LEARNING STRUCTURES TO REPRESENT VERB MEANING

by

Sharon C. Salveter

Computer Sciences Technical Report #294

March 1977

LEARNING STRUCTURES TO REPRESENT VERB MEANING

Sharon C. Salveter
Computer Sciences Department
University of Wisconsin

ABSTRACT: A program that learns structures that represent verb meaning is described. Inputs are combinations of surface sentences and environment snapshots. The program learns to associate surface verbs with conceptual dependency networks that describe the changes that occur in the environment when an action is performed. Learning is performed by one of five processes, determined by how well current knowledge accounts for an input. Each process is described as well as the circumstances under which it is performed and the effects it has on the conceptual dependencies.

DESCRIPTIVE TERMS: Learning, natural language, conceptual dependency, meaning structures, case structures.

This research has been partially supported by a grant from the National Science Foundation (MCS 76-07333).

LEARNING STRUCTURES TO REPRESENT VERB MEANING

Sharon C. Salveter
Computer Sciences Department
University of Wisconsin

ABSTRACT: A program that learns structures that represent verb meaning is described. Inputs are combinations of surface sentences and environment snapshots. The program learns to associate surface verbs with conceptual dependency networks that describe the changes that occur in the environment when an action is performed. Learning is performed by one of five processes, determined by how well current knowledge accounts for an input. Each process is described as well as the circumstances under which it is performed and the effects it has on the conceptual dependencies.

DESCRIPTIVE TERMS: Learning, natural language, conceptual dependency, meaning structures, case structures.

This research has been partially supported by a grant from the National Science Foundation (MCS 76-07333).

INTRODUCTION

Because of the immensity of the data bases needed for language comprehension and other non-trivial domains, many computer scientists have begun to look to programs that learn as a possible solution. The research described here is primarily concerned with how a program might learn conceptual structures that represent verb meaning. The system I am currently building learns the meaning of a verb by distilling the changes it observes in an environment and associating those changes with a surface verb.

Many of the previous programs that learn are designed to recognize visual patterns or play games in a highly restricted problem domain and/or under a fairly good definition of the subject matter to be learned. Most of these programs appear to be environment dependent and cannot be naturally extended to less structured problem areas. Very little research has been done on potentially large domains, such as learning to understand natural language.

The structure used to represent meaning strongly influences what may be learned. Fillmore [2] postulates the grammatical notion of "case" as a basic component of every natural language. A case is a semantically relevant relation between elements of a surface sentence. Schank [8] has extended the concept of case and developed a complex set of cases he calls conceptual cases. His cases participate in interrelationships, called conceptual dependencies, which represent the meaning of sentences. Unlike Fillmore, who defined cases for surface verbs, Schank postulates case structures for basic concepts he calls "primitive acts". He claims that the meaning of any sentence can be represented as an organization of these primitive acts.

Although the idea of building complex meaning from such primitive units is not universally accepted [5], it does enjoy supportive psychological evidence. Gentner [3] reports that children must learn some "simpler" verbs well before they can learn other "more complex" verbs. This implies that some verbs may be extrapolated from or refined out of other more primitive or lower level verb concepts. Schank [9] cites informal evidence that by age one, children exhibit behavior that indicates that they are using some of the primitive acts. Schank makes no claim that these primitives are innate. However, he does claim that all language meaning can be expressed using the primitive acts, that language does not develop until the primitives are available for use and that they are all available by age two.

This research explores the processes a program may use to learn structures based on Schank´s conceptual dependencies. Although no claim is made that this program is a model of human learning, it is designed to be consistant with psychological data with respect to the order in which children learn verbs and the types of mistakes they make while learning. My program is initialized to a base level of a child approximately two years old. This level is chosen because of the psychological evidence that by this age most children have the necessary primitive verb concepts [9], put words together into simple sentences and begin to enter into discourse [1], [10].

Clearly, children do not learn verb meaning in isolation from other learning. They simultaneously learn a grammar, the names for new objects, and general knowledge about the world. In order to study the specific mechanisms responsible for learning verbs, it is necessary to make some assumptions and either restrict or ignore many processes not directly associated with learning verbs. To this end, the initial program makes the following assumptions:

1. The program already knows the concepts for and names of the physical objects encountered in the environment.

2. The system already knows a grammar of the language, basically consisting of actor-action-object. Dale [1] reports that at age two a child indicates syntactic

structure by using word order where the first noun is the subject, the second noun is the object and the content word between them is the verb. This appears to be true even for children learning a highly inflected language such as Russian. The grammar used by this program is somewhat more complex (e.g. prepositional phrases) but does not contain passives and other complex constructions.

