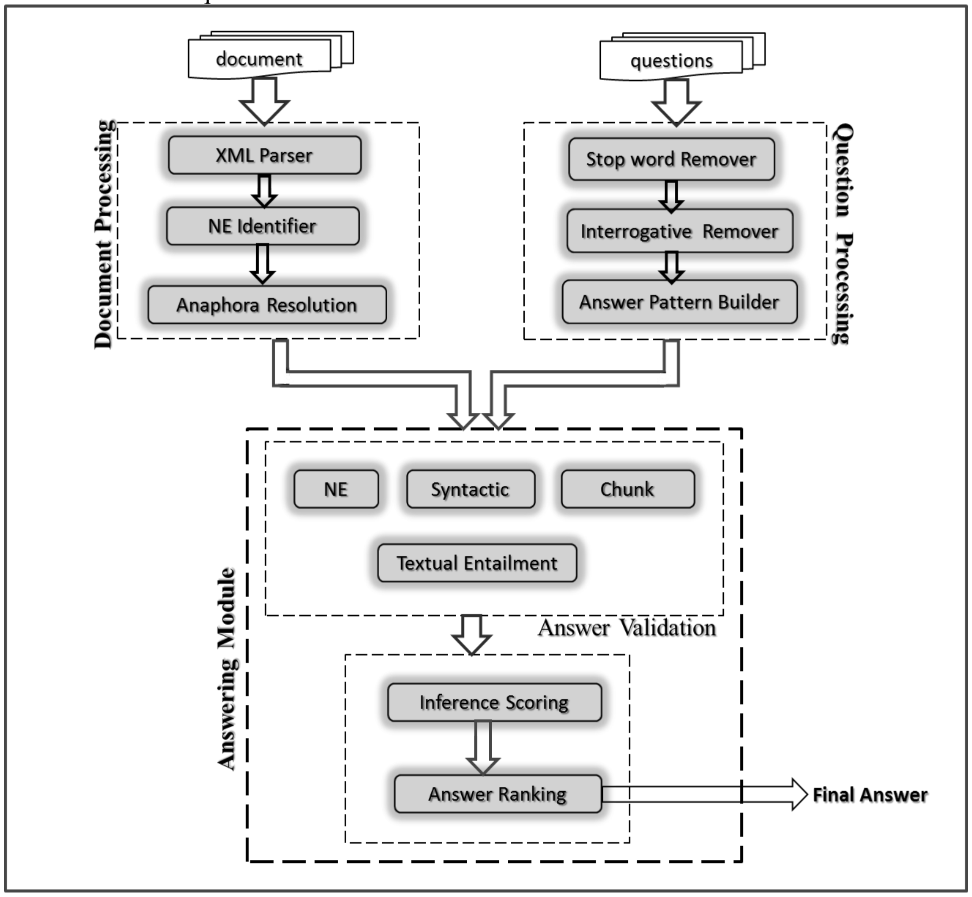

Fig. 1: System Architecture

3.1 Document Processing Module

Document processing module consists of three sub-modules: XML Parser, Named Entity (NE) Identification and Anaphora Resolution.

XML parser. The given XML corpus has been parsed using XML parser. The XML parser extracts the document and associated questions. After parsing, the documents and the associated questions are extracted from the given XML documents and stored in the system.

Named Entity (NE) Identification. For each question, system must identify the correct answer among the proposed alternative answer options. Each generated answer pattern corresponding to a question is compared with each sentence in the document to assign an inference score. The score assignment module requires that the named entities in each sentence and in each answer pattern are identified. The CRF-based Stanford Named Entity Tagger[1] (NE Tagger) has been used to identify and mark the named entities in the documents and queries. The tagged documents and queries are passed to the lexical inference sub-module.

Anaphora Resolution. It has been observed that resolving the anaphors in the sentences in the documents improves the inference score of the sentence with respect to each associated answer option. To resolve the anaphora BART[2] (Beautiful Anaphora Resolution Toolkit) has been used in the present task. BART performs automatic co-reference resolution, including all necessary preprocessing steps.

