UNIVERSITY OF CALIFORNIA

IRVINE

Learning and Using Context in a Connectionist Model of Natural Language Understanding

DISSERTATION

submitted in partial satisfaction of the requirements for the degree of

DOCTOR OF PHILOSOPHY

in Information and Computer Science

by

David Alan Schulenburg

Dissertation Committee:

Professor Michael J. Pazzani, Chair

Professor Dennis F. Kibler

Professor Benjamin N. Colby

1995

The dissertation of David Alan Schulenburg is approved,

and is acceptable in quality and form

for publication on microfilm:

Committee Chair

University of California, Irvine

1995

Dedication

For my family

and L., L., L., & M.

Without your love, support, encouragement, and patience,

it never would have happened.

Contents

List of Figures

List of Tables

Acknowledgments

First, I thank my advisor, Michael Pazzani; I am convinced that there can be no better advisor anywhere. Mike let me do what I wanted to do and even encouraged me on my chosen topic. He gave much needed guidance and showed considerable patience. I learned a lot from you Mike; thanks ever so much. I also thank the other members of my doctoral committee, Dennis Kibler and Nick Colby, for their comments and guidance on my work.

Being a graduate student for these many years, I have come to know many wonderful people who have helped to make graduate student life truly enjoyable. In particular, I thank my friends, Jim Fradkin, Randy Jones, Jim Kipps, Lilace Mellin, Bernd Nordhausen, Lorraine Hinz Walsh, Jim Wogulis, and Hadar Ziv for all the years of fun, travel, friendship, poker, and home-brewing. Volleyball was kind of fun too. I thank the following colleagues/friends for making graduate school an enjoyable and stimulating place to be: David Aha, Kamal Ali, John Allen, Cliff Brunk, Tim Cain, Piew Datta, Dan Frost, John Gennari, Rogers Hall, Tim Hume, Wayne Iba, Chris Merz, Patrick Murphy, Don Rose, David Ruby, Stephanie Sage, Kevin Thompson, and Tony Wieser. The staff and computer support people here in the Department of Information and Computer Science have made this a good place to be a student, especially Caroline Ehrlich, Mary Day, and Susan Moore.

Thanks go to the Aerospace Corporation, which gave me employment during much of my graduate student career. Special thanks to those in the Computer Assurance Section, past and present: Mark Bouler, Jeff Cook, John Doner, Ivan Filippenko, Ranwa Haddad, Beth Levy, Leo Marcus, and Telis Menas.

Special thanks go to Universal Press Syndicate for allowing me to reprint gratis the *Calvin and Hobbes* comic strips found herein. Also to mention is the permission from Chronicle Features for allowing me to reprint the *The Far Side* comic strip.

Most importantly, I thank my family: my parents Ed and Judy, my brothers Dan and Butch, my sister Sande and all the rest for all of their love and support throughout my graduate student career.

Curriculum Vitae

David Alan Schulenburg

1976–1978	Programmer, Sperry Univac MCO
1977	B.A. in Mathematics, B.S. in Information and Computer Science, University of California, Irvine
1979	MCO Product Analyst, Sperry Univac (Australia) Ltd.
1980–1981	Systems Analyst, Johnson Controls, Inc.
1981–1982	Research assistant, Department of Information and Computer Science, University of California, Irvine
1982	M.S. in Information and Computer Science, University of California, Irvine Thesis: *Generating Alternate Interpretations: Directing Inference During Story Understanding* Professor Richard H. Granger, Jr., Irvine, Chair
1982–1983	Scientist, Naval Ocean Systems Center
1983–1984	Research Scientist, Cognitive Systems, Inc.
1985–1990	Research assistant, Teaching Assistant, Instructor, Department of Information and Computer Science, University of California, Irvine
1987	Engineering Intern, MAI Basic Four
1990–present	Member Technical Staff, The Aerospace Corporation
1995	Ph.D. in Information and Computer Science, University of California, Irvine Dissertation: *Learning and Using Context in a Connectionist Model of Natural Language Understanding* Professor Michael J. Pazzani, Irvine, Chair

Publications

Schulenburg, D. (1995). Learning and Using Context in a Connectionist Model of Natural Language Understanding. To appear in *IJCAI Workshop on Context in Natural Language Processing.* Montreal, Canada.

Pazzani, M. J., P. Murphy, K. Ali, and D. Schulenburg. (1994). Trading off coverage for accuracy in forecasts: Applications to clinical data analysis. In *Working Notes: AAAI Spring Symposium: AI in Medicine: Interpreting Clinical Data.* Stanford, CA.

Schulenburg, D. (1993). Using context to interpret indirect requests in a connectionist model of NLU. In *Proceedings of the Fifteenth Annual Conference of the Cognitive Science Society.* Hillsdale, NJ: Lawrence Earlbaum Associates, Inc.

Schulenburg, D. (1992). Sentence processing with realistic feedback. In *Proceedings of the International Joint Conference on Neural Networks.* Piscataway, NJ: IEEE.

Schulenburg, D. (1992). *Sentence Processing with Incremental Feedback.* Technical Report 92-51. Department of Information and Computer Science, University of California, Irvine.

Schulenburg, D. (1991). Using a connectionist model to learn indirect speech act interpretation (extended abstract). In *Working Notes: AAAI Spring Symposium: Connectionist Natural Language Processing.* Stanford, CA.

O'Rorke, P., S. Morris, & D. Schulenburg (1990). Theory formation by abduction: A case study based on the chemical revolution. In *Computational Models of Scientific Discovery and Theory Formation*, Jeff Shrager and Pat Langley, Eds. San Mateo, CA: Morgan Kaufman.

Schulenburg, D. & M. Pazzani (1989). Explanation-based learning of indirect speech act interpretation rules. In *Proceedings of the First International Workshop on Lexical Acquisition.* Detroit, MI.

Pazzani, M. & D. Schulenburg (1989). The influence of prior theories on the ease of concept acquisition. In *Proceedings of the Eleventh Annual Conference of the Cognitive Science Society.* Hillsdale, NJ: Lawrence Earlbaum Associates, Inc.

O'Rorke, P., S. Morris, & D. Schulenburg (1989). Abduction and world model revision. In *Proceedings of the Eleventh Annual Conference of the Cognitive Science Society.* Hillsdale, NJ: Lawrence Earlbaum Associates, Inc.

O'Rorke, P., S. Morris, & D. Schulenburg (1989). Theory formation by abduction: Initial results of a case study based on the chemical revolution. In *Proceedings of the Sixth International Workshop on Machine Learning.* Palo Alto, CA: Morgan Kaufmann.

Abstract of the Dissertation

Learning and Using Context
in a Connectionist Model
of Natural Language Understanding

by

David Alan Schulenburg

Doctor of Philosophy in Information and Computer Science

University of California, Irvine, 1995

Professor Michael J. Pazzani, Chair

Natural languages are ambiguous. This is especially true for non-literal figurative constructs such as metaphors and indirect speech acts. Even literal text suffers from problems of ambiguity as exemplified by text containing words having multiple meanings. Understanding such ambiguous text is a fairly simple task for us humans; it is well recognized that context is often used by people to aid in the resolution of these problems of ambiguity.

This dissertation presents a discussion of a computational model, PIP, which was designed to address the issue of ambiguity in natural language understanding. By incorporating a textual context during the understanding process, PIP is able to disambiguate text containing ambiguous constructs such as metaphors and indirect speech acts. Furthermore, the entire understanding process is uniform in the sense that the exact same mechanisms are used to process both literal and non-literal (ambiguous) text; no special processing "rules" are necessary to deal with the ambiguity.

The underlying computational model of PIP is the feed-forward artificial neural network. The incorporation of the textual context is accomplished by a recurrent relation between the context that is being constructed and the network input

for processing the words of the text. In this fashion is PIP able to use the context directly as it processes text.

PIP has demonstrated its effectiveness on sets of text which include ambiguous lexemes, metaphors, and indirect speech acts. By using the context constructed from earlier sentences in these texts, PIP is able to derive the intended meaning of the ambiguous sentences at the end of the texts. It is shown that without use of the context, PIP is unable to produce the intended meaning and in many cases, cannot decide on any meaning to give to the ambiguous sentences.

Chapter 1

Defining a Problem

> "Begin at the beginning," the King said, very gravely,
> "and go on till you come to the end: then stop."
>
> —LEWIS CARROLL, *Alice's Adventures in Wonderland*

Imagine yourself walking down the avenue; you are late for a meeting. You notice that a stranger coming the other way is wearing a watch and you say to him, "Excuse me, do you know what time it is?" There are, of course, many ways he could answer you. He could say "yes," and walk away; or maybe "no, I don't." These seem like rather strange responses, particularly since you know that he has a watch. Yet, they could be absolutely true. The stranger has answered your question as you asked it; after all, you did ask whether or not he *knew* the time. You, on the other hand, would probably be puzzled, maybe thinking to yourself: "Gee, doesn't this person know that what I am really asking is for him to tell me the time, possibly looking at his watch first?" Could the problem lie in the manner in which the question was asked? With a "yes" or "no" response, the stranger has failed to understand the *intent* of your query.

Contrast the example above with the following. Imagine that you are scheduled to attend a meeting presently. You stop by your partner's office, who is obviously engrossed in some task at her desk. You walk into her office and interrupt her with "Do you know what time it is?" Startled and perturbed, she mumbles "Yes, and I'm not going to the meeting. Now, go away and let me work."

What is going on here? In the first example, a "yes" or "no" response seems grossly inadequate; yet, in the second, the "yes" response to the exact same question is acceptable. The problem of determining how to answer a question such as Do you know what time it is? lies in the *ambiguous* nature of the question. Should this question be interpreted literally? That is, in asking someone Do you know what time it is?, is one really asking whether or not someone knows what the time is as in the second example above? Or, should the question be interpreted in some other fashion, as if one is asking someone else to relate the time, as in the first example above? The answer depends, in part, upon the *context* in which the question was asked.

1.1 Ambiguity in Natural Language

There are three broad areas in the study of natural language,[1] at least from the computational standpoint: syntax (the structure of language), semantics (the meaning, at a micro level, of the symbols or words of language), and pragmatics (the meaning, at a macro level, of language, i.e., how people use language to communicate). Each of these areas suffers from problems of potential ambiguity.

[1]Natural languages such as English and Spanish are distinguished from artificial languages such as those used to program computers, e.g., Lisp and Prolog.

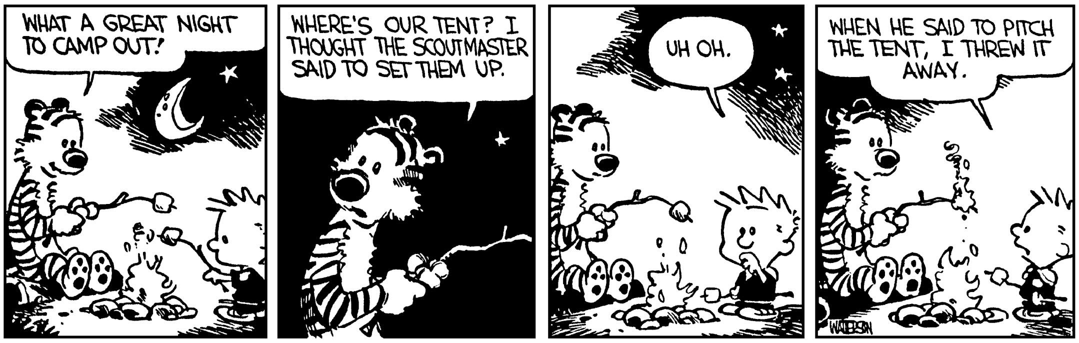

Figure 1.1: Example of lexical ambiguity. CALVIN AND HOBBES ©Watterson. Dist. by UNIVERSAL PRESS SYNDICATE. Reprinted with permission. All rights reserved.

