Abstract

An automatic true-false question answering system over meeting transcripts was developed using a speaker-directed lexical similarity algorithm that includes n-grams matching and lexical extensions. The main function of this system is to determine the true and false statement in a pair of complementary statements. These statements were created using a well-defined methodology to capture facts related to meetings as the Browser Evaluation Test method, namely BET [1]. Our system is the first attempt at building an automatic meeting browser.

First of all, our system uses a lexical similarity algorithm to locate one passage that is the most likely to contain information about question. For this, all passages are compared with each other using passage scoring. The passage scoring is calculated not only based on the sum of scores of matched words between question and passage but also on the speaker of these words. This technique pays more attention to the features of a conversational document as meeting transcript. Based on two retrieved passages corresponding to two question in the pair, one question is considered to be true if the score of its corresponding passage is higher than that of the other question.

The performance of this system is evaluated by answering approximately two hundred BET questions, which were constructed by independent observers during two meetings of the AMI Meeting Corpus [2]. Experimental results show that around 58% of retrieved passages are correct while the chance of randomly guessing one correct passage is less than 4%. The proportion of correct answers finally achieved is around 61%. This result is better than the result from answering true-false questions by chance whose proportion of correct answers is only 50%. In addition, the performance of the algorithm is also evaluated on transcripts that were generated by Automatic Speech Recognition (ASR) , as well as on meeting summaries based on ASR transcripts. These transcripts are noisier and the proportion of correct answers decreases for passage retrieval.

The last evaluation is performed by comparing BET scores obtained by human subjects with scores obtained by the system over the same BET questions. The BET scores by human subjects were obtained with the Transcript-based Query and Brower Interface (BET4TQB)[3]. A comparative analysis shows that human subjects generally answer questions that require a deduction better than an automatic question answering system does.

In conclusion, this system should be integrated into existing meeting browsers as an assistant tool that helps humans answer such type of questions by locating the relevant passage rather than find the true-false answers.

Keywords: Question Answering, Meeting Browser Evaluation, Passage Retrieval, BET questions, True-False Answering, N-gram Matching, Lexical Similarity.

Acknowledgements

The first person I would like to express my gratitude to is my supervisor, Dr. Andrei Popescu-Belis [4], senior researcher at the Idiap Research Institute in Switzerland. He gradually guided me through my research in general and for this project in particular. I recognize that I have learned a lot from him. During the 6 months internship in Idiap, he always encouraged me, which motivated me and made me enjoy my work.

Secondly, I would like to thank all of professors and teachers at the Faculty of Computer Science, University of Namur (FUNDP) for their interesting and valuable courses, which have equipped me with the background knowledge needed to complete this project. My special thanks should be sent to Prof. Jean-Paul Leclercq [5], my supervisor at FUNDP. He is not only my teacher, but also a good friend who gives me useful advices for my works.

This project is part of the Interactive Multimodal Information Management project (IM2, http://www.im2.ch), funded by the Augmented Multiparty Interaction Training Programme (AMIDA, http://www.amiproject.org). I wish to send my great thanks again to Dr. Andrei Popescu-Belis, Prof. Jean-Paul Leclercq and the Dean of Computer Science Department, University of Namur, Jean-Marie Jacquet [6] for their letter of recommendation so that I was accepted for this project.

In addition, I acknowledge the financial support of the Belgian Development Cooperation Agency (BTC, http://www.btcctb.org) for my 2-year course for Computer Science Masters Degree in Namur, Belgium.

Last but not least, I thank my wife and my son for their understanding and support during my studies.

Contents

List of Figures

List of Tables

List of Algorithms

Chapter 1

Introduction

1.1 Context

Meetings have become more and more essential in the workplace in order to exchange information and to make decisions. It has recently become possible to save meeting information including videos, audio files, transcripts and slides in multimedia archives of meeting recordings so that humans can find relevant information from past meetings, for instance, using tools such as meeting browsers [7]. A meeting browser can be defined as follows: *"A meeting browser is a system that enables a user to navigate around an archive of meetings, efficiently viewing and accessing the full multimodal content, based on automatic annotation, structuring and indexing of those information streams"* [8].

One approach to meeting browsing is to design general-purpose meeting browsers that help users to locate the information that is searched for [9], for instance, the meeting browser named Archivus at the University of Geneva and at the EPFL [10], Ferret in Idiap Research Institute [11], Transcript-based Query and Browsing Interface (TQB) at the University of Geneva [12], etc. However, another possibility is to design browsers that locate information automatically, for instance for verification (fact checking) purposes.

1.2 Goal

The goal of this project is to design an automatic browser following a question-answering approach, and to assess its performance on a set of pairs of true-false statements, which have been initially used to evaluate human-directed browsers.

In other words, the goal is to design and to implement a system that determines automatically the true and the false statement in each pair based on searching facts on meeting transcripts, to evaluate its performance over a set of about two hundreds such pairs (over two recorded meetings), and to compare it with human subjects who used meeting browsers. A comparative analysis of the system and the human scores on specific questions should indicate whether or not system and humans have the same difficulties answering such questions. This work will thus show whether such a system should be developed as a full automatic browser that gives an exact answer for user's question or only help users locate relevant information in meeting recordings (thus, functioning as an assistant tool).

The pairs of true-false statements used for this system were created using a well-defined methodology to select facts related to a meeting for the Browser Evaluation Test method, namely the BET question [1]. The BET method and the BET question are presented in detail in the section 3.1.

1.3 Approach

The proposed system is developed with a number of specific techniques that are suitable for the nature of the data and of the task. It proceeds in three stages as follows:

The first stage is the pre-processing of the pair of BET questions and of the meeting transcript for the purpose of transforming them into an uniform data.

Then the second stage aims at identifying separately the passage of the transcript that is most likely to contain the answer for each question in a pair using a lexical similarity algorithm. For this, all passages in the transcript are compared with each other using passage scoring, which is a sum of scores of matched words between the passage and a question. Regarding the matched word score, it is computed based on a complex score of lexical similarity, which is not only based on matched words but also on the speaker of these words. This technique pays more attention to the features of a conversational document as meeting transcripts.

Finally, the third stage compares two BET statements in the pair based on the passage found for each question and hypothesizes that one statement is true if the score of its corresponding passage is higher than that of the other one. In case that they have the same scores, the distance entre matched words for each question as well as the proportion of the length of each question to its corresponding passage length are used to give the answer.

1.4 Evaluation methods

The performance of this system is evaluated by answering nearly two hundred BET questions related to two meeting transcripts named IB4010 and IS1008c (see Chapter 3 for more details) from the AMI Meeting Corpus [2]. Furthermore, the performance of the algorithm is also evaluated on ASR transcripts which were generated by Automatic Speech Recognition [13] as well as meeting summaries based on ASR transcripts.

The last evaluation is performed by comparing BET scores by human subjects, which are from the BET method for Transcript-based Query and Brower Interface (BET4TQB)[3], with scores obtainned by the system over the same BET questions and the transcripts. This task is to answer the question whether the human subjects and the system have the same difficulties to answer such questions.

Based on these results, the evaluation should help to provide information as to whether the system should be developed as a full automated system or an assistant tool that helps humans answer questions concerning meeting information using meeting transcripts.

1.5 Structure of the report

This report contains 8 chapters. The first chapter is an introduction while the rest of the report provides detailed information. Chapter 2 reviews a number of available approaches that are widely applied in many question answering systems. Chapter 3 presents a brief description of the BET method as well as data used to test this system. Chapter 4 consists of three sections that describe the three main stages of our approach as mentioned above. Chapter 5 describes an evaluation method using reference answers in order to assess answers returned by the algorithm. Chapter 6 presents experiments on both manual and automatic (ASR) meeting transcripts. Moreover, at the end of chapter 6, we conduct a comparison and an evaluation of two specific areas: (i) BET human results are compared with those of the system in order to show limitations of an automatic answering as well as difficulties for both human and machine; (ii) the system and its questions are used to measure the quality of Automatic Speech Recognition (ASR) summaries. Chapter 7 suggests specific directions for future research. Finally, the last chapter provides the main conclusions.

Chapter 2

Related Work

A question answering system allows users to ask a question in natural language and receive an exact and succinct answer in place of a list of documents that contain the answer [14, 15]. Since the first article that addressed a textual question answering system by computer was presented by Simmons (1965)[16], many systems have been developed and some of the approaches have been widely used in a number of applications, for instance Okapi BM2 [17]. A typical question answering system is showed in the Figure 2.1.

In a general question answering system, there are four major components [15, 18–20]:

- Question analysis: There are two tasks in this component. First, questions in natural language asked by a user need to be converted into queries that are needed by the subsequent parts of the system. The queries created from user's question contain terms likely to appear in documents containning an answer, for instance for such question as *What is the capital of Vietnam?*, the corresponding query is *capital + vietnam*. Secondly, the expected answer type for this question is detected in this stage so that it helps to narrow the space of searched answers. For example, questions with "When" always relate to *time*, thus those terms concerning time are expected such as *date*, *hour*, etc.

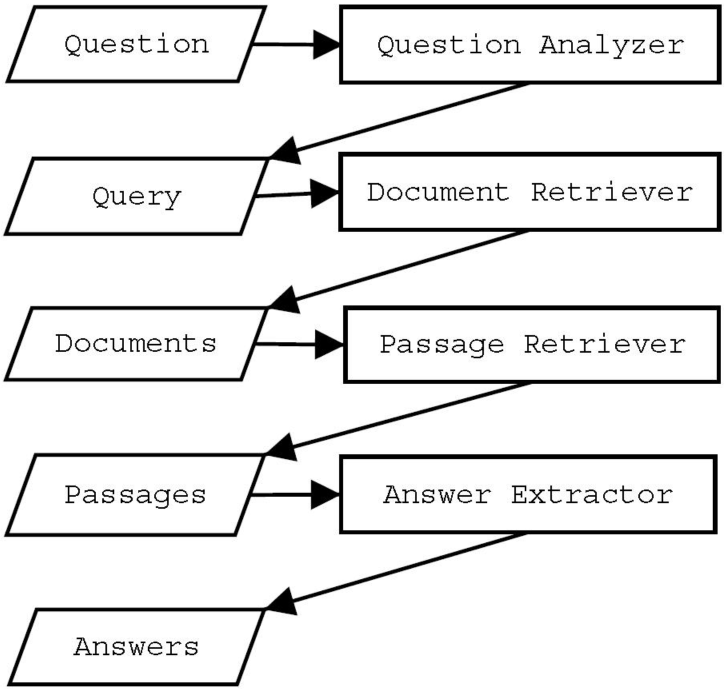


Figure 2.1: A typical Question Answering Architecture

.

- Document retrieval: This task is to retrieve documents from the corpus that may be taken from Internet by a search engine or archived documents likely to contain answers to the query.

- Passage retrieval: A passage can simply be defined as a sequence of words regardless of individual sentences or paragraphs. Some text-based information retrieval systems define a passage as a fixed-length block of words. [21]. Passage retrieval algorithms take a document and a question and try to return a list of passages from the document that are most likely to contain the answer information. For this, all passage are compared with each other using passage scoring. In text-based question answering system, the score of a passage is based on the score of its words with respect to question words. The score of a question word found in a passage is computed based on definition of this word and/or relations of this word with other words in the text [19].

- Answer extraction: Based on the question analysis and retrieved passages, the system extracts phrase/phrases representing an answer.

We are interested only in the passage retrieval stage and not in the document retrieval, the question analysis nor in the answer extraction. It is because we have already documents as meeting transcripts and the type of questions as the BET questions that are statements to be assessed. In our case, the type of answers is simply one bit ”true” or ”false” for each question. We do not need an answer as a short text extracted from the retrieved passage in which we will still have to compare two answers in order to determine the true statement. Moreover, the answer extraction is a difficult task and it can not avoid some errors. We would rather use the full retrieved passages in determining the true statement. Thus, the stage of answer extraction is not suitable to develop in our system. For this reason, we will present only state-of-the-art methods related to passage retrieval including both traditional methods and modern methods.

Most passage retrieval algorithms calculate passage scoring based on words from the passage that are found in the question, namely matched words. However, the method of computing matched word score is different for each algorithm.

The simplest algorithm for this approach was proposed by Light[22], in which a passage score function counts the number of words from the question found in the passage as the score for this passage. That means all words are treated equally and are given the same importance. Many question answering systems use this method as a baseline score to evaluate their performance.

To date, there are many passage retrieval methods that have been presented in the Text Retrieval Conference (TREC) http://trec.nist.gov. These methods can be classified into two groups. One group includes traditional methods and assigns scores to each matched words independently.

That means there are no relations between two matched words. Another group considers relations amongst matched words to assign scores to them.

For the first group, typical approaches use parts-of-speech and/or frequencies of a word in order to calculate a score. Meanwhile for the second group, dependency relation among matched words in a phrase is computed to give a score for this phrase instead of words. This makes it more suitable to capture semantic properties than the first group.

Take SiteQ's passage retrieval algorithm [23] as an example. In this system a passage consists of some consecutive sentences segmented by punctuation and passage score is calculated by summing the weights of individual sentence in the passages. Each sentence is given a score by a formula that combines both the parts-of-speech method and the query term density method. The weight of the matched words is assigned as follows: A proper noun (a common noun recognized by a capital letter) has a higher score than a verb, an adjective and/or an adverb. The term density is defined as the distance among matched words. If two sentences have the same number of matched words the sentence with smaller distance will receive a higher score.