3.2 Question Processing Module

This module responsible for deciding answers or not answers to a question. We do not answer comparative questions e.g., "*How was Mary different from Susan?*". Table-1 shows the question types that we have answered. At first, stop words and interrogatives have been removed from question text to build query terms (QT). Then, an answer pattern is built by (*QT*, OPTION_T) pair; where OPTION_T refers the T-th answer option. Each answer pattern is considered as a hypothesis H. So, five hypotheses have been built for each question.

[1] http://nlp.stanford.edu/ner/index.shtml
[2] http://www.bart-coref.org/

Table 1. Answered Question Type.

Word/Phrase	Question Type
What happened	FACT
What did	REASON
What	OBJECT
How	WAY/MANNER
Why	REASON

3.3 Answering Module

Answering module is responsible for calculating the score of the each answer option.

Answer Validation. The corpus is in XML format. All the XML test data has been parsed before indexing using our XML Parser. The XML Parser extracts the sentences from the document. After parsing the documents, they are indexed using Lucene, an open source full text search tool.

Query Word Identification and Sentence Retrieval. After indexing has been done, the queries have to be processed to retrieve relevant sentences from the associated documents. Each answer pattern or query is processed to identify the query words for submission to Lucene. Each hypothesis has been submitted to Lucene after removing *stop words* (using the stop word list[3]). The remaining words are identified as the query words. Query words may appear in inflected forms in the question. For English, standard Porter Stemming algorithm[4] has been used to stem the query words. After searching using Lucene, a set of sentences in ranked order are retrieved.

First of all, all query words are fired with AND operator. If at least one sentence is retrieved using the query with AND operator then the query is removed from the query list and need not be searched again. The rest of the queries are fired again with OR operator. OR searching retrieves at least one sentence for each query. Now, the top ranked relevant ten sentences for each query are considered for further processing In case of AND search only the top ranked sentence is considered. Sentence retrieval is the most crucial part of this system. We take only the top ranked relevant sentences assuming that these are the most relevant sentences in the associated document for the question from which the query has been generated.

Each retrieved sentence is considered as the Text (T) and is paired with each generated hypothesis (H). Each T-H pair identified for each answer option corresponding to a

[3] http://members.unine.ch/jacques.savoy/clef/
[4] http://tartarus.org/~martin/PorterStemmer/java.txt

question is now assigned a score based on the NER module, Textual Entailment module, Chunking module, Syntactic Similarity module and Question Type module.

NER Module, It is based on the detection and matching of Named Entities (NEs) [9] in the Retrieved Sentence (T) - generated Hypothesis (H) pair. Once the NEs of the hypothesis and the text have been detected, the next step is to determine the number of NEs in the hypothesis that match in the corresponding retrieved sentence. The measure NE_Match is defined as NE_Match = number of common NEs between T and H/Number of NEs in Hypothesis.

If the value of NE_Match is 1, i.e., 100% of the NEs in the hypothesis match in the text, then the T-H pair is considered as an entailment. The T-H pair is assigned the value "1", otherwise, the pair is assigned the value "0".

