Chapter 2: Literature Review

2.1 Outline

This chapter presents a general background to the research field of question answering, its aims, issues, history, the typical architecture of question answering systems, an overview of the question answering track in the Text REtrieval Conference (TREC), and current approaches to open-domain question answering. The following section provides theoretical background on definitions and is followed by a review of previous work on answering definition questions. Finally, the definition question subtask in TREC-12 is described.

2.2 Question Answering

2.2.1 The Aims of Question Answering

An information retrieval system 'merely informs on the existence (or non-existence) and whereabouts of documents relating to' the request of the user (Lancaster, 1968, cited in van Rijsbergen, 1979). On the other hand, a question answering system returns an exact answer in response to the request. More specifically, question answering systems attempt to allow users to ask a question in natural language and receive a concise answer, possibly with enough validating context (Hirschman and Gaizauskas, 2001).

It is more natural for users to type a question in a human language, such as 'Who wrote the Pope's Rhinoceros?' than to type an expression like **`(write OR written OR author) AND (Pope's Rhinoceros)`** (Radev et al., 2002). Indeed, a look at the query logs of Web search engines reveals that 12-15% of the queries consist of questions in natural language (de Rijke and Webber, 2003), even though search engines often reduce such questions to a 'bag of words'. For example, the search engine Google (Google, 2003) will treat the question 'When did the Tasmanian tiger become extinct?' as a keyword-based query (**`did AND Tasmanian AND Tiger AND become AND extinct`**), dropping the stopwords 'When' and 'the'. In the process, syntactic subtleties are lost as well. For example, submitting the query 'What is Limerick?' would

produce the exact results as 'What is a limerick?'. Sometimes, as in the Tasmanian tiger example, the search engine returns a page with the answer among the top-ranking pages, but the tedious task of pinpointing the answer within the document is left to the user (Kwok, Etzioni and Weld, 2001). Question answering systems could therefore reduce the information overload of users.

A recent roadmap document for question answering research (Hirschman and Gaizauskas 2001; Burger et al. 2002) identified five standards that users may expect from question answering systems:

1. **Timeliness**. The system should answer a question in real-time, even when accessed by thousands of users, and the data sources should be kept up-to-date.
2. **Accuracy**. Imprecise, incorrect answers are worse than no answers. The system should also discover and resolve contradictions in the data sources.
3. **Usability**. The knowledge in the system should be tailored to the needs of the user.
4. **Completeness.** When the answers are distributed across one or multiple documents, answers should be fused coherently.
5. **Relevance**. The answer should be relevant within a specific context. The evaluation of question answering system must be user-centred.

2.2.2 Dimensions of the Question Answering Problem

Hirschman and Gaizaukas (2001) list six dimensions of question answering research:

1. Applications,
2. Users,
3. Question types,
4. Answer types,
5. Evaluation,
6. Presentation.

1. Applications can be based on different sources of answers: structured data (databases), semi-structured data (for instance, health records or legal documents), or free text (the focus of current research in question answering). The source of answers can be fixed, as in the case of the TREC collection (see section 1.3), or dynamic, as for

the World-Wide Web. Another distinction between question answering applications is whether they are domain-specific or domain-independent. In addition, the source can be in a form other than free text, such as speech data, annotated images, or even video footage (Katz, Lin and Chris Stauffer, 2003).

2. Different users require different interfaces. Burger et al. (2002) name four levels of user sophistication. The level of user sophistication intersects with other issues in question answering. For example, Table 2.1 illustrates the problem of generating a coherent answer from distributed information for the four types of users.

3. Question may be of different types. Current research in question answering focuses on questions for which the answer is a brief fact or factoid, for example, 'What is the capital of Ireland?'. More difficult types of question include those which ask for opinion, Why and How questions, which require understanding of causality or instrumental relations, What questions which provide little constraint on the answer type, definition questions (see below), questions that require implicatures, disambiguation and reformulation, questions phrased as commands, questions with no answers, and questions requiring a list as an answer. (Moldovan et al., 2003) classify question answering systems into five increasingly sophisticated types according to the kind of methods used: systems based on factoids, systems with simple reasoning mechanisms, systems which can fuse answers from different documents, interactive systems, and systems capable of analogical reasoning. They argue that the more sophisticated apparently may allow the more difficult types of questions to be answered.

4. Answers may vary in length. Justification requires a longer answer, and recent research (Lin et al., 2003) indicates that users prefer a paragraph-sized chunk of text to just an exact phrase. An answer can be created by cutting and pasting original snippets (extraction) or by coherent synthesis of snippets from multiple sentences (generation).

5. Evaluation methods can use different criteria for judging answers (for example, relevance, correctness, completeness, conciseness).

6. The presentation of question answering can be in the form of a dialogue in which users gradually narrow their search. Indeed, speech input (and output) could be used.

	Level 1 *"Casual Questioner"*	**Q:** When was Queen Victoria born?	**Text 1:** Queen Victoria (1854-1889) ruled Britain with an iron fist …. **Text 2:** British monarchs: Victoria 1832-1889 Edward 1874-1946 Elizabeth 1923- **Answer: 1832**
	Level 2 *"Template Questioner"*	**Q:** How many casualties were reported last week in Fredonia?	**Text 1:** Last Monday two people were killed on the streets of Beautiville, Fredonia, after a bomb exploded **Text 2:** The terrorists murdered a family with a small child in Fredonia last Friday, near its border with Evilonia. The father just returned home the day before. **Answer: five people**
	Level 3 *"Cub reporter"*	**Q:** How many U.S. households have a computer?	**Text 1:** Two families in three are connected to the Internet in the U.S. **Text 2:** Last year, IRS has received 150 million individual return forms. **Answer: 90 million**
	Level 4 *"Professional Information Analyst"*	**Q:** Why there were hacker attacks on the computers at University of California, Santa Barbara?	**Text 1:** U.S. colleges have powerful computing facilities. **Text 2:** Computer hackers need speedy processors to break security passwords. **Answer: To use their computers for password cracking**

Table 2.1: Illustration of different challenging answer extraction instances at four levels of questioner sophistication (taken from Burger et al, 2002).

2.2.3 Historical Background of Question Answering

Natural Language Question answering is not a new area. Research began as early as 1959, while attempts to build machines to test logical consistency date back at least to the Catalan mystic and poet Ramon Lull (ca. 1235-1316) who built a set of wheels called the *Ars Magna* (Great Art), a machine supposed to answer all questions (Nilsson, 1998). Logic machines for testing the validity of propositions were built in the nineteenth century, but they did not deal directly with natural languages. Simmons

(1965) reviews no fewer than fifteen systems, built over the preceding five years, which answer some type of English question. The systems included a conversational question answerer (The Conversation Machine, The Oracle), front-ends to structured databases (SAD SAM, DEACON, BASEBALL), and text-based systems (Protosynthex, Automatic Language Analyzer).

BASEBALL was one of the most sophisticated of the early question answering systems. It answered questions about scores, teams, locations, and dates of baseball games (Green et al., 1961). The system syntactically analysed questions such as 'Where did the Red Sox play on July 7?' to the extent necessary to create a frame-like representation called a 'specification list'. It then evaluated that specification against a hierarchically organised database, and returned an answer in outline form (Grosz, Jones and Webber, 1986). BASEBALL was modular; question read-in, dictionary look-up, syntactic analysis, specification list construction, database evaluator and responder modules processed the input in strict sequence. Its creators were not concerned with transportablity, extensibility or speed, but BASEBALL proved that question answering systems were possible.