The main idea of using word frequencies is that if a word appears many times in a current passage but only a few in other passages, its score in the current passage is higher [24]. That means the importance of a word increases proportionally to the number of times this word is found in a passage, but inversely to the number of times this word is found in the overall document. The simplest method for this idea is the *term frequency inverse document frequency* $tf \times idf$. In which, the *tf* is the term frequency that measures the important of the term t_i within the document d_i and *idf* is the inverse document frequency that measures the important of the term t_i in the whole collection of documents. The Okapi BM25 [17, 19, 25–27] presents state-of-the-art method for assigning weights to matched words using word frequencies. In fact, Okapi BM25 is a ranking function that is used to rank matching documents according to their relevance to a given query. Thus, it is often used in the Document Retrieval stage of question answering systems. However, according to the Okapi BM25 presented in the TREC-4 [17], this function is also used for passage determination and searching. In addition, it is a complex function which was developed from the function of term frequency-inverse document frequency *tf*idf* [24].

More approaches in the second group consider dependency relation among matched words, in which n-gram matching is the simplest case. The n-gram method pays more attention to the order of matching words. Accordingly, those in order are better. More specifically, this method is used to estimate similarity between two strings by examining all n-word substring matching instead of word matching [28]. Another simple method is to use word density presented in the SiteQ's algorithm above. A more complex method for these approaches presented by Cui [29] uses a dependency tree to assign scores to sentences. Given the reason that one sentence in English can be written in different ways by exchanging position of words in the sentence without changing its meaning. For instance, with the sentence *John wrote a science fiction book*, it may be written in some different ways but the meaning will remain unchanged. The different ways of writing the sentence include: 1) *A science fiction book was written by John*; 2) *John wrote a book of science fiction*; 3) *A book of science fiction was written by John*; and 4) *John wrote a science fiction book.* These sentences can build a dependency tree that represents correctly positioned word relations in the sentences, so that a given query will be compared with this tree instead of one initial sentence.

In order to enhance the performance of question answering systems, lexical extensions for queries are added such as stemming, lemma, synonyms [22], [18],[20], [23].

Chapter 3

Data description

Our system is designed for data with a specific format and type so that the proposed algorithm can make use of them . In this chapter, two meeting transcripts and questions for these two meetings will be described. An analysis of this data, which also give the reasons to build our algorithm, is also presented.

The questions used to test the system were created by the BET method. Thus, in order to have an overall view of this project as well as to understand features of the questions, we present a brief description of the BET method and the BET question in the first of the chapter.

3.1 The BET method

One method proposed originally by Flynn, M. and Wellner, P. [30] is the Browser Evaluation Test (BET). This method evaluates a meeting browser based on user performance rather than subjective judgment. According to the BET, the act of browsing a meeting recording is to an attempt to find a maximum number of *observations of interest* in a minimum amount of time [1], in which *observations of interest* is defined as interesting to the meeting participants or to people who missed the meeting. Thus, the task of evaluating of a meeting browser is to collect a set of *observations of interest* and then ask human subjects to verify these observations as binary-choice test questions in a fixed amount of time by using a meeting browser to access the meeting. A *good* meeting browser will help human subjects find as many correct answers as possible in a shortest amount of time possible. Information on human subjects is described as answer precision (known as *effectiveness*) and answer speed (known as *efficiency*) which is used to evaluate the performance of the meeting browser. In detail, an *observation of interest* is formed as a complementary pair of statements, one true and one false about a fact related to a meeting recording and human subjects are asked to determine which statement is true and which statement is false in the pair. The answer precision is calculated by dividing the number of correct answers over total answers. In terms of the answer speed, it is computed as the average time required to answer a question. Analyzing this information and comparing it among different meeting browsers provides scores for the performance of a meeting browser.

This is a time-consuming method that requires investment in collecting, preparing the observation and performing experiments with subjects. However, this observation collection is independent to browsers so that the observations collected can be extended to be used for the evaluation of other meeting browsers in the future. [31].

The stages of the BET method is presented in the Figure 3.1.

3.2 The BET Questions

The pairs of statements used for the BET method, called the BET questions, are produced by a set of neutral observers, who independently watch selected meetings from corpus. These observers are native English speakers from the University of Sheffield. They are students, researchers and lecturers. The

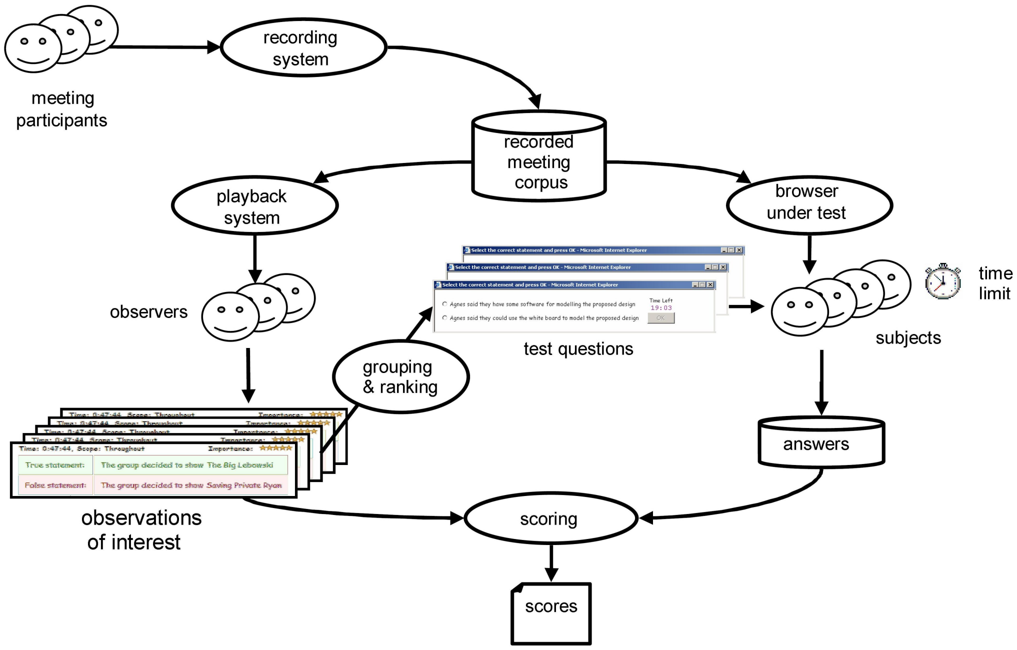
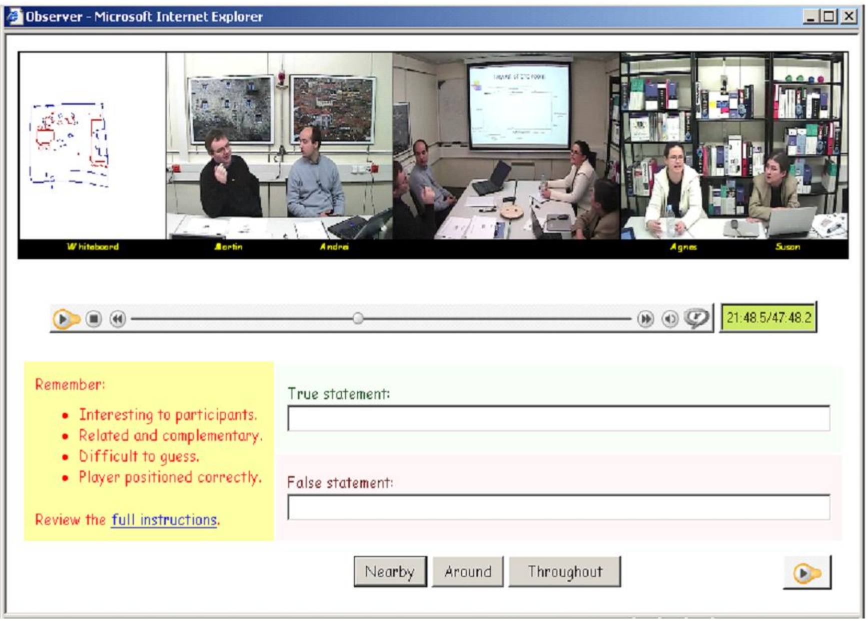

meeting
participants
recording
system
recorded
meeting
corpus
playback
system
browser
under test
observers
Select the correct statement and press OK - Microsoft Internet Explorer
○ Agnes said they have some software for modelling the proposed design
○ Agnes said they could use the white board to model the proposed design
Time Left
19:03
OK
test questions
time
limit
subjects
grouping
& ranking
Time: 0:47:44, Scope: Throughout
Importance: ★★★★★
True statement:
The group decided to show The Big Lebowski
False statement:
The group decided to show Saving Private Ryan
answers
observations
of interest
scoring
scores

Figure 3.1: Stages in the design and execution of a BET evaluation [7]

observers have unlimited time and access to the full recordings from such media sources as videos, audios, in parallel with paper printouts of the slides that participants worked on for the meeting. At first, an observer collects a list of observations that are true statements about facts or events that may interest meeting participants or people who missed the meeting. The collected statements should not be easy to guess without using the meeting information. Then, for each true statement, a false counterpart statement is created so that a pair of complementary statements is generated. The observations should be simple and concisely stated.

An interface for observation collection is presented in the Figure 3.2. As seen in the Figure, there are three buttons ”Nearby”, ”Around” and ”Throughout” that indicate the position of information required to answer the question in the transcript. One observation is marked as *Nearby* or *Here* if it is pertinent to that particular moment; marked as *Around* if it covers at least a minute of the meeting around the point the observer have selected; and marked as *Throughout* if it broadly covers the whole meeting. However, in this system, the questions whose type is *Throughout* are avoided because it is difficult to determine relevant passages which contain information required to answer the questions using an automatic system. After that, the collected observations are examined by experts to reject repeated or inappropriate ones.



Remember:

- Interesting to participants.
- Related and complementary.
- Difficult to guess.
- Player positioned correctly.

Review the full instructions.

True statement:

False statement:

Figure 3.2: Interface for observers

3.3 Data used to test the system

Meeting transcripts

Two meeting transcripts used to test the system were taken from the corpus built by the AMI project [2]. Both of them are in English, involving four participants, native or non-native English speakers. The first meeting, IB4010, lasted 50 minutes in which managers of a movie club discussed how to select the movie for the next show. In the second meeting, IS1008c, a team discussed the design of a remote control for 26 minutes. There are two versions of the meeting transcripts, including manual and automatic ones.

Unnecessary information from the original transcripts from AMI Corpus was eliminated such as time of utterances and notations of episodes. Consequently, the most important information left in the two manual meeting transcripts is speaker name and utterances that are showed in the Tables 3.1 and 3.2.

In a conversational document such as a meeting transcript, information about speakers plays an important role in answering the questions that verify a statement with respect to the speaker of one/some utterances. That is why in the proposed algorithm, in order to calculate passage scoring, we pay more attention to the name of speakers in both questions and transcript. For instance, score for a matched word as speaker name is the highest compared to other matched words.

Table 3.1: Meeting transcript IB4010

Andrei	Hi everyone.
Denis	So I don't know if you all received the the a- agenda for this meeting. Do you no?
Mirek	No, I haven't.
Denis	Here it is.
Mirek	Thank you.
Agnes	I haven't.
Denis	So um um the goal for today are um - We have two goals. Uh - First is to decide a movie for uh the next projection for our movie club.
Mirek	Mm-hmm.
Andrei	Mm-hmm.
...	...

Table 3.2: Meeting transcript IS1008c

Agnes	Why was the plastic eliminated as a possible material?
Christine	Because um it gets brittle, .. Cracks We want - we expect these um uh these remote controls to be around for several hundred years. Good expression.
Ed	Good expression.
Christine	I don't know, speak for yourself, I'm planning to be around for a while.
Agnes	Although I think - $ I think with wood though you'd run into the same types of problems, wouldn't you? I mean, it chips, it- if you drop it, uh it's - I'm not Sure $
Sridhar	So so you're not convinced* about the the wood, yes.
...	...

Questions

The BET Questions have been mentioned above. However, some information of the questions used in this system are eliminated from full versions of original BET questions, which consists of miscellaneous information such as observation time, mediate time, important level, scope, etc. [1]. The system needs only the true and the false statement in each question, which are considered as input data to distinguish one from another. Examples of some pairs are in the Tables 3.3 and 3.4.

For the two meetings IB4010 and IS1008c, 222 and 217 raw observations were collected by 9 and 6 observers respectively. After being filtered and corrected, the results are only 116 and 50 final pairs of true/false statement.

Table 3.3: The BET questions IB4010

True	Mirek had not received the agenda for the meeting
False	Andrei had not received the agenda for the meeting
True	None has seen the Shawshank redemption
False	Only two have seen the Shawshank redemption
True	Denis informed the team that the first objective was to choose a film and the second was to discuss an advertising poster
False	Denis informed the team that the first objective was to choose a film and the second was to discuss a date for the film to be shown
...	...

Table 3.4: The BET questions IS1008c

True	One of the features under consideration is speech recognition.
False	One of the features under consideration is fingerprint identification.
True	The product is expected to last over several hundred years.
False	The product is expected to last more than 5 but less than 15 years.
True	Christine eliminated plastic as too brittle over time.
False	Christine eliminated plastic as it would flex and damage the chips.
...	...

The fact that the questions considered as statements is one main feature that makes this system different than other question answering systems, which normally consists of different questions like ”How”, ”Why”, ”When”, etc. For this reason, it is not necessary to apply an existing complex algorithm, which was widely used to deal with various type of questions in other question answering systems. Therefore, our proposed algorithm is designed to try to fit the type of the BET questions. In this case, a typical question ask to verify information spoken by a speaker, for instance "Mirek asks who has seen Schindlers List". In this case, they have a speaker name at the beginning of the sentence. According to our statistics, there are over 55% of such questions (28/50 such questions for IS1008c and 87/116 such questions for IB4010). This is an important remark that score of one matched word spoken by speaker whose name is mentioned in both the transcript and the question should be higher than other matched words.

