IITD-IBMIRL System for Question Answering using Pattern Matching, Semantic Type and Semantic Category Recognition

Ashish Kumar Saxena, Ganesh Viswanath Sambhu, L. Venkata Subramaniam*, Saroj Kaushik
Indian Institute of Technology, New Delhi, India
*IBM India Research Lab, New Delhi, India
E-mail:ashish.ksaxena@yahoo.co.in, ganesh.sambhu@gmail.com, lvsubram@in.ibm.com, saroj@cse.iitd.ernet.in

October 17, 2007

Abstract

A Question Answering (QA) system aims to return exact answers to natural language questions. While today information retrieval techniques are quite successful at locating within large collections of documents those that are relevant to a user's query, QA techniques that extract the exact answer from these retrieved documents still do not obtain very good accuracies. We approached the TREC 2007 Question Answering task as a semantics based question to answer matching problem. Given a question we aimed to extract the relevant semantic entities in it so that we can pin-point the answer. In this paper we show that our technique obtains reasonable accuracy compared to other systems.

1 Introduction

The goal of the TREC QA task is to foster research on QA systems to improve the state of the art. A QA system works by returning exact answers to natural language questions. A question such as "In 2003 who was the Secretary General of the United Nations," has only one exact answer. Given such a question a QA system should return the exact answer to it.

Our team took part in the TREC QA main task of 2007. This was the first time our team participated in the QA task. The main task was the same as in TREC 2006, in that the test set consisted of question series where each series asked for information regarding a particular target. These targets included people, organizations, events and other entities. The questions on each target comprised of three types, factoid questions, list questions and one "Other" question. Factoid questions have exactly one correct answer. A list question has a list as its answer and the answer to the "Other" question is to be interesting information about the target that is not covered by the preceding factoid and list questions in the series.

The major difference between the 2007 main task and the 2006 main task was that questions were asked over both blog documents and newswire articles, rather than just newswire. A blog document is defined to be a blog post and its follow-up comments (a permalink). The blog collection contains well-formed English as well as badly-formed English and spam, and mining blogs for answers introduced significant new challenges in at least two aspects that are very important for functional QA systems: 1) being able to handle language that is not well-formed, and 2) dealing with discourse structures that are more informal and less reliable than newswire.

In this paper we describe our approach to the TREC 2007 QA Task. We approached the TREC 2007 Question Answering task as a semantics based question to answer matching problem. Given a question we aimed to extract the relevant semantic entities in it so that we can pin-point the answer. Overall our scores were well above the median score of the TREC 2007 runs from all teams.

2 System Overview

Our Question answering system consists of several phases that work in sequential manner. Each phase reduces the amount of data, the system has to handle from then on. The advantage of this approach is that progressive phases can perform more expensive operations on the data. The system is broadly divided into two main modules,the **Question Processing Module**, and the **Answer Retrieval Module**.

2.1 Question Processing Module

This is the module which takes the input set of questions and converts into a form that can be processed by the answer retrieval module. This module consists of question pre-processing, keyword generation, significant keyword selection and question classification.

Question pre-processing contains stopword removal and stemming. In question classification the questions are classified to yield their expected answer types and in Query Generation a parser is used in order to identify certain significant words that are given more weightage than the normal keywords in the construction of the query for both document retrieval and answer retrieval. Figure 1 illustrates this module.

2.2 Answer Retrieval Module

This module takes as input the keywords along with keyword significance scores and expected answer type all produced by the Question Processing Module. Using the keywords the answer retrieval module first finds the documents relevant to the question. Only the top N documents are used for the next step. This greatly reduces the amount of text that need to be handled in subsequent steps. Next from these documents we select the relevant sentences. In this sentence selection phase, all sentences are scored against the question and only the most relevant sentences are picked. In the final phase, we pin-point the answer within a sentence. As per the TREC requirement, this answer should be exact and correct. Figure 2 illustrates this module.

3 Question Processing

In this section we explain in more detail our question processing module introduced in the previous section. We will explain the various sub parts of this module in detail here. Our goal is to not only extract the keywords but also as much semantic information as possible from the question that helps in getting the exact answer.

3.1 Question Pre-processing

This stage implements stop word elimination and stemming. A list if frequently occurring words is considered for stop word removal and Porter's Stemming algorithm [20] was used in order to stem the words. Our stop word list comprised of 50 common English words such as it, the, at, etc. The output of this step comprises the set of all keywords from the question.