A similar user-friendly front-end to structured data was LUNAR, which allowed access to chemical data on lunar material collected during the Apollo moon missions (Hirschman and Gaizauskas, 2001). The system consisted of a general purpose grammar and parser, a rule driven semantic interpretation component, and a database retrieval and inference module (Woods, 1978). The system contained a dictionary of about 3,500 words and two databases: a table of chemical analyses with 13,000 entries, and a topic index to documents with about 10,000 postings. The system also contained components for morphological analysis of regularly inflected words, for maintaining discourse directory of anaphoric expressions and for determining what information to display. During a lunar science convention in 1971 it answered 90% of questions posed by geologists (sophisticated, demanding users), without limiting the question phrasing. However, like BASEBALL, LUNAR was restricted to a narrow domain and its building was labour-intensive.

The LADDER and TEAM systems were developed in the 1970s (Grosz et al., 1986). LADDER (Language Access to Distributed Date with Error Recovery) provided access to information distributed over various databases. It consisted of three modules: 1.

INLAND, translating a query into a command list of constraints on and requests for database field values, 2. IDA, translating the command list into a sequence of queries against individual files, 3. FAM, locating the files and managing access to them. LADDER extended the system's linguistic coverage through synonyms and paraphrases. TEAM (Transportable English database Access Medium) descended from LADDER and tried to maximise transportability. TEAM had a domain-dependent parser, a grammar, a basic vocabulary, semantic representation routines, a basic sort/type taxonomy, pragmatic processes for resolving vague predicates, a quantifier scope algorithm, and a schema translator. None of these had to be changed when moving to a new domain or database.

Systems like BASEBALL, LUNAR, LADDER, and TEAM were limited to a structured knowledge base, while the current focus of question answering research is interrogation of open-ended collections of unstructured texts. However, these systems involved valuable work on the syntactic and semantic analysis of questions and pragmatics of user-system interaction.

Two other types of early question answering systems were dialogue systems and reading comprehension systems. Dialogue systems are related to Alan Turing's test. In 1950, Turing proposed that a machine should be considered intelligent if a human being, communicating with it by teletype, could not distinguish it from another human being (Covington, 1994). In 1966, Joseph Weizenbaum published a computer program called ELIZA which appeared to pass Turing's test. It carried on a dialogue with the user, in English, as a psychological counsellor. ELIZA did not understand the input, but recognised patterns of words and responded, sometimes convincingly, with appropriate 'canned' answers. For example, if the user typed 'You are X' (where X is an adjective), ELIZA responded 'What makes you think I am X?'.

Early dialogue systems were built to explore the issues involved in modelling human dialogue and operated in a narrow domain. The work of (Winograd, 1972) on the SHRDLU system (named after the approximate order of frequency of the most commonly used letters in the English language, ETAOIN SHRDLU) made the artificial intelligence community consider for the first time that natural language was a serious part of their field (Schank, 1980). SHRDLU operated in the world of a toy robot with a simple arm. The arm could manipulate toy blocks on a table containing simple objects.

A user could ask the system through a dialogue to manipulate the objects (for example, 'Put a small one onto the green cube which supports a pyramid'), and question it about current configurations of the blocks, about events during the discussion and to a limited extent about its reasoning (for example, 'Why did you clear off the cube?'). SHRDLU displayed a simulated robot world on a television screen and conversed with a human on a teletype. Restricting the subject addressed the general issues of how language is used in a framework of physical objects, events and ongoing discourse. The system included a syntactic parser, a collection of semantic routines and a cognitive deductive program. Another set of routines generated appropriate English responses. The system answered questions (for example, 'How many blocks are not in the box?'), but also followed commands

Another early dialogue system was GUS (Genial Understander System), which was intended to engage a sympathetic and cooperative human in a dialogue, directed towards a specific goal within a restricted domain of discourse (Bobrow et al., 1977). GUS acted as a travel agent helping a user to plan a trip to a city in California. In the design of the system the emphasis was on modularity, because its designers wanted to see if a dialogue system, despite its complexity could be made modular. GUS understood incomplete and indirect utterances because it set up strong expectations about the user's response (based on the question, the limited domain, and the user's assumed goals).

Turning to reading comprehension systems, researchers realised that the same tests given to children could be used to evaluate natural language understanding systems. The most salient work in this field was that of Roger Schank and Wendy Lehnert (Schank and Abelson, 1977; Lehnert, 1978). Lenhert used Schank's framework of scripts and plans, which modelled human story understanding, to develop a theory of question answering and implement it in QUALM. This system answered questions about stories understood by two other understanding systems, PAM (Plan Applier Mechanism) and SAM (Script Applier Mechanism). Lenhert wanted to move away from question answering systems which regarded the natural language interface as merely a front-end, independent of the information retrieval component. To understand questions, QUALM had to interface with a program that parsed the question into its Conceptual Dependency representation. To produce answers in English, QUALM also needed a generator that could translate Conceptual Dependency representations into English. All the processing specific to answering questions was done on a language-independent conceptual level,

so theoretically QUALM could interface with a parser in language X and a generator in language Y, and therefore understand questions in X and produce answers in Y. The interpretation of the questions also involved categorising it into one of thirteen Conceptual Categories such as Causal Antecedent, Goal Orientation, Enablement, Quantification, Judgemental etc. For example, a question such as 'Do you have a light' should not be classified as a Verification, but as a Functional Request; Otherwise the system may answer the question with Yes or No. Another inference mechanism in QUALM added constraints on what constitutes an appropriate answer. For example, the system should not give an exhaustive list of the world's population as the answer to question "Who isn't coming to the party?". The significance of QUALM was in recognising that an appropriate answer is not always completely determined by the literal meaning of the question and in trying to reason about the function of the question and the appropriateness of particular answers.

Evaluating systems with reading comprehension school tests was revived recently with systems such as Deep Read (Hirschman et al., 1999) and Quarc (Riloff and Thelen, 2000). Both these systems return the sentences which answer the question best. Quarc uses heuristics to looking for lexical and semantic rules. Despite the lack of deep language understanding, the system finds the correct answer 40% of the time. Deep Read uses pattern matching with linguistic processing and achieves a similar performance of 30-40%.

The recent interest in question answering, rekindled largely by the Text REtrieval Conference (see below) and the World-Wide Web, has focused on open-domain queries (Radev et al., 2002). Early systems such as MURAX (Kupiec, 1993) used highly edited knowledge bases containing material such as newspaper articles and encyclopaedias to answer open-domain, mostly factoid type of questions. However, an increasing number of systems today use the Web as one source of knowledge. The Web offers a vast amount of freely available unstructured text and therefore data redundancy (multiple, differently phrased answer occurrences) which allows the finding of transparent answer strings and the selecting of answers based solely on their frequency (Dumais et al., 2002). However, the challenge is to cope with the large amount of irrelevant data. START (Katz, 1997) was one of the first systems with a Web interface, but focused on questions about geography and the MIT Infolab, whereas MULDAR (Kwok et al., 2001) was the first automated question-answering system that used the full Web as its

knowledge base. The commercial service Ask Jeeves (Ask Jeeves, 2003) provides a natural language question interface to the web, but it relies on human editors to match question templates with authoritative sites. Ask Jeeves is not strictly a question answering system, because it returns documents, not answers. Section 2.2.7 describes some of the open-domain systems developed in recent years.

2.2.4 Question Answering and Information Extraction

Information Extraction is defined as '…the activity of filling predefined templates from natural language texts, where the templates are designed to capture information about key role players in stereotypical events.' (Hirschman and Gaizauskas, 2001).

For example, a template for a film could include 'slots' for its title, year of release, director, producer, actor names, language, and genre. The result would be a structured database about films. The database can then be used for database queries, mining, and summarisation. We can view information extraction as a type of question answering in which the questions (templates) are static and the data source for the answers is a text collection.