Another feature of the questions is the similarity of two statements in a pair. In most pairs, two statements are different from each other by only one or two words. Therefore, at the Passage Retrieval stage of the proposed algorithm the probability that the two found corresponding passages coincide is very high. In this case, the true and the false statement can be distinguished by the similarity between each candidate statement with the corresponding passage. In other words, passage score for each statement is compared to determine the true statement/the false statement.

Chapter 4

Proposed algorithm

The proposed algorithm was developed using lexical similarity algorithm combined with some of existing techniques, which are n-gram matching, lexical extensions. However, the way of computing passage scoring pays more attention to the specifications of meeting transcripts, which will be presented in detailed in the following sections.

Our system proceeds in three stages: (i) In the first stage, known as a pre-processing, the two questions and meeting transcripts are normalized and reorganized in order to enhance the performance of the algorithm; (ii) The second stage identifies a section of the meeting transcript which is most likely to contain the answer (i.e. evidence deciding the true and the false statement); and (iii) The third stage compares the two candidate statements with respect to the identified paragraph(s), and returns the true one.

Figure 4.1 gives an overview of the 3 stages.

4.1 Pre-processing

There are two main tasks for this section. First, questions and transcript are transformed into the same form of written text so that they can be compared word by word later. Secondly, in order to enhance the probability of matching

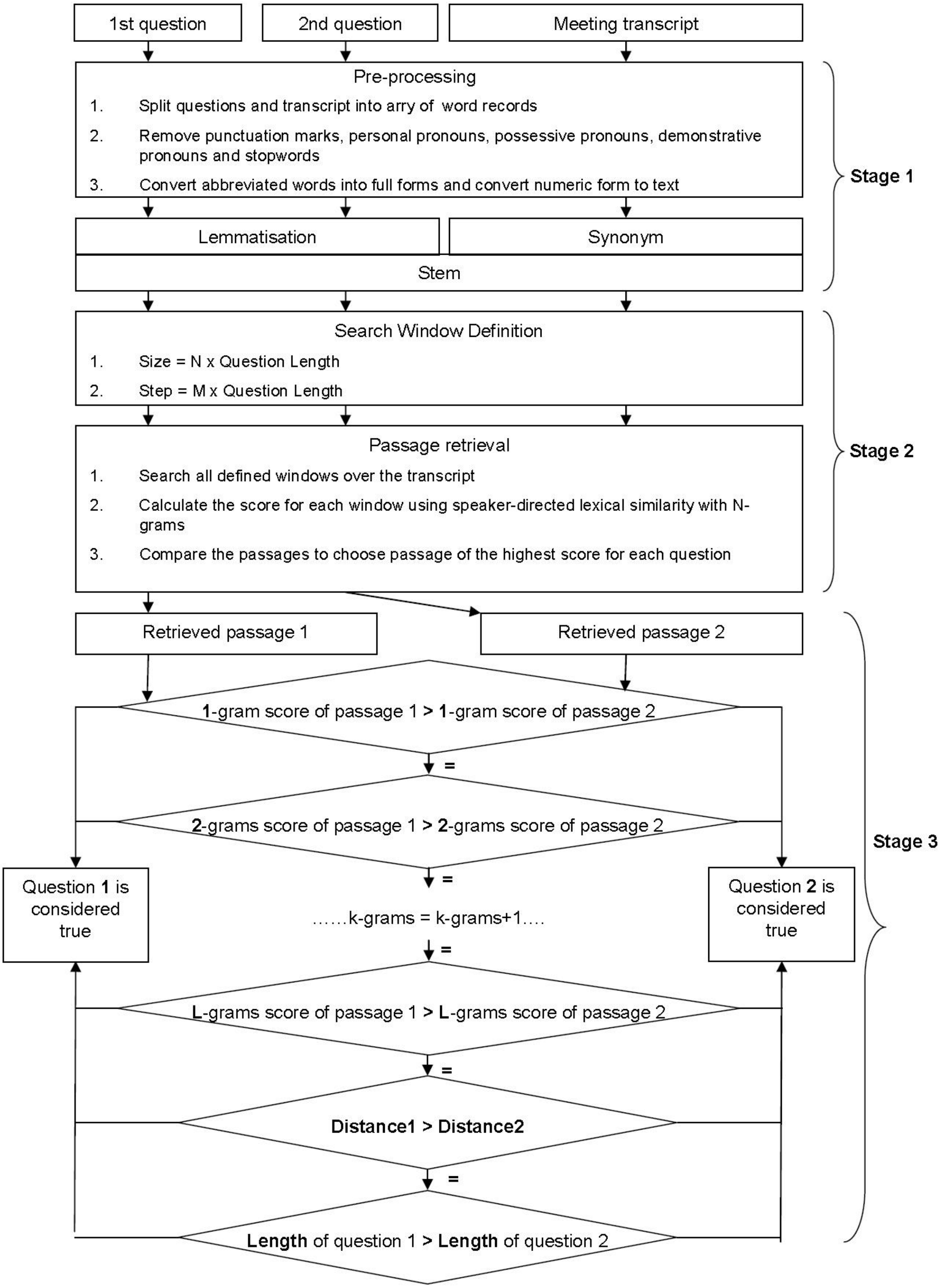

1st question
2nd question
Meeting transcript
Pre-processing
1. Split questions and transcript into arry of word records
2. Remove punctuation marks, personal pronouns, possessive pronouns, demonstrative pronouns and stopwords
3. Convert abbreviated words into full forms and convert numeric form to text
Lemmatisation
Synonym
Stem
Stage 1
Search Window Definition
1. Size = N x Question Length
2. Step = M x Question Length
Passage retrieval
1. Search all defined windows over the transcript
2. Calculate the score for each window using speaker-directed lexical similarity with N-grams
3. Compare the passages to choose passage of the highest score for each question
Stage 2
Retrieved passage 1
Retrieved passage 2
1-gram score of passage 1 > 1-gram score of passage 2
=
2-grams score of passage 1 > 2-grams score of passage 2
=
Question 1 is considered true
Question 2 is considered true
……k-grams = k-grams+1….
=
L-grams score of passage 1 > L-grams score of passage 2
=
Distance1 > Distance2
=
Length of question 1 > Length of question 2
Stage 3

Figure 4.1: Overview of the system

between two words, each word is extended by lemma, stem and synonyms.

The first processing is done by five operations as follows:

1. Removing characters as punctuation marks: comma, dot, quotation mark, semicolon or exclamation, . . . because these characters do not have any effects on the proposed algorithm based on lexical similarity.

2. Removing stopwords such as ”the”, ”to” or ”for” that generally add little or no information regarding the subject matter of text [32]. This helps to reduce cost of computing as well as to take precaution that they muddle the signal from the more content words [33]

3. For the reason that the questions are indirect-speech statements meanwhile the the meeting transcript is a direct-speech report, words that may be changed for a transformation from direct speech to indirect speech should be avoided from counting matched words between a question and a passage because this leads to lexical mismatches. They are personal pronouns (I, me, myself, you, . . .), possessive pronouns (my, your, . . .), demonstrative pronouns (this, that, . . .).

4. Converting numeric forms to text forms: 34 → thirty four, 2nd → second, etc. . . . so that they are written in the same way, this prevents an unnoticed matching while the algorithm executes.

5. Converting abbreviated words into full forms: We’ve → We have, I’ll → I will, etc. . . . . This operation helps the system treat the text in the same way.

All words are transformed into lower-case forms so that nouns, pronouns and verbs have the same level of importance. That means parts-of-speech will not be used to assign scores to words in the Passage Retrieval stage.

The proposed algorithm uses lexical similarity to find a relevant passage that contains answer information (Section 4.2). Thus, a lexical extension should be added so that the system has more than one matching possibility between two words. Take the words *production* and *product* as examples, they match each other because they have the same stem as *produc*.

Three lexical extensions are added to each word, including lemma, stem and synonym.

- Lemma is the canonical form of the word.
- Stem is used to remove inflectional affixes.
- Synonyms have same meaning with the original word.

However, it is not suitable to apply all these extensions to both questions and the transcript. For instance, if both question words and answer word are extended by a set of synonyms, it can create a so called ”redundant information”. In addition, it may make the result of the algorithm inaccurate if they are considered similar because of similarity of an intermediate synonym. For example: *meeting* and *challenge* have one synonym *contest* in common, but they do not have the same meaning. For that reason, a question word is extended by adding its lemma, meanwhile the transcript words are extended by adding a set of synonyms. Meanwhile, stem is applied as the last operation on each word. This may increase the signal from words. For instance with such a question word as the verb *modified* and such a transcript word as the noun *modification*, there are not any matching between these words, even after adding a lemma and a synonym. In detail, *modified* becomes *modify* by a lemmatisation and *modification* has five synonyms *alteration, adjustment, qualifying, limiting, change* (returned by WordNet [34]). But after stemming by Snowball [35], these two words will have a same form as *modifi*, so they are matched.

In practice, for each word, the program runs a Stemming API of Porter, called Snowball [35] to obtain a stem and WordNet API [34] to obtain a lemma and a set of synonyms.

A set of synonyms is reduced to be more coherent with the original word by using parts of speech (PoS) tool named QTAG API [36], which aims at removing synonyms that do not have the same PoS as the original word.

Therefore, each word will be treated from now on as a record of many fields. For a question word, it has three fields: original word, lemma of the word and stem of the lemma. They are described in the Table 4.1. The format of input transcript as described in the Tables 3.1 and 3.2 helps the program identify the speaker of any word in the transcript. Hence, a record of a transcript word has five fields: original word, stem of word, name of speaker name who spoke this word, set of synonyms and part of speech. They are described in the Table 4.2. The name of speakers is also stemmed in order to compare with a question word which may be a speaker name. This field is very important to assess a statement with respect to a specific speaker.

The fields *synonyms* and *lemma* are structured as a set of words because they may have more than one element. For example, lemma of "better" has two words *good* and *well*, synonyms of *better* are *break*, *improve*, *amend*, *ameliorate* and *meliorate*.

For instance, for the pair of such questions as *Mirek had not received the agenda for the meeting* and *Andrei had not received the agenda for the meeting*, after removing stop-words, they remain *Mirek had not received agenda meeting* and *Andrei had not received agenda meeting*. Their lexical extensions are presented as following table:

Table 4.1: Word splitting and lexical extensions for questions

Question 1					Question 2			
Position	**Word**	**Lemma**	**Stem**		**Position**	**Word**	**Lemma**	**Stem**
1	Mirek	Mirek	Mirek		1	Andrei	Andrei	Andrei
2	had	have	have		2	had	have	have
3	not	not	not		3	not	not	not
4	received	receive	receiv		4	received	receive	receiv
5	agenda	agenda	agenda		5	agenda	agenda	agenda
6	meeting	meet	meet		6	meeting	meet	meet

One example for a snippet of transcript as below:

```
.....
 Andrei Hi everyone.
 Denis  So I don’t know if you all received the the a- agenda for this meeting.
 Denis  Do you - no?
 Mirek  No, I haven’t.
....
```

After the processing, it remains:

```
.....
denis   9 not 10 know 11 if 12 you 13 all 14 received 18 agenda 21 meeting
denis   24 no
mirek   25 No 27 have 28 not
....
```

and it is transformed into the following table:

Table 4.2: Word splitting and lexical extensions for transcript

Position	Word	Stem	Speaker	Synonyms (stemmed)	PoS
...	...				
9	not	not	deni	non	XNOT
10	know	know	deni	cogniz experi live acknowledg recogn ...	VB
11	if	if	deni		CS
13	all	all	deni	entir complet total altogeth whole ...	PDT
14	received	receiv	deni	have get find obtain ...	VBD
18	agenda	agenda	deni	docket schedul agendum ...	NN
21	meeting	meet	deni	fill match ensembl contact ...	NN
24	no	no	deni	nobelium	DT
25	no	no	mirek	nobelium	DT
27	have	have	mirek	receiv get own possess ...	HV
28	not	not	mirek	non	XNOT
...	...				

All of the steps above are processed in a procedure separated from execution of the main algorithm, which consists of passage retrieval and true-false answer. The questions and the transcript pre-processed from this stage are stored on hard disk so that the programme can load them into RAM before running the main algorithm. This saves us much time to experiment on different configurations of the algorithm that requires the repeatition of execution with the same input data.

4.2 Passage retrieval

The main goal of this stage is to narrow the space of searched answer by locating a passage which is considered the most likely to contain all the information about the answer, like evidence that helps to discriminate the true statement from the false statement in a pair. It also gives a numeric score indicating the similarity level between a found passage and the question so that two complementary statements in a pair may be compared in the subsequent stage.

In order to find the relevant passage, the system has to compare all possible passages in the transcript. Information being compared is passage scoring, which is a numeric value used to measure the level of similarity between a passage and a question. This task is done by moving a search window from one place to another over the entire transcript. If we assume that the transcript is a cloth stretched to be ironed, then the search window is the iron. In the same way of using an iron, the search window passes over all possible passages that have the same size as the search window in the transcript. All of the retrieved passages by this way are compared in order to provide the passages of highest score [37].