Consider I saw the man with the binoculars. The syntactic ambiguity of this sentence deals with the prepositional phrase with the binoculars. Should this prepositional phrase be attached to the verb saw (as in I saw, by using the binoculars, the man) or to the noun man (as in I saw the man who had some binoculars)? Prepositional phrase attachment is one example of syntactic ambiguity.

The easiest way to illustrate semantic ambiguity is to look at individual words which have multiple meanings. Consider the use the word pitch in Figure 1.1. Pitch is one such word having multiple meanings, two of which are referenced in the comic strip: the intended meaning of "set up" as used by the absent scoutmaster and Calvin's use of the "throw away" meaning. The juxtaposition of the different meanings of pitch forms the underlying comic aspect of this comic strip, allowing it to be funny. This is an example of what is known as *lexical ambiguity*, i.e., the ambiguity associated with lexemes (or words) having multiple meanings.

Metaphor, the use of one thing or concept to represent another in language, is one example of the ambiguity that can be found in the area of pragmatics.

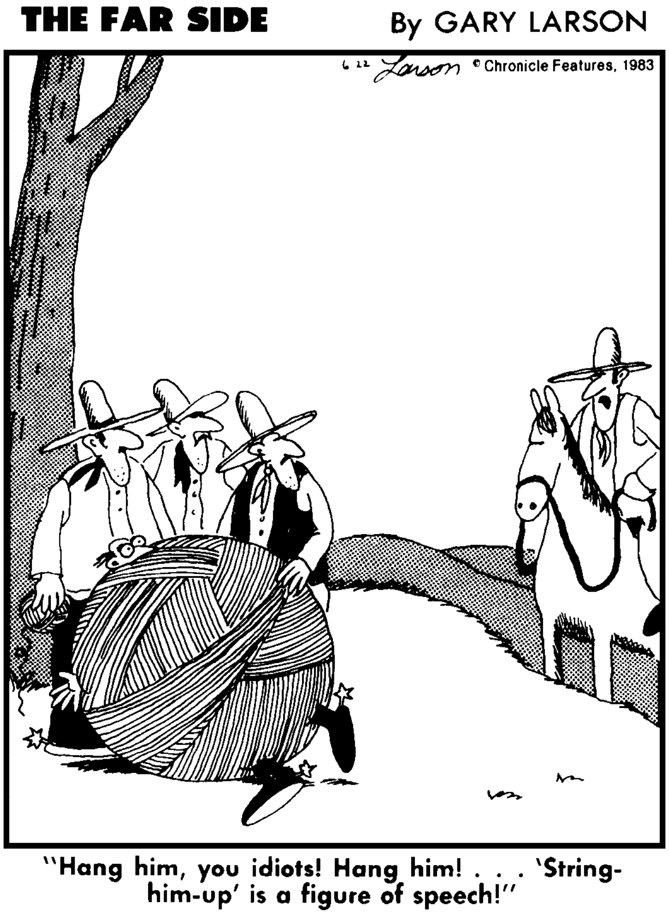

Figure 1.2: Example use of metaphor. 1983 The Far Side cartoon by Gary Larson is reprinted by permission of Chronicle Features, San Francisco, CA. All rights reserved.

Consider the comic strip in Figure 1.2. String-him-up, as explicitly mentioned in the strip, is a figure of speech, used here as a metaphor. It is not intended to be interpreted literally, as the three cowpokes apparently have interpreted it. It is the literal interpretation by the three cowpokes of their leader's implied instruction to string-him-up instead of the intended metaphorical interpretation to hang him that gives the reader the sense of whimsey.

Figure 1.3: Example of an indirect speech act. CALVIN AND HOBBES

Another example of ambiguity in pragmatics is what is known as the indirect speech act (Searle, 1975), where one thing is stated but something else is really intended. Figure 1.3 presents an example of the use of an indirect speech act. The second frame of the comic strip contains the question What are you doing to the coffee table? It is obvious that the mother knows what Calvin is doing to the coffee table. What the mother is *really* asking Calvin is Why are you pounding nails into the coffee table? Being puzzled by the "obvious" (a literal interpretation of the question), Calvin does not bother to answer that question. He replies by asking another question indicating his puzzlement. Calvin has interpreted his mother's question literally and does not understand why she would ask such a "silly" question. What enables the comic aspect of this strip is that Calvin has interpreted the indirect speech act of the mother literally, instead of using the intended indirect meaning. Asking one question by asking another in this fashion is an example of an indirect speech act, another form of ambiguity found in pragmatics.

As can be seen by these examples, natural language is rife with ambiguity. How are we to determine which of the potentially many different interpretations should be given to some bit of text? One answer is to incorporate *context* into the understanding process. In the comic strip examples, context is provided mainly by visual cues. Yet, in narrative, only the text itself is available to create context. Thus, by incorporating the textual context during the understanding process, a system might be able to solve problems of ambiguity found in natural languages. This dissertation addresses this problem.

1.2 PIP

PIP is a computational model for understanding natural language. What makes PIP different from other natural language understanding (NLU) systems is that it has the ability to incorporate a context immediately when processing text. That is, during the actual text processing task, PIP uses the current context to aid development of an interpretation for the current sentence being processed; context is not referenced as an afterthought in some post-processing scheme. This allows PIP to disambiguate possibly ambiguous text in a rather unique fashion. The typical NLU system will process some text to create a *literal interpretation* and, when that interpretation is deemed inappropriate (typically, by applying some set of *rules*), it will submit that initial interpretation to further processing (again, by applying a set of rules).[2] PIP's ability to incorporate context directly alleviates

[2]In the dissertation, this is referred to as the two-stage processing model; see Section 2.2.

it from the necessity of reprocessing an initial literal interpretation that may be incorrect.

The main motivation for PIP's approach to processing language comes from the psychological literature, which suggests that people do not process language by first creating a literal interpretation and then submitting that initial interpretation to further processing. Rather, by incorporating context, people can apparently interpret text ascertaining the intended meaning directly.

The underlying *processing* model used by PIP is the feed-forward artificial neural network (ANN). This processing model was chosen for three main reasons. First, the ANN is a model of learning; i.e., it has the ability to learn different functions, one of which, as will be shown, is the ability to learn to process language. Second, by using a simple recurrent relationship, there is a straightforward way to incorporate a context during language processing. Finally, because PIP uses the ANN architecture as its underlying processing model and by the way in which the context is incorporated into that model, the computational complexity of handling context during text interpretion is uniform; i.e., PIP presents a unified approach to processing natural language which does not need special methods for handling context. Furthermore, PIP is able to apply this architecture to problems of lexical ambiguity, metaphor, and indirect speech acts thereby solving these problems with the same computational methodology as used for literal text.

1.3 Coming Attractions

This dissertation presents a detailed discussion of PIP. The current chapter has introduced various forms of ambiguous constructs found in language, be they syntactic, semantic, or pragmatic and has briefly introduced PIP, a computational model which addresses these problems. The rest of the dissertation is organized as follows.

- **Chapter 2: Psychological Findings on Lexical Access, Metaphor Resolution, and Indirect Speech Act Interpretation** presents some of the underlying psychological motivation for the way PIP processes natural language.
- **Chapter 3: Connectionist Foundations** discusses some of the previous work upon which PIP is built. This work includes the backpropagation algorithm and various recurrent neural network architectures.
- **Chapter 4: A Connectionist Architecture for Text Processing** gives a detailed description of PIP, including a discussion of the artificial neural network architecture and the manner in which that architecture is used to process language.
- **Chapters 5, 6, & 7: Experimental Results** present experimental results of applying PIP to different types of ambiguous language constructs: lexical ambiguity, metaphor, and indirect speech acts, respectively.
- **Chapter 8: Other Work** introduces some other computational models of natural language processing and compares PIP to these systems.

- **Chapter 9: The Present and Future of** PIP discusses both the accomplishments and pitfalls of PIP. Finally, directions for potential future research using PIP are given.

Chapter 2

Psychological Findings on Lexical Access, Metaphor Resolution, and Indirect Speech Act Interpretation

"But what am *I* to do?" said Alice.

"Anything you like," said the Footman,
and began whistling.

—LEWIS CARROLL, *Alice's Adventures in Wonderland*

Psychological studies have shown the utility of context during natural language processing. Much of the debate between researchers studying such phenomena centers around how and when context is used for disambiguation. Some of the studies relating to issues of lexical ambiguity resolution (or lexical access), metaphor resolution, and indirect speech act interpretation are discussed here. This is done with the hope that the data gathered, the results reported, and the ensuing claims of these reports might provide constraints useful in creating computational models to deal with these ambiguous lexical constructs. As such, the studies provide much of the motivation for the creation of PIP both in terms of what it does and how it does what it does.

2.1 Lexical Ambiguity

As stated in Chapter 1, lexical ambiguity deals with words (or lexemes) having multiple meanings. For example, WordNet (Miller, 1990), an on-line lexical database, gives 25 distinct senses to the word line, three of which can be described as: "a line of text, as from a poem"; "a queue (or line) of people"; and "a product line of some company" (Voorhees, Leacock, & Towell, 1995). Being able to choose which meaning is appropriate for a given situation and how that meaning is chosen has been the subject of a number of psychological studies.

There are two basic schools of thought on how a particular meaning for a given word is chosen. One school of thought states that context restricts the choice of meanings to the correct one and that no other meanings are considered (the *Prior Decision Hypothesis*; Foss & Jenkins, 1973). The other states that all possible meanings are temporarily considered (if only for a very short period) and that context is used to filter out those which are not appropriate (the *Post Decision Hypothesis*; Swinney & Hakes, 1976).

Early studies on lexical access (i.e., resolving lexical ambiguity) seemed unable to resolve the issue. The Post Decision Hypothesis has gained more favor with later results (Adriaens & Small, 1988). However, both schools predict the use of context for lexical ambiguity resolution – one applies context before access and the other after access. Thus, context is important for accurate lexical access. Two studies, one supporting the Prior Decision Hypothesis and the other supporting the Post Decision Hypothesis, are briefly discussed here. They provide some bit of history concerning the evolution of the acceptance and rejection of these competing hypotheses.

Swinney & Hakes (1976) presented results of an experiment which supports the Prior Decision Hypothesis. The experiment was an attempt to measure the processing load on the experimental subjects while they were comprehending sentences which included an ambiguous word or an unambiguous control word. Prior to presentation of the ambiguous word, the text would include either an immediate context identifying the appropriate meaning for the ambiguous word, a distant context, or no specific context appropriate for disambiguation. For example, Swinney & Hakes give these example texts used in the experiment.[1]

None: Rumor had it that, for years, the government building had been plagued with problems. The man was not surprised when he found several *bugs* in the *c*orner of his room.

Immediate: Rumor had it that, for years, the government building had been plagued with problems. The man was not surprised when he found several spiders, roaches, and other *bugs* in the *c*orner of his room.

Distant: Rumor had it that, for years, the government building had been plagues with spiders, roaches, and other problems. The man was not surprised when he found several *bugs* in the *c*orner of his room.