3. The system has a built-in body of knowledge of the world. Although one must be careful not to build in the knowledge one is trying to teach, it is often necessary to use general knowledge to determine what is going on. World knowledge, here, includes both attributes of known objects and physical limitations and characteristics of the world.

REPRESENTATION

The structure used to represent verb meaning is a variation on Schank's conceptual dependencies. It describes the visual effects of a verb and contains descriptive slots for cases required by that verb. For want of a better name, I call these structures Conceptual Meaning Structures (CMSs).

A CMS is composed of two parts. The first part is a set of case slots describing the noun concepts that may participate in the action. Each such selectional restriction has associated with it a frequency count indicating how often the restriction has occured in the program's experience. The second part of a CMS is a description of the effects of the verb in the form of a list of the changes that occur in the environment when the action is carried out.

The system stores all the CMSs associated with surface verbs in a body of knowledge called the verbworld, which is an interrelated net of the CMSs. This organization allows the meaning of one verb to be a component of the meaning of other verbs.

THE LEARNING ENVIRONMENT

The system learns the meaning of a verb (the CMS to associate with a verb) through interaction with the environment. The environment is flexible in that it can be made as sparse or as rich as desired. The environment used in the initial program is a single room containing objects, people and a number of reference locations. To avoid dealing with perceptual complications, this environment is presented to the system as a list structure. An unordered collection of triples of the form (object relation value) describes the environment at any given instant. The program is given an initial description of the environment at time T0 (called a snapshot), followed by a further sequence of one or more snapshots at times T1, T2,...Tn. The system creates a separate net to represent the environment at each snapshot. The program also inputs an English sentence describing the action that took place in the snapshot sequence. The sentence is parsed to determine the subject (actor), verb (action), and objects, and to mark any unknown words. Additionally, the features of each known word are looked up in the world knowledge base and made available to the learning processes.

The program determines what event has taken place during the sequence of snapshots by comparing each environment at time Ti with the environment at time T(i+1) for i=0 to n-1. For each pair of environment snapshots a list is made containing each triple that is in the first environment snapshot but not in the next snaphot and another containing those that are in the second but not in the first. Then the program attempts to "explain" the event by associating each triple in the first list with one in the second and vice versa. Changes discovered in the environment can be ambiguous. For example, change can be seen as occuring either in the first or third position of a triple. However, differences occuring in the third position of a pair of triples (the "value" position) are considered to be more likely in "explaining" the event than differences in the first position (the "object" position). If the change that happened is:

```
                (BOOK AT LOCA) (CUP AT LOCB)
```

in the snapshot at time Ti and

```
                (BOOK AT LOCB) (CUP AT LOCA)
```

in the snapshot at time T(i+1), then this difference will be "explained" by:

```
              (BOOK AT LOCA)---->(BOOK AT LOCB)
               (CUP AT LOCB)---->(CUP AT LOCA)
```

rather than by:

```
              (BOOK AT LOCA)---->(CUP AT LOCA)
              (CUP AT LOCB)---->(BOOK AT LOCB)
```

In other words, it is more plausible that the objects have changed values than that the values have changed objects. Additionally, since words explicitly used in the input sentence are assumed to be important, the English input sentence is used both to help direct difference detection and explain the differences.

After the program has extracted this information from its two external sources, the environment and descriptive sentence, it uses that information to find the CMS most closely accounting for both the input sentence and the changes in the environment. If the system has already learned this surface verb, then it will retrieve its CMS. If more than one CMS is associated with the input verb the closest one will be chosen. In either case, the retrieved CMS will already be associated with the surface verb. If, however, the input verb is not known at all to the program, then the system must attempt to locate an existing CMS, currently associated with another surface verb, which most closely accounts for the environmental changes.

Many problems arise in attempting to match the observed meaning of an input verb to existing CMSs. These problems arise both in choosing among different structures associated with the same surface verb and in choosing the closest structure among all known verbs. In particular, more than one CMS may account for all the changes, several CMSs may account for different subsets of the changes, or no CMS may account for the changes. The differences found in the environment must strongly influence the determination of which CMS is closest. Currently the system measures closeness by simply counting the number of environment changes a CMS accounts for. Clearly, it is necessary to develop appropriate measures of similarity and devise heuristic algorithms to evaluate them.

TYPES OF LEARNING

There are five processes by which learning may take place: confirmation, synonym, minor adjustment, major adjustment and definition creation. The type of learning is determined by the extent of similarity between the changes in the environment and the constraints on the CMS. The program must associate with the input verb a CMS that describes the observed changes in the environment.