Information extraction can indeed support question answering (Srihari and Li, 1999). The Message Understanding Conferences (MUCs), an evaluation exercise which ran between 1987 and 1998, set the standards in information extraction and provided the basis for using information extraction in question answering. MUC divided information extraction into distinct tasks: Named Entity (NE), Template Element (TE), Template Relation (TR), Co-reference (CO), and Scenario Templates (ST). Tagging text with an expanded list of MUC-defined NE types (PERSON, ORGANISATION, LOCATION, TIME, DATE, MONEY, and PERCENT) is a strategy used in question answering in order to identify possible answers to a question. For example, the answer to a Who question is likely to be an instance of a PERSON entity.

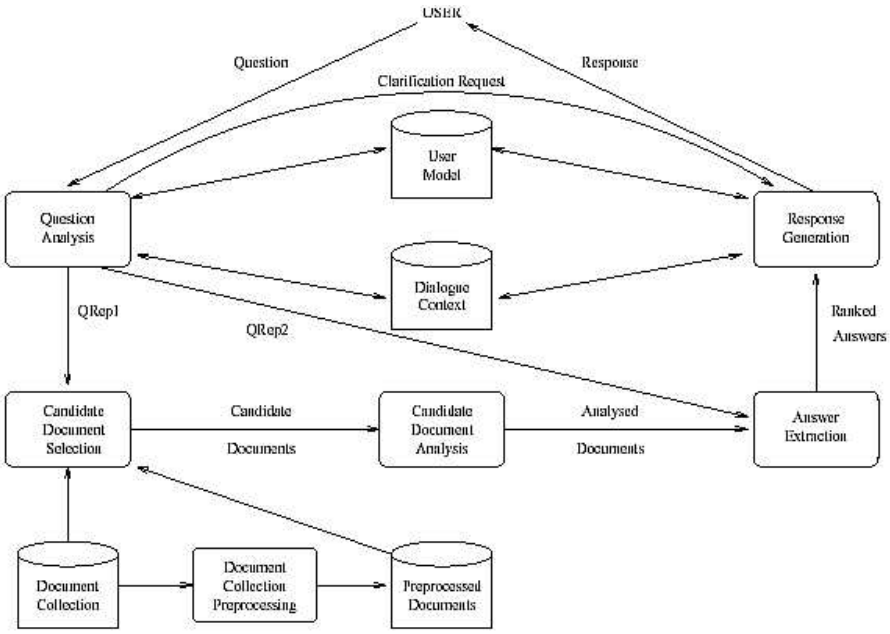

Figure 2.1: Generic architecture for a question answering system (Hirschman and Gaizauskas, 2001)

2.2.5 Typical Architecture of Question Answering Systems

Question answering systems typically employ a single pipeline architecture which consists of three main components: question analysis, search and answer selection (Chu-Carrol et al., 2002). Recent work has tried to improve performance by breaking away from this and either incorporating feedback loops or using multiple answering agents to process the same question in parallel.

Figure 2.1 shows a generic architecture for a question answering system (Hirschman and Gaizauskas, 2001). Not all question answering systems implement the full model (especially not the user or dialogue components). The following is a brief description of each stage:

1. **Question Analysis**. A question in natural language is analysed into forms used by subsequent parts of the system.
2. **Document Collection Pre-processing**. The collection is processed into a form which will allow question answering in real-time.
3. **Document Selection**. A subset of documents, likely to contain the answers, is

selected. The subset is typically several orders of magnitude smaller than the entire collection.

4. **Document Analysis**. A detailed analysis of the candidate documents may be needed if the pre-processing was superficial.
5. **Answer Extraction.** Answers are extracted from the documents and ranked according to the probability of being correct.
6. **Response Generation**. The system returns a response, possibly in a dialogue context.

Echihabi and Marcu (2003) claim that at their core all known question answering systems are a pipeline of only two modules: 1. An information retrieval engine that retrieves a subset of documents/sentences which may contain answers to a given question; 2. An answer identifier module that identifies a substring of the sentence that is likely to be an answer and assigns a score to it.

2.2.6 Question Answering in TREC

The Text REtrieval Conference (TREC) is sponsored by the American National Institute (NIST) and the Defense Advanced Research Projects Agency (DARPA) (Voorhees, 2003c: Voorhees, 2003d). The series of workshops started in 1992 in order to foster research within the information retrieval community and particularly to provide infrastructure needed for large-scale evaluation of text retrieval methods.

TREC contains different areas of focus called 'tracks'. Examples include Cross-Language Retrieval Track, Interactive Retrieval Track and Web Retrieval Track. Every year the organisers of TREC add new tracks (for example the Genome Track and High Accuracy Retrieval from Documents (HARD) Track in 2003) and modify existing ones.

TREC introduced an open-domain question answering track in 1999 (TREC-8). The task given to participants was to answer 200 fact-based short-answer questions such as 'How many calories are there in a Big Mac?', 'Who is the voice of Miss Piggy?', 'What language is commonly used in Bombay?'. Each question was guaranteed to have at least one document in the collection with an explicit answer. The test collection included

material from the Financial Times, the Federal Register, Foreign Broadcast Information, and the Los Angeles Times. The answer consisted of either a 50 or 250 byte snippet along with the identification number of the document (doc id.) from which it was retrieved. Systems were allowed to return up to five answers to each question.

Human assessors read the snippets and decided if they contained an answer to the question in the context of the document referred to. The score computed for a submission was the Mean Reciprocal Rank (MRR), the mean of the inverse rank at which the first correct answer was found. The answers were judged as either Correct, Unsupported or Incorrect. An answer string was judged as Correct when it was responsive to the question (i.e., it answered it), and the document supported the answer. An answer was judged Unsupported if it was correct but the document cited did not support the answer. Otherwise, the answer was judged Incorrect. Despite differences in judgements between assessors, the relative MRR between question answering system remains consistent.

The test set in TREC-9 (2000) consisted of 693 questions: 500 were collected from logs of search engines (independently of the TREC document collection), while 193 were reformulations of some of the 500.

The question answering track in TREC-10 (2001) introduced two new tasks. The List Task required systems to provide an answer consisting of items located in multiple documents. List questions are harder than the ones in the main task because the system is expected to avoid duplication and report each item once. Examples of list questions from TREC-10 are ‘Name 10 different flavors of Ben and Jerry's ice cream’ and ‘Name 30 people who have composed an opera’. There were 500 questions in the main task, and 25 in the list task. The second task added was the context task, which evaluated question answering within a particular scenario. Questions were grouped into different series, and the systems had to track the discourse objects across the questions. The track produced no interesting results and was dropped. TREC-10 also modified the main task to make it more realistic. The length limit of answer strings was reduced from 250 bytes (which proved not challenging enough) to 50 bytes. In addition, the questions in the main task were no longer guaranteed an answer in the document collection, and systems were expected to return the string `NIL` when no answer was found (marked as correct if the collection contained no known answer).

The TREC-11 (2002) track repeated the main and list tasks (with 500 and 25 questions, respectively), but now demanded the systems to return exact answers. The new demand stemmed from the difficulty to differentiate between responses which all contained the correct answers but varied in the amount of extraneous text and coherence. As a result, the human assessors could now assign the judgement Inexact to responses, which meant that the answer string contained the correct answer and correct document id, but contained more than just the answer, or an incomplete answer.

The track in 2002 also introduced a new document collection known as the AQUAINT Corpus of English Text. The corpus consists of about one million documents (3 gigabytes of text) from three sources: The AP newswire from 1998-2000, the New York Times newswire from 1998-2000 and the English portion of the Xinhua News Agency from 1996-2000.

In 2001 about quarter of the questions were unintentionally definition questions such as 'Who is Duke Ellington?' and 'What is angiotensin?', but because of the difficulty in evaluating their answers, questions of this type were dropped in 2002.