3.2 Significant Keywords Selection

The quality of the answer-retrieval engine depends on the richness of the query that is given to it. In order to obtain the most relevant answers for the questions, the query tries to highlight certain words present in the question as significant words. The following words within the keywords are considered significant:

1. Words referring full or part of the target (As the target is already provided identifying this is very easy)
2. Words that refer to the object of the question (Question Object)
3. Noun phrases in the absence of question object.

3.2.1 Question Object

We have observed that the object of the question is often one the significant words and hence that has to be identified. This can be detected with the help of a Parser (We have used Stanford Parser [17] for all parser implementations). We have exploited the feature of this parser that recognizes the dependencies and detects the object.

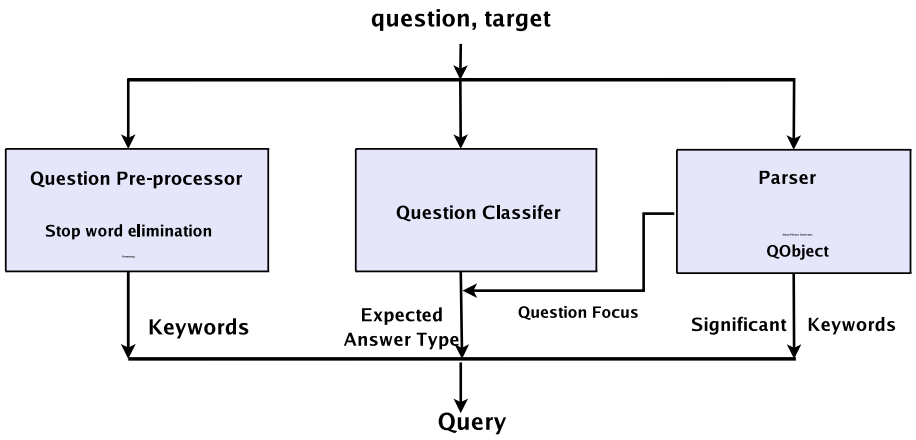

Figure 1: Block Diagram of Question Processor

For example in the question, *What company produces his records?* , the parser detects that *produces* and *records* are connected and that *records* is the object.

3.2.2 Significant Noun Phrases

Even though most in most of the questions we are able to identify the object, there are exceptions. In such a case, we try to use all the Noun Phrases in the question minus the stop words as significant words. These noun phrases are obtained with the help of the parser.

3.3 Question Classification

Question classification is the process of categorizing the question into one of the predetermined classes. This stage is needed because we need to know the expected answer type before returning an answer. We have used a rule based classifier that classifies a question into fine grained categories and their corresponding coarse categories.

3.3.1 Categories

There are 8 coarse grained categories and 59 fine-grained categories. The coarse-grained categories we selected were PERSON, LOCATION, ORGANIZATION, NUMBER, TITLE, JOBTITLE, DATE and MONEY. We deemed these as the important categories based on past TRECs. The Named Entity Recognizer in the retrieval part has the same names as these 8 types and therefore, this is helpful in directly matching the indexed documents. The fine-grained types provide additional information for the answer retrieval module. The fine grained classes are almost similar to those present in the UIUC Dataset [19] but with few additions and deletions. The categories used in our system are listed in Table 1. To classify a question into the coarse and fine grained classes we defined a set of rules. Our rule set comprised of 300 ordered rules. The ordering implied that certain rules had precedence over others. We give some examples here,

3.3.2 PERSON

All *Who* questions are classified as requiring answer type PERSON. Questions containing *What, Which and Name* with words like architect, engineer, artist etc. were for example categorized as requiring answer type PERSON - INDIVIDUAL.

3.3.3 LOCATION

All *Where* questions are classified as LOCATION. Further questions containing *What, which and Name* with country, state, city, town, ocean or river refer to more specific locations and can be put in the respective fine grained categories.