<u>Confirmation Learning</u>. Confirmation learning occurs when the program has located a CMS for the input verb and that CMS accounts for the input. That is, the program already knows the input word, and the current CMS description is in agreement with the input. Confirmation learning involves adjusting frequency counts associated with each of the restrictions in a CMS. During the learning process, it is desirable to keep track of how often a hypothesis is confirmed. Then if conflicting information is encountered, it may be determined if it constitutes a strange case. Although every learning mode must adjust the frequency counts, in confirmation learning that is the only process carried out and the counts are only incremented, never decremented.

Suppose the program has developed a meaning for the verb ´carry´ where the subject of the sentence has always been male, the object of the verb has always been a toy and the action is identical location changes for both the subject and object. Further suppose that the program now encounters a snapshot sequence in which the subject and object undergo identical location changes and the input sentence is ´Figaro carries the ball´. The system would then retrieve the CMS associated with ´carry´ and determine that the input satisfied both the environment changes and case restrictions of the paradigm. It then increments the frequency counts associated with the "maleness" of the subject and the "toyness" of the object. That is, the program has verified its current understanding of ´carry´.

Synonym Learning. Synonym learning (the name describes a stereotype case) is the process of associating an additional name to an already known situation or sequence of events. No change is made to any CMS. Rather, an already existing CMS becomes accessible by another name. If the program has no CMS associated with the input verb but can retrieve a CMS associated with another surface verb that accounts for the input sentence and all the changes in the environment, then synonym learning occurs. Because exact synonyms are quite rare, the program actually copies the CMS and associates it with the new word, rather than simply associating the new word with the retrieved CMS. Future modifications of the CMS for this new word will be made only to the copy.

For example, suppose the program is given the same snapshot sequence it encountered in the above example of ´carry´, followed by the sentence ´Figaro moves the ball´.

Then the program retrieves the CMS associated with ´carry´ as the closest one since it satisfies both the case and environment change restrictions. A copy of the CMS is associated with the word ´move´ and stored in the verbworld.

Minor Adjustment Learning. Synonym and confirmation learning both occur when there exists a CMS that accounts for the input. When the closest CMS cannot account totally for the input, the CMS must be modified. Minor adjustment learning does not make structural changes to a CMS, but only modifies restrictions as to who or what may participate in the action by expanding or contracting the sets pointed to by the case restrictions. The frequency counts associated with each restriction are adjusted: the new restrictions are added with initial counts and any confirmed restrictions are incremented.

For example, suppose the program is given the sentence ´Lucy carries the orange´ and a snapshot sequence showing Lucy and the orange undergoing the same location change. Further suppose that the CMS for ´carry´ is retrieved as closest. The description of the environment changes fits the input, but the case restrictions are not satisfied. The program uses minor adjustment learning to modify the case restrictions. It uses world knowledge to find the smallest superset of the instances attributes. In this case ´male´

and ´female´ might yield ´human´ and ´toy´ and ´orange´ may become ´physical object´. This process is simply a loosening of restrictions and is equivalent to moving up ISA links in a hierarchy. At this point, the meaning of ´carry´ has changed from "a male causes himself and a toy to change from one location to another" to "a human causes itself and a physical object to change from one location to another".

Major Adjustment Learning. Major adjustment learning is more drastic than minor adjustment learning in that it involves making structural changes to a CMS, such as adding or deleting restrictions. The CMS is structurally modified to account for the input by altering the description of changes that occur in the environment when the action happens. The information for this type of learning comes from the "explanation" of the differences found in the environment during the snapshot sequence.

For example, suppose the program is given the sentence ´Greg throws the ball´. A snapshot of the environment at T0 shows Greg and the ball at the same location and Greg is in physical contact with the ball. The snapshot at time T1 shows Greg still at the old location, the ball at a different location and Greg no longer in physical contact with the ball. The program does not have a CMS for the surface verb ´throw´. For purposes of illustration, let us suppose that the system retrieves the CMS for 'carry' as closest. The program modifies a copy of the CMS since it has a different surface verb. The case restrictions are satisfied since Greg is human and a ball is a physical object. However, the observed environment changes do not fit the paradigm. The program discovers that if it deletes the requirement that the human change location along with the object and adds the restriction that the subject break physical contact with the object, then it will have an adequate description. Thus the program derives the following meaning of 'throw': "a human causes a physical object to change location and also goes from the state of being in physical contact with it to not being in physical contact with it".