In 2002 systems were again required to return exactly one answer per question but in the submission file the questions had to be ordered from the most confident response to the least confident response. Systems could also submit a justification string, optionally. In 2002, 34 groups participated in the question answering track and submitted 75 runs: 67 in the main task, 8 in the list task. The best system in TREC-11 answered 415 of questions in the main task correctly (83%). In the list task the best system achieved average accuracy of 0.65, while others achieved poor results (0.06 to 0.15).

In 2003, the question answering track reintroduced definition questions and included 50 of them along with 37 list questions among the 500 questions of the main task (Voorhees, 2003a). This meant that for the first time there was a significant participation in the definition and list subtasks. The factoid and definition questions were drawn from the logs of AOL and MSNSearch logs, and NIST assessors created the list questions. The track added a passage task, which allowed systems to return text segments containing answers to factoid questions. In the main task, the type of question was tagged (factoid, definition, or list). Each question was evaluated separately, but a final

score was computed by combining the scores for each type of question. In 2003 the best system answered 70% of the factoid questions correctly and achieved an F measure score of 0.396 in the list task. Section 2.5.3 discusses the evaluation of definition questions in TREC-12. The final score for the main task run weighted the average of the three component scores:

FinalScore = ½*FactoidScore + 1/4 * (ListScore + DefinitionScore)
(Voorhees, 2003b)

The weight for the List and Definition sub-tasks was large enough to encourage participation in them.

2.2.7 Current Approaches to Open Domain Question Answering

Current state-of-the-art question answering systems are extremely complex (Echihabi and Marcu, 2003) and consist of tens of modules which retrieve information, parse sentences, pinpoint question-types, analyse semantics, reason, access external resources and rank answers. To demonstrate this complexity, we describe four leading question answering systems that participated in TREC (three from TREC-12 in 2003, and one system from TREC-11 in 2002). This section does not cover answering definition questions, which are discussed in Section 2.5.

The system of BBN Technologies (Xu et al., 2003a; Xu, Licuanan and Weischedel, 2003b) used an Hidden Markov Model-based information retrieval system to select documents that are likely to contain answers to a question. The question is then classified as one of 30 types such as persons, locations, numbers, and monetary amounts. Occurrences of named entities in the top documents that match the type of question comprise the pool of candidates for answers. BBN's information retrieval engine ranks the candidates by scoring every text window that has the candidate at the centre against the question. The candidate receives the score of the highest-scoring window. The candidates are then re-ranked using the following constraints: If the question asks for a number, the answer should quantify the same noun in the answers' context as in the question. For example, the number in the answer to 'How many dimples does a regulation golf ball have?' should quantify the noun 'dimples'. If the question asks for a sub-type of locations (for instance, state, city), the answer should be of that sub-type.

BBN's system in 2003 boosted the score of answers occurring multiple times in the corpus and introduced more constraints on answers to questions asking for dates, names of authors, inventors, and measurements. WordNet was used to match verbs (for example, 'Who killed X' is matched to 'Y shot X').

Both in 2002 and 2003, BNN used the World Wide Web to supplement the TREC corpora for question answering in two runs. The system submitted the question to the search engine Google in an exact form, rewritten as a declarative sentence, and in a non-exact form—as a conjunction of all the content words in the question. The most frequent appropriate entity was selected from the top 100 summaries returned by Google. The system then looked for the same entity in the highest ranked document from the TREC corpus to return the document identification number required in the track.

In 2003, using the Web improved BBN's score from 0.177 to 0.208 for factoid questions and from 0.087 to 0.097 for list questions.

The MultiText Group participating in TREC-11 (Clarke et al., 2002), used a statistical approach to answer selection, supported by a lightweight parse which categorises questions and generates search queries. The queries were then fed to a passage retrieval system. The group developed a passage-retrieval algorithm that can retrieve any document substring from a document. The score of the sub-string depends on its length, the number of question terms in it, and the relative weight assigned to each term. Each sub-string was expanded by *n* words on each side to provide context. The answer selection algorithm took into account the location of the answer candidates relative to the original sub-string (or 'hotspot') within each larger passage. The system retrieved passages from four corpora: TREC, a local terabyte Web corpus, a 27MB corpus with 330,000 trivia questions and answers, and a query-specific corpus which was generated by querying the AltaVista search engine. In the passages, answers were considered to be word n-grams of an appropriate named entity type corresponding to the question category. If the question could not be assigned to one of 48 categories, the answer candidate was an n-gram of one to five words within 30 words of the 'hotspot'. The entity extractor eliminated unacceptable or unlikely n-grams if they appeared only once in a single passage, began or ended with prepositions, or consisted primarily of stopwords and question terms. The ranking of n-gram answer candidates combined

redundancy (the number of distinct passages in which a candidate occurred) and information about the location of the candidate relative to the 'hotspot' in the passage.

The MultiText system supplemented its basic statistical approach with an Early Answering strategy. The Early Answering subsystem answered questions by referencing a structured database gathered from the Web (for example, biographies, trivia questions and answers, airports, country locations, country capitals and population, currency by country, animal names). If an answer could be found in the database, the system's task was reduced to finding a document in the TREC corpus where the question and the answer keywords appear in close proximity. Answers generated by the Early-Answering subsystem were always given precedence and were ranked first.

In the TREC-11 run, the early-answering sub-system answered 65 questions, of which 44 were correct. In the best run (36.8%) 126 uncategorised questions were answered by a purely statistical approach, and 27 of these were answered correctly. When applying the statistical strategy to the entire set of 500 TREC questions, 73 (14.6%) were answered correctly.

IBM's PIQUANT system (Chu-Carrol et al., 2002; Prager et al., 2003) adopted an easily extensible, modular architecture which allowed multiple agents to answer the same question in parallel so that the results could be combined. The system in 2003 included the following answering agents (the agents responsible for answering definition questions are described later in Section 2.5.4):

The **Predictive Annotation Answering Agent** was based on pre-processing of the text in which the corpus is indexed not only with keywords but with predictively annotated semantic classes such as person names, locations, and dates. The Question Analysis module determined the semantic type of the answer candidates, along with a set of keywords, which are used in a weighted search engine query.

The architecture of the **Statistical Answering Agent** was also of pipeline type. During query analysis, the query was classified into one of 32 types based on features like words, part-of-speech tags, bigrams, and question markers. Web pages were then retrieved from a search engine, and answers were extracted to be added to the query for retrieval from the AQUAINT corpus. The agent selected answers using a maximum

entropy model for chunk selection trained from true sentences of previous evaluations, followed by a maximum entropy chunk ranking model trained on the system's output for 5K questions.

The **Knowledge Server Portal** provided access to external structured knowledge sources by adapting the sources and presenting consistent choices to all the question answering components. For certain classes of questions (for example, populations and capitals) the answering agent recognised a number of ways to ask these questions and formulated a query to the structured knowledge base, which included public databases such as the US Geological Survey and WordNet. The agent then formulated a query that included the answer found in the knowledge base.

Sometimes adding the answer as a search term is not effective when the answers can be expressed in many different forms (for example the size of population) in the corpus. Instead, the system generated a set of answer candidates and then validated their range with the Cyc sanity checker (integrated with the Cyc knowledge base) which returned either 'in range', 'out of range' or 'don't know' verdicts. In 2003 the sanity checker was expanded to include more predicates and to return verdicts validating the correctness of answers (in addition to range validation). The expanded sanity checker was invoked in TREC-12 for 51 out of the 420 factoid questions, returning a result in 30 cases which would have been wrong without sanity checking. It rejected over 1,000 answers and validated about 100, only one of which was validated incorrectly. The system's score for factoid questions in 2003 was 0.298.