As listed in the Chapter 2, there are many ways to calculate the score of a passage with respect to the question. According to the experimental results of Stefanie [19] with different algorithms of passage retrieval, the performance of the algorithms is different from each other depending on input data. That means each method is suitable for only certain cases. Moreover, existing passage retrieval algorithms use input retrieved by a document retrieval that returns unknown-typed documents such as a forum website, a commercial website or an online newspaper, etc. Additionally, these approaches have to analyze the type of questions such as *How* or *When* before the stage of passage retrieval [15, 19, 38, 39]. This effects the final results of the passage retrieval algorithm because each step accumulates a certain error. In our case, the type of document and the type of questions are defined before, as described in Chapter 3. That is why we do not apply any existing complex algorithms, but rather develop our own algorithm based on the basic one presented by Light et al. [22]. The method of Light is the simplest method to calculate a passage scoring, which is the number of matched words between the passage and the question. Based on this method, we added some certain extensions to the original algorithm, including n-gram matching, lemmatisation, stem, synonyms and various scores assigned to matched word. Lexical extensions such as lemmatisation, stem and synonyms have been presented in the previous stage while the application of n-gram matching and different scores are to be explained specifically in following sections.

As described by Light et al, the score of a passage is based on the number of matched words. However, there are three score levels assigned to matched words. As illustrated in the table 4.2, each word from the transcript corresponds to one speaker's name. Therefore, if a matched word is spoken by a speaker whose name is addressed in the question, the score assigned to this word must be higher than other matched words.

In our experiment system, as presented in the pseudo code 4.2, different values of score are assigned to a matched word as follows:

1. If a word from the question matches the name of the speaker in the passage, then it receives the highest score (e.g., 4.0)

2. If a word from the question matches a word from the passage (lemmas), and this word is spoken by a speaker mentioned in the question, then it receives the second highest score (e.g.,2.5)

3. Otherwise, if a word from the question matches a word from the passage (lemmas) then it receives the "normal score" (e.g.,1.0)

4. If a word (lemma) from the question matches one of the synonyms of a word from the passage, then it receives a "low" score (e.g.,0.5)

The numeric values listed above are set by the author based on the intuition about the importance of each matching, and might not be optimal for this task. No automatic optimization (statistical learning) could have been attempted because the amount of data was insufficient.

As mentioned above, a search window is used to seek and calculate all possible passages on the transcript. The size of the search window is defined as a multiple of the question length. Meanwhile, the distance between two consecutive windows is known as window step and also defined as a multiple of question length. For instance, window size = 5 x question size and window step = 2 x question size. Thus, parameters of search window are dynamic depending on the input question length. This method seems suitable to retrieve relevant passages using the lexical similarity algorithm because when the length of the question increases, the information that the question demands is larger. As a result, it is necessary to enlarge the size of search window.

The passage retrieval algorithm returns a list of passages at the same highest score. Nevertheless, we would like only one passage for the next stage of the algorithm because it is likely that only one relevant passage corresponds to each BET question. In order to choose the most relevant passage, 2-gram score, 3-gram score and so on until n-gram score are compared to reduce the number of passages in the returned list, in which n is the number of question words. A k-gram score is calculated as follows: Instead of working with matched words between question and answer, the program will work with matched substrings of k words between them. If two passages have same 1-gram score, their 2-gram score are compared with each other to return higher-score passages. If they have the same 2-gram score, their 3-gram score are compared with each other. This continues in the same way until one passage has a higher score than others or all n-gram scores are compared with each other, in which n is the number of words in the question. If they still have same n-gram score, the first passage is returned. In fact, n-gram matching is the simpler variant of the dependency relation approach among matched words that the matching of words in order is better.

The implementation of passage scoring calculation is described by pseudo code 4.2 with comments in detail. The following is a list of the most impotant features of this implementation:

- A passage is defined as a class that has 5 properties: 1) Passage score, which indicates the similarity between this passage and the current question. The passage score is computed as the sum of the scores for each matched word; 2) Position of the passage in the transcript; 3) Size of passage; 4) Set of positions of matched words in the transcript; and 5) Distance among matched words (density). This distance is calculated as the sum of absolute distance between any two matched words. This task is done at the end of the passage retrieval to prepare for the True-False Answer stage.

- The order of priority is to "name matching", then "stem matching" and lastly "synonyms matching" in order to compute the highest score for the current passage.

- The frequency of one matched word is not used to increase the score. However, in the case of "multiple matching", if one word is repeated more than one time in both the question and the passage, the number of matched words will be counted as the minimum number of appearance of this word in both the question and the passage.

Algorithm 4.1 Passage Retrieval

```
Require: Question //Array of word records as described in the table 4.1
Require: Transcript //Array of word records as described in the table 4.2
Require: WindowSize //Size of search window (in word unit)
Require: WindowStep //Distance between two consecutive windows (word unit)
$Passage \leftarrow Empty$ //Initiate a new passage as current passage(the passage class is defined above)
$BestPassage \leftarrow Empty$ //Initiate a new passage as the best retrieved passage at one moment
$Position \leftarrow 0$ //Initialized position of the search window
while $Position < Transcript.length - WindowSize$ do //Search for all passages to choose the best passage
    $Passage.position \leftarrow Position$ //Position of current passage
    $Passage.size \leftarrow WindowSize$ //Size of current passage
    $Passage.WordList \leftarrow Transcrip[Postion \div Position + WindowSize]$ //List of word records is extracted from the transcript[Position,Position+1,..,Position+Size of window]
    $Ngrams \leftarrow 1$ //Firstly, passage score is calculated using unigram matching
    $Passage.MatchedList \leftarrow getMatchedList(Passage, Question, Ngrams)$ //that is from the procedure 4.2
    $Passage.score \leftarrow getPassageScore(Passage, Question, Ngrams)$ //that is from the algorithm 4.2
    if $BestPassage.score < Passage.score$ then //The current passage is better
        $BestPassage \leftarrow Passage$ //remember it as the best passage
    else if $BestPassage.score \equiv Passage.score$ then //If they have the same score
        while $BestPassage.score \equiv Passage.score \wedge Ngrams \leqslant Question.length$ do //Recalculate their score using bigrams, trigrams,.. until their score is different from each other or Ngrams is over the length of question
            $Ngram \leftarrow Ngrams + 1$ //Increase Ngrams by 1
            $BestPassage.score \leftarrow getPassageScore(BestPassage, Question, Ngrams)$ //Recalculation with new Ngrams
        end while
        if $BestPassage.score < Passage.score$ then //Finally, if current passage have better score, remember the current passage as the best passage until now
            $BestPassage \leftarrow Passage$
        end if
    end if
    $Position \leftarrow Position + WindowStep$ //Move the search window forward
end while
$Passage.distance \leftarrow 0$
for $i = 0$ to $Passage.WordList.Length - 1$ do
    for $j = i + 1$ to $Passage.WordList.Length$ do
        $Passage.distance \leftarrow Passage.distance + abs(Passage.MatchedList[i] - Passage.MatchedList[j])$
    end for
end for
return BestPassage
```

Algorithm 4.2 Passage Score Calculation

```
Require: $Question[] \neq null$ //Array of word records as described in the table 4.1
Require: $Passage.WordList[] \neq null$ //Array of passage word records as described in the table 4.2
Require: $Ngrams \geqslant 1$ //1 for unigram; 2 for bigram; 3 for trigram
Score ← 0 //Initialization value for passage score.
$Speaker \leftarrow Null$ //Name of a speaker that is mentioned in the question
$PositionsSet \leftarrow Null$ //Set of matched word positions
$AvailQues[] \leftarrow True$ //Available status of each question word to match with passage words
$AvailPas[] \leftarrow True$ //Available status of each passage word to match with question words
//Firstly, search for a speaker name that is containned in both question and passage
for $i = 0$ to $Question.length$ do
    $Matching \leftarrow False$
    $j \leftarrow 0$
    while $!Matching \wedge j \leqslant Passage.WordList.length$ do
        if $Question[i].Stem \equiv Passage.WordList[j].Speaker$ then //If one exists
            $Score \leftarrow Score + 4.0$ //Then passage score increases 4.0 points
            $Speaker \leftarrow Question[i].Stem$ //Remember this name for step later
            $AvailQues[i] \leftarrow False$
            $Matching \leftarrow True$
        end if
        $j \leftarrow j + 1$
    end while
end for
//Checking for a N-grams matching
for $i = 0$ to $Question.length$ do
    $Matching \leftarrow False$
    $j \leftarrow 0$
    while $!Matching \wedge j \leqslant Passage.WordList.length \wedge AvailQues[i] \wedge AvailPas[j]$ do
        $Matching \leftarrow True$
        for $k = 0$ to $Ngrams$ do
            if $Passage[j + k].stem \nsubseteq Question[i + k].lemma$ then
                $Matching \leftarrow False$
            end if
        end for
        if $Matching$ then //If one exists
            if $Speaker \equiv Passage.WordList[j].Speaker$ then //Speaker of matching words is mentioned in the question
                $Score \leftarrow Score + 2.5$ //Then the score increases 2.5 points
            else
                $Score \leftarrow Score + 1.0$ //Other cases, the score increases 1.0 point
            end if
            $AvailQues[i] \leftarrow False$ //These words will be disable from next matching process
            $AvailPas[j] \leftarrow False$
            $PositionsSet \leftarrow PositionsSet \cup j$ //Save position of matching to list
        end if
        $j \leftarrow j + 1$
    end while
end for
//Checking for a synonym matching
for $i = 0$ to $Question.length$ do
    for $j = 0$ to $Passage.WordList.length$ do
        if $Question[i] \subseteq Passage.WordList[j].Sysnonyms \wedge AvailQues[i] \wedge AvailPas[j]$ then
            $Score \leftarrow Score + 0.5$
            $PositionsSet \leftarrow PositionsSet \cup j$ //Save position of matching to list
        end if
    end for
end for
return Score, List
```

4.3 True-False Answer

Based on two retrieved passages corresponding to two input statements from the previous stage, the purpose of this stage is to identify the true statement in a pair.

At the first sight, this system seems simpler than other question answering systems, which must analyze the type of question, such as *Who* or *How* before extracting answer words in retrieved passages as mentioned in the Chapter 2. In our case, two questions in a pair are formed as statements and the answer is simply *true* or *false* one bit for each question. However, because two questions are very close with each other, that means that in most cases they are different from each other by only one or two words. Thus, it is not easy to distinguish one question from another, even when the retrieved passage corresponding to the true question is correct (the retrieved passage corresponding to the false question is always false because information of this statement does not exist in the transcript). The only hopefulness is that the passage corresponding to the false statement can help the system tell that the statement is false due to their number of matched words is less.

For passages retrieved from the previous stage, we have two cases: a passage corresponding to the true question is evaluated as incorrect or correct. In the first case, when both retrieved passages are incorrect, thus we cannot identify the true statement by any reasonable algorithm. This is because that the database, which are the passages, used to give the answer is not correct. In the second case, we also have two possibilities: (i) two passages are identified; and (ii) two passages are different from each other. For the second possibility, there is not relation between the first question and the second question. They are totally different. Their retrieved corresponding passages are therefore two different ones. As a result, the only way to answer which question is correct is to measure independently the correctness of each question based on their passage. Then, the question whose accuracy is higher is considered to be true. At this time, the question of how to measure them has to be answered? If we use passage score to evaluate them, there is not any reason that the number of matched words of the false question would be less than those of the true question. So this step needs a deeper analysis on semantic text to answer the question persuasively. We have not yet found an effective method to assess which question is "more" correct.

The proposed algorithm is applied rather in the case that both the two passages have the same position in the transcript. The possibility of this is high because the similarity of two statements in a pair as mentioned in Chapter 3. That means the same passage retrieved for both the true and the false candidates as well as the passage corresponding to the true question are correct. In this case, it is more probable that passage score of the false statement is less than this of the true statement. It is also the main idea of the proposed algorithm for this step. Despite the simplicity of this algorithm, experimental results are much better than results by chance (see section 6.2).

In detail, all scores for 1-gram matching, 2-gram matching, ..., n-gram matching of two passages are used to assess two questions. If two passages have the same 1-gram score, their 2-gram score will be compared, continuing in the same way until one passage have higher score or all n-gram scores were compared. In this case, n is number of words from the smaller question. In fact, using n-gram matching is the simplest version of using dependency relations between neighbor words. The way of calculating n-gram matching has been presented in the previous section Passage Retrieval. In the case they still have the same score, the number of words from two passages are then compared with each other. To explain, one question will be considered to be true if the number of matched words over the total number of question words is smaller.

Pseudo code of the algorithm is described as follows:

Algorithm 4.3 True-False Answer

```
Require: $Passage1$ //That is the best passage for question1 which is from the algorithm 4.1
Require: $Passage2$ //That is the best passage for question2 which is from the algorithm 4.1
Require: $Question1$ //Array of word records as described in the table 4.1
Require: $Question2$ //Array of question2 word records as described in the table 4.1
  if $Passage1.score > Passage2.score$ then //If the score of passage1 is higher than that of passage2
    return 1 //The question1 is considered as one true statement
  else if $Passage1.score \equiv Passage2.score$ then //If they have the same score
    if $Passage1.distance < Passage2.distance$ then //compare their distance among matched words as calculated in the algorithm 4.1
      return 1 //The question1 is considered as one true statement
    else if Passage1.distance ≡ Passage2.distance then //If they still have the same distance
      $Ngrams = 2$
      while $Passage1.score \equiv Passage2.score$ do //Recalculate the passage scores using Ngrams matching
        $Passage1.score \leftarrow getPassageScore(Passage1, Question1, Ngrams)$ //That is from the algorithm 4.2
        $Passage2.score \leftarrow getPassageScore(Passage2, Question2, Ngrams)$
        $Ngrams \leftarrow Ngrams + 1$ //Increase Ngrams by 1
      end while
      if $Passage1.score > Passage2.score$ then //If the score of passage1 is higher than the score of passage2
        return 1; //The question1 is considered as one true statement
      else
        return 2; //The question2 is considered as one true statement
      end if
    end if
  end if
```

In such case, the distance among matched words is calculated in the previous stage, (see the pseudo codes 4.1).