Definition Creation Learning. Definition creation learning is the creation of an entirely new CMS. The new CMS has case slots prescribed by the classes to which the input words belong and a list of the observed changes in the environment. Suppose the program encounters the sentence 'John carried the election' and a snapshot sequence showing John going from campaigning for election to winning the election (admittedly beyond the environment as it has been described so far, but for purposes of illustration let's assume it can be done). The system retrieves the CMS for 'carry'. However, the paradigm does not at all fit the input. Not only is ´election´ not a physical object, but there is no location change of the subject or of the object. The program does not want to make such drastic changes to a hypothesis so often reconfirmed by previous experience. Therefore, it doesn´t modify the retrieved CMS at all. Instead it creates an entirely new CMS, taking case restrictions from the input words and environment change restrictions from the difference lists. The program now has two CMSs associated with ´carry´. Anytime a further instance of ´carry´ is encountered, it must determine which CMS is closest to the perceived situation.

There are, then, five learning processes used in three situations.

1) When a CMS accounts for the input but is associated with another surface verb, synonym learning occurs.

2) When the CMS accounts for the input and is associated with the input verb, confirmation learning occurs.

3) If the CMS does not account for the input, whether or not the word is known, the increasingly drastic measures of minor adjustment, major adjustment and definition creation are successively tried until a satisfactory CMS is generated. The CMS retrieved for modification may be currently associated with the input verb or with another surface verb. In the former case the retrieved CMS is modified, in the latter a copy of the retrieved CMS is

modified. The program first tries minor adjustment learning. If that is inadequate, major adjustment is tried. If too extensive major adjustment is required, the program uses definition creation to create a new CMS. Definition creation is also used when alternate word senses are encountered. The current system uses a fixed limit on the number of changes allowed before another type of learning is tried. Eventually the program must also consider the severity of the change when deciding at what point to abandon minor adjustment for major adjustment and major adjustment for definition creation.

It is interesting to note that Norman and Rumelhart [7] have recently classified learning into three different types. They also feel there are different types of learning and the type required depends on the amount of modification required to account for new information. Roughly, their accretion corresponds to what I call synonym and confirmation learning, their tuning to my minor adjustment learning and their restructuring to my major adjustment and definition creation learning.

CONCLUSION AND EXTENSIONS

The present system is designed as a framework for experimentation with a variety of learning techniques. The above discussion provides an outline of the system, its general organization, the major problems I am initially addressing and how I am attacking these problems. It should be viewed as a first approximation, a bootstrap system to facilitate experimentation with various learning mechanisms and decision procedures. It is expected that as implementation proceeds, more light will be shed on the different types of learning processes, the conditions under which the various processes should be applied, their results and implications.

It is already apparent that there are other techniques and learning mechanisms that need to be explored and possibly integrated into the basic program. It would be nice to add an interactive component that would allow the system to ask pertinent questions of a human informant. Even children are not able to get all the information they need simply by observing their environment. They constantly ask questions about what they know, what they think they know, how two situations are different or similiar, etc. Topics to explore include knowing when to ask a question, what questions are relevant to the problem and how the resultant information should be integrated into the existing knowledge bases.

An additional, and related, desirable feature would be to have the program take time out to closely inspect its knowledge from time to time. Such a component would allow learning to take place that is too complex or time-comsuming to occur during the general learning cycle. This phase would allow complex generalization, identification of recurring patterns and general consolidation. At this time the system could also utilize its question-asking ability to query its human informant and verify its generalized structures and knowledge.

REFERENCES

1. Dale, P.S. Language Development. Hinsdale, Illinois: The Dryden Press, Inc., 1972.

2. Fillmore, C.J. The Case for Case. In Bach and Harms (Eds.), Universals in Linguistic Theory. Chicago: Holt, Rinehart and Winston, 1968.

3. Gentner, D. Evidence for the Psychological Reality of Semantic Components: The Verbs of Possession. In 6.

4. Jordan, S. Learning to Use Contextual Patterns in Language Processing. Univ. of Wisconsin Computer Sciences Dept. Tech Report No. 152, 1972.

5. Kintsch, W. The Representation of Meaning in Memory. Hillsdale, N.J.: Lawrence Erlbaum Associates, 1974.

6. Norman, D. and Rumelhart, D. (Eds.), Explorations in Cognition. San Francisco: W.H. Freeman and Co., 1975.

7. Norman, D. and Rumelhart, D. Accretion, Tuning and Restructuring: Three Modes of Learning. University of Calif., San Diego Center for Human Information Processing Report 63, 1976.

8. Schank, R.C. Identification of Conceptualizations Underlying Natural Language. In Schank and Colby (Eds.), Computer Models of Thought and Language. San Francisco: W.H. Freeman and Co., 1973.

9. Schank, R.C. The Development of Conceptual Structures in Children. Stanford University Computer Science Dept. Memo 203, 1973.

10. Weir, R. Language in the Crib. The Hague: Mouton, 1972.

11. Winston, P.H. Learning Structural Descriptions from Examples. In Winston, P.H. (Ed.), The Psychology of Computer Vision. New York: McGraw-Hill, 1975.