Amsterdam Colloquium, Institute for Logic, Language and Computation, Universi
of Amsterdam, pp. 93-112.

Comprehension Grammars Generated from Machine Learning of Natural Languages

Patrick Suppes, Michael Böttner and Lin Liang

1 Introduction and Overview

We are developing a theory of machine language learning in the context of robotic instruction of elementary assembly actions. More specifically, our situations of language learning concern actions like moving to objects, picking up objects and moving objects in environments. Typical objects are screws, nuts, washers, and sleeves of various colours, sizes, and shapes, related to each other by common spatial relations. Typical commands to be comprehended are *Get a screw! Go to the black washer behind the screw! Put the screw in the left hole!* The system developed so far deals successfully with: English, Chinese, and German, as well as several other languages, which for a lack of space we will report on separately.

The overlap between the utterances that a person is able to understand and those that a person is able to produce may be small. A comprehension grammar would, e.g., be able to derive *Go to the left of the nut screw!* but not *Put a nut the near screw!* although neither of these utterances should be derivable in an English production grammar. A purely comprehension grammar should parse a superset of the set of strings we would expect a production grammar to derive.

Our axioms, given in Section 2, for machine learning of comprehension are long and complicated, even though we analyze here the learning of relatively small fragments of the natural languages considered but they do faithfully reflect the computer implementation, as well as extend the axioms given in our earlier paper (Suppes, Liang, Böttner, 1992), where the relevent literature is discussed. We do believe the learning principles expressed in the axioms can serve as the basis for learning much more extended fragments and represent principles that must be present in any system that starts with no knowledge of the natural language to be learned. The axioms naturally fall into three parts. First are those concerning computations using short-term memory based on learned associations; second those dealing with changes in the state of long-term memory, which include computations updating the comprehension grammar; and third the axiom describing the process of comprehension when more learning is not required to understand a particular command.

In Section 3 we describe the internal language, which is not learned, and then in Section 4 we provide an extensive commentary on the axioms, using, as would be expected, the resources of the internal language. In Section 5 we present detailed results on the comprehension grammars generated, and in Section 6, the final section, we review a number of significant problems we have not yet solved.

The results for the comprehension grammars of English, Chinese and German are linguistically surprising. Once congruence computations (see Axiom 2.5) are made, 9 grammatical rules are common to all three languages, 2 to Chinese and English, 7 to English and German, and 0 for Chinese and German. The surprise is that no special rules for English are required. Eight special rules are required for Chinese and 4 for German. As these numbers suggest and the details confirm, our comprehension grammars are vastly simpler than the corresponding production grammars.

2 Learning System

What we want to describe now is the process of learning in terms of the various events that happen on a given trial. Initially the robot has no knowledge of the lexicon or grammatical structure of the language to be learned. The robot begins a trial in a given state of memory. There are three parts of this memory that are changed due to learning: the probabilistic association relation, the set of grammatical forms and short-term memory of the current command and association computations.

1. The first is the association relation between words of a given language and internal symbols that represent denotations of members of categories in the internal representation language. The semantic categories of action, object, property and spatial relation, as well as the internal language (LISP), are given in advance and not learned. For example, the action of putting will be represented internally, let us say, by the symbol $\$p$ and will have three different linguistic representations in English, German and Chinese. The problem will be to learn in each language what word is properly associated with the internal representation $\$p$. The same process of association must take place for the internal symbols for objects, properties and spatial relations (Axiom 1.2). But knowing such associations is not enough, contrary to some naive theories of associationism.

2. It is also important to have grammatical forms in long-term memory (Axiom 3). We will not try to lay out everything about grammatical forms that we have in our fully stated theory, but it will be useful to give some examples. Consider the verbal command *Get the screw!* This would be an instance of the grammatical form *A the O!*, where *A* is the category of actions and *O* is the category of objects. As might be expected we do not actually have just a single category of actions, but several subcategories, depending upon the number of arguments required, etc. The central points here, however, are that: (i) the grammatical forms are derived only by generalization from actual instances of verbal commands given to the robot, and (ii) the grammatical forms are used in parsing any new command.

3. The third part of memory that varies is the short-term memory that holds a given verbal command for the period of the trial on which it is effective. This memory content decays and is not available for access after the trial on which a particular command is given (Axiom 1.6). This is also true of any association computations (Axioms 1.1–1.5).

Once a verbal command is given to the robot, the learning program looks up the associations of the words in the verbal command that has been given (Axiom 3a). If associations exist for any of the words, the categories of the associations, which are the categories of the internal interpretation, are also retrieved (Axiom 3b). The categories are substituted for the associated words and an effort is then made to generate recursively the resulting grammatical form (Axiom 3c). For example, if mistakenly the word *get* had been associated to $\$s$, the internal symbol for *screw,* and *screw* had been associated with $\$g$, the internal symbol for *get,* then the grammatical form that would have been generated and now found upon a second occurrence of the verbal command would be *O the A!* Now if there were no such grammatical form generated, once the associations were formed such a grammatical form would be created by the process of generalization which is used to generate grammatical forms (Axiom (1.3).

When a grammatical form is stored in memory, also associated with this grammatical form in memory is its internal representation (Axiom 2.6). This is an important part of the memory that changes with learning as well. If the grammatical form is generated, the internal representation is then used to execute the verbal command that has been given (Axioms 1.4 and 3b). If the command is executed correctly then the robot is ready for a new learning trial.

The important case of learning is when no grammatical form can be generated recursively from the grammatical forms in memory to match the form of the given verbal command. In this case the robot is unable to make a response. The correct response must be coerced (Axiom 1.1). On the basis of this coercion, a new internal representation is formed. At this point the critical step comes of a probabilistic association between words of the verbal utterance and the internal denoting symbols of the new internal representation (Axiom 1.2). After this association is made, a new grammatical form is generated (Axiom 1.3), and possibly because of the new associations at least one of the old grammatical forms is deleted, for, according to our axioms, a word or internal denoting symbol can have exactly one association, so when a new association is formed for a word any old associations must be deleted (Axiom 2.7). With the new associations formed and old ones possibly deleted, the robot is ready for the next trial in a new state of memory.

The learning process as described so far must be extended to the dynamic computations, recursive in nature, for updating the grammar as new commands that cannot be parsed by the existing grammar are encountered. Axiom 1.5 formulates the recursive procedure for generating new associations of parts of a grammatical form with parts of the associated internal form. Axiom 2.6 then characterizes the computations for dynamically changing the current grammar by adding a new rule or deleting a now redundant old rule.