Textual Entailment Module (TE), This TE module [8] is based on three types of matching, i.e., WordNet based Unigram Match and Bigram Match and Skip-bigram Match.
a) WordNet based Unigram Match: In this method, the various unigrams in the hypothesis for each Retrieved Sentence (T) - generated Hypothesis (H) pair are checked for their presence in the retrieved text. WordNet synsets are identified for each of the unmatched unigrams in the hypothesis. If any synset for the H unigram match with any synset of a word in the T then the hypothesis unigram is considered as a successful WordNet based unigram match. If the value of Wordnet_Unigram_Match is 0.75 or more, i.e., 75% or more unigrams in the H match either directly or through WordNet synonyms, then the T-H pair is considered as an entailment. The T-H pair is then assigned the value "1", otherwise, the pair is assigned the value "0".
b) Bigram Match**:** Each bigram in the hypothesis is searched for a match in the corresponding text part. The measure Bigram_Match is calculated as the fraction of the hypothesis bigrams that match in the corresponding text, i.e., Bigram_Match=(Total number of matched bigrams in a T-H pair /Number of hypothesis bigrams). If the value of Bigram_Match is 0.5 or more, i.e., 50% or more bigrams in the H match in the corresponding T, then the T-H pair is considered as an entailment. The T-H pair is then assigned the value "1", otherwise, the pair is assigned the value "0".
c) Skip-grams: A skip-gram is any combination of n words in the order as they appear in a sentence, allowing arbitrary gaps. In the present work, only 1-skip-bigrams are considered where 1-skip-bigrams are bigrams with one word gap between two words in a sentence. The measure 1-skip_bigram_Match is defined as
1_skip_bigram_Match = skip_gram(T,H) / n,
where skip_gram(T,H) refers to the number of common 1-skip-bigrams (pair of words in order with one word gap) found in T and H and n is the number of 1-skip-bigrams in the hypothesis H. If the value of 1_skip_bigram_Match is 0.5 or more, then the T-H pair is considered as an entailment. The text-hypothesis pair is then assigned the value "1", otherwise, the pair is assigned the value "0".

Chunk Module, The question sentences are pre-processed using Stanford dependency parser. The words along with their part of speech (POS) information are passed through a Conditional Random Field (CRF) based chunker [11] to extract phrase level chunks of the questions. A rule-based module is developed to identify the chunk boundaries. The question-retrieved text pairs that achieve the maximum weight are identified and the corresponding answers are tagged as “1”. The question-retrieved text pair that receives a zero weight is tagged as “0”.

Syntactic Similarity Module, This module is based on the Stanford dependency parser [9], which normalizes data from the corpus of text and hypothesis pairs, accomplishes the dependency analysis and creates appropriate structures.

Matching Module. After dependency relations are identified for both the retrieved sentence and the hypothesis in each pair, the hypothesis relations are compared with the retrieved text relations. The different features that are compared are noted below. In all the comparisons, a matching score of 1 is considered when the complete dependency relations along with all of its arguments match in both the retrieved sentence and the hypothesis. In case of a partial match for a dependency relation, a *matching score* of 0.5 is assumed.

a. Subject-Verb Comparison: The system compares hypothesis subject and verb with retrieved sentence subject and verb that are identified through the *nsubj* and *nsubjpass* dependency relations. A matching score of 1 is assigned in case of a complete match. Otherwise, the system considers the following matching process.
b. WordNet Based Subject-Verb Comparison: If the corresponding hypothesis and sentence subjects do match in the subject-verb comparison, but the verbs do not match, then the WordNet distance between the hypothesis and the sentence is compared. If the value of the WordNet distance is less than 0.5, indicating a closeness of the corresponding verbs, then a match is considered and a *matching score* of 0.5 is assigned. Otherwise, the subject-subject comparison process is applied.
c. Subject-Subject Comparison: The system compares hypothesis subject with sentence subject. If a match is found, a score of 0.5 is assigned to the match.
d. Object-Verb Comparison. The system compares hypothesis object and verb with retrieved sentence object and verb that are identified through dobj dependency relation. In case of a match, a *matching score* of 0.5 is assigned.
e. WordNet Based Object-Verb Comparison: The system compares hypothesis object with text object. If a match is found then the verb corresponding to the hypothesis object with retrieved sentence object's verb is compared. If the two verbs do not match then the WordNet distance between the two verbs is calculated. If the value of WordNet distance is below 0.5 then a *matching score* of 0.5 is assigned.
f. Cross Subject-Object Comparison: The system compares hypothesis subject and verb with retrieved sentence object and verb or hypothesis object and verb with retrieved sentence subject and verb. In case of a match, a *matching score* of 0.5 is assigned.