TextMap was the question answering system of the group from the University of Southern California (Echihabi et al., 2003). In common with the other systems, the question analyser determined the answer type, such as PROPER-PERSON, PHONE-NUMBER, or NP. The group built a typology of 185 types, organised into classes (Abstract, Semantic, Relational, Syntactic etc.). To bridge the gap between the wording of the question and that of the answer, TextMap reformulated the question following patterns. The reformulations generated more focused TREC and Web queries. The reformulation collection contained 550 assertions. The system in TREC-12 produced between one and more than 40 reformulations per question (5.03 reformulations on average). It submitted the queries to Google and TREC to retrieve 100 Web and 100 TREC documents, and a sentence module selected 100 sentences that were most likely

to contain correct answers from each collection.

Three answer selection modules pinpointed the correct answers in the resulting 200 sentences and scored them. The selection by the knowledge-based module was based on the degree of matching at the syntactic/semantic level between question and answer parse trees, and heuristics, which penalise answers for reasons such as semantic mismatch, vagueness, negation. This module had a limited amount of external knowledge: the WordNet hierarchy, internal quantity and calendar conversion routines, and abbreviation routines.

The development of the pattern-based answer selection module followed the surprising success of a system by Soubbotin and Soubbotin (2001) in TREC-10 which achieved the top MRR of 0.68 by using an extensive list of surface patterns (Ravichandran and Hovy, 2002). Unlike Soubbotin and Soubbotin, however, the system of the University of Southern California learnt patterns automatically. The learning consisted of two steps: Firstly, given an answer type from the TextMap ontology and a few instances of <Question; Answer> pairs, all the patterns (templates) that contained such a pair were extracted from the Web. Secondly, the precision of each pattern was calculated, keeping the patterns of high precision.

The pattern-based answer selection module used the patterns to find possible answers. The answers were ranked using a maximum-entropy-based framework.

The statistics-based answer selection model implemented a noisy-channel model. The model explains how answer-sentence parse trees are mapped into questions through a sequence of stochastic operations. The probability model is trained using a parameter estimation package which was developed for statistical machine translation.

A maximum-entropy-based re-ranker (using 48 feature functions) combined the output of the three answer selection modules into a ranked list. The features used could be component specific, redundancy-specific (i.e., a count of answer candidates in the collection), answer type specific (some answer selection modules answer particular question types better than others), or blatant error specific (e.g., a negative rule stating that answers usually do not contain personal nouns).

In the official TREC-12 evaluation TextMap scored 33.7% for factoid questions and 11.8% for list questions.

Further experiments using the Web as a corpus showed that good answer selection modules require as many sources of knowledge as possible together with good methods for their integration. The sources used by the knowledge-based answer selection module had more impact on the answer selection than the automatic training parameters in the pattern- and statistics-based systems. Yet, proper weighting of the contribution of the various knowledge sources was equally important.

Echihabi and Marcu (2003) observed that given the complexity of current question answering systems, it is difficult to assess the contribution of each module to performance. However, LCC (Language Computer Corporation), one of the highest scoring groups in TREC, analysed their system (Moldovan et al., 2003) in depth and concluded that their performance is directly related to the depth of natural language processing resources and depends on the tools used for answer finding. In the LCC system the performance bottlenecks were found to be question classification and keyword expansion.

2.3 Definitions

2.3.1 Motivation for Defining Terms

According to Swartz (1997) definitions improve our use of language by increasing our vocabulary, eliminating some kinds of ambiguity and reducing vagueness.

2.3.2 Types and Theories of Definitions in Philosophy

Swartz (1997) describes seven types of definitions:

1. **Stipulative definitions** specify how a term is to be used. They are sometimes used to introduce wholly new terms or to restrict a meaning in a particular context;
2. **Lexical, or dictionary, definitions** report common usage of terms;
3. **Precising definitions** refine the meaning of an established term whose meaning is vague in a context and which needs improvement;

4. **Theoretical definitions** define a term in relation to scientific hypotheses, but in fact such definitions are not limited to science;
5. **Operational definitions** define terms by the steps or operations used to measure them;
6. **Recursive definitions** consist typically of two parts: a 'basis' clause in which the term does not occur, and an 'inductive step' in which it does;
7. **Persuasive definitions** are intended to influence attitudes and 'generally do violence to the lexical definitions';

Philosophers also distinguish between defining intention—specifying a set of logically necessary and jointly sufficient conditions for the application of the term—and defining by extension—sampling things which are described by a term. Some philosophers argue that terms describing sense-perceptions such as colours or smells can only be understood by presenting instances from their extensions.

Sometimes it is difficult to determine the 'width' of intensional definitions. If an intensional definition admits too many members to the extension of that term, the definition is said to be too wide or broad. Some intensional definitions can be both too wide and too narrow if they admit things to the extension of a term which do not properly belong there and exclude things which do.

Early philosophers, in particular Plato and Aristotle, believed that words (definiendum) had 'true' meaning waiting to be discovered. In the modern sense of this theory of 'real' definition, discovering the definition of terms means discovering what criteria most language users adopt in applying the term.

The Classical Theory of Definition is a theory in modern philosophy with two tenets: 1. A 'proper' intensional definition states the logically necessary and sufficient conditions of the application of the term; 2. There are intensional definitions for each of the class terms (e.g., 'horse', 'house', 'musical instrument') used. Often though we cannot specify the intension even if we know the extension of a term very well. For example, we know what 'lemon' means but if we list 'yellow' as one of the necessary and sufficient conditions to make an object a lemon, this would mean that a fruit which is like a lemon in all other respects except its colour could not be a lemon. So in this case the term is 'cluster-concept'—'made of a number of conditions which generally are not singly

necessary and are jointly oversufficient'.

2.3.4 Definitions in Technical Writing

In technical writing, clear and accurate definitions are critical and ensure that readers understand key terms and concepts (Alred, T. Brusaw and Oliu, 2000). Technical writers can define terms formally or informally. A formal definition of a term classifies it and then specifies features that distinguish it from other members in the same class or category. This type of definition is common in dictionaries (see next section) and is called genus/species or genus/definiendum (Landau, 2001). For example, the term 'spoon' would be placed in the category 'an eating utensil' with the distinguishing features 'that consists of a small, shallow bowl on the end of a handle'.

In an informal definition, a familiar word or phrase is used as a synonym for an unfamiliar word or phrase. For example, 'an invoice is a bill' or 'Plants have a symbiotic, or mutually beneficial, relationships with certain kinds of bacteria'.

Sometimes, simple dictionary-like definitions are not enough, and the definition needs to be expanded with details, examples, comparisons, or other explanatory devices. The most common devices are

1. Extended definition, which explores a number of qualities of the item being defined,
2. Definition by analogy (useful when the readers are non-specialists), which links an unfamiliar concept with a simpler or more familiar one,
3. Definition by cause, which is used when a term can be defined best by an explanation of its cause,
4. Definition by components, which breaks down a concept into its parts to make a formal definition of a concept simpler,
5. Definition by exploration of origin, which in certain cases, especially when defining terms with Greek or Latin roots, can clarify the meaning of a term and make it more memorable,
6. Negative definition, which is effective only when the reader is familiar with the item with which it is contrasted.

2.3.4 Lexical/Dictionary Definitions

While philosophers are concerned with the internal coherence of the system of definition, lexicographers are concerned with practical definition and have the readers in mind (and space at a premium). Dictionaries define words, not concepts described by them. Definitions in dictionaries should adhere to the following principles (Landau, 2001):

- The term (definiendum) should be defined according to the class the word belongs to;
- The term should be distinguished from all other things in that class;
- The definition should capture the essence of the things defined concisely and unambiguously;
- The definition should be positive;
- All the words within a definition must be explained;
- The definition should not contain words more difficult to understand than the word defined—simpler should define difficult, not vice versa;
- The defined word should not appear in its definition, nor should derivations or combinations of the defined word unless they are separately defined. However, one part of speech may be used to define another;
- The definition must correspond to the part-of-speech of the definiendum;
- Circularity must be avoided—no word can be defined from its own family of words unless the related word is defined independently;
- The definition should be self-contained.