2.1 Learning Principles

To facilitate the statement of principles governing the learning process just described, certain notational conventions are useful. First, generally we use Latin letters to refer to verbal commands or their parts, whatever the natural language, and we use Greek letters to refer to internal representations or their parts. The letters a, a_i, a'_i etc. refer to words in a verbal command, and the Greek letters α, α_j, α'_j, etc. refer to internal denotations. The Roman letter t, as well as t_i, t'_i refer to terms of a verbal command, and correspondingly, τ, τ_i, τ'_i to terms of an internal representation, i.e., in the present setup, LISP expressions. The symbols s, s', and $s(t)$, showing that t is a term of s, refer to entire verbal commands, and correspondingly σ, σ', $\sigma(\tau)$ to entire internal representations. Grammatical forms – either sentential or term forms – are denoted by g or also $g(X)$ to show a category argument of a form; correspondingly the internal representations of a grammatical form are denoted by γ or $\gamma(X)$. We violate our Greek-Latin letter convention in the case of semantic categories or category variables X, X', Y, etc. We use the same category symbols in both grammatical forms and their internal representations. In order to avoid enumeration of cases, when we use $\gamma(X)$ or $\tau(X,Y)$, for example, it is not required that X, or X and Y, occur as free variables in the grammatical form.

Axioms of Learning

1. Association Computations using Working Memory

1.1 *Association by contiguity.* If on any trial verbal command s is contiguous with a coerced action that has internal representation σ, then s is associated with σ, i.e., in symbols $s \sim \sigma$.

1.2 *Probabilistic association.* On any trial n, let s be associated to σ in accordance with Axiom 1, let $\{a_i\}$ be the set of words of s not associated to any internal denoting symbol of σ, let $d_n(a_i)$ be the current denotational value of each such a_i and let $\{\alpha_j\}$ be the set of internal denoting symbols not currently associated with any word of s. Then

i. an element α_j is uniformly sampled without replacement from $\{\alpha_j\}$,

ii. at the same time an element a_i is sampled without replacement from $\{a_i\}$ with the sampling probability

$$p(a_i) = \frac{d_n(a_i)}{\sum_{\{a_i\}} d_n(a_i)}.$$

iii. The sampled pairs are associated, i.e. $a_i \sim \alpha_j$.

iv. Sampling continues until either the set $\{a_i\}$ or the set $\{\alpha_j\}$ is empty.

1.3 *Form generalization.* If at any step of an association computation on any trial, $g(t(Y)) \sim \gamma(\tau(Y))$, $t(Y) \sim \tau(Y)$ and $X \rightarrow \tau(Y)$, then $g(X) \sim \gamma(X)$, where $X \rightarrow \tau(Y)$ is a grammatical rule of the internal language.

1.4 *Form specification.* If at any step of an association computation on any trial, $g(X) \sim \gamma(X)$, $t(Y) \sim \tau(Y)$ and $X \rightarrow \tau(Y)$, then $g(t(Y)) \sim \gamma(\tau(Y))$.

1.5 *Form association.* Let $g \sim \gamma$ at any step of an association computation on any trial,

(a) If X occurs in g and $(f o\ X\ *)$ occurs in γ, then $X \sim (f o\ X\ *)$.

(b) If

i. wX is the n-th occurrence of a substring of g with $g \sim \gamma$ and w is a **minimal** substring, possibly empty, of occurrences of nondenoting words, and

ii. $\tau(X)$ is the **maximal** LISP form of γ, containing the occurrence of X and no other occurrence of denoting categories,

then $wX \sim \tau(X)$.

(c) If

i. $X_1 w_1 X_2 \cdots w_n X_n$ is a substring of g, where the $X_i, i = 1, \ldots, n$ are not necessarily distinct category names and $w_i, i = 1, \ldots, n+1$ are substrings, possibly empty, or words that have no association to internal symbols on the given trial,

ii. $\tau(X_{\pi(1)}, \ldots, X_{\pi(n)})$ is the minimal LISP form of γ containing $X_{\pi(1)}, \ldots, X_{\pi(n)}$.

then $X_1 w_1 \cdots w_n X_n \sim \tau(X_{\pi(1)}, \ldots, X_{\pi(n)})$,
where π is a permutation of the numbers $1, \ldots, n$.

1.6 *Delete contents.* The content of working memory is deleted at the end of each trial. (Axiom 1.2).

2. Changes in State of Long-term Memory

2.1 At the beginning of trial 1, the association relation $\sim$, the congruence relation and the set of grammatical rules G are empty. Moreover, the denotational value $d(a_i)$ is the same for all words a_i.

2.2 The semantic categories and the semantically interpreted internal language I.L., as characterized in Definition 1, are stored in long-term memory and remain unchanged from trial to trial.

2.3 *Denotational Learning Computations.* If at the end of trial n a word a_i in the presented verbal stimulus is associated with some denoting internal symbol α_j of the internal representation σ of s at the end of the trial, then

$$d_{n+1}(a_i) = (1-\theta)d_n(a_i) + \theta,$$

and if a_i is not so associated

$$d_{n+1}(a_i) = (1-\theta)d_n(a_i).$$

Moreover, if a word a_i does not occur on trial n, then

$$d_{n+1}(a_i) = d_n(a_i),$$

unless the association of a_i to an internal symbol α_j is broken on trial n, in which case

$$d_{n+1}(a_i) = (1-\theta)d_n(a_i).$$

2.4 *Grammar computations.* First, if on trial n $g \sim \gamma$ on the basis of Axiom 1.5, then

(a) if no substring g' of g is already in long-term memory with $g' \sim \gamma'$, $g \sim \gamma$ is stored in long-term memory,

(b) if a substring g' of g is already in long-term memory with $g' \sim \gamma'$, then g and γ are reduced to $g(X) \sim \gamma(X)$ where X is the category generating γ'.

(c) This recursive reduction continues until no further reduction is possible. Then the reduced $g'' \sim \gamma''$ is added to long-term memory if not already present.

Second, if $g \sim \gamma$ is in long-term memory at the beginning of trial n but a substring $g' \sim \gamma'$ of g is generated on the basis of Axiom 1.5, then the reduced $g(X) \sim \gamma(X)$ is stored in memory and g, γ and $g \sim \gamma$ are deleted from long-term memory.