For instance, with two questions presented in the table 4.1, the algorithm gives the following results:

Passage score for the first question *Mirek had not received agenda meeting* is 12.5, corresponding to the retrieved passage below:

```
denis   9 not 10 know 11 if 12 you 13 all 14 received 18 agenda 21 meeting
denis   24 no
mirek   25 No 27 have 28 not
```

In this case, the score for such five matched words as *not*, *receive*, *agenda*, *meet* and *have* is 1.5 while the score for speaker name matching *Mirek* is 4.0. However, because the word *have* is spoken by *Mirek* and the word *Mirek* is found in the question, a bonus 1.0 is added to the total score. Thus, the total score is 12.5. Meanwhile passage score for the second question is 11.5 corresponding to the following retrieved passage:

```
andrei  4 hi 5 everyone
denis   9 not 10 know 11 if 12 you 13 all 14 received 18 agenda 21 meeting
```

In this case, the score for five matched words *have*, *not*, *receive*, *agenda* and *meet* is 1.5. Meanwhile, the score for speaker name matching ”Andrei” is 4.0.

Then, the total score is 11.5. Consequently, the first question higher passage score is considered as true.

Chapter 5

Evaluation methods

The performance of the proposed algorithm is evaluated based on the results of both principal phases: Passage Retrieval and True-False Answer.

5.1 Passage Retrieval Evaluation

In the first phase, the correctness of a retrieved passage is evaluated by comparing it with a reference corresponding passage, which was annotated by hand. The information of the reference passage includes the position of its first word in the transcript and its size counting in word unit. These correspond to two properties of a passage defined in the section Passage Retrieval of the previous chapter.

For example for the question and the transcript in the table 4.1 and 4.2, the reference passage found for this question is (25,28) so that it contains only three words ”25 No 27 have 28 not”. The name of speaker who spoke these words is always integrated as a field of word record, thus it is not necessary to show the speaker name in the reference passage and even in the retrieved passage.

The size of a reference passage is reduced to be as small as possible, but this reference passage still contains the most essential words to answer the question. For the example above, a reference passage may be (9,28) that help us understand the context of answer, but the keywords are only "25 No 27 have 28 not" spoken by Mirek.

If candidate passage and reference passage have a non-empty intersection, candidate passage is considered to be correct. The number of overlapped words is fixed as one word in this case. If the system is used to help users locate the position of answer information, one overlapped word will be accepted as well. For the experimental system, the size of the retrieved passage is reduced to region which contains matched words instead of the size of the search window. For above example, with the first question, the size of the search window is 5 x number of question words = 35 words. But the size of the found passage is 14 instead of 35 as size of window. The reason is that the position of the passage is defined as the position of the first matched word in the transcript whereas the size is defined as the distance between the first matched word and the last matched word. In this case, the position of the first matched word and of the last matched word are respectively 3 and 17.

5.2 True-false Question Answering Evaluation

It is throught that we know the true question and the false question in a pair in the input database. Therefore, a question, which is considered to be true by the system, is evaluated by its index and returned by the program. In this case, in the database, the first question in a pair is known as true. Thus, if the system returns 1 as an answer, this answer is correct. On the contrary, if the system returns 2 as an answer, this answer is incorrect.

5.3 Cross-Validation method

The 5-fold Cross-Validation method [40] is applied to give the average of the best scores for the proposed algorithm. This method is suitable in the case that the algorithm does not have enough data to test. Hence, it hides a part of the data as unknown data in the future. First, the algorithm builds the best configuration for the known data. After that, it uses these hidden data to test the built configuration. The data used to train the system is called training data and the data used to test the system is called test data.

In this case, a configuration is a pair of parameters of the search window: size of window and step of window that have been addressed in the section Passage Retrieval. According to the Cross-Validation method, questions are partitioned into 5 subsets, in which four subsets are used as training data and the remainning subset is used as test data. The algorithm is iterated five times so that each subset will be used as test data one time. Thus, the final result is the average result after running the algorithm five times in this way.

For each iteration, the system returns a pair search window parameters that are considered the most suitable for training data, in which the values tested for both search window parameters are from 1 x question size to 10 x question size. A pair of parameters helps the system obtain the highest score with training data; then it will be tested on the test data. The result obtainned on test data is used to estimate the average score of the algorithm.

Chapter 6

Experimental Results

The goal of this chapter is to discuss the experimental results obtained by the proposed algorithm using evaluation methods as presented in Chapter 5, as well as the results obtainned from intermediate steps such as Pre-Processing and Parameter Optimisation.

The input data used for the experiments is the two meeting transcripts IB4010 and IS1008c with the BET questions for these meetings as described in Chapter 3. In addition to the experimental data, automated transcripts generated by the Automatic Speech Recognition, namely ASR transcripts and some automatic summaries based on ASR transcripts for two meetings are also used.

Because that two meetings may have different difficulties, we analyzed the results for each meeting separately.

6.1 Data Processing

Above all, we present the result for the pre-processing stage, questions and transcript are processed by removing unuseful information as punctuation marks, stopwords and pronominal words and being transformed into a standard form to enhance the performance of lexical similarity algorithm as described in the Section 4.1.

After removing the un-useful words, the length of IB4010 is 4872 words compared to 9488 words in the original transcript. At the same processing, the length of IS1008c is 2059 words compared to 4000 words in the original transcript. This means that nearly half of the total amount of words are removed. This helps at least doubles the algorithm speed when it searches relevant passages in the transcript.

For the questions, after the pre-processing, the average length of the questions is 8 words compared to 12 words (in original questions). Reducing the length of the questions also plays an important role in speeding up the algorithm, because it is used as the base unit to define size of the parameters for a search window (see the definition of search window in the section 4.2).

6.2 System Performance

This section shows the results of the algorithm for both IB4010 and IS1008c using the evaluation methods presented in the Chapter 5.

The experimental results show the performance of both principal phases of the algorithm, Passage Retrieval and True-False Answer. The contribution of each technique in the algorithm such as n-gram matching and speaker-directed scores assigned to matched words are also demonstrated by showing obtainned scores for each technique separately.

In order to insist on the effectiveness of the algorithm, the scores obtainned by guessing the answers randomly, which is also calculated and compared with the results of the algorithm. The "random" scores for true-false answers are 50% because there are only two values "True" or "False" for an answer. The "random" scores for passage retrieval are calculated as shown below:

With a defined search window and input data, we can compute the total of possible passages in the transcript based on transcript size, search window size and search window step. The search window moves on through the transcript from one place to another. At one time, the position of the window defines a passage that have the same position and same size as the window. Thus, the number of window movements is the total number of passages. Two consecutive positions of the search window movement is defined as the *step* of search window. Consequently, the total passages = [(transcript size - window size)/window step] + 1. If a candidate passage and corresponding reference passage overlap each other by one word, the candidate passage is considered to be true and the average of reference passage is 10 words, we have total number of correct passages is 10 x (window size / window step). Therefore, "random" score is calculated by dividing the number of correct passage by the total number of passages. In the section 6.1, we have the average of question size is 8 words, the length of transcript IB4010 is 4872 and the length of transcript IS1008c. From this numbers, calculated "random" score for IB4010 is 1.65% and for IS1008c is 3.9%.

The method of Cross-Validation is used to calculate the average of scores as described in the pseudo codes 6.1.

Algorithm 6.1 Calculate average score based on Cross-Validation method

```
$Score = 0$
**for** each $TrainingData$ from 1 to 5 **do** //There are 5 pairs (Training, Test)
  **for** each $WindowSize$ from 1 x $QuestionSize$ to 13 x $QuestionSize$ **do**
    **for** each $WindowStep$ from 1 x $QuestionSize$ to $WindowSize$ **do**
      Find relevant passage over Training Data by the algorithm 4.1
      Save passage $P\_max$ whose score is highest until now
    **end for**
  **end for**
  Using the pair $(size, step)$ corresponding to $P\_max$
  to find relevant passage $P$ over corresponding Test Data by the algorithm 4.1
  $Score = Score + P.Score$
**end for**
**return** $\frac{Score}{5}$
```

Results for the two processing stages of the automatic BET question answering algorithm are given in the Table 6.1, for three variants of the algorithm: using only unigram matching when computing the similarity score (and no weighting of speaker-specific words), then with N-gram matching, and finally with additional weighting of matched words spoken by a speaker mentioned in the question, as explained in the Chapter 4.2.

Passage retrieval provides excellent results compared with the chances of randomly locating the correct passage, with scores of 0.55 ± 0.14 for IB4010 and 0.62 ± 0.16 for IS1008c (obtainned with 5-fold cross validation). The automatic system is of course much faster than humans+browsers, at less than 1s per question.

When combined with the question discrimination, the performance increases only slightly. The expected score should be an average of the score on the questions for which the passage was correctly identified (55% and 62%) with a 50% chance for the questions on which the passage was incorrectly identified, so about 77% for IB4010 and 81% for IS1008c.

The fact that the actual scores are lower (though above chance) shows that the algorithm needs improvement for this stage.

Table 6.1: Performance of the algorithm over two meetings

Condition	Passage Retrieval				True-False Answers			
	IB4010		IS1008c		IB4010		IS1008c	
	Acc.	Stdev	Acc.	Stdev	Acc.	Stdev	Acc.	Stdev
Random	0.17	n/a	0.39	n/a	0.50	n/a	0.50	n/a
Unigram matching	0.27	0.15	0.54	0.21	0.37	0.14	0.36	0.21
N-gram matching	0.32	0.15	0.50	0.19	0.43	0.17	0.42	0.11
N-gram + speaker	0.55	0.14	0.62	0.16	0.57	0.06	0.64	0.18

The standard deviation is calculated based on results from the Cross-Validation method using the formula 6.1 as follows:

$$Standard\ Deviation = \sqrt{\frac{1}{N}\sum_{i=1}^{N}(x_i - \bar{x})^2} \tag{6.1}$$

In this case, N = 5 corresponds to five test data from Cross-Validation , $\bar{x}$ and is the average value of five test data results and x_i is the value of each result among five results of test data.

Our proposed algorithm, based on the algorithm of Light using n-gram matching and speaker-directed scores assigned to matched words in the passage retrieval, demonstrates the effectiveness of these techniques by the results displayed in the table. Despite the simplicity of the second phase of the algorithm, its final scores are still a lot better than "random" scores.

6.3 Experiments with ASR transcripts

In this section, the performance of the system will be evaluated on ASR transcripts that are generated by an automatic speech recognition. In this case, two ASR transcripts are generated by using the M4 recognition system developed by Krafiat et al [13]. We will evaluate the obtainned results using the ASR transcripts by comparing them with those using the manual transcripts.

According to the report of Karafiat, the quality of these transcripts are not very good. Logically, scores obtainned over the ASR transcripts should be worse than those over the manual transcripts, but remain in a similar range. The experimental results obtainned by our algorithm using the ASR transcripts as described in the table 6.2 show that the results are good as expected. First, the remaining number of words in the ASR transcripts are not much changed compared with those of the manual transcripts. Secondly, although the rate of correct answers over the ASR transcripts reduces logically, but the reduction is only about 8% correct answers compared with the manual transcripts for both principle phases of the algorithm. This may be explained by a fact that when word error rate of the ASR transcripts affects overall text of the transcripts so that the score of all passages reduces together. The algorithm always chooses the passage of highest score for its answer, thus the accuracy is not much changed. That means the algorithm works very well with the ASR transcripts. This is a promising results for building a full automatic assistance tools over ASR meeting transcrips.

For IB4010, passage retrieval accuracy drops to 0.46 ± 0.13 (from 0.55 ± 0.14) and true-false question accuracy drops to 0.52 ± 0.09 (from 0.57 ± 0.06). For IS1008c, passage retrieval drops to 0.60 ± 0.33 (from 0.62 ± 0.16) and true-false question drops to 0.56 ± 0.19 (from 0.64 ± 0.18).

The two following tables present the results in detail. The method of 5-fold Cross-Validation as described in the Section 5.3 is applied to get the average results over two ASR transcripts. The standard deviations are also calculated according to the formula 6.1.

Table 6.2: Experimental results for ASR transcripts

	IB4010 transcript		IS1008c transcript	
	Manual	ASR	Manual	ASR
Original length	9488	9393	4000	3927
Processed length	4872	4624	2059	1957
Passage retrieval	55%±14%	46%±13%	62%±16%	60%±34%
True-false answer	57%±6%	52%±9%	64%±18%	56%±19%

6.4 Summarization Results

A meeting transcript summary is a transcript that is shorter than the original, but on that still contains the main information of the meeting.

In this section, the system is tested on summaries of two meeting transcripts IB4010 and IS1008c based on ASR transcripts, named ASR summaries. There are two purposes for this test. First, it aims to measure the quality of an ASR summary by its scores when it is used to replace the original transcript. Secondly, it helps to evaluate the proposed algorithm that should give the number of correct answers that decrease little by little when the length of summaries reduces because some important information are losted from the processing of the ASR.

Moreover, for an evaluation purpose, for each ASR summary, we generate a corresponding "random" summary in order to compare scores of ASR summaries with those of random summaries. If the summarization algorithm is good, the score schema of its summaries must be different from that of the random summaries. A random summary is created by repeating the elimination of a transcript word randomly until this summary has the same length of the ASR summary. In order to increase the precision of the results over random summaries, for each known summary length, we create 100 random summaries and calculate their average scores.

The ASR summaries used in these experiments were created by an automatic summarization system presented by Gabriel Murray and Steve Renals [41] using term-weights. Within these, each dialogue act is ranked by a score of importance level, namely ranking score. Based on these ranking scores, we create different summaries by eliminating utterances whose ranking score is less than a defined threshold.