All stimuli for the study were presented aurally. The task assigned to the subjects was that they should understand the text because they would later be quizzed

[1]The italicized word *bugs* is the ambiguous word of record. For these examples, *insects* was used as the control unambiguous word. The target phoneme is the *c* of corner.

on the content. Furthermore, they were to identify a predetermined phoneme which was to occur somewhere in the text. This phoneme actually occurred after the presentation of the ambiguous or control word. If the subjects of the study took less time to recognize the phoneme when the context aided the disambiguation of the ambiguous word, Swinney & Hakes hypothesized, support for the Prior Decision Hypothesis would be given. The argument was that, given a smaller reaction time, the subjects would be engaged in less processing of the ambiguous word. The reason given for this was that the subjects would have used the context to restrict which of the meanings of the ambiguous word were being considered. Results of the reaction time data supported this hypothesis thus giving weight to the Prior Decision Hypothesis of lexical access.

Swinney (1979; Onifer & Swinney, 1981) changed direction and presented experimental evidence supporting the Post Decision Hypothesis model of lexical access. He dismissed his earlier work by arguing that the methodology of that experiment was measuring "some process that occurs *following* access" and not actual lexical access. The second experiment used a cross-modal priming task in an attempt to measure actual lexical access. The experiment can be described as follows. Subjects were presented with some text aurally, in a fashion similar to that of Swinney & Hakes (1976) as described above. In addition, at some (strategically appointed) time during this aural presentation of the text, a visual presentation of a letter string (a target word) would appear. The subjects were asked to decide whether or not the letter string was a word. Reaction time for this decision was measured. The timing of the presentation of the letter string coincided with the aural presentation of the lexically ambiguous word found in the text. Furthermore, the target word was classified either as contextually related (e.g., for the example texts given above, ant), contextually inappropriate (spy), or unrelated (sew). If the reaction times were insignificantly different for the contextually related and contextually inappropriate targets, Swinney hypothesized, support would be given to the Post Decision Hypothesis model. Furthermore, reaction time for the unrelated target word should be significantly different. The argument was that if multiple meanings indeed were accessed, there should be no significant difference in reaction times for recognizing the different possible meanings of the ambiguous word thus suggesting that the subjects did actually access the different meanings. The data supported this argument.

2.2 Indirect Speech Acts

As with the two competing theories for lexical access, there are two general competing theories for indirect speech act interpretation as well. They can be classified in a smiliar manner; the Prior Decision Hypothesis equivalent for the domain of indirect speech acts (i.e., the *one-stage processing model*) would state that context is used during the interpretation process to yield the indirect interpretation directly (and possibly the literal interpretation as well, in parallel). The Post Decision Hypothesis equivalent (the *two-stage processing model*) can be formulated by stating that a literal interpretation is derived first, followed by application of the context to yield the indirect interpretation.

Several psychological studies have been conducted on the interpretation of indirect speech acts. For some of these studies, conclusions drawn by the differing researchers support one or the other interpretation hypothesis.[2] This section will present two such studies, one supporting the two-stage processing model, presented first, and the other supporting the one-stage processing model.

Clark & Lucy (1975) test the hypothesis presented by Gordon and Lakoff (1971) that the interpretation of *indirect requests* conveyed via (direct) questions[3] requires three pieces of knowledge: (1) the literal meaning of the sentence, (2) the perceived context, and (3) a so-called conversational postulate (Grice, 1975). A hearer arrives at the intended meaning of a request by first deriving the literal meaning of the request. The literal meaning is then compared to the context to determine whether or not it seems plausible. If it is deemed plausible, that is the preferred interpretation; if not, the literal meaning is combined with an "appropriate" conversational postulate and by some deductive process, the intended meaning is acquired. Clark and Lucy (1975) present the following example.

> Consider Would you mind opening the door? spoken by S (the speaker) to H (the hearer). Its literal meaning, the first ingredient of the recipe, might be stated as follows: "S is asking H whether or not H would object to opening the door." A likely context, the second required ingredient, might be stated as follows: "S nevertheless believes that H would not object to opening the door." Since the perceived context is directly contradictory to the literal meaning (why should S inquire about something he already believes to be true?), the sentence must be taken in some other sense. Such a reinterpretation in turn requires the literal meaning to be considered in conjunction with an appropriate conversational postulate, namely, "If S questions H's willingness to

---

[2]Other studies have addressed issues relating to child interpretation of indirect speech acts (Ackerman, 1978; Garvey, 1975; Shatz, 1978). The consensus on the child studies can be summed up by Shatz's (1978) hypothesis "that the child has some simple action-oriented discourse rule ...; e.g., MOTHER SAYS; CHILD DOES," a rule which is very similar to those proposed by Selfridge (1986).

[3]E.g., Do you know what time it is? is a direct question but the indirect interpretation is usually Please tell me what the current time is. This is what is known as an indirect request.

> do something when in fact H's willingness is not in doubt, then S is requesting H to do that something." By deduction this postulate leads to the correct conveyed interpretation: "S is requesting H to open the door."

To test the two-stage model, Clark and Lucy performed an experiment measuring the reaction time of people presented with a comprehension task involving indirect requests. Three predictions were made concerning the subject's speed of comprehension:

> 1: The listener should show evidence that he had come to the literal interpretation of a sentence before he had come to its conveyed [indirect] interpretation.
> 2: The listener should take longer whenever the intended meaning is different from the literal meaning, all other things being equal.
> 3: The listener should show evidence that his final representation of a sentence is its intended meaning.

Subjects were presented with sentences such as: Why not color the circle blue? and Why color the circle blue? Notice that the first sentence has a "negative" polarity for the literal interpretation and a "positive" polarity for the indirect interpretation; i.e., the indirect interpretation actually asks that the circle be colored blue (or to have a blue circle) whereas the literal interpretation implies that the circle should not be blue. The opposite is true for the second sentence. Each sentence was paired with a colored circle. The subjects were asked whether or not the color of the circle matched the request.

Analysis of results indicated evidence supporting predictions one and three. Regarding prediction three, for example: "...they [the subjects] did not hesitate in taking all of the sentences in their conveyed sense. They found the task sensible and made very few errors...." There is no difficulty accepting that the data supported this prediction. Support for prediction one came from pairs of sentences of the form I'll be very happy if you make the circle blue and I'll be very sad unless you make the circle blue. Reaction times were consistently longer for unless than for if. "Apparently, the difference [in reaction times] can only be attributed to a difference in their literal interpretations" implying that the literal interpretations were indeed constructed. An alternative explanation for the longer reaction times would be that it may take longer to process the "negative" connotations associated with unless; and in this case only, because of this "negative" effect, did the subjects require more processing time (cf. McMahon, 1963). The evidence does not support the implied claim that people constructed a literal interpretation when if was present.

The second prediction was not tested directly with this experiment. The methodology of the experiment did not allow for both literal and indirect interpretations of the same sentence, "all other things being equal." Gibbs (1983) did perform such an experiment. *Do people always process the literal meanings of indirect requests?* Gibbs poses this question and hypothesizes that the answer is *no*, offering the results of his experiment as supporting evidence.

To test the hypothesis, Gibbs presented subjects with short (one paragraph) stories. The last line of the story contained a priming sentence such as Can't you be friendly? as seen in the following two stories from Gibbs (1983).

Indirect story: Mrs. Norman was watching her kids play in the backyard. One of the neighbor's children had come over to play. But Mrs. Norman's kids refused to share their toys. This upset Mrs. Norman. She angrily walked outside and said to one of her children, "Can't you be friendly?"

Literal story: Rod was talking with his psychiatrist. He was having lots of problems in establishing relationships. "Everyone I meet I seem to alienate," Rod said. "I just turn very hostile for no reason," he continued. The shrink said, "Can't you be friendly?"

Depending on the context of the story, the priming sentence could be interpreted either literally or indirectly. Subjects were then presented with one of three potential target sentences: the literal interpretation, the indirect interpretation, or a meaningful but unrelated sentence. Continuing the example above, these were:

Indirect: Please be friendly to other people.
Literal: Are you unable to act friendly?
Unrelated: Running is excellent for the heart.

They were asked whether or not this target sentence "was a meaningful English sentence." Subjects' reaction time for their determination was measured.

Results indicated that subjects took less time to determine that the target sentences indicating an indirect interpretation were recognized as being meaningful for the two cases when the story indicated that the priming sentence should be interpreted literally and when the story indicated that the priming sentence should be interpreted indirectly. That is, when the story indicated that the priming sentence should be interpreted literally, the subjects took less time to indicate that the target sentence with the indirect interpretation was meaningful than to indicate that the target sentence with the corresponding literal interpretation was meaningful. Also, when the story indicated that the priming sentence should be interpreted indirectly, the subjects took less time to indicate that target sentence with the corresponding indirect interpretation was meaningful than to indicate that the target sentence with the literal interpretation was meaningful. The conclusion drawn is that people tend to be biased toward an indirect interpretation of the (conventional) indirect request, even when primed by an apparently literal interpretation. (It should be noted that Gibbs deals with conventional indirect requests. Apparently, no one has tried to run a similar experiment with unconventional indirect requests.)

As with the study of Clark and Lucy, the published example of Gibbs suffers from the possibility that the negativeness of unless as seen in the literal interpretation target sentence might be the cause of subjects' greater response time in determining the meaningfulness of the target sentence. Clearly, more careful studies need to be done on language processing itself before conclusions can be drawn from these types of experiments. One such study might be to use the texts of Gibbs and switch the priming and target sentences. The "new" target sentences would then be uniformly the same and any results would not be subject to the criticism above.

2.3 Metaphor

The same one-stage vs. two-stage processing model dichotomy of indirect speech act interpretation also applies to metaphor resolution. When interpreting metaphors, do people exhibit behavior indicating that they perform more processing for metaphors than for literal text (given that the context will determine which interpretation to give to the text)? This question has been addressed by numerous studies presented in the psycholinguistic literature (Gildea & Glucksberg, 1983; Keysar, 1989; Shinjo & Myers, 1987).

The key to the discussion of metaphor resolution from the psycholinguistic perspective centers around the available context. When context is "adequate" (i.e., there is sufficient context available to aid in metaphor resolution), the studies indicate that metaphor resolution requires no more processing time than does literal text, in the same context. When context is less than adequate, processing time for the metaphor increases. This was shown in the study discussed here.

Ortony, Schallert, Reynolds, & Antos (1978) hypothesized that

> whether or not a target sentence [metaphor] requires a relatively large amount of processing time is a function of how easily it can be interpreted in the light of contextually determined expectations rather than a function of its nonliteralness.

To test this hypothesis, Ortony et al. developed a set of textual contexts which were devised to give a target sentence either a literal or a metaphorical interpretation. Examples of the full contexts given by Ortony et al. follows.

Literal inducing context: *Approaching the enemy infantry*, the men were worried about touching off landmines. They were very anxious that their presence would be detected prematurely. These fears were compounded by the knowledge that they might be isolated from their reinforcements. The outlook was grim.

Metaphorical inducing context: *The children continued to annoy their babysitter.* She told the little boys she would not tolerate any more bad behavior. Climbing all over the furniture was not allowed. She threatened to spank them if they continued to stomp, run, and scream around the room. The children knew that her spankings hurt.

The target sentence for these contexts was:

Target: Regardless of the danger, the troops marched on.

Shorter contexts were constructed by using only the first sentence of each full context, or a part thereof, as indicated by the italicized portion of the examples above. One thing to note here is that the target sentence is the same for all contexts; this is different from the case for the indirect speech act experiments described in Section 2.2.