2.5 *Congruence computations.* If two G.F.s g and g' are such that

(a) $g \sim \gamma$ and $g' \sim \gamma$,

(b) g and g' have only category symbols as denoting terms,

(c) g and g' have exactly the same sequence of category symbols,

then g is γ-congruent to g'.

2.6 *Probabilistic associations stored in long-term memory.* This association is stored in long-term memory.

2.7 *Form memory trace.* The first time a form generalization (Axiom 1.3) is formed, the word associations on which the generalization is based are stored with it in long-term memory. This memory trace is inherited under the grammar computations (Axiom 2.4).

2.8 *Delete prior associations.* When a word in a verbal command or an internal symbol is given a new association (Axiom 2.3), any prior associations of that word or symbol are deleted from long-term memory.

2.9 *Delete form association.* If $a \sim \alpha$ is deleted (Axiom 3), then any generalization for which $a \sim \alpha$ is a memory trace is also deleted from long-term memory, as well as any G.F.s that inherited the memory trace.

3. **Comprehension-and-Response axiom**

If command s occurs at the beginning of trial n, then:

(a) Long-term memory is accessed for a possible associated internal symbol for each word of s;

(b) The category of each associated internal symbol is retrieved from long-term memory;

(c) A grammatical form g of s is computed by replacing each word a_i of s, which has an associated internal symbol α_j, by the symbol for the semantic category of α_j, in accordance with Axiom 1.3.

(d) Using the parser Π on the set G_n of grammatical rules in long-term memory, a parse of g is attempted.

(e) If exactly one parse g is found, the associated semantic form γ is computed in accordance with Axiom 1.4, and γ is then executed if possible;

(f) If more than one parse is found, say, the set $\Pi(g)$ of parses, then each associated constructed semantic form γ is tested for compatibility with the given physical environment E_n, and parses incompatible with E_n are deleted from the set $\Pi(g)$; one parse is selected at random from the reduced set $\Pi'(g)$ and the associated γ is executed if possible; if execution is not possible, no other parses are sampled.

(g) If no parse is found, no response is made.

We comment in detail on the learning process described by the axioms after the next section on the internal language.

3 Internal Language

The internal language is a LISP regimentation of English lexical semantics, but not, of course, English syntax. For example, the internal symbols for spatial relations correspond more closely to the semantics of English prepositions than to that of any of the other languages we have considered. It is an objective of our work to eliminate in the future this English lexical bias. We believe that much can be accomplished in this direction by forming for each set of concepts a partition fine enought to cover most if not all the languages we would be likely to consider. This is not to suggest that such partitions solve a host of other problems. Internal representations will therefore be LISP-expressions. A LISP-expression is any string $(FE_1, \ldots, E_n)$ where F is a function symbol and $E_1, \ldots, E_n$ are either atoms or LISP-expressions. The atoms refer to properties, actions, spatial relations, and objects of the environment and fall into corresponding categories. In particular, we have category P of properties like \$small, \$black, \$square; category OBJ of special properties serving to single out objects like \$screw, \$nut, \$washer, \$plate, \$hole, \$sleeve; category R of binary relations like \$front, \$back, \$left, \$right; category S of set of objects: $*$; action categories: A_1 like \$get, \$open, \$close, A_2 like \$go, A_3 like \$put.

In addition to these we have categories of expressions that are not atoms: category A of actions like (fa1 \$get (io (fo1 \$screw))); category O of objects like (fo1 \$screw $*$); category G of regions like (fo1 \$near (io (fo1 \$screw $*$).

Expressions denoting subsets of these categories are derived by the semantic operations listed below, but committment to a set-theoretical framework is not essential.

Grammar of Internal Language

1.	fo1 (focus-on-object-1)	S	→	(fo1 OBJ ∗)
2.		S	→	(fo1 P S)
3.		S	→	(fo1 G S)
4.	fo2 (focus-on-object-2)	S	→	(fo2 R S)
5.	io (identify object)	O	→	(io S)
6.	so (select-object)	O	→	(so S)
7.	fa1 (form-action-1)	A	→	(fa1 A_1 O)
8.	fa2 (form-action-2)	A	→	(fa2 A_2 G)
9.	fa3 (form-action-3)	A	→	(fa3 A_3 O G)
10.	fa4 (form-action-4)	A	→	(fa4 O)
11.		A	→	(fa4 D O)
12.	fdir (form-direction)	D	→	(fdir R)
13.	fr1 (form-region-1)	G	→	(fr1 R O)
14.	and (form-and-property)	P	→	(and P P)
15.	or (form-or-property)	P	→	(or P P)
16.	not (form-not-property)	P	→	(not P)

The robot's internal language is the set of expressions of category A. Notice that the operations also return expressions of our initial categories, in particular P and S which means that these categories are recursive. So we will have in our internal language symbols for each content word of a command. The English command *Get a screw!* has the following internal representation (fa1 $g (so (fo1 $s ∗))), where $\$g$ and $\$s$ occur in the internal language as counterparts of the English content words *get* and *screw*. The internal representation given above has three more symbols: *fo1*, *so*, and *fa1*. These symbols are abbreviations of the semantic operations *focus on object*, *select object*, and *form action*. The purpose of these operations is to provide what we think is the procedural structure of the natural language command mentioned above. Before it can perform the action denoted by $\$g$, the robot has to determine the object of this action. The object that is intended to become the object of the action is returned by the operation *so*. This operation is nondetermistic: it just selects one out of a set of objects. The input for *so* is a set of objects. This set of objects is the output of the operation *fo1*: it takes a property and the set of objects in the robot's environment and returns a set of objects. The set of objects in the robot's environment is represented by the symbol ∗ in our internal language. In our example, the property the environment is checked for is the presence of screws. The procedure *so* then takes this set of all screws of the environment as input and returns an arbitrary screw from this set. This particular screw together with the action g is the input of the operation *fa1*.

To illustrate our internal language we give some examples for basic types of action and object structures together with their associated English counterparts.