In reality, we define 10 different thresholds. The obtainned results are displayed in two tables 6.3 and 6.4 for IS1008c and IB4010. In the tables, the first column is the percentage of summary length compared with the length of the original transcript. The second column is threshold, which is used to created a corresponding summary by eliminating all of the utterances whose score is less than this threshold in the transcript. The scores were assigned to each utterance in the transcript by an automatic summarization. When threshold is zero in the first row, the results are presented for original transcript. The four remaining columns are percentages of correct passages and true-false answers for ASR summaries and random summaries, respectively.

Based on the experimental results, we have some remarks, as follows:

- The number of correct passages for ASR summaries decreases linearly and more quickly than random summaries do when the number of utterances removed from the summaries increases. The graph of summary lengths and the number of correct passages for ASR summaries have the same bias (ie, they are parallel with each other). This says that eliminated utterances for ASR summaries contain important information. This is contrary to the rule of an automatic summarizer that have to remove firstly utterances whose information is less important. The random summaries are even better than the ASR summaries according to the results of correct passages. For evaluation of the algorithm performance, the algorithm works well, in that the number of correct passages decreases linearly for both type of summaries.

- For true-false answers, both type of summaries have the same behaviour. The number of true-false answers decreases logically at first. After that it does not decrease but it tends to a random result. That means the proportion of the correct answers is always around 50%, meanwhile we know that the probability of a correct true-false answer by chance is also 50%. This is explained by a fact that true-false answers were answered by the algorithm based on a comparison between two similarity levels, which are obtainned by considering each question and its corresponding passage retrieved from the phase Passage Retrieval of the algorithm. At first, decreasing transcript size affects both both true and false question in a pair, the score of corresponding passages decreases accordingly so that the number of correct answers decreases. However, after that, when the important information related to the questions was removed from the transcript due to the automatic summarization, there were not enough facts to distinguish one question from another or in other words, returned answer tends to be random. This demonstrates that the results from the phase True-false Answer are not suitable for aiming to measure the quality of a summary.

Consequently, in this case, the way to eliminate dialogue acts in order to generate a ASR summary did not work very well because it eliminated some important utterances that are necessary to answer the BET questions. In fact, the automatic summarization used to produce the ASR summaries is not official version and this result has not yet been able to be used to judge the fruit of authors who wrote the automatic summarization. Thus, the results are only used to evaluate the stability of our system. On this aspect, its performance for the passage retrieval stage is stable over all summaries so that it can be used to measure the quality of a summary in this way.

Table 6.3: Results for IS1008c summaries

%Original Length	ASR Summaries			Random Summaries	
	rank score	%cpassage	%canswer	%cpassage	%canswer
100	≥ 0.00	68	64	68	64
85	≥ 0.05	62	62	60	60
74	≥ 0.10	56	54	58	60
64	≥ 0.15	52	52	54	58
57	≥ 0.20	50	48	50	56
54	≥ 0.25	46	48	50	56
52	≥ 0.30	38	46	50	54
49	≥ 0.35	36	46	50	56
45	≥ 0.40	28	48	48	52
41	≥ 0.45	24	46	46	52
38	≥ 0.50	24	46	46	50

Original length of ASR transcript for IS1008c is 1957 words

Table 6.4: Results for IB4010 summaries

%Original Length	ASR Summaries			Random Summaries	
	rank score	%cpassage	%canswer	%cpassage	%canswer
100	≥ 0.00	45	62	45	62
85	≥ 0.05	34	51	44	59
74	≥ 0.10	26	55	38	56
65	≥ 0.15	21	52	37	56
58	≥ 0.20	21	54	37	56
54	≥ 0.25	20	52	36	56
51	≥ 0.30	17	53	36	54
47	≥ 0.35	17	53	36	55
44	≥ 0.40	16	52	33	54
40	≥ 0.45	14	50	35	52
36	≥ 0.50	15	51	33	52

Original length of ASR transcript for IB4010 is 4624 words

6.5 Parameter Optimization

The task of the parameter optimisation is to find parameters which are the best fit for each transcript, IB4010 and IS1008c. The parameters are search window size and search window step.

In order to do this, we used the 5-fold Cross-Validation method as presented in the previous chapter to build a statistic table. This table consists of columns and rows which present values of window step and values of window size correspondingly. For instance, for position (2,5), the step of search window is 2 x input question size and the size is 5 x input question size. In this table, each position (row,column) of the table presents the number of partitions as training data of the Cross-Validation method that obtain maximal scores using the value of parameters corresponding to row and column of this position. For instance, the first partition obtains maximal scores at (2,3), (2,5) and the second partition obtain maximal score at (2,4), (2,5) and the others partitions obtain maximal scores at other pairs of parameters, then value of the position (2,5) of the table is 2 corresponding to two partitions. That means when search window size is 2 and search window is 5, there are two partitions over all five partitions obtain maximal score. The maximal scores are the maximal number of true passages retrieved by the algorithm in the first phase. For this experiment, we only use the first phase Passage Retrieval because it is the most essential of the proposed algorithm.

The following tables present results obtainned for IB4010 and IS1008c in detail:

Table 6.5: Parameter Optimization for IB4010

Step Size	**1**	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**	**10**	**11**	**12**	**13**
1													
2													
3													
4													
5													
6													
7													
8	1			1									
9	4	3											
10	4	2	5 *										
11	3	2	5										
12	1												
13		1	2								1		

* This position is chosen

Table 6.6: Parameter Optimization for IS1008c

Step Size	**1**	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**	**10**	**11**	**12**	**13**
1													
2	1												
3	1	1	1										
4	4 *	1	1	1									
5			1	1									
6	2	2	3	1	2	1							
7	1		3		2	1							
8								2					
9	3	1	1	1	1	1							
10	2		1	1	1	1				1			
11	1	1				1	1			1			
12	5	3	3		2	4	3			1	1		
13	1	2	3		3		1			1	1		

* This position is chosen

According to the way of building the table above, parameters are considered good if they help as many training data as possible obtain the maximal scores. Therefore, we will choose parameters at a position whose value is the largest in the table as the relevant parameters. As seen in the table, for IB4010 there are two maximal values at (10,3) and (11,3) and for IS1008c the value of position (12,1) is the largest. These values of parameters help all training data of Cross-Validation method obtain the best scores. However, it is evident that the more the size of a search window increases, the higher the probability that a passage becomes correct. When the size of search window is equal to the size of the transcript, then it certainly contains the information of the question. So the returned passage is always true. In this task, we want to find parameters that help programme obtain the maximal number of correct passages but the objective of the passage retrieval is to reduce the search space. For this reason, we should choose the smallest size of search window that is suitable for the most partitions. In the table of IS1008c, the position (4,1) is also accepted because it is satisfy 4 over 5 training data. That means the size of the search window is 4 times the question size and the step of the search window is 1 time the question size, and are thus the best fit for the BET questions and the transcript IS1008c. In the table of IB4010, the pair of parameters (10,3) has the best value, thus the size of search window is 10 and the step of search window is 3 and these are chosen as the best fit for the BET question and the transcript IB4010.

The chosen parameters will be used in the next sections.

6.6 Comparison with BET scores by human subjects

The main goal of this comparison is to discover whether automatic machine and human subjects have the same difficulties in answering the BET questions. By analyzing the scores obtained by the system and BET scores, we can also identify in which case this system is useful to help humans answer the BET questions.

The BET scores used for this comparison are results from BET for the TQB interface [3] known as a Transcript-based Query and Browsing Interface which was developed by Andrei Popescu-Belis et al. The TQB is considered a meeting browser tool for searching and browsing multi-modal recordings of group meetings. And the BET method is used to evaluate the performance of this meeting browser over two meetings IB4010 and IS1008c.

According to the BET method, human subjects that had not worked with TQB before, were tested by answering the BET questions using TQB. They were 28 students at the University of Geneva, mainly form the School of Translation and Interpreting. Half of the subjects started with IB4010 and continued with IS1008c, and the other half did the reverse order, thus allowing for differentiated results depending on whether a meeting was seen first or second. That means when subjects worked on the first meeting, they were trained with the TQB interface, so that they would answer BET questions better on the second meeting. In fact, the average of precision is a bit higher for the second meeting. In this experiment, both BET scores of the first meeting and the second meeting are used for comparison with the results obtained by the system. However, only 8 BET questions for each meeting are used for this comparison. We are interested in only two pieces of information from the BET scores: average answering time and precision of each answer.

In order to set up the configuration of the system, defined parameters of search window are obtainned from the previous section Parameter Optimisation. They are (search window size = 10 x question size, search window step = 3 x question size) for IB4010 and (search window size = 4 x question size, search window step = 1 x question size) for IS1008c.

The BET scores by human subjects and scores obtainned by the system are showed in detail in two tables 6.7 and 6.8. In each table, for the scores by human subjects, *Precis1* and *avg time1* in millisecond are average precision and average time as the first meeting, *Precis2* and *avg time2* are average precision and average time in millisecond as the second meeting. For scores obtainned by the system, *#cpassage* and *#canswer* are number of correct passages and number of correct true-false answers correspondingly. However, number of answers for each question is only one. *time* in millisecond is time for answering one question.

Table 6.7: Comparison with BET scores by human subjects for IB4010

curQuid	Humans				System		
	Precis1	avg time1	Precis2	avg time2	#cpassage	#canswer	#time
1	0.93	303.14	0.71	143	0	0	24
2	0.93	105.36	1.00	66.14	1	1	22
3	0.71	118.14	1.00	89.21	1	1	40
4	0.86	207.5	0.86	206.43	1	1	32
5	1.00	64.71	0.93	37	0	1	16
6	0.93	57.79	1.00	53.21	1	1	17
7	0.93	60.93	0.71	52	1	1	24
8	0.71	129.5	0.79	85.29	1	1	19
	0.88	**130.88**	**0.88**	**91.54**	**0.75**	**0.88**	**24.25**

According to the BET for TQB [3], average precision to answer all BET questions for IB4010 is 0.85 ± 0.05 and 0.70 ± 0.10 for IS1008c. Thus, for human subjects, we can divide the BET questions into two groups: easy and less easy. A BET question belongs to easy group as the first group if average precision of its answers as first meeting or second meeting is more than 0.85 for IB4010 and 0.70 for IS1008c, otherwise it belongs to the second group. Meanwhile, for the system, easy group includes all questions that their number of correct passage or number of correct true-false answer is 1. This help us have a standard to compare the BET scores by humans with scores obtainned by the system.

We first examine the results for IB4010. According to the convention above, for human subjects there is only one less easy question (question number 8), while there are two less easy questions for the system (questions number 1 and 5). In fact, all of these questions are deductive questions that require deep comprehension rather than a search of lexical similarities. In detail, question number 1 is *throughout*. That means it is necessary to read all of the transcript before answering the question. That is why the system could not identify the correct passage using a small search window that does not cover all the information of the transcript as it requires for this type of question. Consequently, the true-false answer is determined by chance (false in this case). The question number 5 also require a deduction for its true statement ”No one had seen Goodfellas”. In the meeting, when all meeting participants said ”No” for the question ”Have you seen Goodfellas?”, it is easy for human subjects to understand the answer of the participants. However, this is really a difficult task for an automatic system. Therefore, the system identified the incorrect passage. Consequently, its true-false answer is determined by chance, that is true in this case. Question number 8 is also a deductive question because it is not easy to match the question ”I dislike Quentin” with the text ”I am not a huge fan of Quentin”. However, the system gave the true answer for this question while it was difficult for human subjects. That is because the keyword Quentin appears only one time in the transcript and the system was based on this word, but not based on the meaning of the essential phrase to identify the correct passage.

Table 6.8: Comparison with BET scores by human subjects for IS1008c

curQuid	Humans				System		
	Precis1	avg time1	Precis2	avg time2	#cpassage	#canswer	#time
1	0.86	410	0.93	127.36	1	1	13
2	0.67	298.58	0.86	129.5	1	1	45
3	0.82	78.09	0.93	67.5	1	1	15
4	0.89	80.22	0.93	103.93	1	1	16
5	0.63	66.38	0.69	63.92	1	0	20
6	0.67	44	0.73	62.18	0	0	10
7	1.00	24	0.82	48	1	0	11
8	0.67	66	0.64	93.55	0	1	11
	0.77	**133.41**	**0.81**	**86.99**	**0.75**	**0.63**	**17.63**

For IS1008c, as defined above, for easy and less easy questions there are two less easy questions for human subjects. They are questions numbered 5 and 8. For the system, it incorrectly answered three questions that are questions numbered 5, 6,7 and 8, in which the questions numbered 5, 6 and 8 are deductive questions. For question numbered 5, whose true statement is *Agnes express her opinion that ...*, the correct passage should be *Agnes: I think ...* . Two different expressions make it difficult to understand for both human subjects and the automatic system. With regard to question number 6, which is ”Agnes notes some reasons to not have a display”, Agnes showed a list of reasons in the transcript but there is few matched words between question string and answer string. This is similar with question number 8. However, the true-false answer for question numbered 8 is correct by chance. Question number 7 is not difficult, so the system gave correct passage but incorrect true-false answer. That means true-false answers by the system are not as stable as correct passage answering.

In conclusion, although both human subjects and the system encounter difficulties to answer deductive questions, it seems be more difficult for the automatic system. For IB4010, there are 3 deductive questions and the automatic system incorrectly answered 2 out of 3 questions, while at the same time, the human subjects have difficulty only answering 1 of the 3 questions. For IS1008c, there are also 3 deductive questions. The system gave the wrong answers for all three questions, meanwhile the human subjects had difficulty answering two questions. In fact, deductive questions are equivalent to How and Why questions that are difficult for all question answering systems [42, 43].