Subjects were instructed to read each sentence as presented on a computer display indicating comprehension by pressing the space bar of the computer keyboard. The reaction time for comprehension of the target sentence was measured. Results indicated support for the hypothesis; namely, subjects took more time to comprehend both the literal and metaphorical interpretations of the target sentence when context was short than when the context was full. Furthermore, there was an insignificant difference in comprehension time between the metaphorical and literal interpretations when context was full and the opposite when context was short. The conclusion drawn was that, when context is adequate, metaphor processing is on par with literal processing due to the expectations set up by the context. When context is inadequate, metaphor processing suffers because of the lack of contextual expectations. These results were successfully replicated by Inhoff, Lima, & Carroll (1984).

2.4 Wrap Up

This chapter has presented some of the psychological studies motivating the underlying processing model of PIP. These all tend to show that context of some form is necessary when attempting to disambiguate natural language. As Rumelhart (1979) states, "Linguistic utterances are always interpreted in some context. The context of utterance, along with any knowledge available to the listener, may potentially be employed in the process of constructing an interpretation of the utterance." PIP was designed to have the direct application of a textual context during the processing of natural language. PIP also rejects the two-stage processing model in favor of a unified approach to text processing; the one-stage processing model used by PIP is applied to all types of text, be they literal or figurative. The next two chapters introduce the reader to backpropagation followed by the details of PIP's processing model.

Chapter 3

Connectionist Foundations

This was such a new idea to Alice ...

—LEWIS CARROLL, *Alice's Adventures in Wonderland*

The previous two chapters have introduced the general problems associated with ambiguity and pragmatics and have discussed various psychological studies which address these problems. This chapter introduces the general *feed-forward artificial neural network* architecture, the *back-propagation* learning algorithm, and the *recurrent network* architecture.

3.1 Error Back-propagation

The most basic aspect of PIP's architecture is the error back-propagation algorithm of Rumelhart, Hinton, and Williams (1986), or *backprop* as it is more popularly known. The motivation for the development of the backprop algorithm has its roots in the inability of the perceptron model (an early artificial neural network model based on linear computations, cf. Minsky and Papert, 1969) to compute certain functions requiring an internal representation. The classic example here is exclusive-or (XOR).

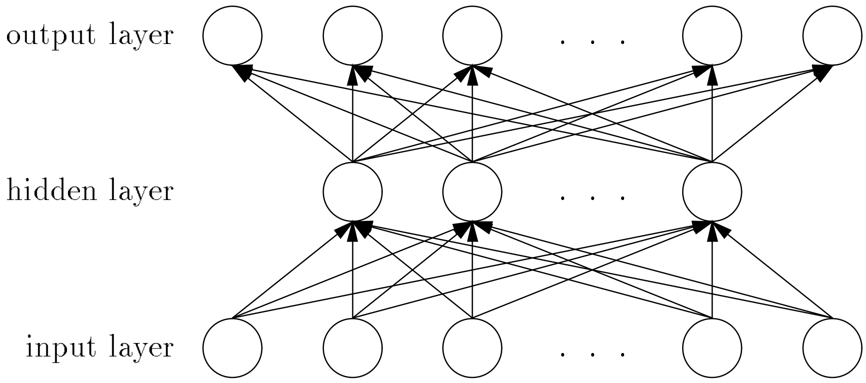

Figure 3.1: Fully connected generic ANN.

The specific problem of the perceptron model with respect to the necessity of an internal representation (commonly known as a *hidden layer*) is its inability to train networks with hidden layers using the *perceptron convergence rule* (also referred to as the *delta rule*). Rumelhart et al. (1986) present the *generalized delta rule* which allows a network to be trained and thus to learn the function mapping its inputs to its outputs. Before discussing backprop, however, a short introduction to processing in a feed-forward artificial neural network is presented.

Consider the generic fully connected network with one hidden layer as seen in Figure 3.1. Input to the network is a vector of real numbers, typically in the closed range $[0, 1]$ and often further constrained to be either zero or one. Output values depend on the chosen *activation function*; the standard logistic activation function (see below), which limits the values of its output units to the open range $(0, 1)$, is used by the network described in this dissertation. Each arrow connecting units has associated with it a *weight*. The input value of each unit is computed by taking the weighted sum of all of its inputs; e.g., to compute the input for one of the hidden units, the algorithm multiplies output values for each input unit by its corresponding weight and sums these products. Furthermore, to compute the actual output value of a unit, its weighted sum is subjected to an activation function which has the effect of normalizing the output value. The input and output values for the input layer degenerate to the values of the input vector.

Backprop is a "supervised" learning algorithm; i.e., for each input vector, it requires a corresponding target output vector. Training occurs by propagating the error (difference between the actual output and the target output) backwards through the network. The effect of this propagation is that the weights connecting the units are modified. This can be described by the following equations.[1] First, the generalized delta rule specifies how the weights connecting the units i to the units j should be modified; the change Δ_p for weight w_{ji} given input/output pair p is proportional to the product of the error signal δ_{pj} and the actual output o_{pi}:

$$\Delta_p w_{ji} = \eta \delta_{pj} o_{pi}$$

where η is the learning rate. The error signal δ_{pj} for an output unit is computed by:

$$\delta_{pj} = (t_{pj} - o_{pj}) f_j'(net_{pj})$$

where t_{pj} is the target output and $f_j'(net_{pj})$ is the first derivative of the activation function applied to the unit's net input (i.e., the weighted sum). For non-output units u_j, the error signal is computed recursively by:

$$\delta_{pj} = f_j'(net_{pj}) \sum_k \delta_{pk} w_{kj}$$

where δ_{pk} and w_{kj} are the error signals and weights of the units (u_k) to which u_j is directly connected. The activation function used by PIP and most backprop simulators is the logistic activation function:

$$o_{pj} = \frac{1}{1 + e^{-net_{pj}}}$$

[1]The equations are from Rumelhart et al. (1986).

where

$$net_{pj} = \sum_i w_{ji} o'_{pi} + \theta_j$$

and o'_{pi} is the output on the units from the previous layer and θ_j acts as a bias and is similar in function to a threshhold in linear neural networks. This activation function has, as its first derivative:

$$\frac{\partial o_{pj}}{\partial net_{pj}} = o_{pj}(1 - o_{pj}).$$

Finally, a *momentum* term (α) is added which has the effect of increasing the rate of the direction of learning (not to be confused with η) and also prevents the weights from oscillating between values. This yields the following generalized delta rule for computing the change in the weights for learning cycle $(n + 1)$.

$$\Delta w_{ji}(n + 1) = \eta(\delta_{pj} o_{pi}) + \alpha \Delta w_{ji}(n)$$

3.2 Recurrence in Artificial Neural Networks

Natural language processing is inherently a sequential task; i.e., it is natural to process the words of a text sequentially, one after the other. Early artificial neural network models of NLP mapped the sequential aspect of language processing onto a spatial representation in the input layer of units (McClelland & Kawamoto, 1986; Voorhees et al., 1995). That is, all of the words of a sentence, for example, were represented as one input vector and processed by the network en masse. The obvious drawback to this approach is that the size of the sentence is limited by the size of the input vector. It would be much better to have the sequential aspect of the task represented by the processing in the network itself. This can be accomplished by use of a recurrent network.

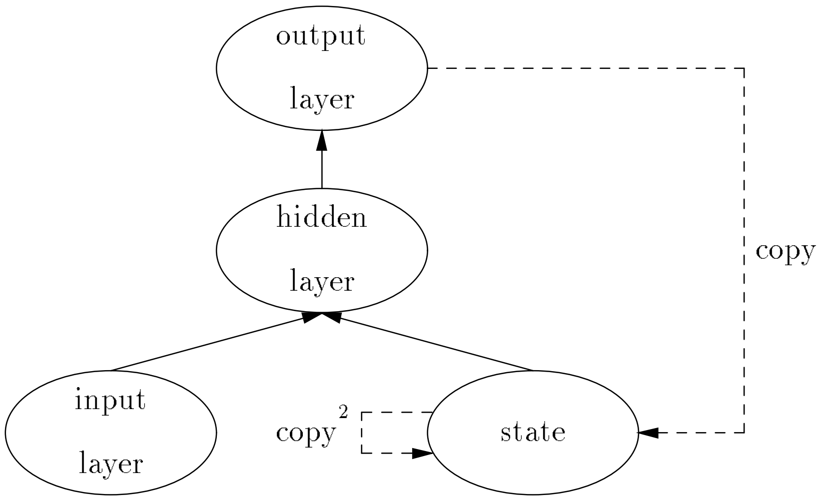

Figure 3.2: Jordan net – basic topology.

Jordan (1986) introduced a network architecture that is capable of producing a sequence of outputs when given a fixed set of inputs. (Jordan's application is described as having a fixed *plan* as input while the output is a sequence of *actions*; the domain is that of speech production.) This is accomplished by use of a recurrent relation between the output units and a layer of *state units* (Figure 3.2). This gives the input the benefit of the previous output during processing. The state units also have a recurrent relation giving the network the previous state as additional input. Thus the output is a function of the (in this case, fixed) input units, the previous output units, and the previous state.

Elman (1990) takes Jordan's idea and simplifies it in two ways yielding the simple recurrent network (SRN): he removes the previous state and uses the activation from the hidden units as the basic recurrent relation (Figure 3.3). The *context* layer of units (equivalent to Jordan's *state* units) is completely "internal" to the SRN because it references only the *hidden* units; it does not associate with

[2] Typically, the weights on these connections will be 0.5 indicating a decay of the previous state.

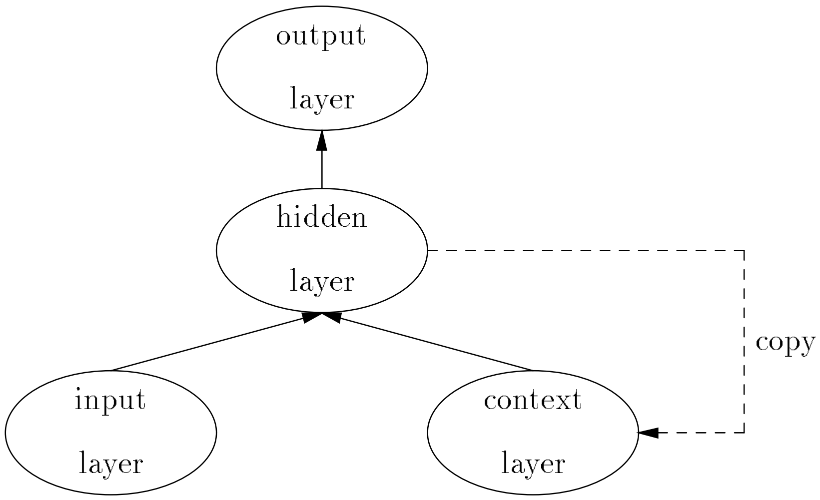

Figure 3.3: Simple Recurrent Network topology.

the "outside world" at all, the outside world being the input and output units. With the SRN, Elman was able to demonstrate its usefulness on a set of tasks involving sequential input.

PIP incorporates both of these ideas into its architecture, to a more or less extent. By using these ideas, PIP is able to process text sequentially, one word at a time. Section 3.3 presents the story gestalt representation (St. John, 1992), which uses the SRN with the ("hidden") gestalt layer supplying the recurrent context or state. PIP applies Jordan's network (without the recurrent state units) to its intra-sentence processor (Section 4.1). Schulenburg (1992) presents evidence that using the activations from the output layer as the recurrent information (as did Jordan, although without the recurrent state units) trains the network faster and more accurately than using the activations from the hidden layer.