Action Structures

Get the screw!	∼	(fa1 $get (io (fo1 $screw ∗)))
Go near the screw!	∼	(fa2 $go (fr1 $$near_2$ (io (fo1 $screw ∗))))
Put the screw behind the nut!	∼	(fa3 $put (io (fo1 $screw ∗)) (fr1 $$behind_3$ (io (fo1 $nut ∗))))

Object Structures

screw	~	(fo1 $screw *)
large screw	~	(fo1 $large (fo1 $screw *))
screw which is large	~	(fo1 $large (fo1 $screw *))
screw which is not large	~	(fo1 (fp- $large) (fo1 $screw *))
large black screw	~	(fo1 $large (fo1 $black (fo1 $screw *)))
large and black screw	~	(fo1 (fp and $large $black) (fo1 $screw *))
left screw	~	(fo2 $left (fo1 $screw *))
left square screw	~	(fo2 $left (fo1 $square (fo1 $screw *)))
screw behind a nut	~	(fo1 (fr1 $behind (so (fo1 $nut *))) (fo1 $screw *))
large screw behind a nut	~	(fo1 $large (fo1 (fr1 $behind (so (fo1 $nut *))) (fo1 $screw *)))

Putting object structures into action structures we may derive structures of arbitrary degrees of complexity, as e.g.,

(fa3 $put (io fo1 (fr1 $near (io (fo1 $nut *))) (fo1 $screw *)) (fr1 $behind (io (fo2 $left (fo1 $hole *)))))

which corresponds to the English command

Put the screw that is near the nut behind the left hole!

The operations *fo1* and *fo2* have in common that they focus the robot's attention on objects of a certain kind in the set of objects that can be perceived by the robot on the occasion of being given a particular command: *fo1* takes a property and a set of objects and returns the objects that are referred to by the element of O. So the expression

(fo1 $screw *)

returns the set of all objects from the set in the robot's perceptual environment that are screws. In

(fo1 black (fo1 screw *))

fo1 operates recursively: it first returns the set of all screws in the robot's environment and then returns the subset of all black screws. The operation *fo2* focusses on a set of objects that bear another certain relation to a set of objects. A typical example is the English expression *left screw* which, in our internal language, is given the structure

(fo2 $left (fo1 $screw *))

A region is a set of locations that stand in a certain relation to at least one object. Regions are built from a binary spatial relation and an object by *fr1*. The operation *so* selects an arbitrary object from a set of perceptually given objects. The operation *io* identifies a unique object satisfying certain properties in a set of perceptually given objects. Not surprisingly a number of operations arise in the context of actions. This has to do with the various subclasses of actions depending on the different valencies exhibited by actions.

4 Detailed Comments on the Axioms

An important and distinguishing feature of the comprehension grammars we generate is that the denoting words of a language, as defined just above, are assigned to semantic rather than syntactic categories. For example, the English word *left* is in the category of spatial relations independent of its differing syntactic roles in the following phrases: *Move left! Go to the left! Pick up the left screw! Pick up the screw which is left of the box!* Similar examples appear in the other languages. Moreover, the nondenoting words are not assigned to any category at all except in the formal sense of the general category of being nondenoting, so that the usual syntactic distinctions that would apply to such words are not made. On the other hand, nondenoting words occur in different roles in grammatical forms.

4.1 Creativity

To get a better understanding of the axioms let us consider two examples that show that the grammars generated allow to comprehend new commands and an even infinite number of new commands. The first is to show that sentences can be understood by the grammar that are "new" to the learner. The second is to show that the grammar can analyse sentences that are longer than any of the sentences of the finite sample of the language to be learned.

Predication test. Once the generalized associations are established the internal representations of "new" commands can be derived. For instance, on the basis of having been learned correctly, *Get the screw!*, *Go to the nut!*, and *Get the nut!*, the following command can be understood by the comprehension grammars, and thus executed by the robot: *Go to the screw!*

Recursivity. Our system does not just derive new structures of the same length but is also able to cope with recursivity by understanding commands longer than any of the commands occurring in the learning sequence. On the basis of having learnt correctly

Get the nut!
Get the plate!
Get the red washer!
Get the nut which is left of the screw!
Get the screw behind the nut!

it would be able to to derive the following command:

Get the red screw behind the plate which is left of the washer!

Assume that, on the basis of previous experience, the robot's long-term memory holds the following associations:

nut ~ $n; *get* ~ $g ; *screw* ~ $s ; *plate* ~ $p; *washer* ~ $w;
red ~ $r; *behind* ~ $b; *left* ~ $l

Coerce:

Get the nut! ~ (fa1 $g (io (fo1 $n *))) (1)

Get the red washer! ~ (fa1 $g (io (fo1 $r (fo1 $w *)))) (2)

Get the nut which is left of the screw! ~ (fa1 $g (io (fo1 (fr1 $l (io (fo1 $s *)))
(fo1 $n *)))) (3)

Get the screw behind the nut! ~ (fa1 $g (io (fo1 (fr1 $b (io (fo1 $n *)))
(fo1 $s *)))) (4)

Form generalization applied to (2)–(5) extends memory by

A_1 *the OBJ!* ~ (fa1 A_1 (io (fo1 OBJ *))) (5)

A_1 *the P OBJ!* ~ (fa1 A_1 (io (fo1 P (fo1 OBJ *)))) (6)

A_1 *the OBJ which is R of the OBJ'!* ~ (fa1 A_1 (io (fo1 (fr1 R
(io (fo1 OBJ' *))) (fo1 OBJ *)))) (7)

A_1 *the OBJ R the OBJ'!* ~ (fa1 A_1 (io (fo1 (fr1 R (io
fo1 OBJ' *))) (fo1 OBJ *)))) (8)

Form association applied to (5)–(8) enables the robot to extend its memory by the following factorizations:

$OBJ \sim (fo1\ OBJ\ *)$ (9)

$the\ OBJ \sim (io\ (fo1\ OBJ\ *))$ (10)

$P\ OBJ \sim (fo1\ P\ (fo1\ OBJ\ *))$ (11)

$R\ of\ the\ OBJ \sim (fr1\ R\ (io(fo1\ OBJ\ *)))$ (12)

$R\ the\ OBJ \sim (fr1\ R\ (io\ (fo1\ OBJ\ *)))$ (13)

Form generalization applied to (7), (12) and Internal Production Rule 13 which we abbreviate IL Rule 13) yields

$$A_1\ the\ OBJ\ which\ isG! \sim (fa1\ A_1\ (io\ (fo1\ G\ (fo1OBJ*)))) \quad (14)$$

Form generalization applied to (8), (13) and IL rule 13 yields:

$$A_1\ the\ OBJ\ G! \sim (fa1\ A_1\ (io\ (fo1\ G\ (fo1OBJ*)))) \quad (15)$$

Form Association applied to (14) and (15) yields:

$OBJ\ which\ is\ G \sim (fo1\ G\ (fo1\ OBJ\ *))$ (16)

$OBJ\ G \sim (fo1\ G\ (fo1\ OBJ\ *))$ (17)

