AutoBET: towards automatic answering of BET questions for meeting browser evaluation

Quoc Anh Le[1,2], Andrei Popescu-Belis[1]
[1] University of Namur
[2] Idiap Research Institute

Objectives

- Design an automatic BET question answering tool: AutoBET
- Use the AutoBET to:
  1. compare with BET scores of humans and assess their performance;
  2. build an assistant that help humans answer BET questions;
  3. assess the feasibility of a fully-automatic question-answering system for meeting transcripts.

The Browser Evaluation Test: BET

- Method for collecting browser evaluation questions
  - human observers create pairs of T/F statements
  - about 500 “questions” for 3 meetings + inter-scorer agreement
- Method for browser evaluation
  - human subjects answer questions using a browser
  - average performance: speed and precision

AutoBET Procedure

1st question

1
Here
Denis was interested in whether participants have received an agenda for the meeting
---
2
Around
Mirek had not received the agenda for the meeting

……

2nd question

1
Here
Agnes was interested in whether participants have received an agenda for the meeting
---
2
Around
Andrei had not received the agenda for the meeting

……

Meeting Transcript

Start time	End time	Speaker	Dialogue
[0:48]	[0:49]	Andrei	Hi everyone.
[0:49]	[0:56]	Denis	So I don’t know if you all received the the a- agenda for this meeting.
[0:49]	[0:56]	Denis	Do you – no?
[0:55]	[0:56]	Mirek	No, I haven’t.
[0:57]	[0:57]	Denis	Here it is.
[0:57]	[0:57]	Mirek	Thank you.
……			

Pre-processing

1. Remove characters not relate to words: *( ) ^ ! - …*
2. Convert abbreviated words into full forms: *I'll = I will, haven’t -> have not, …i*
3. Convert numeric form to text: *34 -> thirty four, 2nd -> second, …*
4. Remove stop-words: *a, an, and, are, as, …*

Lemmatizer

1st question

Position	Word	Lemma	Stem
1	Mirek	Mirek	Mirek
2	had	have	have
3	not	not	not
4	received	received/receive	receiv

2nd question

Position	Word	Lemma	Stem
1	Andrei	Andrei	Andrei
2	had	have	have
3	not	not	not
4	received	received/receive	receiv

Words splitting and Indexing

Position	Word	Stem of word	Stem of speaker	Stem of synonyms
14	received	receiv	deni	have get find obtain experi invit welcom meet encount …
18	agenda	agenda	deni	docket schedul agendum …
21	meeting	meet	deni	see converg play receiv contact touch encount suffer …
……				

Calculate score for each search window — Phase 1

Lexical Similarity Algorithm:

```
for each word_question in the question
for each index in the table of transcript
    if ( word_question == speaker) { score = score + 2.0; break; }
    if (word_question == stem) { score = score + 1.0; break; }
    if (word_question belongs to set of synonyms) { score = score + 0.2; break; }
```

Remarks:

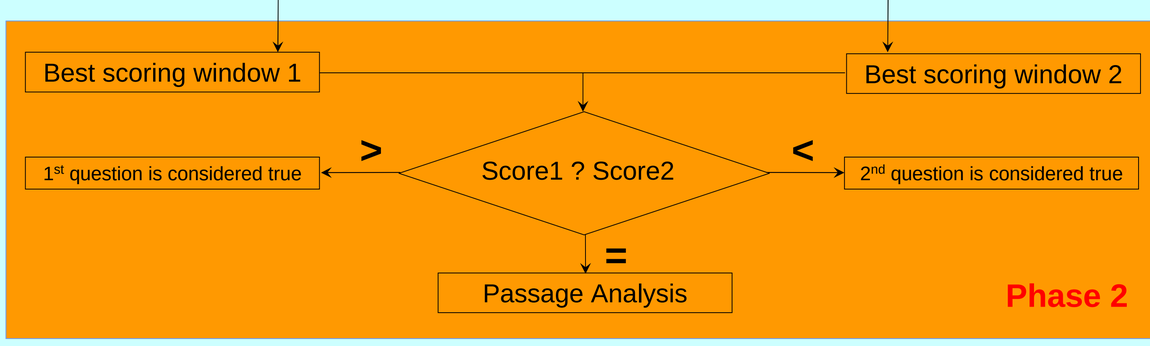
1. The priority order is given to “speaker matching”, then “stem matching” and lastly “synonyms matching” in order to bring the highest score for the search window.
2. The frequency of one found word is not used to increase the score but in the case “multiple matching”, if one word repeats more than one time in both the question and the transcript, the score will be calculated as the minimum number of appearance of this word between the question and the transcript.

Search window size and Search window step

1. Unit of calculation may be sentence or word.
2. Two methods to set the parameters for search window size and search window step: dynamic method depends on the size of the question and the size of transcript and static method is contrary to the dynamic method.
3. For example: Search window size = 5* number of words of the question or search window step = number of words of the question.

Evaluation of retrieved window

- Comparing reference answer (made by hand) and candidate answer (made by algorithm)
- if two answers are overlapped each other, the candidate answer will be considered correct.
- The number of overlapped words is defined as one word or multiple words. If the system is used to help users to answer the questions, one overlapped word will be accepted as well.



Phase 2

Passage Analysis

1. Repeat the lexical similarity algorithm in the phase 1 in which the lexical similarity will be calculated with two or three consecutive words instead of one word.
2. Analyse the context: who is the subject in the question and list of subjects in the retrieved window, talk about what,...
3. Study the way to calculate score for the order of common words as well as the distance of common words

Final result: which is true question in the pair

Results

- Implemented successfully the passage retrieval algorithm
- Tested **window correctness** on two meeting transcripts:
  - (i) IS1008c, 26’, 4000 words, 50 questions ➔ **66% correct (33/50)**
  - (ii) IB4010, 46’, 9500 words, 116 questions ➔ **53% correct (62/116)**
- Chances of guessing randomly the correct window**: < 1% !**

Future work

- Improve the passage retrieval algorithm by analysing and classifying the questions in order to add particular techniques to the algorithm
- Experiment with the algorithm using variable parameters of search window size and search window step
- Design an acceptable algorithm for phase 2, evaluate the two phases

References

1. Popescu-Belis A. and Georgescul M. “TQB: Accessing Multimedia Data Using a Transcript-based Query and Browsing Interface”, Proceedings of LREC 2006, International Conference on Language Resources and Evaluation), Genoa, Italy, p.1560-1565.
2. Popescu-Belis A., Baudrion P., Flynn M. and Wellner P.: “Towards an Objective Test for Meeting Browsers: the BET4TQB Pilot Experiment”, Machine Learning for Multimodal Interaction IV, LNCS 4892, Springer-Verlag, Berlin/Heidelberg, p108-119.
3. Wellner, P., Flynn, M., Tucker, S., Whittaker, S.: “A meeting browser evaluation test” In: CHI 2005 (Conference on Human Factors in Computing Systems), Portland, OR, pp. 2021-2024 (2005).
4. BET resource website: “http://www.idiap.ch/mmm/tools/bet-data”
5. Marius Pasca and Sanda M. Harabagiu: “The Informative Role of WordNet in Open-Domain Question Answering”, The Second Meeting of the North American Chapter of the Association for Computational Linguistics, NAACL 2001, Carnegie Mellon University, Pittsburgh, PA, USA, 2-7 June 2001