g. Number Comparison: The system compares numbers along with units in the hypothesis with similar numbers along with units in the retrieved sentence. Units are first compared and if they match then the corresponding numbers are compared. In case of a match, a *matching score* of 1 is assigned.
h. Noun Comparison: The system compares hypothesis noun words with retrieved sentence noun words that are identified through *nn* dependency relation. In case of a match, a matching score of 1 is assigned.
i. Prepositional Phrase Comparison: The system compares the prepositional dependency relations in the hypothesis with the corresponding relations in the retrieved sentence and then checks for the noun words that are arguments of the relation. In case of a match, a *matching score* of 1 is assigned.
j. Determiner Comparison: The system compares the determiner in the hypothesis and in the retrieved sentence that are identified through *det* relation. In case of a match, a *matching score* of 1 is assigned.
k. Other relation Comparison: Besides the above relations that are compared, all other remaining relations are compared verbatim in the hypothesis and in the retrieved sentence. In case of a match, a *matching score* of 1 is assigned.

API for WordNet Searching RiWordnet[5] provides Java applications with the ability to retrieve data from the WordNet database.

Each of the matches through the above comparisons is assigned some weight.

Inference Score Module. In this module, we have got the weight from Named Entity Recognition (NER) Module, Textual Entailment (TE) Module, Question Type Analysis Module, Chunk Boundary and Syntactic Similarity Module.
Each sentence in the associated document is assigned an inference score with respect to each (QT, OPTION_T) pair.

Answer Pattern Generation for Inference Score. Each question has five answer options and the task is to identify the best answer to the question from an associated document. Each question in the system is identified as the (question, document) pair represented as {q_i, d_id} where i=1…5. There are 5 questions corresponding to each document. Each answer option is represented in the system as {d_id, q_id_i, a_id_j}, where, d_id=document id, q_id_i= i th query, where i=1…5, a_id_j= j th answer option, where j=1…5.

Each query frame is defined in the system as (DOC, QT, OPTION_T) where,

DOC= Give Document to be used for verifying answer options
QT= Query Term, is a list of words after removing the stop words and interrogative word from the given question.
OPTION_T= T-th answer option

5 http://www.rednoise.org/rita/wordnet/documentation/index.htm

Scoring Assignment. This module takes query frame as input and returns score as output. The algorithm *InferenceScore* describes the scoring procedure.

Table 2. Algorithm InferenceScore (Sentence, QT, $OPTION_T$)

```
Algorithm InferenceScore (sentence, QT, $OPTION_T$)
Step 1: [Initialization]
            score = 0
            keywordmatched = 0 // count no of  matched keyword
Step 2: [Check whether (QT, $OPTION_T$) matches in a sentence]
            If (QT, $OPTION_T$) matches in a sentence then
               Score = 1
               goto step 5
Step 3: [Check each keyword in $OPTION_T$]
            For each keyword in $OPTION_T$
                 If keyword matches in a sentence then
                      Score = score + 1 / (number of keywords -1)
                      Keywordmatched = keywordmatched + 1
Step 4: [Check whether all the keywords have matched]
            If (keywordmatched = = total keywords – 1) then
               Score =1
Step 5: Return score
End
```

Answer Ranking Module. Now, for each given answer option a score is calculated and the answer option with highest score is taken as correct answer for the given query. The algorithm *RankigAnswerOption* describes the option selection procedure.

Table 3. Algorithm SelectAnswerOption (Answer Set)

```
Algorithm SelectAnswerOption(answer set)
Step 1: [Initialization]
  correct_option= ∞ // not answered
Step 2: [Calculate score for each sentence]
  For each sentence $S_i$ € Sentences and answer option $q_j$€ Q
  Where, j=1…5
     $A_{ji}$=AnswerScore($S_i$, QT, OPTION)
   End For
Step 3: [Assign score to each option]
    For answer pattern $q_j$€ Q
       $AQ_i$=maximum evaluated score for {$S_1$,$S_2$,….$S_n$};
       Where $AQ_i$ is the score of i^th option
    End For
Step 4: [ Applying Matching Score($M_{score}$)]
    For each answer option $AQ_j$€ AQ
      $AQ_j$ =InferenceScore($AQ_j$) + $M_{score}$
    End For
Step 5: [Select the answer option]
    correct_option= index of maximum AQ={ $AQ_1$, $AQ_2$, $AQ_3$, $AQ_4$, $AQ_5$ }
END
```