Form generalization applied to (16), (9) and IL rule 1 yields:

$$S\ which\ is\ G \sim (fo1\ G\ S) \quad (18)$$

Form generalization applied to (17), (19) and IL rule 1 yields:

$$S\ G \sim (fo1\ G\ S) \quad (19)$$

Form generalization applied to (9), (10) and IL rule 1 yields:

$$the\ S \sim (io\ S) \quad (20)$$

Form generalization applied to (9), (11) and IL rule 1 yields:

$$P\ S \sim (fo1\ P\ S) \tag{21}$$

Form generalization applied to (12), (20) and IL rule 5 yields:

$$R\ of\ O \sim (fr1\ R\ O) \tag{22}$$

Form generalization applied to (13), (20) and IL rule 5 yields:

$$R\ O \sim (fr1\ R\ O) \tag{23}$$

Form generalization applied to (5), (10) and IL rule 7 yields:

$$A_1\ O \sim (fa1\ A1\ O) \tag{24}$$

Grammatical Computation reduces the memory to the subset of associations (9), (18) - (24). Assume now that the following command is given to the robot:

$$Get\ the\ red\ screw\ behind\ the\ plate\ which\ is\ left\ of\ the\ washer! \tag{25}$$

This command is more complex than any of the stimulus commands. Notice that its G.F. does not occur in any of the original G.F.s:

$$A_1\ the\ P\ OBJ\ R\ the\ OBJ'\ which\ is\ R'\ of\ the\ P\ OBJ''! \tag{26}$$

Let us start from (24). We now make 9 successive applications of *Form specification.* We show only the associations and internal production rules used. By (20) and IL Rule 5 applied to (24):

$$A_1\ the\ S \sim (fa1\ A_1\ (io\ S))$$

By (19) and IL Rule 3:

$$A_1\ the\ S\ G \sim (fa1\ A_1\ (io\ (fo1\ G\ S)))$$

By (23) and IL Rule 13:

$$A_1\ the\ S\ R\ O \sim (fa1\ A_1\ (io\ (fo1\ (fr1\ R\ O)\ S)))$$

By (20) and IL Rule 5:

$$A_1\ the\ S\ R\ the\ S' \sim (fa1\ A_1\ (io\ (fo1\ (fr1\ R\ (io\ S'))\ S)))$$

By (18) and IL Rule 3:

$$A_1\ the\ S\ R\ the\ S'\ which\ is\ G \sim (fa1\ A_1\ (io\ (fo1\ (fr1\ R\ (io\ (fo1\ G\ S')))\ S)))$$

By (22) and IL Rule 13:

$$A_1\ the\ S\ R\ the\ S'\ which\ is\ R'\ of\ O \sim$$
$$(fa1\ A_1\ (io\ (fo1\ (fr1\ R\ (io\ (fo1\ (fr1\ R'\ O)\ S')))\ S)))$$

By (20) and IL Rule 5:

$$A_1\ the\ S\ R\ the\ S'\ which\ is\ R'\ of\ the\ S'' \sim$$
$$(fa1\ A_1\ (io\ (fo1\ (fr1\ R\ (io\ (fo1\ (fr1\ R'\ (io\ S''))\ S'))))\ S)))$$

By (21) and IL Rule 2:

$$A_1\ the\ P\ S\ R\ the\ S'\ which\ is\ R'\ of\ the\ S'' \sim$$
$$(fa1\ A_1\ (io\ (fo1\ (fr1\ R\ (io\ (fo1\ (fr1\ R'\ (io\ S''))\ S')))\ (fo1\ P\ S))))$$

By (9) and IL Rule 1:

$$A_1\ the\ P\ OBJ\ R\ the\ OBJ'\ which\ is\ R'\ of\ the\ OBJ'' \sim$$
$$(fa1\ A_1\ (io\ (fo1\ (fr1\ R\ (io\ (fo1\ (fr1\ R'\ (io\ (fo1\ OBJ''\ *)))$$
$$(fo1\ OBJ'\ *))))\ (fo1\ P\ (fo1\ OBJ\ *)))))$$

Notice that this association matches the G.F. of (26).

4.2 Contiguity

Our learning algorithm is based on the idea that the structure of the internal language can be used to derive the constituent structure of natural language utterances. This idea easily invites the objection that the algorithm is successful for languages with only contiguous constituents but inevitably fails for languages with noncontiguous constituents. A case in point is the German command

Heb die Schraube hoch, die gross ist! (1)

Note that in this command *Schraube die gross ist* forms one constituent from the standpoint of our internal language. This constituent, however, is not contiguous since it is interrupted by the constituent *hoch*. Now there is in German also a way to express the command expressed by (1) with contiguous constituents:

Heb die Schraube, die gross ist, hoch! (2)

(We disregard for the moment that one could argue that (2) has also noncontiguous constituents since *heben* and *hoch* are forms of one word *hochheben*.) But (2) is considered less natural and the algorithm should be able to come up with a grammar on the basis of both (1) and (2). In fact, our algorithm is able to derive a grammar in both cases. The grammar derived from (2) will have among its rules the following one:

$$A_4\ O\ ist\ D \sim \text{(fa4 } A_4 \text{ D O)} \quad (3)$$

The grammar derived from (1), however, will not have this rule but the following rule instead:

$$A_4\ die\ S\ D\ die\ P\ ist \sim \text{(fa4 } A_4 \text{ (io (fo1 P S)) D)}$$

Note that (2) is less general than (1). All the rules specific to German are of that kind.

4.3 Context-free, but not Regular Languages

Our axioms can generate grammars for languages that are context-free but not regular. This may happen because of the presence of so-called center-embedding structures exhibited by the following German example:

1. *Nimm die nahe der Mutter befindliche Schraube!*
2. *Nimm die nahe der hinter dem Loch befindlichen Mutter befindliche Schraube!*
3. *Nimm die nahe der hinter dem vor der Platte befindlichen Loch befindlichen Mutter befindliche Schraube!*

...

Our axioms are able to deal with center-embedding structures by having the following associations in memory:

$$R \textit{ der } S' \textit{ befindliche } S \sim (\text{fol } (\text{fr R } (\text{io S}'))\ \text{S}).$$

This amounts to having the following rule in a grammar for German:

$$S \rightarrow R \textit{ der } S' \textit{ befindliche } S.$$

4.4 Concept of Denotational Value

In the probabilistic theory of machine learning of natural language which we have been developing, we have encountered in a new form a standard problem in the analysis of the semantics of natural language, namely, how to handle words that are nondenoting. We do not mean nondenoting in some absolute sense, but relative to our standard set of semantical categories. It may well be that in some elaborate set-theoretical semantics of natural language, nondenoting words like the definite article *the* denote a complicated set-theoretical function. We have something simpler and closer to the common man's view of what denotations are. We take as denoting words color words, common nouns, familiar concrete action words, etc.