2.3.5 Appropriate Content of a Definition

Sarner and Carberry (1988) analysed cooperative dialogues in which the expert's goal is to help an information-seeker to solve a problem and suggested the following Principle of Usefulness:

1. The definition should be at a level which is high enough to be meaningful to users, is easily understood and is not more detailed than is appropriate for their current focus of attention;
2. The definition should be at a level which is low enough to be helpful to users but does not inform them of something they already know or give them

information which is unrelated to their current focus of attention.

2.4 Previous Work on Answering Definition Questions

Hearst (1992) identified a set of lexico-syntactic patterns in domain-independent free text that discover hyponymic lexical relationship between two or more noun phrases. Such patterns are frequent in many text genres, reliable in indicating the relation of interest (hyponymy), and can be recognised with little or no pre-encoded knowledge. One example is `such` NP_1 `as` NP_2 (where NP_2 would be a hyponym of NP_1). Hearst suggested the following algorithm for discovering new patterns automatically:

1. Decide on a lexical relation (not necessarily hyponymy);
2. Create a list of term pairs which fit this relation;
3. Find places where the two terms appear near each other and record the environment;
4. Find common environments and hypothesise that they yield new patterns;
5. Use the new pattern to gather more instances of the target relation and go to Step 2.

When applied to meronymy (part/whole relation), the algorithm was not as successful as it was for hyponymy (evaluated by comparing results to the noun hierarchy of WordNet). The quality of the relations was high, but their number small compared to the size of the text used (8.5M words of encyclopaedia text). Other problems encountered were under-specification, metonymy, atypical relations, and over general hypernyms.

Joho and Sanderson (2000) built a descriptive phrase retrieval system which could be thought of as a specialised question answering tool for answering 'Who is' and 'What is' questions. The system retrieved all documents and then all the sentences containing the query noun. The sentences were ranked according to three criteria: the presence of key phrases (an expanded list of Hearst's patterns—see Table 2.2) without any parsing, the number of terms co-occurring across documents with the query noun, and the ordinal position of the sentence in the document (higher score was given to earlier sentences). The top five and top ten sentences were ranked for relevance. A response was considered successful if at least one sentence in the ranked list was a correct or partially correct answer

(DEF such| such DEF) as TERM
TERM (and|or) other DEF
DEF especially TERM
DEF including TERM
TERM (DEF) or (DEF) TERM
TERM (is|was|are|were)(a|an|the) DEF
TERM, which(is|was|are|were) DEF
TERM (a|an|the) DEF
TERM, (a|an|the) DEF(.|?|!)
TERM, DEF,(is|was|are|were)

Table 2.2 Key phrases (patterns) used by Joho and Sanderson (2000). TERM is query noun. DEF is the descriptive phrase. The first four were used by Hearst (1992) to detect hyponyms, while the rest detect acronyms, 'is a' type descriptions, and appositions parenthesised by commas.

In the tuning phase it was found that all patterns are rare, though the comma parenthesised apposition and `as such` were most frequent, and `and other` proved most accurate. The accuracy specific to patterns was combined in the final weighted score.

In an evaluation of the system with 50 queries over a corpus of LA Times articles (475 Mb) the system ranked a sentence with a description within the top ten for 94% of the queries. Using the three criteria (as above) and a formula combining the three achieved better results than random retrieval of sentences containing the query term. Key phrase presence was only second to using the combination formula. The co-occurring word counting method proved better than the key-phrase for high levels of recall. Experimenting with different percentages of the document collection showed, as expected, that using smaller random samples of the collection reduced precision. In a later experiment (Joho, Liu and Sanderson, 2001) the system performed significantly better (across 96 queries at least one relevant sentence was found in the top five), when the Web was used as the document collection.

In earlier work, Radev and McKeown (1997) described PROFILE, a system extracting

noun-phrase descriptions of entities such as people, places, and organisations. The system was created as an information extraction tool and as a utility to generate functional descriptions to be re-used. After entity names were extracted, variable noun phrases on either side of the entity (either pre-modifiers or appositions) were matched as descriptions. WordNet was used to categorise extracted descriptions, which were then organised in a database of profiles. The precision of 611 descriptions of randomly selected entities was computed manually and found to be 90%.

Recently, Fleischman, Hovy and Echihabi (2003) used part-of-speech patterns to extract offline concept-instance relationships: common noun/proper noun constructions and appositions. Different machine learning algorithms filtered the regular expression patterns, achieving precision of over 90% (based on evaluation of a sample of 100 concept-instance pairs from the 2,000,000 pairs extracted). The look-up of extracted concept-instance pairs resulted in 8% more partially correct answers and 25% more entirely correct answers than TextMap (a state-of-the-art system, among the top ten in TREC-11). The look-up took only ten seconds for 100 questions, compared to the nine hours it took for the question answering system.

Lie, Wee and Ng (2003) studied the task of helping users to learn about a new topic on the Web. They assumed that finding definitions would be one of the first steps in learning about a new topic. Despite the diversity of the Web, the team managed to identify patterns that are suitable for Web pages (see Figure 2.2). In addition, HTML structuring clues and hyperlinks also identified definitions. For example, if a page begins with one header that contains a concept, the page is assumed to include a description/definition of the concept. Using the above heuristics to find definitions achieved better average precision (61%) in returning definitional documents in the top 10 pages over 28 search topics compared to Google (18%) and Ask Jeeves (17%).

DEFINDER is a rule-based system that mines consumer-oriented full text medical articles for terms and their definitions (Klavans and Muresan, 2001a; Klavans and Muresan, 2001b). It was developed at Columbia University to provide clear, lay definitions for technical terms. The system addresses the difficulty of building online dictionaries for technical terms and their incompleteness.

- {is | are} [*adverb*] {called | known as | defined as} {*concept*}
- {*concept*} {refer(s) to | satisfy(ies)} ...
- {*concept*} {is | are} [*determiner*] ...
- {*concept*} {is | are} [*adverb*] {being used to | used to | referred to | employed to | defined as | formalized as | described as | concerned with | called} ...
- {What is} [*determiner*] {*concept*}?
- {*concept*} {- | :} {*definition*}
- <dt> {*concept*} <dd> {*definition*}

Legend:
{ } - compulsory field
[ ] - optional field
adverb - e.g., usually, normally, generally,...
determiner - e.g., the, one, a, an, ...
definition - definition of a concept

Figure 2.2: patterns used to identify definitions of concepts in Web pages (Liu, Wee and Ng, 2003)

The system identifies by shallow text processing initial contexts for pattern extraction such as cue phrases (for example, `TERM is the term for DEFINTION`, `DEFINITION is called TERM`) and some text markers (for example, the dash in `TERM–DEFINITION`). The shallow parsing involves part-of-speech tagging, NP chunking, filtering of misleading patterns and pattern identification. A rich, dependency-oriented lexicalist grammar analyses appositions, complex patterns of text markers, and syntactic complements of the verb 'to be'.

A human-determined 'gold standard' consisting of 53 definitions was created by four subjects annotating terms and their definitions in nine patient-oriented articles. DEDINDER identified 40 of them, achieving 87% precision and 75% recall.

Eight non-expert subjects judged the usefulness, readability, and completeness of DEFINDER's definitions for 15 medical terms compared to definitions of two specialised on-line dictionaries. The usefulness and readability of the DEFINDER definitions were rated significantly higher than the definitions of the online dictionaries. The coverage of DEFINDER was evaluated against three on-line medical dictionaries by looking up 93 terms and their definitions (extracted by DEFINDER). Two of the specialised online dictionaries had a coverage of 60% and 76% each, while a non-specialised glossary had a coverage of 21% compared to DEFINDER.