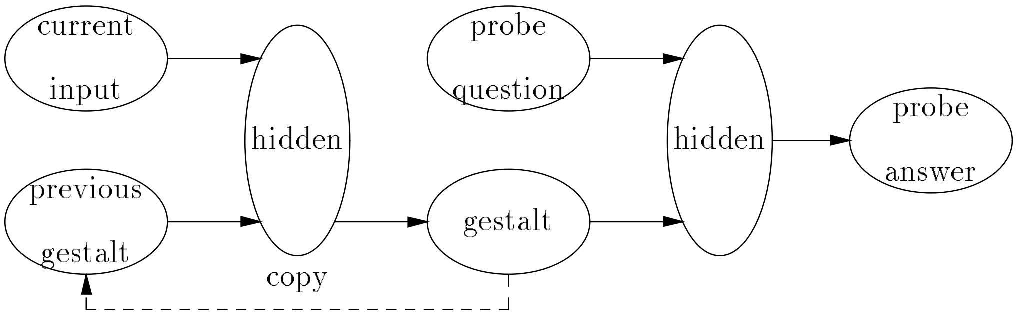

Figure 3.4: Sentence & story gestalt networks

3.3 The Sentence & Story Gestalts

St. John (1992; St. John & McClelland, 1988; St. John & McClelland, 1990) presents two general models of text comprehension: the SENTENCE GESTALT and the STORY GESTALT. The basic tasks are to construct a meaning representation of a sentence or text and to use that representation to answer questions about the sentence or text. Furthermore, the meaning representations are learned. That is, rather than giving an explicit representation, the GESTALT models learn the meaning representation of sentences or texts and use these representations to answer questions about those sentences or texts.

Figure 3.4 illustrates the general network topology used by the SENTENCE GESTALT and the STORY GESTALT. Training of both networks is similar. Presented first is a discussion of how the STORY GESTALT is trained. Following that, differences between the two are briefly outlined.

3.3.1 Training the STORY GESTALT

For the following discussion, consider this story from St. John (1992):

(a) Andrew and Roxanne decided to go to the park. The weather was sunny. Andrew ran through the park. He threw the frisbee to Roxanne.

The basic ideas are that the *gestalt* layer (cf. Figure 3.4) contains a representation of the text and that it acts as a trainable "hidden" layer. (Since the *gestalt* layer is a hidden layer, it is "internal" to the network; thus, the "copy" recurrent relation from the *gestalt* layer to the *previous gestalt* layer acts as the SRN aspect of the gestalt model.) Furthermore, it is trained by a question and answer process. This is accomplished as follows. For the STORY GESTALT, the *current input* layer contains the representation for the sentence currently under consideration.[3] The first sentence in story (a) might be represented as

```
[(pred decided-to-go) (actors andrew roxanne) (to park)]
```

and the appropriate units in the *current input* layer would be activated.[4] At the beginning of processing, the *previous gestalt* layer is empty; i.e., all of the units have zero activation. Standard feed-forward processing occurs in order to obtain output activations in the *gestalt* layer. This, then, is the current representation of the story.

To train the gestalt representation, "questions" are "asked" of the system and "answers" are provided. Questions are asked by activating, in the *probe question*

[3]A role/filler representation is used.

[4]St. John argues that a preprocessor such as the SENTENCE GESTALT could be used to produce this representation leading to a network topology very similar to that of Miikkulainen and Dyer (1991). However, the STORY GESTALT itself does not do this.

layer, the unit corresponding to the *predicate* role associated with the question. For the current example, the `decided-to-go` unit would be activated as for asking a question such as "Who decided to go to the park?"[5] The "question" (i.e., the activations in the *probe question* layer) and the current story representation (as found in the *gestalt* layer) are fed forward through the network producing an output activation in the *probe answer* layer. This output activation is the system's answer to the question. It is compared to the target output (the target output forms the "true answer" to the question and is the representation for the entire proposition representing the answer to the question) forming the *error*, which becomes part of the error signal δ (see section 3.1). Continuing the example, the target output would be those units corresponding to the representation for

```
[(pred decided-to-go) (actors andrew roxanne) (to park)].
```

The error backpropagation training algorithm is then applied across all layers and connections from the *probe answer* layer back to the *current input* and *previous gestalt* layers. The rest of the sentences of the story are processed in a similar fashion with the exception that the output activations from the *gestalt* layer are copied into their corresponding units in the *previous gestalt* layer as in the simple recurrent network.

The questions are asked in an incremental fashion. That is, only questions pertaining to those sentences already processed are used to train the network.

The usefulness of this training schema is that, typically, in computational models of natural language understanding, systems illustrate their understanding

[5]The astute reader will note an inherent drawback with this approach. Only one question is allowed to be used per sentence for training purposes. Furthermore, no two sentences in the story can have the same predicate.

of some text by the ability to answer questions about that text. With the STORY GESTALT, once the network is trained, sentences can be processed to produce the gestalt representation of the story. Then, the system can demonstrate its understanding of the story by using the gestalt representation to answer questions supplied in the *probe* layer.

3.3.2 The SENTENCE GESTALT

The *gestalt* layer for the SENTENCE GESTALT (St. John & McClelland, 1988; St. John & McClelland, 1990) is created in a similar fashion. The differences are: 1) Instead of using representations for an entire sentence as input to the *current input* layer, the words of the sentences are input sequentially;[6] and 2) The probes (i.e., questions) used are the roles and fillers and the answers are the appropriate role/filler pairs. Unlike for the STORY GESTALT, the questions asked here are "batch"; i.e., the set of questions asked as each word or phrase is input considers all possible role/filler pairs for the entire sentence, even those applicable to words or phrases not yet parsed. Consider again the first sentence from text (a). Table 3.1 shows the entire set of probes and their associated answers for the example sentence. As indicated, if the probe were `to`, the answer would be `(to park)`; if the probe were `park`, the answer would be `to`. Thus, the entire set of probes and answers given in Table 3.1 are presented to the SENTENCE GESTALT for each word or phrase of the first input sentence.

[6]The SENTENCE GESTALT actually presumes that the sentences have been preprocessed into appropriate phrases. For example, the prepositional phrase `to the park` would be encoded as a single input unit for the sentence.

Table 3.1: Probes and answers for Andrew and Roxanne decided to go to the park.

probe	answer
`pred`	`(pred decided-to-go)`
`actors`	`(actors Andrew Roxanne)`
`to`	`(to park)`
`decided-to-go`	`(pred decided-to-go)`
`Andrew Roxanne`	`(actors Andrew Roxanne)`
`park`	`(to park)`

3.4 Transition

This chapter has briefly discussed various artificial neural network topologies that may be used for natural language understanding. The next chapter presents the details of PIP, a feed-forward artificial neural network model for natural language understanding. Backprop, the recurrent network, and the gestalt model are the building blocks upon which PIP is built.

Chapter 4

A Connectionist Architecture For Text Processing

> The Hatter's remark seemed to her to have no sort of meaning in it,
> and yet it was certainly English.
> "I don't quite understand you," she said,
> as politely as she could.
>
> —LEWIS CARROLL, *Alice's Adventures in Wonderland*

This chapter presents PIP, a new computational model which utilizes a simple recurrent feed-forward artificial neural network. The neural net architecture used by PIP builds upon previous research on backprop, recurrent networks, and the gestalt model as presented in Chapter 3.

4.1 PIP's Architecture

The main idea behind PIP's architecture is the realization of the need to incorporate into the language understanding process the context obtained from previously processed text. There are two major aspects of PIP which distinguish it from previous connectionist models of natural language understanding. PIP uses "incremental feedback" and a "global context" while processing the words of sentences.

PIP has two processing phases: one handles the individual words of a sentence and is referred to as the *intra-sentence* processing level; the other deals with the representations of sentences and is called the *inter-sentence* processing level (Figure 4.1). This section discusses the intra-sentence processing level first followed by a discussion of the inter-sentence processor and relates them to the notion of incremental feedback and global context.

4.1.1 Sentence processing (with incremental feedback): The intra-sentence processing level

Section 3.2 introduced recurrent networks. PIP applies the modified Jordan network (i.e., without the recurrent state units) to accomplish the sequential processing of the words in a sentence, as can be seen by the outlined portion of Figure 4.1 labeled as "intra-sentence processing."

Previous feed-forward connectionist models which process sentences have done so in a "batch" mode. That is, using the error back-propagation algorithm, they train the network by supplying, as their target output, the representation for the entire sentence (McClelland & Kawamoto, 1986; Miikkulainen & Dyer, 1991; and, although the case is slightly different, St. John, 1992[1]). This is done even

[1]The STORY GESTALT uses incremental feedback for processing at the inter-sentence level; but, since it does not do intra-sentence processing, the feedback issue does not apply.

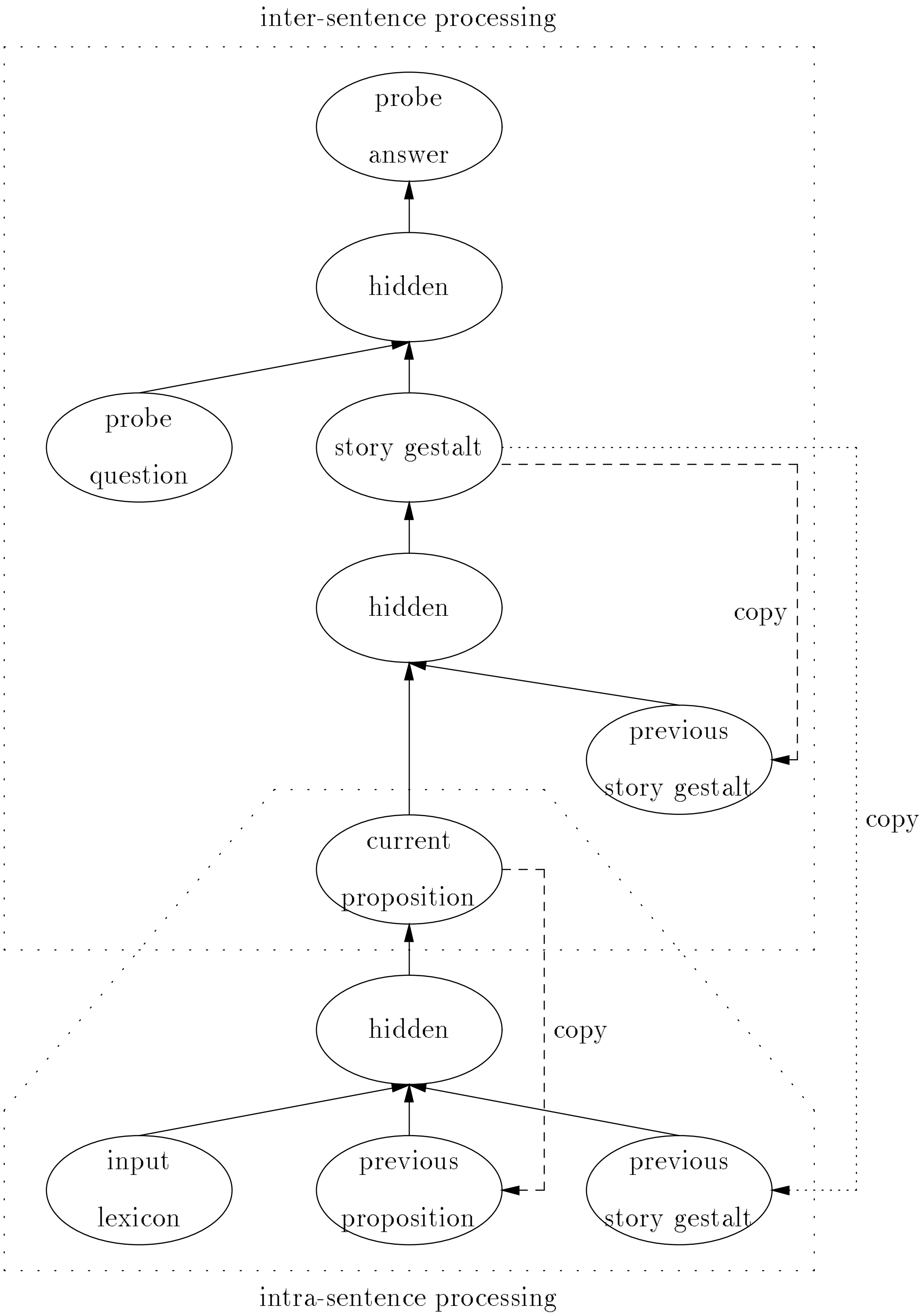

Figure 4.1: PIP's network topology.