When a child learning a first language or an older person learning a second language first encounters utterances in that new language, there is no uniform way in which nondenoting words are marked. There is some evidence that various prosodic features are used in English and other languages to help the child. For example, in many utterances addressed to very young children the definite or indefinite article is not stressed but rather the common noun it modifies, as in the expression *Hand me the cup!* But such devices do not seem uniform and in any case are not naturally available to us in our machine-learning research, where we use written input of words without additional prosodic notation. A central feature of our approach to machine learning is the probabilistic association between words of the natural language being learned and internal symbols that denote, as explained earlier.

It is appropriate that at the beginning all words are treated equally and so the associations are formed from sampling based on a uniform distribution. On the other hand, after many words have been learned and a good piece of language has been acquired by the robot, it is very unnatural, and also inefficient, if the robot is now given the esoteric command *Get the astrolabe!* to have the internal symbol *$ast* to be associated with equal probability with the definite article *the* and *astrolabe*–we assume here that the association of *get* is already correctly fixed. After much experience, what we want is that there is very little chance of associating the definite article *the* with any denoting symbol.

To incorporate such learning, many variant learning models are easily formulated. We have restricted ourselves in Axiom 2.3 to a familiar linear learning model (Estes and Suppes, 1959). We begin with each word w of the given natural language having the value $d_1(w) = 1$. It is obvious from Axiom 2.3 that for all trials n, $0 \leq d_n(w) \leq 1$. For reasons we shall not expand upon here we chose as the value of the learning parameter $\theta = 0.03$. We show in

English		Chinese		German	
word	d-value	word	d-value	word	d-value
plate	0.997	ban	0.999	Platte	0.998
nut	1.000	luomu	0.999	Mutter	0.999
large	0.967	dade	1.000	gross	1.000
place	0.986	fang	0.970	bring	0.989
in	0.972	jing	0.999	in	0.889
the	0.003	nage	0.007	die	0.002
a	0.003	yige	0.005	eine	0.002
		ba	0.001		
		na4	0.127		
		na4li	0.246		

Table 1: Selected Denotational Values

Table 1 the denotational values of selected words from English, Chinese and German after approximately 400 learning trials using the corpus of commands described in Section 5. As desired the denotational value of the denoting words in the sense defined above remain close to 1, and the nondenoting words approach 0.

4.5 Congruence of Nondenoting Words

To reduce the number of grammatical rules and facilitate comparison of rules for different languages, we introduced in the last part of Axiom 2.5 the concept of comprehension congruence, applied here only to the limited case of nondenoting words. The sense of congruence use has, as the formulation of the axiom shows, a specific and definite content. The idea of congruence used here was developed earlier in Suppes (1973, 1991). To illustrate ideas, consider this German case:

Nimm die *Schraube!* $\sim$ (fa1 $get (io (fo1 $screw *)))
Nimm den *Ring!* $\sim$ (fa1 $get (io (fo1 $washer *)))
Nimm das *Loch!* $\sim$ (fa1 $get (io (fo1 $hole *)))

Now generalize to the grammatical forms without denoting words, only category symbols, and we have:

A_1 *die OBJ !* $\sim$ (fa1 A1 (io (fo1 Obj *)))
A_1 *den OBJ !* $\sim$ (fa1 A1 (io (fo1 Obj *)))
A_1 *das OBJ !* $\sim$ (fa1 A1 (io (fo1 Obj *)))

Then, using γ-congruence, we may write as a single grammatical form

$$A_1\ [die]\ OBJ! \sim (\text{fa1 A1 (io (fo1 Obj *)))}$$

where

$$[die]_i = (die,\ den,\ das)$$

and i is the rule number, which indexes the internal semantic form γ. We also note that in many cases, we include in the congruence class $[w]_i$ the empty string ϵ, if the corpus has generated such a variant grammatical form. For example, prior to computing congruence, we generated for Chinese the following three rules:

$$
\begin{array}{lll}
A \rightarrow & O\ A_1 & \sim \text{ (fa1 A1 O)} \\
A \rightarrow & ba\ O\ A_1 & \sim \text{ (fa1 A1 O)} \\
A \rightarrow & xianzai\ ba\ O\ A_1 & \sim \text{ (fa1 A1 O)}
\end{array}
$$

Computing γ-congruence, we reduce these three rules to a single one:

$$A \rightarrow [ba]\ A_1\ O \sim \text{(fa1 A1 O)}$$

where $[ba]_i = (ba,\ xianzai\ ba,\ \epsilon)$.

In reporting the grammatical results for English, Chinese and German, we use an ordered triple notation for a given congruence class of nondenoting words of a rule common to all three languages: first the English words, then the Chinese words and finally the German words, with ϵ occurring as needed for each language. Thus the first common rule listed in Section 5 is

$$O \rightarrow [DA]\ S,$$

where $[DA]$ =*(the; nage, zhege; das, den, dem, der, die).*

5 Grammars Generated

Comprehension grammars have been generated so far for English, Chinese, German, French, Dutch and Korean. Mainly because the first languages of the authors are English, German and Chinese respectively, we will concentrate on these three languages with remarks about the others. The corpus used has consisted of 456 commands, the last 60 of more complicated ones. In the first group commands range from the simple *Get the screw!* to *Place the screw on the plate !* In the last 60 a typical more complicated command is the following: *put a small nut on the screw behind the washer left of the plate*; *ba yige xiaode luomu fang zai nage ban zuobian de dianquan houmian de nage luosiding shang*; *tu eine klein-e Mutter auf die Schraube hinter dem Dichtungsring links von der Platte.*

Close translations of the 456 commands were used for the other languages. It was intended that the translations would be semantically equivalent but would be faithful to idiomatic constructions in the given language.