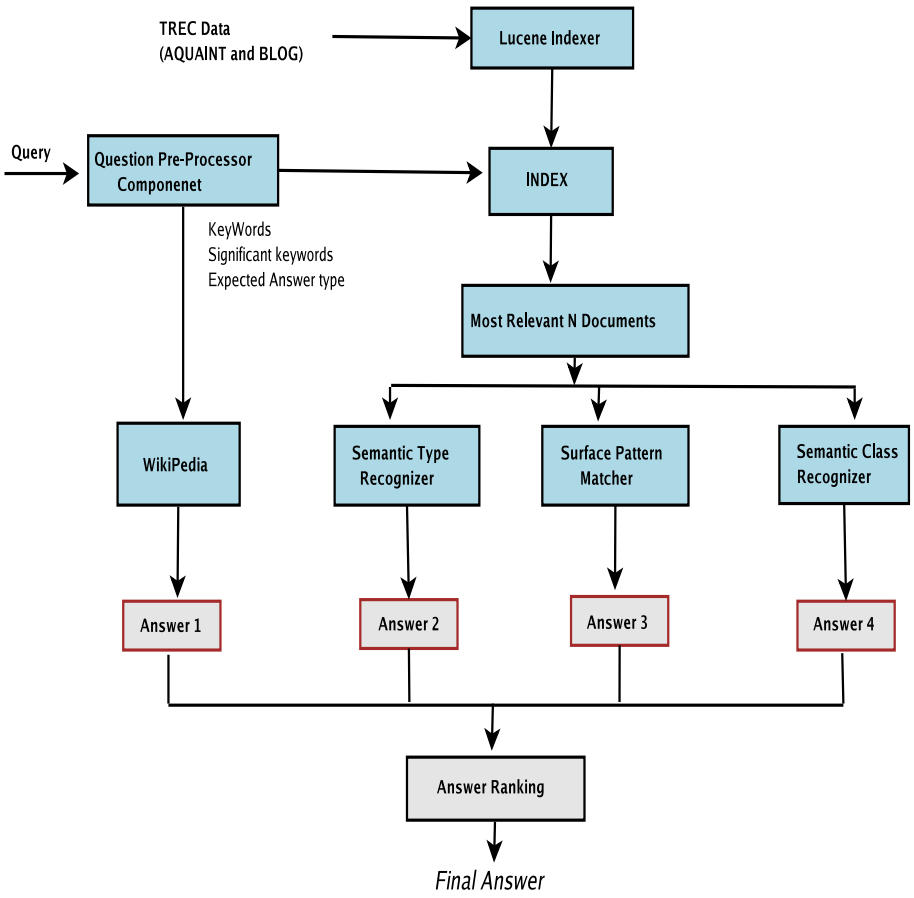

Figure 2: Architecture of Answer Retrieval Module

3.3.4 DATE

All *When* questions are classified as DATE. Specific mentions of year, month or time along with *What and Which* are assigned to finer grained classes.

3.3.5 NUMBER

How questions mostly refer to numbers. They are detected by checking for other words like many, far, long, deep, fast, hot, etc. For example, *How many* always refers to a count.

3.3.6 ORGANIZATION

what, which and Name with other words like team, league, organization, institution, school, college, etc., are assigned to this class.

3.3.7 ENTITY

Entity types are detected mostly with the specific entities like colour, creature, product etc. that occur along with *what, which and Name* questions.

We built these rules using past years questions. We also used wordnet and wikipedia to associate entities with their categories. For example, town is assigned to the class LOCATION - CITY using wordnet hypernym.

3.4 Question Focus

In the absence of an expected answer type we need some focus to be determined for retrieving the correct answer. For us the question focus is the word that is associated with the question phrase like what, when, etc. Generally it is the Noun Phrase

Table 1: Coarse and fine-grained classes

Coarse	Fine		
ABBR	abbreviation	expansion	
DATE	date	month	other
	week	year	
ENTY	animal	body	book
	color	currency	disease
	event	food	instr
	language	letter	medicine
	movie	music	physt
	plant	position	prize
	product	religion	song
	sport	substance	symbol
	tvshow	url	vehicle
	word		
JOB	jobtitle		
LOC	city	country	geo
	other	state	
NUM	age	count	distance
	duration	money	percent
	phone	size	speed
	temp	time	volume
	weight	zip	
ORG	organization		
PER	individual	name	
TITLE	title		

attached to the question phrase (what, when etc.,). For example, for the question *What part of the Soldiers' anatomy reminded the Indians of the buffalo?*, the Focus is *part*. There was no rule to classify this question, but using question focus, we could classify this as ENTY:part. We found this module to succeed in a few cases where the question classification failed. This module only fired if the classification didn't return any results.