though the entire sentence has not yet been processed. There are two main reasons why this batch approach is less than appropriate. First, it seems counter-intuitive to the way that people process sentences. People do not necessarily attempt to determine the interpretation of a sentence after seeing only the first word, for example. What, after all, would be the interpretation of this sentence after seeing only the first word What? It is indeterminate. Secondly, were the sentence processing task a prediction task, "batch feedback" would seem to be a viable approach. While prediction is clearly important during natural language understanding because parsing "expectations" encountered early on can constrain later word processing (Granger, 1977; Small & Rieger, 1982), these constraints are typically general in form. For example, a general constraint might be to expect a "physical object" to appear as the filler of an object slot; a connectionist model with batch feedback, on the other hand, might "predict" that a specific object, such as "John's bike," would fill that slot. PIP avoids these problems by building the sentence meaning representation incrementally in the output layer, as the words of the sentence are sequentially processed. This implies that the network is trained only on that part of the sentence meaning representing that portion (i.e., those words) of the sentence processed up to that time.

Consider the following example: The girl ate the spaghetti with the fork.[2] This sentence could be represented by the following role/filler assignments:

```
[(pred eat) (agent girl) (object spaghetti) (instrument fork)]
```

where there would be some indication that the act is past tense. In the typical connectionist NLP system, the network would be trained to predict the full meaning representation of the entire sentence after the first word The or phrase

[2]From McClelland and Kawamoto (1986).

The girl is seen even though much of that information has not yet been processed. That is, upon being presented with The or The girl, the target output (for the back-propagation algorithm) would be the entire sentence meaning representation instead of a non-committed "nothing" for The or possibly just `(agent girl)` for the phrase The girl.

In contrast, PIP builds a role/filler meaning representation of the sentence in the output layer incrementally, adding to the meaning representation the information from the sentence as the words are processed. Input to PIP is a sentence in English with each input unit representing an English word. That is, PIP uses a *localist* representation in the input layer (and also in the output layer) where each unit represents a complete concept. For the input layer, the concepts are words (cf. Figure 4.2); for the output layer, the concepts are role/filler pairs. The sentence is processed one word at a time, left to right; thus, to process the entire sentence requires n passes through the network where n is the number of words (and punctuation marks – each punctuation mark has its own input unit) in the sentence. While there is no principled reason for using a localist representation in the output layer, there is for the input layer. Having each word represented as a unit in the input layer allows PIP to provide a word-level context when processing sentences at the intra-sentence level. (This context is different from the global context discussed below.) This word-level context manifests itself in a "sliding window" across the words of a sentence. The word currently being processed (the *focus*) is at the center of the window and receives a full activation. The words on either side of the focus form the rest of the window and each receives a partial activation. Full activation is represented by the value 1.0 on the input units; partial activation is the value 0.5. The focus changes as the window slides across the words of the sentence.

Training of a new sentence in PIP's network is illustrated in Figures 4.2 and 4.3 for the sentence Can you reach the book? In PIP, this sentence would be represented by the following role/filler assignments (in Lisp notation):[3]

```
[(pred question) (mode ability)
                 (focus [(pred moveb) (actor you)
                                      (object hand)
                                      (to book)])]
```

In Figure 4.2, the current focus word is you; hence, it receives a full activation on the input layer. The surrounding words, can and reach, each receive partial activation. The recurrent portion of the input layer receives the activation of the output units from the previous cycle. The target output activation indicates how the sentence meaning representation is built incrementally; only the information from that part of the sentence which has been processed so far (i.e., Can you) is incorporated into the meaning. Thus, the first `(pred question)`, second `(mode ability)`, and fifth `(focus->actor you)` output units are activated. When reach becomes the focus (Figure 4.3), PIP incorporates into the sentence meaning the associated information by adding full activation for the fourth `(focus->pred moveb)` and ninth `(focus->object hand)` output units. Finally, when book becomes the focus, PIP would activate the sixth `(focus->to book)` output unit. The sentence ending

[3]Each unit in the output layer represents a role/filler pair in the meaning representation, which can be mapped into a Lisp list notation. The notation used here, e.g., `(pred question)` indicates that `pred` is the role and `question` is the filler. The notation `focus->actor` indicates the `actor` role of the filler for the `focus` role. An easy way to think of this is that `focus->actor` represents a path from the `focus` role to the `actor` role.

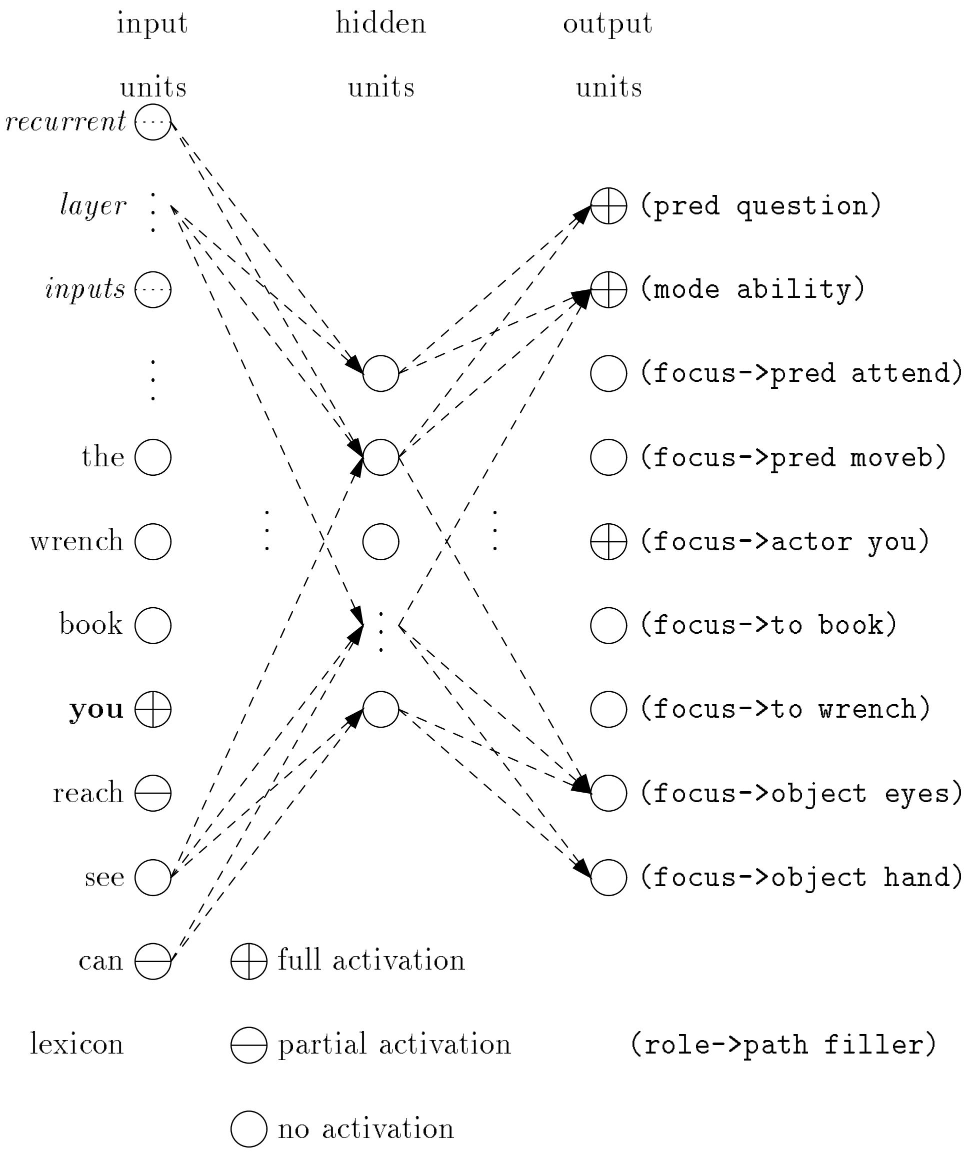

Figure 4.2: Example of intra-sentence processing. The current word of *focus* is you.

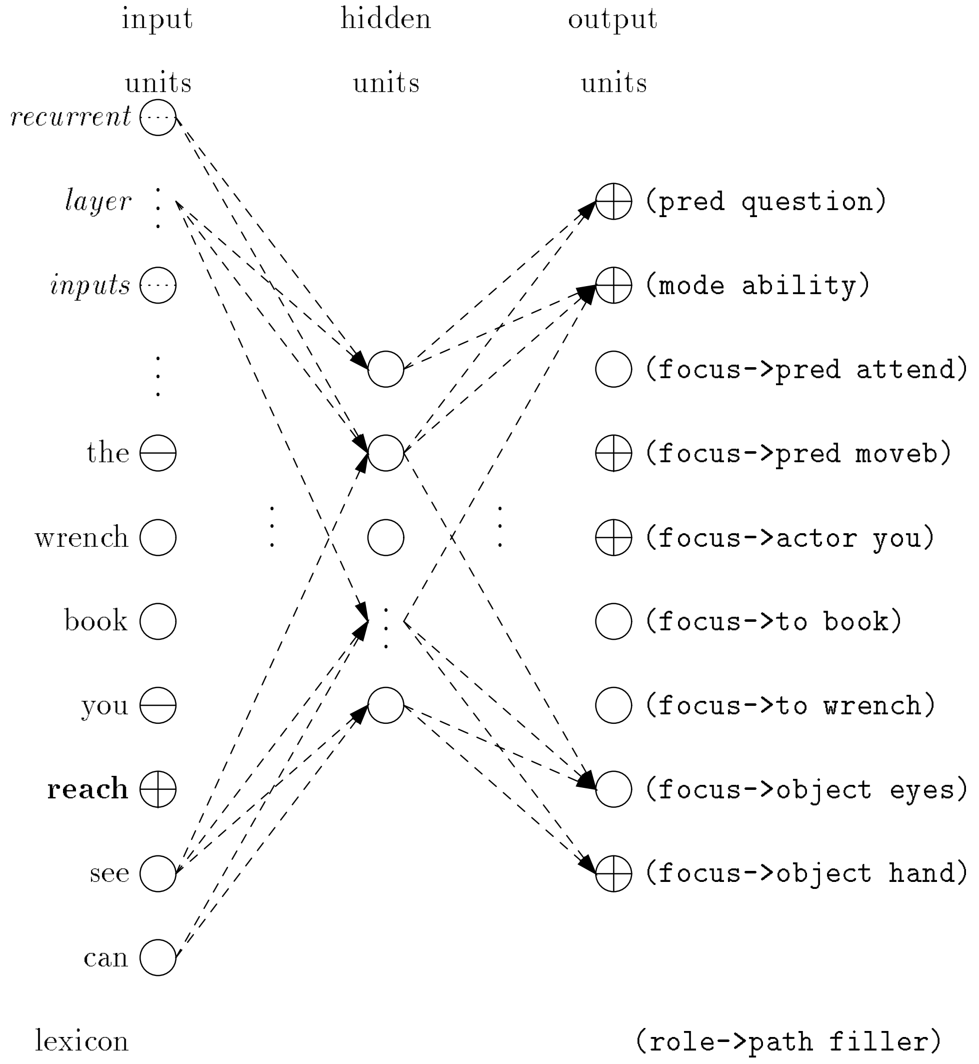

Figure 4.3: Example of intra-sentence processing. The current word of *focus* is reach.

period would conclude PIP's processing of this sentence yielding the final role/filler representation in the output layer. PIP would then incorporate that sentence into the *story gestalt* representation as described in the next section.