The details of the learning curves that are generated from our procedure of machine learning are as described in the axioms but will not be analyzed here, because we already have a very thorough discussion of learning curves in our earlier paper (Suppes, Liang, Böttner, 1992). We also do not analyze here in detail the learning of denotational value as characterized in Axiom 2.3. We do show in Table 1 the denotational values for some typical words, denoting and nondenoting in each of the three languages, English, Chinese and German, after approximately 400 learning trials for English and Chinese, and 800 for German. It is evident from the table that the denoting words have a denotational value that is close to 1, and the nondenoting words a value close to 0 as desired. The empirical details of the denotational model are being written up in another paper, but it is important to emphasize that just as we assume no grammar for any of the languages we study, we also do not assume in our machine learning an a priori classification of words as denoting or nondenoting.

We now turn to the three grammars generated for English, Chinese and German. The grammatical rules are written in context-free notation, with the associated internal semantic forms shown on the right. The congruence classes are listed separately. The generation particularly uses Axiom 2.4 on grammar computations. Using the axiom on congruence

(Axiom 2.5) to collapse rules that differ only in the occurrence of semantically equivalent nondenoting words, the number of rules for English is 18, for Chinese 19, and for German 20. What is of perhaps greater interest is the analysis of the structure of common rules in comparison with specials rules for the particular languages. The basic numerical data are these. There are 9 rules that are common to English, Chinese and German where 'commonality' means that the category such as definite article now includes words from each of the three languages, in this particular case for example, *the, zhege, nage, den, das, die, dem,* and *der*. In addition, there are 2 rules common to English and Chinese that are not in the German grammar, there are 7 rules common to English and German that are not used in Chinese, and no rules common only to Chinese and German.

The 9 rules common to English, Chinese and German are the following:

1. $O \rightarrow [DA]S \sim$ (io S)
2. $O \rightarrow [IA]S \sim$ (so S)
3. $S \rightarrow P\ S \sim$ (fo1 P S)
4. $S \rightarrow OBJ \sim$ (fo1 OBJ *)
5. $G \rightarrow R\ [PO]O \sim$ (fr1 R O)
6. $D \rightarrow R \sim$ (fdir R)
7. $P \rightarrow P\ [\&]P \sim$ (and P P′)
8. $P \rightarrow [\neg]P \sim$ (not P)
9. $P \rightarrow P\ [\vee]P \sim$ (or P P′)

$[DA]_1$ = *(the; nage, zhege; das, dem, den, der, die),*
$[IA]_2$= *(a; yige; eine, einem, einen, einer),*
$[PO]_5$= *(of, ϵ; ϵ; von, ϵ),*
$[\&]_7$= *(and; he; und),*
$[\neg]_8$= *(not; busi; nicht),*
$[\vee]_9$= *(or; huozhe; oder)*

where the congruence classes for nondenoting words are shown below the nine rules.

What is remarkable about the above 9 rules is that none of them are high level rules for the generation of complete commands. The first 4 deal with the generation of object phrases, Rule 5 with the generation of a description of a region, Rule 6 with a direction, and the last 3 with properties.

There are two common rules for English and Chinese not used in the German grammar:

1. $A \rightarrow [ADV]\ A_4\ D\ O \sim$ (fa4 A4 D O)
2. $A \rightarrow A_4\ O \sim$ (fa4 A4 O)

$[ADV]_1$= *(now, so, ϵ; ϵ)*

Note that these two rules are high level rules for generating commands.

The 7 rules common to English and German are the following:

1. $A \rightarrow [ADV]\ A_1\ O\ [COP] \sim$ (fa1 A1 O)
2. $A \rightarrow [ADV]\ A_2\ G\ [COP] \sim$ (fa2 A2 G)
3. $A \rightarrow G\ A_3\ O \sim$ (fa3 A3 O G)
4. $A \rightarrow A_3\ O\ [COP]\ G \sim$ (fa3 A3 O G)
5. $A \rightarrow [ADV]\ A_4\ [COP]\ O\ D \sim$ (fa4 A4 D O)
6. $S \rightarrow S\ [which\ is]\ P \sim$ (fo1 P S)
7. $S \rightarrow S\ [which\ is]\ G \sim$ (fo1 G S)

$[ADV]_1$ = *(now, so, ϵ; nun, ϵ)*
$[COP]_1$ = *(ϵ; ist, ϵ)*
$[ADV]_2$ = *(now, so, ϵ; dann, jetzt, nun, ϵ)*
$[COP]_2$ = *(ϵ; ist, ϵ)*
$[COP]_4$ = *(ϵ; ist, ϵ)*
$[ADV]_5$ = *(now, so, ϵ; ϵ)*
$[ADV']_5$ = *(ϵ; jetzt, nun, ϵ)*
$[RP]_6$ = *(that is, which is; der, die, ist die)*
$[RP]_7$ = *(that is, which is, ϵ; der, die, die sich, ϵ)*

Note that 5 of the 7 are high-level rules for generating commands. The remaining 2 are for generating object phrases.

The rules special for German and the congruence classes are the following:

1. $A \rightarrow A_4$ *[einen] S D [der] P [ist]* ~ (fa5 A_4 D (so (fo1 P S)))
2. $A \rightarrow A_4$ *[nun die] S D die G ist* ~ (fa4 A_4 D (io (fo1 G S)))
3. $A \rightarrow A_4$ *die S R von dem S D der G ist* ~ (fa4 A_4 D (io (fo1 (fr1 R (io (fo1 G S))) S)))
4. $A \rightarrow A_4$ *die S R der S D die P ist* ~ (fa4 A_4 D (io (fo1 (fr1 R (io (fo1 P S))) S)))

[der]$_1$ = *(der, die)*
[einen]$_1$ = *(einen, jetzt eine)*
[ist]$_1$ = *(ist, ε)*
[nun]$_2$ = *(nun die, die)*

The 8 rules special for Chinese, are the following:

1. $A \rightarrow$ *[ba] O* A_1 ~ (fa1 A1 O)
2. $A \rightarrow$ *[xianzai] G [na4]* A_2 ~ (fa2 A2 G)
3. $A \rightarrow$ *zai G* A_3 *O* ~ (fa3 A3 O G)
4. $A \rightarrow$ *[ba] O* A_3 *[dao1] G* ~ (fa3 A3 O G)
5. $A \rightarrow$ *[ba] O* A_4 *D* ~ (fa4 A4 D O)
6. $O \rightarrow$ *[ba] S* ~ (io S)
7. $S \rightarrow$ *G [de] S* ~ (fo1 G S)
8. $G \rightarrow$ *O [de] R* ~ (fr1 R O)

[ba]$_1$ = *(ba, xianzai ba, ϵ)*
[xianzai]$_2$ = *(chao, name, xianzai)*
[na4]$_2$ = *(na4, na4li)*
[ba]$_4$ = *(ba, ϵ)*
[dao1]$_4$ = *(dao1, zai, ϵ)*
[ba]$_5$ = *(ba, xiazai ba, ϵ)*
[ba]$_6$ = *(ba, dao1, xianzai, zai, ϵ)*
[de]$_7$ = *(de, de nage)*
[de]$_8$ = *(de, ϵ)*

Five of these 8 are high-level rules for generating commands. Rules 6 and 7 are for generating object phrases, and Rule 8 for generating region description.