The question processing step gives as its output the list of all keywords, the list of significant keywords and the expected answer type. The significant keywords and the expected answer type are the extra semantic information that have been extracted to aid in the answer retrieval.

4 Answer Retrieval Module

In this module given a question and a set of documents, we retrieve the exact answer to the question from the given documents. It is possible that there is no answer in the given document collection for a given question. In this section we enumerate the steps to getting the answer to a question.

4.1 Document Retrieval Module

The first step is the information retrieval task. From the set of all given documents we identify the top relevant documents for a given question. The document retrieval module enables this in a fast and efficient manner.

4.1.1 Indexing

The goal of storing an index is to optimize the speed and performance of finding relevant documents for a search query. Without an index, the search would scan every document in the corpus, which would take a considerable amount of time and computing power. For example, an index of 1000 documents can be queried within milliseconds, where a raw scan of 1000 documents could take hours. The trade off for the time saved during retrieval is that additional storage is required to store the index and that it takes a considerable amount of time to update. Lucene is a free and open source information retrieval library, originally implemented in Java. It is suitable for any application which requires full text indexing and searching capability. We indexed the complete TREC 2007 QA data collection using Lucene.

4.1.2 Full-text searching

The Document Retrieval Module identifies the documents or paragraphs in the document set that are likely to contain the answer. Using the keywords found by the question processing module we retrieve the relevant documents. We give more weightage to significant keywords determined by the question processing module in ranking the retrieved documents. We keep only the top N ranked documents for a given query.

4.2 Exact Answer Selection

In this module given the relevant documents we now select the exact answer to a given question. We implemented four separate ways of doing this.

4.2.1 Semantic Type Recognition

The semantic type recognizer extracts the answer based on the expected answer type.

Example: *In what year was the Prius concept car introduced?* (Ques. 245.3) has answer *1997* (DATE type)

Who won the 2005 Snooker World Championship? (Ques. 259.5) has answer *Shaun Murphy* (PERSON type)

We also built a NUMBER identifier which is able to identify any NUMBER present in a sentence. For example 10, one hundred, or one million.

Example: *How much wine does Australia export to the U.S.?* (Ques. 279.2) has Answer *19.4 million* (NUMBER)

The semantic type matching is based on a named entity recognizer which extracts all the named entities from the answer text.

We have used statistical model based Named Entity Recognizer (NER) which is trained using newswire training set. NER seeks to locate and classify elements in text into predefined categories PERSON, LOCATION, ORGANIZATION, NUMBER, TITLE, JOBTITLE, DATE, MONEY, etc.

The first step in answer extraction is to get the most relevant sentence that is likely to contain the answer. Two factors are considered for ranking the sentences: number of keywords occurring in the sentence, and whether the sentence contains the same answer type as the question. The Sentences are scored using tf/idf [1]. The tf/idf score was scaled by the count of the query terms that appear within the Sentence. Each Sentence (Sentence of the candidate articles) is scored. Once the most significant sentence has been found, the named entity with the correct answer type is selected as the answer to the given question. The tf/idf weight (term frequency inverse document frequency) is a weight used in information retrieval and text mining. A high weight in tf/idf is reached by a high term frequency (in the given document) and a low document frequency of the term in the whole collection of documents, the weight hence tends to filter out common terms.

4.2.2 Text Pattern Matching

Some questions are difficult to answer using semantic type based strategy. We developed simple patterns for answering such questions. Specifically, we developed patterns for acronym expansion questions, date of birth questions and location questions. These patterns are derived from the answers to the questions of previous years TREC data. We extracted about hundred patterns for these three class of questions. This approach extracts answers from the surface structure of the retrieved documents by relying on an extensive list of patterns [11]. Although building extensive lists of such patterns is time consuming, this approach has high precision. The approach is based on the assumption that answers can be identified by their correspondence to patterns describing the structure of strings carrying certain semantics. These patterns, are like regular expressions.