4.1.2 Text processing within a global context

Figure 4.1 depicts the overall topology of the network used by PIP. There are several things to note about PIP's network topology, as will be described next.

First, the STORY GESTALT architecture is wholly contained within PIP and is labeled as "inter-sentence processing." There is, however, a major difference in how this architecture is used. Section 3.3 explained that, during training of the *gestalt* layer, the *probe question* layer contains only the predicate of the question proposition for the sentences of text and the roles and fillers for the words of a sentence. In contrast, PIP uses a representation for the entire question. This representation would be what the intra-sentence processor could output; yet, in PIP, it is preprocessed. The advantages of this approach are two-fold. First, multiple questions can be asked (and answers supplied) for the same predicate. For example, reconsider story (a) from Section 3.3 repeated here.

(a) Andrew and Roxanne decided to go to the park. The weather was sunny. Andrew ran through the park. He threw the frisbee to Roxanne.

Instead of the single "question" `decided-to-go` (i.e., the predicate of the sentence representation) "asked" by the STORY GESTALT, PIP can ask "complete" questions such as Who went to the park? and Where did Andrew and Roxanne go? and supply answers such as Andrew and Roxanne and to the park. Thus, the questions asked by PIP probe different roles of the representation; in the first example above, PIP is asking a question about the `actor` role while in the second, the question is about the `to` role. Note also that each of these questions contains the same predicate, an impossibility with the STORY GESTALT. The second advantage is that, because PIP can ask complete questions, the text being processed is not limited to sentences all having different predicates.

The next thing to notice in Figure 4.1 is the (dotted) "copy" of the *story gestalt* layer to the *previous story gestalt* layer in the intra-sentence processing portion of the topology. It is this copy of the *story gestalt* layer which makes PIP unique among feed-forward connectionist models of natural language processing. This copy supplies the *global context* which aids in the disambiguation at the word processing level. That is, the learned representation, as captured in the *story gestalt* layer and copied to the *previous story gestalt* layer in the intra-sentence processor, provides information to the intra-sentence processor which allows it to disambiguate the meanings (either literal or non-literal) of ambiguous words and phrases. Without the aid of this global context, the intra-sentence processor would be unable to determine the "correct" interpretation of sentences under conditions where the context of previous text specifies which interpretation should be made. Section 4.2 illustrates this.

The intra-sentence processor works as described above in Section 4.1.1 with the exception of the inclusion of the *previous story gestalt* layer.

Training of PIP's network works as follows. The unit representing the first word of a sentence is activated (as described above in Section 4.1.1). The *previous proposition* and *previous story gestalt* are initialized to have zero activation.

The feed-forward algorithm is applied yielding output activations in the *current proposition* layer. The backprop algorithm is applied on the connections between the *current proposition* layer and the intra-sentence processor *hidden* layer and then on the connections between the *hidden* layer and the *input lexicon*, *previous proposition*, and *previous story gestalt* layers, modifying the weights on all of these connections (again, refer to Figure 4.1). This process is repeated until all of the words of the sentence have been fed through the intra-sentence processor, each time copying the output activations from the *current proposition* layer back into the *previous proposition* layer. As stated, this copy provides a "local context" during sentence processing.

Once the first sentence of the text has been completely processed, it is incorporated into the *story gestalt* layer as described in Section 3.3. When that process is complete, the *story gestalt* layer is copied into the *previous story gestalt* layer of the intra-sentence processor, the activations for the *previous proposition* layer are reset (i.e., set to zero), and the next sentence is sent through the intra-sentence processor. During this process, the input activations of the *previous story gestalt* layer are frozen. This cycle repeats until the entire text has been processed.

Once the network has been trained, testing occurs on untrained texts in a similar fashion to the training process; except, of course, no backprop occurs. The words of a sentence are fed through the intra-sentence processor sequentially with appropriate recurrent layer copying. After a sentence has been processed, its representation in the *current proposition* layer is fed through the network as far as the *story gestalt* layer. Again, this process is repeated for all sentences thus finally producing the "meaning" of the entire text in the *story gestalt* layer.

4.2 PIP in Action

This section presents an example of the processing of a text by PIP. Intra-sentence processing both with and without the aid of the global story gestalt context on the same example will illustrate the necessity of the global context. Assume that the network has been trained already. Consider the following text with a metaphor in the last sentence, as obtained from one of the templates in the Appendix, Section A.2.

(b) Alex hated Max. Max needed a hat. Alex lent Max a hat. Alex is a prince.

The metaphorical interpretation of last sentence in this text would have the following role/filler representation.

`[(pred member) (actor Alex) (class good-person)]`

For the literal interpretation, the filler of the `class` role would be `prince`. The discussion that follows assumes that all but the final sentence of the text has been processed by PIP, thus already creating the global story gestalt context.

Figures 4.4 and 4.5 show the output activations for selected units of the *current proposition* layer as identified by their roles and fillers in the left column. The boxes indicate the relative activation with larger boxes representing units having a greater activation value. An "invisible box" (i.e., a box with zero dimension) indicates an activation value of zero; a "dot" indicates a very small activation value (a box with very small dimension). Each column shows the activation values for the selected units after each word of the sentence is processed; the corresponding word is given at the bottom of each column. (All other units in the *current proposition* layer have activation values very close to zero; they are uninteresting in the

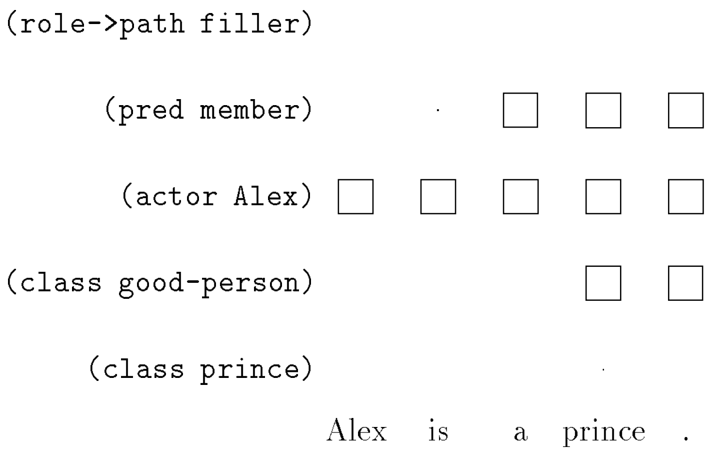

Figure 4.4: Word by word output activations for the last sentence in the metaphorical text (b) with the story gestalt context aiding the intra-sentence processor.

present discussion and thus are not shown.) Note that the sentence ending period is consider a "word" of the sentence.

First, consider the case where the global story gestalt context does aid the intra-sentence processor (Figure 4.4). After the first word of the sentence (Alex) is processed, the `(actor Alex)` role/filler pair is fully activated, correctly inferring that Alex is the subject of the sentence. Nothing concrete can be inferred after is is processed; that verb does not supply enough information. Note, however, that the `(pred member)` role/filler pair has a slight activation; this can be interpreted as a prediction of things to come. When a is processed, the `(pred member)` role/filler is fully activated, confirming the prediction. Next, prince indicates the filler of the `class` role. However, this is a metaphorical interpretation so the filler is actually `good-person` and not the literal (royal) `prince`. This yields the final metaphorical interpretation, as indicated by the activations on the `pred`, `actor`,

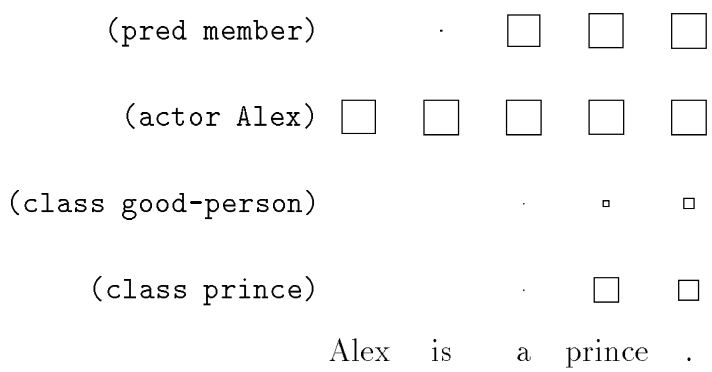

Figure 4.5: Word by word output activations for the last sentence in the metaphorical text (b) without the story gestalt context aiding the intra-sentence processor.

and `class` units. PIP has thus disambiguated the final sentence of the text and settled on an interpretation of that sentence.

Next, consider the case where the global story gestalt context does not aid the intra-sentence processor (Figure 4.5). Processing of the first three words is similar to the case where the global context is present. This is to be expected because there is no apparent ambiguity in the beginning of the sentence. Note however that there is a very slight activation of the `(class prince)` role/filler after the `a` is processed; this is due both to the absence of the global context and the distribution of the training data (there are more `prince`s than there are `good-person`s). Once the word `prince` is processed, both the `good-person` and `prince` fillers of the `class` role show some activation; however, neither is fully activated (the `prince` filler has a greater activation, again, due to the distribution of the training data). After the sentence ending period is processed, PIP is even further unsure of which filler of the `class` role should be activated; the `good-person` filler has a larger activation while the `prince` filler has a smaller activation than after `prince` was processed. Thus, for this detailed example, PIP is unable to disambiguate between a literal

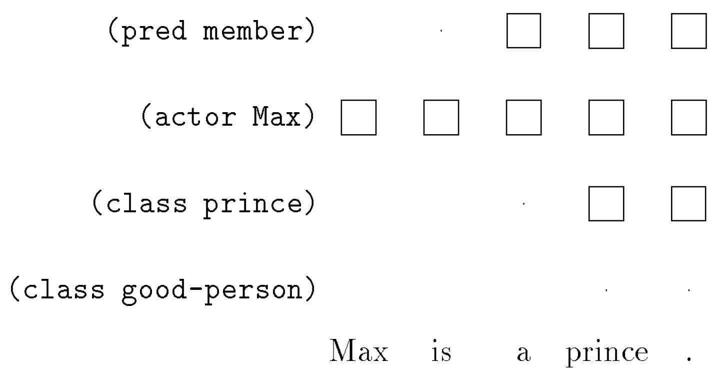

Figure 4.6: Word by word output activations for the last sentence in the literal text (c) with the story gestalt context aiding the intra-sentence processor.

and a metaphorical interpretation for prince when the global story gestalt context is not available to the intra-sentence processor. It is not even able to decide which interpretation to make.