4 Evaluation

The main measure used in this evaluation campaign is c@1, which is defined in equation 1.

$$c@1 = \frac{1}{n}(n_R + n_U \frac{n_R}{n}) \quad (1)$$

where, n_R: the number of correctly answered questions,
n_U: number of unanswered questions
n: the total number of questions

Afterwards, these c@1 scores can be aggregated at topic and global levels in order to obtain the following values:

- Median, average and standard deviation of c@1 scores at test level, grouped by topic,
- Overall median, average and standard deviation of c@1 values at test level.

The median c@1 has been provided under the consideration that it can be more informative at reading level than average values. This is because median is less affected by outliers than average, and therefore, it offers more information about the ability of a system to understand a text.

This approach allows us to evaluate systems in a similar way to the manner new language learners are graded.

Evaluation at question-answering level:

Total QUESTIONs: 46
- Number of questions ANSWERED: 23
- Number of questions UNANSWERED: 23
- Number of questions ANSWERED with RIGHT candidate answer: 13
- Number of questions ANSWERED with WRONG candidate answer: 10
- Number of questions UNANSWERED with RIGHT candidate answer: 0
- Number of questions UNANSWERED with WRONG candidate answer: 0
- Number of questions UNANSWERED with EMPTY candidate: 23

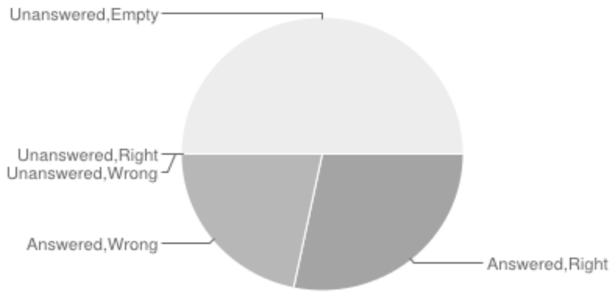

Fig. 3: Pie Chart Representation of Evaluation at QA level

Accuracy (answered with judgment=correct) calculated over all questions:
Overall *accuracy* = 13/46 = 0.28

Proportion of answers correctly discarded: 0/23 = 0.00

Overall *c@1 measure* = (13+23(13/46))/46 = 0.42

Overall c@1 per topic:
c@1 topic t_id '1' = (13+23(13/46))/46 = 0.42

Evaluation at reading-test level:

Median: 0.00 - *Average:* 0.23 - *Standard Deviation:* 0.31 *-calculated over c@1 of all 9 reading tests*
Topict_id='1'-EntranceExams
Median: 0.28 - *Average:* 0.40 - *Standard Deviation:* 0.31 *-calculated over the c@1 of the four reading tests*

- c@1 measure for reading-test r_id '1' = (0+4(0/5))/5 = 0.00
- c@1 measure for reading-test r_id '2' = (1+3(1/6))/6 = 0.25
- c@1 measure for reading-test r_id '3' = (1+1(1/5))/5 = 0.24
- c@1 measure for reading-test r_id '4' = (3+1(3/5))/5 = 0.72
- c@1 measure for reading-test r_id '5' = (1+2(1/5))/5 = 0.28
- c@1 measure for reading-test r_id '6' = (2+3(2/5))/5 = 0.64
- c@1 measure for reading-test r_id '7' = (0+4(0/5))/5 = 0.00
- c@1 measure for reading-test r_id '8' = (2+3(2/5))/5 = 0.64
- c@1 measure for reading-test r_id '9' = (3+2(3/5))/5 = 0.84

5 Conclusion

The question answering system has been developed as part of the participation in the QA4MRE pilot track as part of the CLEF 2013 evaluation campaign. The overall system has been evaluated using the evaluation metrics provided as part of the QA4MRE 2013 pilot track. It has been observed from evaluation results that our proposed model works very well on the reading test – 4,6,8,9**.** And the system performs very poor to handle reading test- 1,7. As the questions of type comparative have not been answered, it affects the evaluation results. But, the overall evaluation results are satisfactory. Future works will be motivated towards improving the performance of the system

Acknowledgements. We acknowledge the support of the DST India-CONACYT Mexico project "Answer Validation through Textual Entailment" funded by DST, Government of India and the Department of Electronics and Information Technology (DeitY), Ministry of Communications & Information Technology (MCIT), Government of India funded project "Development of Cross Lingual Information Access (CLIA) System Phase II".