2.5 Definition Questions in TREC

2.5.1 The Question Test Set

TREC 8, 9, and 10 included definition questions in the main task as part of the factoid questions. As mentioned above, definition questions were dropped in TREC–11 (2002) and reintroduced in TREC-12 (2003). Definition questions are questions such as 'Who is Duke Ellington?' or 'What is bipolar disorder?' They seem to be an important type of question judging from their relatively high frequency in logs of Web search engines (Voorhees, 2003a). The test set in 2003 included 50 questions which originated in the same set of search engine logs from which the factoid questions were drawn. Assessors selected a question from the log and searched the AQUAINT corpus for information about the target. The final set contained 30 questions about a person (real or fictional), 10 questions about an organisation, and 10 questions about some other thing. Any qualification in the log was retained (for example, 'What is pH in biology?').

2.5.2 Evaluating answers to Definition Questions in TREC 2003

Evaluating answers to a definition question is more difficult than evaluating answers to factoid questions, because it is not useful to judge a definition answer as simply right or wrong.

A pilot study (Voorhees, 2003b), planned as a part of the ARDA AQUAINT program (a research initiative sponsored by the U.S. Department of Defense aimed at increasing the types and difficulty of questions systems can answer) demonstrated that human assessors agree generally on concepts that should appear in the answer to a definition questions and can find the concepts in systems' responses. Computing concept recall is simple, based on these judgements—the ratio of the number of correct concepts retrieved to the number of concepts in a list of 'information nuggets' prepared by the assessors. However, the measure of concept precision, the ratio of the number of correct concepts retrieved to the total number of concepts retrieved is difficult, because the denominator value is unknown.

When evaluating answers to a definition question, profiling the intended user is crucial to determine what level of detail is appropriate in the response (see Section 2.3.5). The following scenario was assumed to guide system developers:

> 'The questioner is an adult, a native speaker of English, and an 'average' reader of US newspapers. In reading an article, the user has come across a term they would like to find out more about. They may have some basic idea of what the term means either from the context of the article […] or basic background knowledge […]. They are not experts in the domain of the target, and therefore are not seeking esoteric details…' (Voorhees, 2003b)

In the first step of evaluation, the assessors created a list of 'information nuggets' about the target based on the answer-strings from all the responses and their own research during question development. Information nuggets were defined as facts for which the assessor could make a binary decision as to whether a response contained them. The assessors then decided which nuggets were vital. In the second step assessors went through each system's responses and marked where each nugget appeared. If a nugget was returned more than once, it was marked only once.

The evaluation depended only on the content of the responses, not their structure. Assessors ignored wording or syntactic differences and made conceptual matches between responses and their nuggets. A single string answer within a response was allowed to match multiple nuggets. Occasionally, when a nugget split across a system's strings, the assessors assigned the nugget to a string that contained the main part of the concept.

Following the findings in the pilot evaluation study, the length of the response was used as a crude estimate of precision, based on the intuition that users would prefer the shorter of two definitions which contain the same concepts. Precision could not be ignored, because using only nugget recall as a final score would not reward systems for selectivity.

If the response is longer than an allowance (100 non-white-space characters per correct nugget retrieved), the precision is downgraded from 1 using the function

$$\text{precision} = \frac{\text{length-allowance}}{\text{length}}$$

Nugget recall was computed only over vital nuggets, whereas precision was computed on vital and non-vital nuggets, so as to penalise systems for not retrieving vital nuggets and avoid penalising, or rewarding, them for retrieving non-vital nuggets.

The F-measure was used as the final score for responses. The general version of the F-measure is

$$F = \frac{(\beta^2 + 1)*\text{precision}}{\beta^2*\text{precision}+\text{recall}}$$

In TREC-12, β was set arbitrarily to 5, reflecting the emphasis assessors gave to recall in the pilot study and adjusting for the crudeness of the length approximation to true precision.

2.5.3 Results and Analysis of Evaluation of Answers to Definition Questions

The best run (by BBN Technologies) achieved an average F(β = 5) score of 0.555 with an average length of a response for the run of 2059.20 (measured in non-white-space characters). The median average F score was 0.192, and the worst was 0.000 (Voorhees, 2003a).

Two aspects to the quality of the evaluation in the first year of the definition sub-task were analysed: fidelity and reliability. Fidelity is the extent to which the evaluation measures what it is intended to measure. Reliability is the extent to which an evaluation can be trusted.

The pilot evaluation included 'holistic' evaluation in which the assessor scored the content of the entire response between 0 and 10. Using a β value of five in the F-measure gave a good correlation between the quantitative score and the holistic score, but it is unclear if the average user would prefer recall so strongly. Such strong emphasis on recall may not encourage systems to be selective. The results of a baseline run, in which sentences containing the target of definition questions were retrieved indiscriminately, revealed that the baseline was ranked fourth (with F-measure of 0.493) among all 55 runs when β was set to five. Changing the value of β (hence the

importance of recall versus precision) changed the ranking of runs but did not show which value of β is better for the definition task.

Human error, differing opinions between assessors, and the small sample of questions (50) undermine the reliability of the evaluation. The definition task was new in TREC-12, and the evaluation was more difficult for the assessors. In a comparison of 14 pairs of identical definition components, the inconsistency in the average F (β=5) scores for the pairs ranged between 0.0 to 0.043 and averaged 0.013.

In the track, each question was judged by a single assessor to ensure internal consistency. However, to quantify the effect of different opinions of different assessors on scores, a second assessor judged each question independently. The second assessor was not involved in the question development and was given the initial list of nuggets as a starting point. The second assessor was free to modify the list in any way. Some assessors listed many more nuggets, and there was no agreement as to what information should be returned. The largest difference in original F scores between two runs that were evaluated by different assessors was 0.123. In the light of these findings, the evaluation of definition runs in TREC-12 should be interpreted with caution.

According to Voorhees, increasing the number of questions could stabilise evaluation. Extrapolating an error rate curve over question set size reveals that for a question set of 50 questions, an absolute difference of at least 0.1 in the F(β = 5) scores between runs is needed before the error rate drops below 5%. However, such a difference would not discriminate sufficiently between runs. Adding more questions to the set will increase sensitivity while maintaining confidence in the results.

2.5.4 Approaches to Answering Definition Questions in TREC-12 (2003)

This section describes the approaches to answering definition questions of four systems ranked in the top fourteen groups for this sub-task in 2003.

The system of the Massachusetts Institute of Technology (MIT) employed three parallel techniques to answer definition questions:

Database lookup. A knowledge base containing nuggets about every entity in the

AQUAINT corpus was pre-compiled automatically, so the task of answering definition questions was reduced to database lookup. The surface patterns included a copular pattern (e.g,, 'Karen Mantler is a jazz musician'), appositive pattern (e.g., 'Karen Mantler, a jazz musician'), occupation pattern (common noun preceding proper nouns contains an occupation; e.g., 'jazz musician Karen Mantler'), verb pattern (`NP1 verb NP2`, where verb is from a list of common verbs occurring in biographies; e.g., 'become', 'founded', 'invented'), and a parenthesis pattern. Surface patterns were used offline to pre-compile knowledge nuggets about entities from the AQUAINT corpus. The responses returned additional context by expansion of all nuggets by 100 characters to improve readability and sometimes to return additional relevant nuggets that are not part of the original pattern. To reduce the tremendous amount of redundancy, if two responses shared more than sixty percent of their keywords, one of them was thrown out.

Dictionary lookup. The definition of a term was fetched from the Merriam-Webster website. Keywords from the definition were included in the query to the document retriever. The system chunked all the sentences from the documents retrieved and scored each one based on the keyword overlap with the dictionary definition.