Example: *What does the abbreviation CSPI stand for?* (Ques. 248.1)

The text contains "Former North Carolina basketball coach Dean Smith, former Nebraska football coach Tom Osborne, 246 university presidents, the American Medical Association and the Center for Science in the Public Interest (CSPI) have not...." the pattern matcher for this is based on matching the first letters of adjacent words

Example: *Which year was Mozart born?*

The string "Mozart (1756-1791)" contains answers to the questions about Mozart's birth and death years, allowing construction of the pattern: "capitalized word; parenthesis; four digits; dash; four digits; parenthesis ".

4.2.3 Semantic Class Recognition

For many of the questions the question processing module is not able to return the expected answer type. For such questions we exploited knowledge about hypernym relationship in WordNet. WordNet is the well-known English ontology freely available on the Web and covers the vast majority of nouns, verbs, adjectives, and adverbs from the English language. For questions asking about certain categories such as animal, disease, plant, color, etc., we evaluated each noun or noun phrase in the sentence using knowledge about hypernym relationship in WordNet. For example, *In what US state was Barack Obama born?* (Ques. 272.1) the answer type is state. For each candidate answer, we used WordNet to check whether state is one of its

hypernyms. *Hawaii* has state as it's hypernym, and hence it is chosen as final answer.

4.2.4 External Resource

We also implemented a module for retrieving answers from the WikiPedia [16] InfoBox. We compared keywords from the question with infobox entries to retrieve the exact answer. We use the "target" as the basic element to retrieve the wikipedia page, and then use its infobox to get specific answers.

5 Experimental Setup and Results

We used the data available for participants of the QA track of the 2007 TREC conference. This data includes AQUAINT news-wire data and BLOG data. The questions in TREC 2007 are grouped by topic. The competition consisted of 70 topics, with a total of 515 questions. These questions are divided into three different types: factoid, list and other. Factoid questions require a single fact as answer. Lists asks for a list of answers and other is answered by giving any additional information about the topic. There are 360 factoid questions in the question set.

5.1 Question Processing

As detailed in Section 3 we obtained the keywords, and their significance. We also obtained the expected answer type for each question. To obtain the expected answer type we made rules based on the previous years' TREC questions. 92% of the TREC2006 questions were classified out which 95% were correct.
This year 84% (374) of all the 445 Factoid and List Questions could be classified and the classifier accuracy stood at 95% (358 correct classifications).
Our rule-based classifier produces better results compared with Machine-Learning based classifiers. For example, the Machine-Learning classifier by Li and Roth [18] gives an accuracy of only about 30-40 percent with fine-grained classes.

5.2 Answering FACTOID questions

Factoid questions have only one correct answer. Using the keywords extracted we select the top 50 documents for a question. Next we select the relevant sentences based on keyword matching. These sentences are ranked and the best phrase (with the highest score and matching the expected answer type) is returned as answer to a FACTOID question. We combined the answers returned by the exact answer selection modules in a weighted manner. We got an accuracy value of 0.183 in factoid questions which is higher than the median score which is 0.131. We got 4.3% correct answers from BLOG data and rest 14% correct answers from AQUAINT data. It was expected to get more correct answers from newswire data as we trained and tested our system on newswire data.

5.3 Answering LIST questions

List questions are similar to factoid question, except there is more than one correct and distinct answer to the question. By analyzing candidate answers produced by an existing Question Answering system, we can have multiple answers for a question. Instead of giving single answer, the system will produce a list of top K candidate answers. The Average F score over 85 list questions of 0.125.

5.4 Answering OTHER questions

We observe that Target followed by VB (Verb-Phrase) is a good candidate for giving interesting information about the target. Sentences those are having NUMBERS/DATE, can be considered as priority candidates for answering OTHER question. The average Pyramid F score over 70 'other' questions is 0.208 for our system which is higher than median score of 0.118.

6 Conclusion and Future Work

We believe there is a lot of promise in the semantics based approach that we followed this year. Our system performed reasonably well in the TREC 2007 QA task. Our scores were well above the median.

Table 2: Results of our system compared to other systems

	Run1	Run2	Run3	med
FACTOID				
Accuracy	0.172	0.181	0.183	0.131
incorrect	269	265	262	
unsupported	3	3	3	
inexact	20	21	22	
locally correct	6	6	7	
globally correct	62	65	66	
LIST				
avg F score	0.120	0.123	0.125	0.085
OTHER				
Pyr F score	0.152	0.209	0.208	0.118
Average				
per-ser score	0.150	0.173	0.174	0.108

We observed that we got more answers from the newswire data than the blog data. This could be because we didn't have any modules in place to take care of the informal nature of blog text.