References

1. Anselmo Peñas, Pamela Forner, Richard Sutcliffe, Álvaro Rodrigo, Corina Forăscu, Iñaki Alegria, Danilo Giampiccolo, Nicolas Moreau, Petya Osenova.: Overview of ResPubliQA 2009: Question Answering Evaluation over European Legislation. In Working Notes for the CLEF 2009 Workshop, 30 September-2 October, 2009, Corfu, Greece.
2. Anselmo Peñas, Pamela Forner, Álvaro Rodrigo, Richard Sutcliffe, Corina Forăscu and Cristina Mota.: Overview of ResPubliQA 2010: Question Answering Evaluation over European Legislation. In Working Notes for the CLEF 2010 Workshop, Padua, Italy, 20-23 September 2010.
3. Anselmo Peñas, Eduard Hovy, Pamela Forner, Álvaro Rodrigo, Richard Sutcliffe, Corina Forascu, Caroline Sporleder. Overview of QA4MRE at CLEF 2011: Question Answering for Machine Reading Evaluation, Working Notes of CLEF 2011. (2011)
4. Peñas, A.,Rodrigo, Á. , Sama, V., Verdejo, F.: Overview of the answer validation exercise 2006. Working Notes of CLEF 2006. (2006)
5. Peñas, A., Rodrigo, Á, Verdejo, F.: Overview of the Answer Validation Exercise 2007. Working Notes of CLEF 2007. (2007)
6. Rodrigo, Á., Peñas, A., Verdejo, F.: Overview of the answer validation exercise 2008. Working Notes of CLEF 2008. (2008).
7. Partha Pakray, Pinaki Bhaskar, Santanu Pal, Dipankar Das, Sivaji Bandyopadhyay and Alexander Gelbukh: JU_CSE_TE: System Description QA@CLEF 2010 – ResPubliQA. CLEF 2010 Workshop on Multiple Language Question Answering (MLQA 2010).
8. Pakray, P., Gelbukh, A., Bandyopadhyay, S.: Answer Validation using Textual Entailment. 12th CICLing, Lecture Notes in Computer Science, 2011, Volume 6609/2011, 353-364, DOI: 10.1007/978-3-642-19437-5_29. (2011)
9. E. Briscoe, J. Carroll, and R. Watson.: The Second Release of the RASP System. In Proceedings of the COLING/ACL 2006 Interactive Presentation Sessions.
10. Marie-Catherine de Marneffe, Bill MacCartney, and Christopher D. Manning.: Generating Typed Dependency Parses from Phrase Structure Parses. In 5th International Conference on Language Resources and Evaluation (LREC) (2006)
11. Xuan-Hieu Phan.: CRFChunker: CRF English Phrase Chunker. PACLIC 2006. (2006)
12. P. Pakray, P. Bhaskar, S. Banerjee, B. Pal, A. Gelbukh, S. Bandyopadhyay: A Hybrid Question Answering System based on Information Retrieval and Answer Validation, In: the proceedings of Question Answering for Machine Reading Evaluation (QA4MRE) at CLEF 2011, Amsterdam. (2011)
13. P. Bhaskar, P. Pakray, S. Banerjee, S. Banerjee, S. Bandyopadhyay, A. Gelbukh,: Question Answering System for QA4MRE@CLEF2012. In: the proceedings of Question Answering for Machine Reading Evaluation (QA4MRE) at CLEF 2012, Rome, Italy. (2012)