The big surprise is that no special grammatical rules are required for English. In no sense did we anticipate this outcome.

The congruence class for Rule 6 is intuitively incorrect. The reason is simple to explain. The Chinese particles *xianzai, dao1, zai,* and *ba* are not associated with object phrases, as

the rules suggest, but with other categories of words. For example, *xianzai* has a meaning that is close to the English adverb *now*, the particle *dao1* is associated with verbs, as is *ba*. An example from our corpus of the use of *zai* would be the following. *Get the screw on the plate.* has the following translation in Chinese:

ba	*nage*	*zai*	*(nage)*	*ban*	*shang*	*de*	*luosiding*	*naqi.*
	the		(the)	plate	on		screw	get.

In this example, we mention that the second occurrence of *nage* is deleted because it is awkward to say in Chinese *nage zai nage*. Note also in this example that although *ba* is associated with the verb *naqi*, *ba* comes at the beginning of the sentence, *naqi* at the end which would make for difficulties if we had to assign a specific semantic meaning to *ba*. What is important is that at the level of the commands we are considering, the non-intuitive Rule 6 is satisfactory for the purposes of comprehension. It would of course lead to very bad Chinese if applied to the production of utterances. We emphasize once again that in considering these Chinese examples, we are generating comprehension by uniform procedures that are the same across all languages. We are not claiming this can be done for production or even that for the ultimate reaches of comprehension it will be satisfactory. On the other hand it is important for machine learning purposes to understand how far uniform procedures can be satisfactorily exploited.

6 Problems not solved

Problems arise from the following assumptions: the association relation is one-one, and the internal language is a LISP regimentation of English lexical semantics, as already mentioned. Here are some examples.

1. Contraction of denoting and nondenoting

   (a) German:
   Geh zur *Schraube!* ~ (fa1 $go ($to (io (fo1 $screw *))))
   Geh zu *der Schraube!* ~ (fa1 $go ($to (io (fo1 $screw *))))
   Geh zum *Loch!* ~ (fa1 $go ($to (io (fo1 $hole *))))
   Geh zu *dem Loch!* ~ (fa1 $go ($to (io (fo1 $hole *))))

   (b) French:
   Va au *trou!* ~ (fa1 $go ($to (io (fo1 $hole *))))
   Va à *la vis!* ~ (fa1 $go ($to (io (fo1 $nut *))))

2. Only one word for different internal symbols (French)

   Ferme *la porte!* ~ (fa1 $close (io (fo1 $door *)))
   Ferme *la porte!* ~ (fa1 $lock (io (fo1 $door *)))

3. More than one synonymous word corresponding to same denoting symbol

   Put *the screw near the washer!*
   ~ (fa3 $put (io (fo1 $screw *)) (fr1 $near (io (fo1 $washer *))))
   Place *the screw near the washer!*
   ~ (fa3 $put (io (fo1 $screw *)) (fr1 $near (io (fo1 $washer *))))

4. More than one nonsynonymous word corresponding to same denoting symbol

(a) German:
Leg *die Schraube in die Schachtel!*
~ (fa1 $put (fo $screw *) fr1 $in (io (fo $box *))))
Steck *die Schraube in das Loch!*
~ (fa1 $put (fo $screw *)(fr1 $in (io (fo $hole *))))

(b) French:
vis ronde ~ (fo1 $round (fo1 $screw *))
douille cylindrique ~ (fo1 $round (fo1 $washer *))

5. Concepts that are expressed easily in one language can be more difficult to express in another language. For example, the English *go behind the plate* does not have a natural direct translation in Dutch, even though *go* can generally be translated by *gaan* in Dutch.

As a consequence, we made certain adjustments in the corpora of natural languages:

1. Since our system currently cannot identify different forms of the same denoting word, we mark the boundary between stem and ending by a hyphen, e.g., in French *vis rond-e* or German *dem rund-en Loch*

2. There are no words in French and German that correspond to the English verb *pick up* with respect to its "constructional range". We chose the German translation *hochheben* because it resembles the English construction at least for the imperatives under consideration where the verb gets split into two parts and *heben* can be associated with *pick* and *hoch* can be associated with *up*. On the other hand, in cases like *pick a screw!* the word *heben* does not appear to be usable and we switched to *nehmen*. In French we used *ramasser* as a translation for *pick up* and *saisir* as a translation for *pick*.

3. The English preposition *on* has no unique counterpart in German and French. Some contexts require it to be translated by *auf/sur* as in, e.g., *Put the screw on the plate!* for other contexts a translation by *an/à* would be more natural as in, e.g., *Put the nut on the screw!* We used *auf* and *sur* in all cases even in cases where the visual scene clearly suggests a different preposition.

4. If a language had no single word for a certain English expression but had only an expression which was longer than one word, we used this and marked it by hyphenations in a fashion that the machine would take it for one word: for French we used *de-taille-moyenne* for the English adjective *medium*.

5. In French we used *à* instead of *au* since *à* is needed as a counterpart for a denoting symbol of the IL. We also used *de le*, *de un*, and *de une* instead of *du*, *d'un*, and *d'une* respectively.

7 Acknowledgements

The research of Michael Böttner has been supported by grant 683/1-2 from the Deutsche Forschungsgemeinschaft. We are indebted to Michael Donio for the analysis of French and Paul Meyer for the analysis of Dutch.

8 References

Estes, W. K., and P. Suppes (1959) Foundations of linear models, R. R. Bush and W. K. Estes (eds.), *Studies in Mathematical Learning Theory*, Stanford University Press, Stanford

Suppes, P. (1973) Congruence of meaning. *Proceedings and Addresses of the American Philosophical Association*, 46, 21–38

Suppes, P. (1991) *Language for Humans and Robots*, Blackwell, Oxford

Suppes, P., Liang, L. and Böttner, M. (1991) Complexity issues in robotic machine learning of natural language, L. Lam and V. Naroditsky (eds.), *Modeling Complex Phenomena*, pp. 102-127, Springer, New York