Our QA system is still in the development stage. Some of the subsystems had not been fully tested before the TREC experiments due to time constraints. We are continuing our effort to develop and improve the system.

References

[1] D. Moldovan, M. Pasca, S. Harabagiu and M. Surdeanu, "Performance Issues and Error Analysis in an Open-Domain Question Answering System" *Proceedings of the 40th Annual Meeting on Association for Computational Linguistics*, 33-40, Philadelphia, Pennsylvania, 2001.

[2] E. Hovy, L. Gerber, U. Hermjakob and M. Junk, "Question Answering in Webclopedia", *Proceedings of the TREC-9 Conference*, National Institute of Standards and Technology, 2001.

[3] E. M. Voorhees and Dawnz, "Building a question answering test collection.", *23rd Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 200-207, Athens, August 2000.

[4] N. Wacholder, Y. Ravin and M. Choi, "Disambiguation of Proper Names in Text", *Proceedings of ANLP '97* , Washington, DC, April 1997.

[5] J. Burger, C. Cardie, V. Chaudhri, "Issues, Tasks and Program Structures to Roadmap Research in Question & Answering (Q & A)", (URL: http://vww-nlpir.nist.gov/projects/duc/ roadmapping.html), October 2000.

[6] Warren, H. D. David, Fernando C.N. Pereira, "An efficient easily adaptable system for interpreting natural language queries", *Computational Linguistics*, 8:3-4, 110-122, 1982.

[7] T. Winograd, "Procedures as a representation for data in a computer program for understanding natural language", *Cognitive Psychology*, 3(1), 1972.

[8] D. Zhang and W. S. Lee, "Web based pattern mining and matching approach to question answering", *Proceedings of the Eleventh Text Retrieval Conference (TREC 2002)*, Gaithersburg, MD: National Institute of Standards and Technology, 2002.

[9] R. Srihari and W. Li, "Information extraction supported question answering", *Proceedings of the Eighth Text REtrieval Conference (TREC-8)*, National Institute of Standards and Technology, 1999.

[10] LingPipe Named Entity Recognizer, (URL:http://www.alias-i.com/lingpipe/).

[11] Greenwood and Saggion, "The University of Sheffield's TREC 2004 QA Experiments", 2004.

[12] Q. Armour, N. Japkowicz,and S. Matwin, "The role of named entities in text classification", *Proceedings CLiNE 2005*, Gatineau, Canada, 2005.

[13] M. M. Soubbotin and S. M. Soubbotin, "Use of Patterns for Detection of Answer Strings: A Systematic Approach", *Proceedings of the Eleventh Text Retrieval Conference (TREC 2002)*, Gaithersburg, MD: National Institute of Standards and Technology, 2002.

[14] M. M. Soubbotin and S. M. Soubbotin, "Patterns and Potential Answer Expressions as Clues to the Right Answers", *Proceedings of the Tenth Text Retrieval Conference*, National Institute of Standards and Technology, 2001.

[15] E. Hovy, U. Hermjakob, and D. Ravichandran, "A Question/Answer Typology with Surface Text Patterns", *Proceedings of the DARPA Human Language Technology Conference*, 247-250, San Diego, CA, 2002. .

[16] "Wikipedia, the free encyclopedia", (URL: http://en.wikipedia.org).

[17] D. Klein and C. D. Manning, "Fast Exact Inference with a Factored Model for Natural Language Parsing", *Advances in Neural Information Processing Systems 15 (NIPS 2002)*, MA: MIT Press, pp. 3-10, (URL:http://nlp.stanford.edu/software/lex-parser.shtml), Cambridge, 2003.

[18] X. Li and D. Roth,"Learning Question Classifiers", *Proceedings of the 19th International Conference on Computational Linguistics (COLING '02)*, 2002.

[19] X. Li and D. Roth, "Experimental Data for Question Classification", Cognitive Computation Group, Department of Computer Science, University of Illinois at Urbana-Champaign, (URL: http://l2r.cs.uiuc.edu/%7Ecogcomp/Data /QA/QC/), August 2002.

[20] Porter, "An algorithm for suffix stripping", *Program*, Vol. 14, no. 3, pp 130-137, 1980.