According to the experimental results, the results for the passage retrieval are more logical than the results for true-false answers. That means the system should be developed to help humans answer BET-typed questions by identifying relevant passage instead of giving the final answers. In other words, it is a useful tool for locating the answer, but not necessarily finding it.

Chapter 7

Future Research

The results of the passage retrieval propose a promising assistant tool for meeting browsers. This automatic tool integrated into a meeting browser could help users locate relevant information in a short amount of time so that they can save time to reason out the answer to a question.

This is the first attempt for building an automatic meeting browser following to question answering approach. Thus, this system can be developed by adding a *answer extraction* stage after the passage retrieval stage in order to extract a short phrase that expresses the answer instead of giving an answer *true* or *false*. However, this requires more significant research on semantic text.

Chapter 8

Conclusion

The performance of the automatic true-false question answering system is quite below that of humans using existing browsers. However, the scores of passage retrieval stage are a lot better than random scores, in are obtained in a very short time (less than 1s per question).

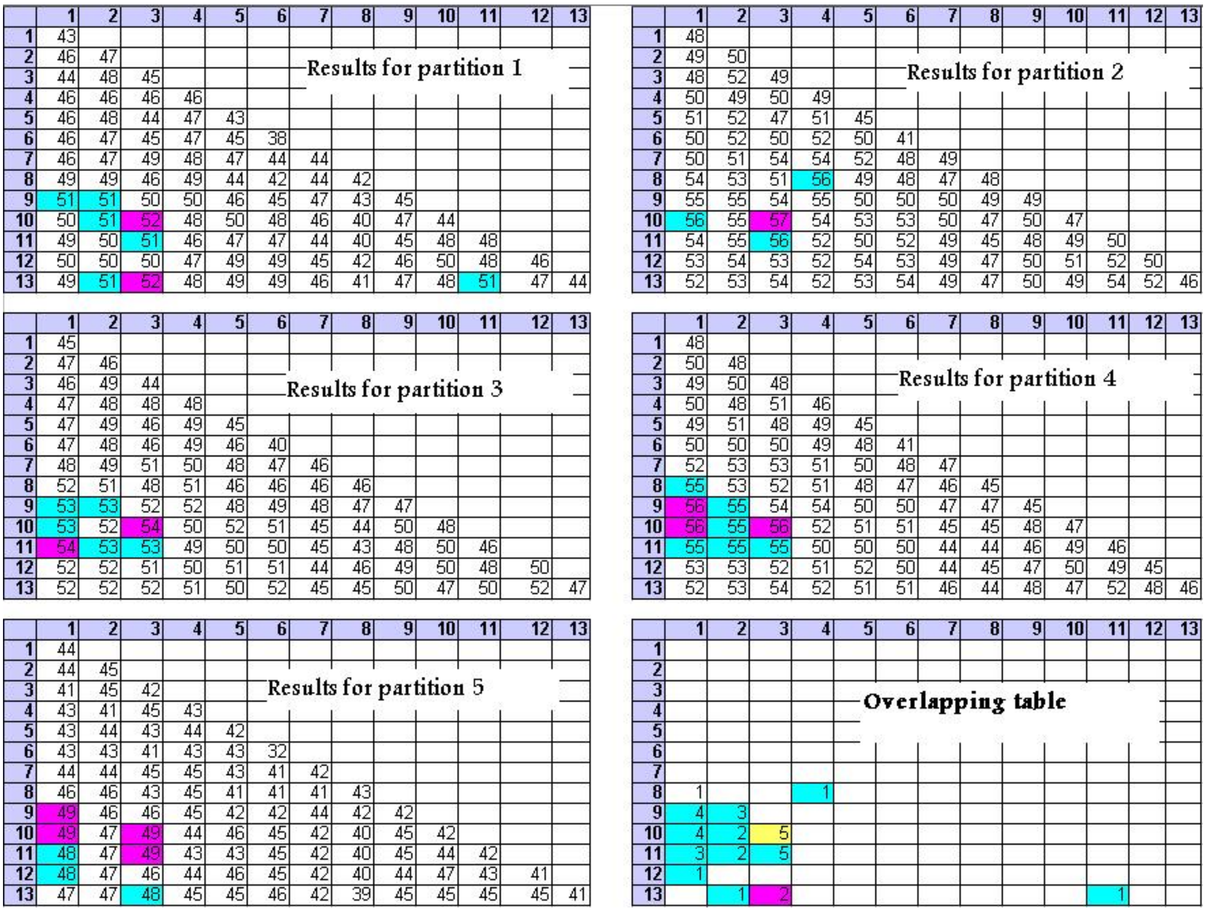
The human subjects answer questions that require a deduction or a reflexion better than the system does, but the system gives the answers much more quickly. Thus, the automatic system should give consultative information for a question given by users rather than return the answer in a fully autonomous way.

In conclusion, this project opens a new starting point to develop the feasibility of a fully-automatic question-answering system for meeting browsers. The lexical similary methodology may not be able to solve completely this problem. Nevetheless, it is an open problem and need further researches. Within 6 months of doing this project, I do not have ambition to study and find out a perfect solution but try to experience with simple and basic one.

Appendix

8.1 Intermediate results for parameter optimisation

8.1.1 IB4010



Results for partition 1

	1	2	3	4	5	6	7	8	9	10	11	12	13
1	43												
2	46	47											
3	44	48	45										
4	46	46	46	46									
5	46	48	44	47	43								
6	46	47	45	47	45	38							
7	46	47	49	48	47	44	44						
8	49	49	46	49	44	42	44	42					
9	51	51	50	50	46	45	47	43	45				
10	50	51	52	48	50	48	46	40	47	44			
11	49	50	51	46	47	47	44	40	45	48	48		
12	50	50	50	47	49	49	45	42	46	50	48	46	
13	49	51	52	48	49	49	46	41	47	48	51	47	44

Results for partition 2

	1	2	3	4	5	6	7	8	9	10	11	12	13
1	48												
2	49	50											
3	48	52	49										
4	50	49	50	49									
5	51	52	47	51	45								
6	50	52	50	52	50	41							
7	50	51	54	54	52	48	49						
8	54	53	51	56	49	48	47	48					
9	55	55	54	55	50	50	50	49	49				
10	56	55	57	54	53	53	50	47	50	47			
11	54	55	56	52	50	52	49	45	48	49	50		
12	53	54	53	52	54	53	49	47	50	51	52	50	
13	52	53	54	52	53	54	49	47	50	49	54	52	46

Results for partition 3

	1	2	3	4	5	6	7	8	9	10	11	12	13
1	45												
2	47	46											
3	46	49	44										
4	47	48	48	48									
5	47	49	46	49	45								
6	47	48	46	49	46	40							
7	48	49	51	50	48	47	46						
8	52	51	48	51	46	46	46	46					
9	53	53	52	52	48	49	48	47	47				
10	53	52	54	50	52	51	45	44	50	48			
11	54	53	53	49	50	50	45	43	48	50	46		
12	52	52	51	50	51	51	44	46	49	50	48	50	
13	52	52	52	51	50	52	45	45	50	47	50	52	47

Results for partition 4

	1	2	3	4	5	6	7	8	9	10	11	12	13
1	48												
2	50	48											
3	49	50	48										
4	50	48	51	46									
5	49	51	48	49	45								
6	50	50	50	49	48	41							
7	52	53	53	51	50	48	47						
8	55	53	52	51	48	47	46	45					
9	56	55	54	54	50	50	47	47	45				
10	56	55	56	52	51	51	45	45	48	47			
11	55	55	55	50	50	50	44	44	46	49	46		
12	53	53	52	51	52	50	44	45	47	50	49	45	
13	52	53	54	52	51	51	46	44	48	47	52	48	46

Results for partition 5

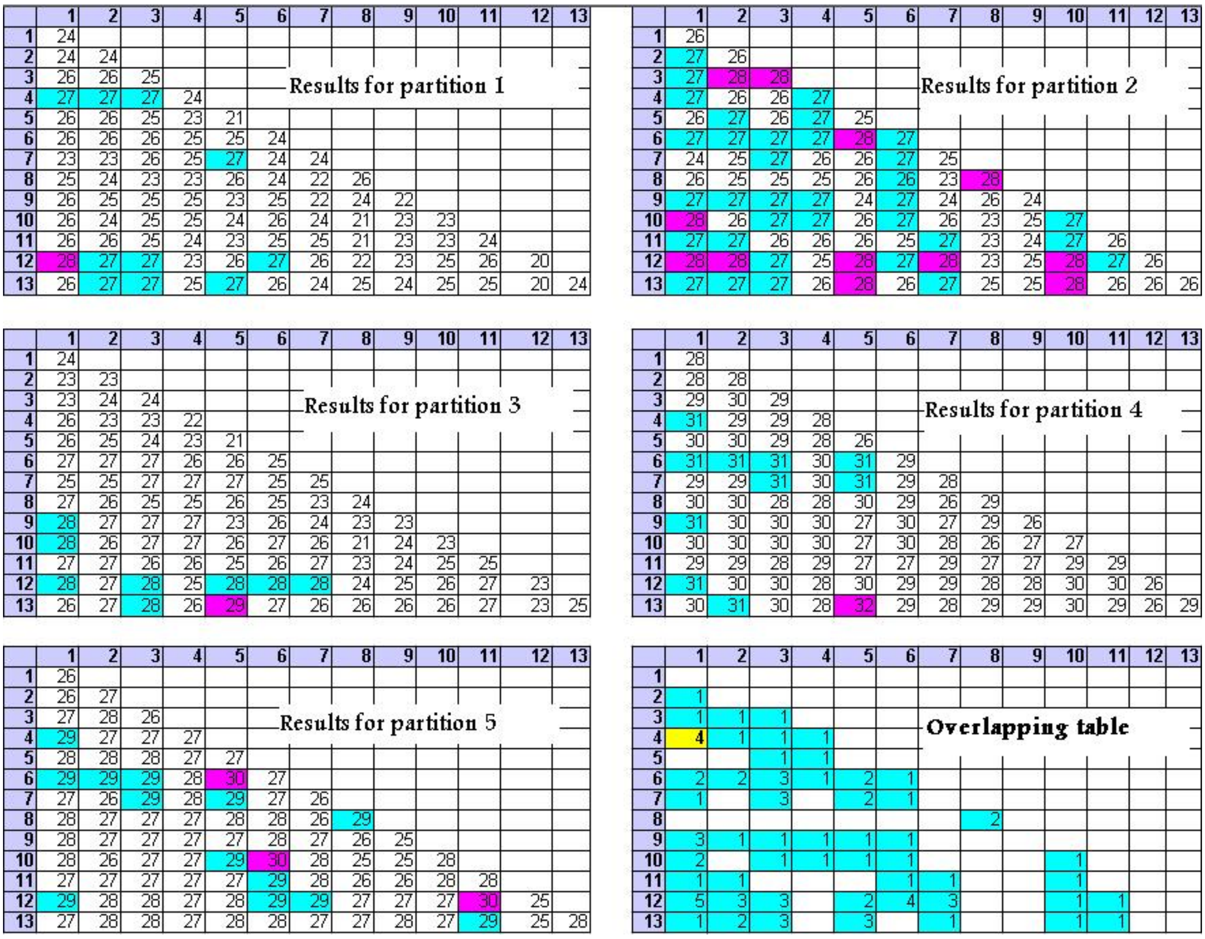
	1	2	3	4	5	6	7	8	9	10	11	12	13
1	44												
2	44	45											
3	41	45	42										
4	43	41	45	43									
5	43	44	43	44	42								
6	43	43	41	43	43	32							
7	44	44	45	45	43	41	42						
8	46	46	43	45	41	41	41	43					
9	49	46	46	45	42	42	44	42	42				
10	49	47	49	44	46	45	42	40	45	42			
11	48	47	49	43	43	45	42	40	45	44	42		
12	48	47	46	44	46	45	42	40	44	47	43	41	
13	47	47	48	45	45	46	42	39	45	45	45	45	41

Overlapping table

	1	2	3	4	5	6	7	8	9	10	11	12	13
1													
2													
3													
4													
5													
6													
7													
8	1			1									
9	4	3											
10	4	2	5										
11	3	2	5										
12	1												
13		1	2								1		

Figure 8.1: Using 5-fold Cross-Validation for parameter optimisation for IB4010

8.1.2 IS1008c



Results for partition 1

	1	2	3	4	5	6	7	8	9	10	11	12	13
1	24												
2	24	24											
3	26	26	25										
4	27	27	27	24									
5	26	26	25	23	21								
6	26	26	26	25	25	24							
7	23	23	26	25	27	24	24						
8	25	24	23	23	26	24	22	26					
9	26	25	25	25	23	25	22	24	22				
10	26	24	25	25	24	26	24	21	23	23			
11	26	26	25	24	23	25	25	21	23	23	24		
12	28	27	27	23	26	27	26	22	23	25	26	20	
13	26	27	27	25	27	26	24	25	24	25	25	20	24

Results for partition 2

	1	2	3	4	5	6	7	8	9	10	11	12	13
1	26												
2	27	26											
3	27	28	28										
4	27	26	26	27									
5	26	27	26	27	25								
6	27	27	27	27	28	27							
7	24	25	27	26	26	27	25						
8	26	25	25	25	26	26	23	28					
9	27	27	27	27	24	27	24	26	24				
10	28	26	27	27	26	27	26	23	25	27			
11	27	27	26	26	26	25	27	23	24	27	26		
12	28	28	27	25	28	27	28	23	25	28	27	26	
13	27	27	27	26	28	26	27	25	25	28	26	26	26