Document lookup. As a last resort, the target term was used in the query to the document retriever. The documents retrieved were chunked into separate sentences, and sentences with the term were returned as responses.

The responses were ranked using an ad-hoc priority scale based on the accuracy of each technique (for example, verb patterns were found to be the most effective). The MIT group decided to return long answers because the length penalty was mild. Long answers could contain additional relevant nuggets. Given *n* responses, the final number of responses was

$$\begin{cases} n & \text{if } n \leq 10 \\ n + \sqrt{n-10} & \text{if } n > 10 \end{cases}$$

MIT submitted three runs which were identical in the definition responses, but two runs scored 0.282, while a third scored 0.309 in the Definition task (eighth among all systems

in TREC).

IBM's PIQUANT system broke down definition questions into auxiliary questions which were assembled into a "dossier". This approach resembles the use of templates in information extraction. The question-answering-by-dossier approach was adapted for TREC-12 to answer definition questions with three types of focus: Person, Organisation and Thing. Each question type invokes a different set of auxiliary questions. The set may include multiple rounds of follow up questions based on answers to earlier questions, but this was not implemented in TREC-12.

The auxiliary questions were of two types: 1. general life-cycle questions that should be applicable to all subjects. 2. speculative, reasonably general follow-up questions.

A survey of obituaries and encyclopaedias determined that a response to PERSON-type questions should include major events in a person's life cycle. Therefore, the auxiliary questions included ones such as 'When was X born?', 'How did X die?', and 'Who was X married to?'.

Different answering agents of PIQUANT (see Section 2.2.7) returned a fixed or variable number of answers to different auxiliary questions. The best answer should exceed a threshold established in training. Two auxiliary Who questions used the Structured Knowledge Agent's relevant data—biographical information from (Who2, 2003)—to find phrases describing what is/was someone famous as or known for. The phrases were then combined with the question subject into a bag-of-words to query the TREC corpus.

The auxiliary questions for THING questions were 'What is X?', 'What is another name for X' and 'X,,<WordNet gloss entry>'. The Definition Agent employed Virtual Annotation (see more in Section 2.4), whereby 'the focus of the question is looked up in WordNet to find all hypernyms, and the ones that are most likely to co-occur with the question focus in the reference collection, penalised by WordNet path length, are returned'. For example, if the focus of the question is 'table', and both 'thing' and 'furniture' co-occur frequently with it, 'furniture' will be selected, because of the two it is the closer hypernym to 'table' in WordNet. The passages retrieved should contain the questions focus and the selected hypernym(s).

For Organisation the auxiliary questions included ‘What does X manufacture?’, ‘Where are the headquarters of X?’, ‘Who is the CEO of X?’.

The system scored an average F of 0.175. In its analysis, the IBM team argued that often their nuggets were unjustifiably judged as incorrect, highlighting the difference between IBM’s interpretation of the somewhat vague evaluation framework and that of NIST and the need to reduce the variability in interpretation. IBM’s self-assessed average F score was significantly higher (0.387).

The question answering system of BBN Technologies (Xu, Licuanan and Weischedel, 2003) answered definition questions in six steps:

1. Classification of the question as Who or What question;
2. Retrieval for each question of the top 1000 documents, using the question target as the query;
3. Use of heuristics to check if a sentence mentions a question target. Matching is done directly (by string comparison) or indirectly (through co-reference);
4. Extraction of ‘kernel’ facts: appositives and copulas (extracted from the parse trees), propositions (based on a list of verb-argument structures), structured patterns (based on 50 handwritten rules such as the regular expression `<TERM>,?(is|was)?also?<RB>?also?<RB>?called|named|known+as<NP>`), binary relations, and finally full sentences when none of the above match;
5. Ranking of the kernel facts based on their type (for example, appositives and copulas were ranked before structured patterns), and on similarity to the profile of the question. The question’s profile was created by searching for existing definitions of the question target in resources such as WordNet, dictionaries, an online encyclopaedia, and online biographies, and using their centroids. If no definition was found for a What question, the centroid of all kernel facts about the question target was used as the question profile.
6. Redundancy removal using the following rules: Two propositions were considered equivalent if they shared the same predicate (verb) and same head noun for each argument. If two or more facts were extracted by the same rule of structured pattern, only one was selected. If the fact consisted of more than 70%

of words which appeared at least once in the set of kernel facts, the fact was considered redundant.

BBN achieved the highest score in definitional question answering at TREC-12. The average for Who questions was 0.577 (30 questions) while for what questions it was 0.522 (20 questions). It is hard, however, to determine if the difference is statistically significant because of the small number of questions. Analysis of the results showed that sources of error were misinterpretation of the question target (in questions which scored zero), faulty redundancy removal, and low recall. While the F-metric in TREC-12 emphasised recall over precision, BBN's system had better precision than recall. Interestingly, assuming perfect recall, while keeping actual precision would increase the F-measure for BBN's best run to 0.79.

Columbia University presented DefScriber, a system dedicated to answering open-ended definition questions and modified for TREC-12 (Blair-Goldensohn, McKeown and Schlaikjer, 2003a; Blair-Goldensohn, McKeown and Schlaikjer, 2003b). The system combined top-down techniques, based on key elements of definitions such as genus (category) and species (properties), with data-driven techniques adapted from work in multi-document summarization.

The system used in TREC-12 identified three predicates: Genus, Species, and Non-specific Definitional, although the entire set of predicates created by the Columbia University group included nine predicates (the additional six were Synonym, Target Partition, Cause, Effect, Historical and Etymology). Based on previous work on definitions, the group concluded that information modelled by its predicates is crucial to descriptive-type definitions.

The system identified sentences containing predicates by feature-based classification and by pattern recognition. The features included ones measuring term concentration, ones for relative and absolute position of a sentence in a document, and ones for presence of punctuation (to detect full-sentence text). Identifying the non-specific definitional predicate, which is crucial as a cue to the presence of other predicates, achieved an accuracy of 81 percent using the rules extracted by machine-learning (the training data consisted of 81 web documents retrieved in response to submitting 14 diverse terms and marked by hand).

Eighteen syntactic and lexical patterns (extracted manually) were used to match sentences containing both Genus and Species predicates (the ones most fundamental to definitions). The patterns were modelled as partially specified syntax trees, not at the word level, and so are flexible.

Summarization techniques, designed to identify common themes in the data, were used to cluster and order the set of non-specific definitional sentences. For TREC-12 the statistical cohesion measures were disabled, but clustering was kept to avoid redundancy.

DefScriber was designed originally to find definitions of objects and concepts, but it can process Who questions as well (with just one minor change).

Originally, users were supposed to specify the number of answer sentences to include in each answer, but for TREC-12 this number was optimised using a linear combination of a minimum number of answers and an adjustment factor based on the number of relevant documents.

DefScriber achieved an average F score of 0.338. However, its designers observed that the judges missed some nuggets in their responses, perhaps due to responses requiring a high level of inference. Other problems pointed out by the group were the lack of support for phrasal queries in the search engine used by the system, resulting in too many irrelevant documents, sub-optimal answer length, and lack of a fuzzy search option (to overcome spelling mistakes).

2.6 Summary

This chapter placed our project within the context of question answering. We began with an overview of the aims, issues and history of the field and described the question answering track in the Text REtrieval Conference (TREC), which has recently rekindled interest in developing open-domain systems. We then reviewed some of these state-of-the-art systems. Focusing on definition questions, we presented theoretical background on definitions and summarised previous work on answering such questions. Finally, we introduced the definition subtask in TREC-12 (2003), the issues involved in evaluating

definitions and current approaches to extracting them. The next chapter presents the domain in which we chose to undertake our research on the answering of definition questions.