Results for partition 3

	1	2	3	4	5	6	7	8	9	10	11	12	13
1	24												
2	23	23											
3	23	24	24										
4	26	23	23	22									
5	26	25	24	23	21								
6	27	27	27	26	26	25							
7	25	25	27	27	27	25	25						
8	27	26	25	25	26	25	23	24					
9	28	27	27	27	23	26	24	23	23				
10	28	26	27	27	26	27	26	21	24	23			
11	27	27	26	26	25	26	27	23	24	25	25		
12	28	27	28	25	28	28	28	24	25	26	27	23	
13	26	27	28	26	29	27	26	26	26	26	27	23	25

Results for partition 4

	1	2	3	4	5	6	7	8	9	10	11	12	13
1	28												
2	28	28											
3	29	30	29										
4	31	29	29	28									
5	30	30	29	28	26								
6	31	31	31	30	31	29							
7	29	29	31	30	31	29	28						
8	30	30	28	28	30	29	26	29					
9	31	30	30	30	27	30	27	29	26				
10	30	30	30	30	27	30	28	26	27	27			
11	29	29	28	29	27	27	29	27	27	29	29		
12	31	30	30	28	30	29	29	28	28	30	30	26	
13	30	31	30	28	32	29	28	29	29	30	29	26	29

Results for partition 5

	1	2	3	4	5	6	7	8	9	10	11	12	13
1	26												
2	26	27											
3	27	28	26										
4	29	27	27	27									
5	28	28	28	27	27								
6	29	29	29	28	30	27							
7	27	26	29	28	29	27	26						
8	28	27	27	27	28	28	26	29					
9	28	27	27	27	27	28	27	26	25				
10	28	26	27	27	29	30	28	25	25	28			
11	27	27	27	27	27	29	28	26	26	28	28		
12	29	28	28	27	28	29	29	27	27	27	30	25	
13	27	28	28	27	28	28	27	27	28	27	29	25	28

Overlapping table

	1	2	3	4	5	6	7	8	9	10	11	12	13
1													
2	1												
3	1	1	1										
4	4	1	1	1									
5			1	1									
6	2	2	3	1	2	1							
7	1		3		2	1							
8								2					
9	3	1	1	1	1	1							
10	2		1	1	1	1				1			
11	1	1				1	1			1			
12	5	3	3		2	4	3			1	1		
13	1	2	3		3		1			1	1		

Figure 8.2: Using 5-fold Cross-Validation for parameter optimisation for IS1008c

8.2 List of stopwords

i, a, about, above, an, are, as, at, am, and, be, been, being, but, by, do, does, done, did, for, he, her, hers, herself, his, him, himself, how, in, is, it, its, itself, me, my, mine, myself, nor, of, on, or, our, ours, ourself, ourselves, so, she, that, the, they, them, their, theirs, these, themself, themselves, this, those, to, uh, um, up, us, really, very, was, were, we, well, will, with, what, when, where, which, who, whom, whose, why, yet, you, your, yours, yourself, yourselves.

Bibliography

[1] P. Wellner, M. Flynn, S. Tucker, and S. Whittaker. A meeting browser evaluation test. In *Conference on Human Factors in Computing Systems*, pages 2021–2024. ACM New York, NY, USA, 2005.

[2] J. Carletta, S. Ashby, S. Bourban, M. Flynn, M. Guillemot, T. Hain, J. Kadlec, V. Karaiskos, W. Kraaij, M. Kronenthal, et al. The AMI Meeting Corpus: A Pre-announcement. *LECTURE NOTES IN COMPUTER SCIENCE*, 3869:28, 2006. URL http://www.springerlink.com/index/yg77316944656462.pdf.

[3] A. Popescu-Belis. Objective Test for Meeting Browsers: the BET4TQB Pilot Experiment. *H., B. and S., R., editors, to appear in Proceedings of Machine Learning for Multimodal Interaction IV, LNCS, Brno, Czech Republic. Springer*, 2007.

[4] Andrei Popescu-Belis. URL http://www.idiap.ch/~apbelis/.

[5] Jean-Paul LECLERCQ. URL http://www.fundp.ac.be/universite/personnes/page_view/01000350/.

[6] Jean-Marie Jacquet. URL http://www.fundp.ac.be/universite/personnes/page_view/01002715/.

[7] A. Popescu-Belis, M. Flynn, P. Wellner, and P. Baudrion. Task-based evaluation of meeting browsers: from task elicitation to user behavior

analysis. URL http://www.lrec-conf.org/proceedings/lrec2008/pdf/580_paper.pdf.

[8] I. McCowan, J. Carletta, W. Kraaij, S. Ashby, S. Bourban, M. Flynn, M. Guillemot, T. Hain, J. Kadlec, V. Karaiskos, et al. The AMI Meeting Corpus. In *Proceedings of Measuring Behavior*, 2005. URL http://www.idiap.ch/~mccowan/publications/mccowan-mb2005.pdf.

[9] D. Lalanne, A. Lisowska, E. Bruno, M. Flynn, M. Georgescul, M. Guillemot, B. Janvier, S. Marchand-Maillet, M. Melichar, N. Moenne-Loccoz, et al. The IM2 Multimodal Meeting Browser Family. *Interactive Multimodal Information Management Tech. Report, Margtiny, Switzerland*, 2005.

[10] A. Lisowska, M. Rajman, and T.H. Bui. ARCHIVUS: A System for Accessing the Content of Recorded Multimodal Meetings. In *In Procedings of the JOINT AMI/PASCAL/IM2/M4 Workshop on Multimodal Interaction and Related Machine Learning Algorithms, Bourlard H. & Bengio S., eds.(2004), LNCS, Springer-Verlag, Berlin*. Springer, 2004.

[11] P. Wellner, M. Flynn, and M. Guillemot. Browsing recorded meetings with Ferret. *Machine Learning for Multimodal Interaction*, pages 12–21, 2004.

[12] A. Popescu-Belis and M. Georgescul. TQB: Accessing Multimodal Data Using a Transcript-based Queryand Browsing Interface.

[13] V. Wan, M. Karafiát, and S. Renals. Speech Recognition on M4. In *Workshop on Multimodal Interaction and Related Machine Learning Algorithms*, pages 21–23, 2004. URL http://www.dcs.shef.ac.uk/~vinny/docs/pdf/asr.mlmi2004-poster.pdf.

[14] T. Kato, F.M. Junichi Fukumoto, and N. Kando. Handling information access dialogue through QA technologiesA novel challenge for open-domain question answering. In *Proceedings of the HLT-NAACL 2004 Workshop on Pragmatics of Question Answering*, pages 70–77, 2004. URL http://acl.ldc.upenn.edu/W/w04/W04-2509.pdf.

[15] L. HIRSCHMAN and R. GAIZAUSKAS. Natural language question answering: the view from here. *Natural Language Engineering*, 7(04): 275–300, 2002. URL http://journals.cambridge.org/production/action/cjoGetFulltext?fulltextid=96168.

[16] RF Simmons. Answering English questions by computer: a survey. *Communications of the ACM*, 8(1):53–70, 1965. URL http://portal.acm.org/citation.cfm?id=363707.363732.

[17] S.E. Robertson, S. Walker, S. Jones, MM Hancock-Beaulieu, and M. Gatford. Okapi at TREC-4. In *Proceedings of the Fourth Text Retrieval Conference*, pages 73–97, 1996.

[18] S. Tellex and B. Katz. *Pauchok: A modular framework for question answering*. Massachusetts Institute of Technology, Dept. of Electrical Engineering and Computer Science, 2003.

[19] S. Tellex, B. Katz, J. Lin, A. Fernandes, and G. Marton. Quantitative evaluation of passage retrieval algorithms for question answering. In *Proceedings of the 26th annual international ACM SIGIR conference on Research and development in informaion retrieval*, pages 41–47. ACM New York, NY, USA, 2003. URL http://portal.acm.org/citation.cfm?id=860445.

[20] M.W. Bilotti, B. Katz, and J. Lin. What works better for question answering: Stemming or morphological query expansion. In *Proceedings*

of the Information Retrieval for Question Answering (IR4QA) Workshop at SIGIR 2004, 2004.

[21] N. Goharian and S.S.R. Mengle. On document splitting in passage detection. In *Proceedings of the 31st annual international ACM SIGIR conference on Research and development in information retrieval*, pages 833–834. ACM New York, NY, USA, 2008. URL http://portal.acm.org/citation.cfm?id=1390528.

[22] M. LIGHT, G.S. MANN, E. RILOFF, and E. BRECK. Analyses for elucidating current question answering technology. *Natural Language Engineering*, 7(04):325–342, 2002.

[23] G.G. Lee, S. Lee, H. Jung, BH Cho, C. Lee, BK Kwak, J. Cha, D. Kim, J. An, J. Seo, et al. SiteQ: Engineering high performance QA system using lexico-semantic pattern matching and shallow NLP. *NIST SPECIAL PUBLICATION SP*, pages 442–451, 2002.

[24] S. Robertson. Understanding inverse document frequency: on theoretical arguments for IDF. *Journal of Documentation*, 60:503–520, 2004.

[25] M. Beaulieu, M. Gatford, X. Huang, SE Robertson, S. Walker, and P. Williams. Okapi at TREC-3. In *Overview of the Third Text REtrieval Conference (TREC-3)*, 1995.

[26] X. Xue, J. Jeon, and W.B. Croft. Retrieval models for question and answer archives. In *Proceedings of the 31st annual international ACM SIGIR conference on Research and development in information retrieval*, pages 475–482. ACM New York, NY, USA, 2008.

[27] P.R. Comas and J. Turmo. Spoken document retrieval based on approximated sequence alignment. In *11th International Conference on Text, Speech and Dialogue (TSD 2008)*. Springer, 2008.

[28] A.M. Robertson and P. Willett. Applications of n-grams in textual information systems. *Journal of Documentation*, 54(1):48–67, 1998. URL http://www.ingentaconnect.com/content/mcb/278/1998/00000054/00000001/art00003.

[29] H. Cui, R. Sun, K. Li, M.Y. Kan, and T.S. Chua. Question answering passage retrieval using dependency relations. In *Proceedings of the 28th annual international ACM SIGIR conference on Research and development in information retrieval*, pages 400–407. ACM New York, NY, USA, 2005.

[30] M. Flynn and P. Wellner. In Search of a Good BET: A proposal for a Browser Evaluation Test. *IDIAP-COM 03*, 11, 2003.

[31] A. Popescu-Belis, P. Baudrion, M. Flynn, and P. Wellner. Towards an Objective Test for Meeting Browsers: The BET4TQB Pilot Experiment. *LECTURE NOTES IN COMPUTER SCIENCE*, 4892:108, 2008.

[32] J. McKechnie, S. Shaaban, and S. Lockley. Computer assisted processing of large unstructured document sets: a case study in the construction industry. In *Proceedings of the 2001 ACM Symposium on Document engineering*, pages 11–17. ACM Press New York, NY, USA, 2001. URL http://portal.acm.org/citation.cfm?id=502190.

[33] L. Hirschman, M. Light, E. Breck, and J.D. Burger. Deep Read: a reading comprehension system. In *Proceedings of the 37th annual meeting of the Association for Computational Linguistics on Computational Linguistics*, pages 325–332. Association for Computational Linguistics Morristown, NJ, USA, 1999.

[34] M. Pasca and S. Harabagiu. The informative role of WordNet in Open-Domain Question Answering. In *Proceedings of NAACL-01 Workshop on WordNet and Other Lexical Resources*, pages 138–143, 2001.

[35] MF Porter. Snowball Stemmer, 2001.

[36] O. Manson. QTAG–A portable probabilistic tagger. *Corpus Research, The University of Birmingham, UK*, 1997.

[37] Le Quoc-Anh and Andrei Popescu-Belis. AutoBET: towards automatic answering of BET questions for meeting browser evaluation. In *IM2 Annual Review Meeting*, 2008.

[38] E.M. Voorhees. The TREC-8 Question Answering Track Report. *NIST SPECIAL PUBLICATION SP*, pages 77–82, 2000. URL http://trec.nist.gov/pubs/trec8/papers/qa_report.pdf.

[39] E.M. Voorhees and D. Harman. Overview of TREC 2001. *NIST Special Publication*, pages 500–250, 2001. URL http://www-nlpir.nist.gov/trec/pubs/trec10/paper.

[40] R. Kohavi. A Study of Cross-Validation and Bootstrap for Accuracy Estimation and Model Selection. In *INTERNATIONAL JOINT CONFERENCE ON ARTIFICIAL INTELLIGENCE*, volume 14, pages 1137–1145. LAWRENCE ERLBAUM ASSOCIATES LTD, 1995.

[41] G. Murray and S. Renals. Term-Weighting for Summarization of Multiparty Spoken Dialogues. *LECTURE NOTES IN COMPUTER SCIENCE*, 4892:156, 2008. URL http://www.springerlink.com/index/8x72j8787044757x.pdf.

[42] J. Prager, E. Brown, A. Coden, and D. Radev. Question-answering by predictive annotation. In *Proceedings of the 23rd annual international ACM SIGIR conference on Research and development in information retrieval*, pages 184–191. ACM New York, NY, USA, 2000. URL http://portal.acm.org/citation.cfm?id=345574&dl=GUIDE,.

[43] E. Brill, J. Lin, M. Banko, S. Dumais, and A. Ng. Data-Intensive Question Answering. *NIST SPECIAL PUBLICATION SP*, pages 393–400, 2002. URL http://student.bu.ac.bd/~mumit/Research/NLP-bib/papers/Brill01.pdf.