It will be instructive to examine briefly what happens when processing a text where the final sentence has a literal interpretation to demonstrate that PIP is able to do the disambiguation correctly for the literal case. Consider this literal text.

(c) Max needed a knife. Max ordered Terry to get the knife. Max is a prince.

Figure 4.6 shows the output activations for selected units from the *current proposition* layer for the literal text above when the global context is present while Figure 4.7 shows the output activations for those same units when the global context is not present for the intra-sentence processor. The processing and conclusions for this literal text are the same as those presented above for the metaphorical text and thus will not be discussed in detail.

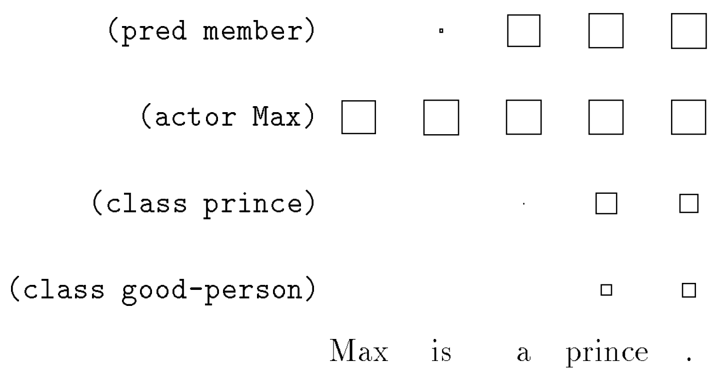

Figure 4.7: Word by word output activations for the last sentence in the literal text (c) without the story gestalt context aiding the intra-sentence processor.

It should be clear by these detailed examples that the global story gestalt context is affecting the output activations of the intra-sentence processor giving PIP the ability to disambiguate metaphorical and literal sentences. Furthermore, PIP is able to generate the correct interpretation directly without having to resort to a two-stage processing model.

4.3 Conclusion & Transition

This chapter has briefly discussed various artificial neural network topologies for natural language understanding. Specifically addressed was the topology used by PIP. The process of training PIP's network was presented along with the way that PIP uses a trained network to understand the individual sentences of a text and the entire text as a whole.

The next three chapters present experimental results of applying PIP to texts illustrating the problems found in the domains of lexical ambiguity, metaphor, and indirect speech acts. Since PIP attempts mainly to address these issues as applied to individual sentences of a text and not those issues specifically associated with the entire text itself, of interest in the analysis to follow is the output of the *current proposition* layer for the individual ambiguous sentences (the ambiguous sentences being those containing lexical ambiguity, metaphors, or indirect speech acts). That is, the analysis addresses the meaning representation produced by PIP (as found in the *current proposition* layer) for those amgibuous sentences from the texts.

Chapter 5

Lexical Disambiguation: A Neural Network Simulation

"Mine is a long and sad tale!"
said the Mouse turning to Alice, and sighing.

"It *is* a long tail, certainly,"
said Alice, looking down with wonder at the Mouse's tail;
"but why do you call it sad?"

—LEWIS CARROLL, *Alice's Adventures in Wonderland*

This chapter presents the results of running PIP on the set of texts found in the Appendix, Section A.1. The final sentence in each of these texts contains one or more words which are lexically ambiguous. For example, green can be interpreted either as the color or as describing inexperience and field can be interpreted either as an area of ground (e.g., a baseball field) or as a group of contestants (e.g., the runners in a foot race). The context set up by the early sentences in the texts determines which of the potential meanings to give to the lexically ambiguous words. For example, consider the following three texts generated from the templates. (The appendix explains how the texts are generated from the templates. The italicized words indicate those which are substitutable.)

(d) *Sande* bought some *knick-knacks.* The *pitcher* was *blue.*

(e) The *batter* hit a *line-drive. He* made it to the base on time.

(f) The *horses* were in the gates. They were *ready.*

The final sentence of text (d) actually has two lexically ambiguous words: pitcher can be either a jug from which water is poured or the baseball player who throws the ball to the catcher; blue can be either the color or can indicate a "sour mood." Out of context, there are thus two plausible interpretations for The pitcher was blue: "the water jug was colored blue" and "the baseball player was in a sour mood." (It does not really make sense to ascribe "sour mood" to the inanimate "water jug" nor to consider the "baseball player" as being "colored blue.") The last sentence of text (e) has the lexically ambiguous word base, which can be interpreted either as a military base or one of the bases found on a baseball diamond (diamond, another lexically ambiguous word! yes, our language is full of them). This sentence also has the pronoun he, the referent of which must be resolved and would determine which interpretion to give to base. As in the final sentence of text (e), the final sentence of text (f) has a pronoun, they, which must be resolved. As will be seen, PIP is able to solve these lexical ambiguity and pronoun reference problems when it has available a context in which to perform its interpretation task. When that context is not used, PIP cannot solve these problems accurately.

5.1 Simulation Specifics

For the data reported in this chapter, PIP's network (cf. Figure 4.1) was configured as follows. The *input lexicon* layer had 69 units; the *current proposition* layer (and thus, the *previous proposition* layer) and the *probe answer* layer each had 92 units. There were 104 units in the *probe question* layer. Each *hidden* layer had 20 units and the *story gestalt* layer had 15 units. Initial weights on the connections were randomly generated in the interval $(-0.3, 0.3)$. The learning rate (η) and momentum (α) values were constant throughout training and were 0.03 and 0.7, respectively.

In all, 160 different texts were generated by the lexical ambiguity templates. The texts were divided into training and testing sets. Each text was generated with a 0.75 probability of being included in the training set; if a text was not in the training set, it was in the testing set. Thus, the training and testing sets were disjoint. For the data reported, there was an average of 120.8 texts in the training set (or, approximately 75.5% of the total) and 39.2 texts in the testing set (or, approximately 24.5%).

A series of ten simulation pairs was run on this data; each simulation pair consisted of the same training and testing sets with the difference being that one of the pair included the story gestalt context to aid the intra-sentence processing while the other did not. The data reported in Figures 5.1, 5.6, and 5.7 are averaged over the ten pairs of simulations. The data from Figures 5.6 and 5.7 are after the networks had been trained for 2000 epochs. The data in Figures 5.3, 5.4, and 5.5 are for the single trial where the network had been trained for 6000 epochs.

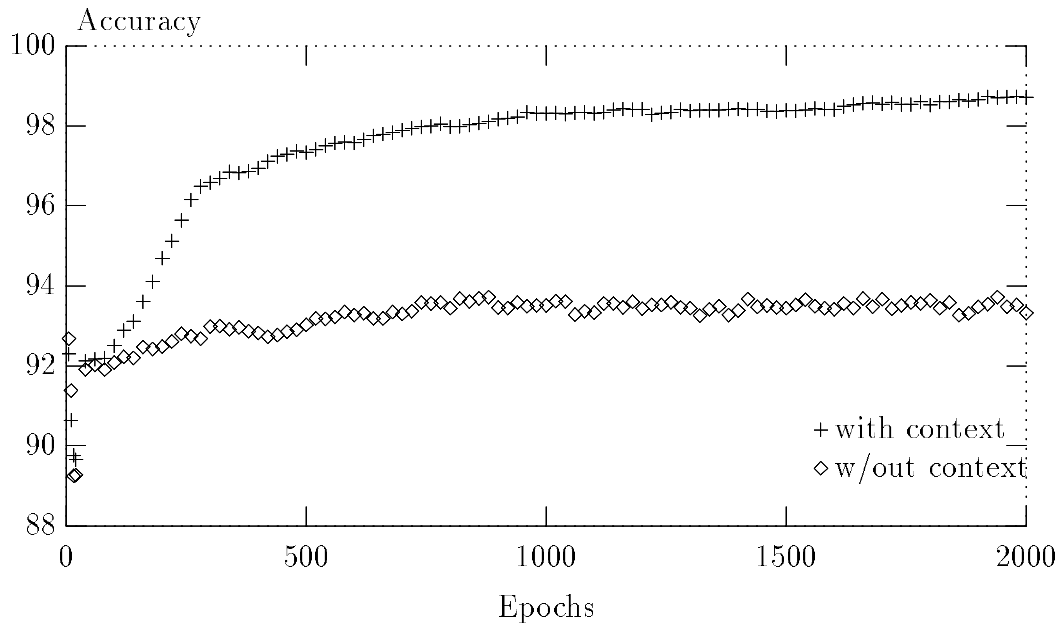

Figure 5.1: Percentage of correct units for the final sentences in the lexical ambiguity testing texts averaged over ten trials.

5.2 Simulation Results

The graph depicted in Figure 5.1 shows the percentage of correct units[1] (i.e., accuracy) averaged over all of the units in the *current proposition* output layer for the final sentences in the testing sets of texts. Two curves are shown in Figure 5.1: one plots the accuracy for the simulations when the global story gestalt context is present during intra-sentence processing while the other plots the accuracy when the global context is not present. As the data in the figure show, the simulations when context is present are, on the average, almost 5% more accurate than those simulations when context is not present to aid the intra-sentence processor indicating that a global context is indeed useful for lexical disambiguation.

[1]Output units are "correct" if their activation is within 0.2 of their target value.

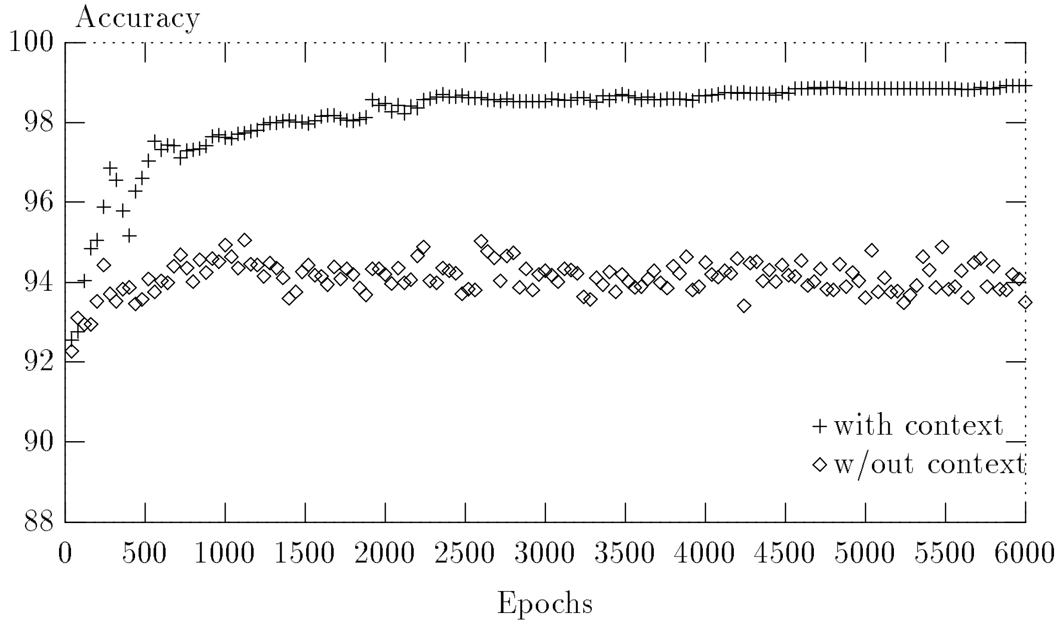

Figure 5.2: Percentage of correct units for final sentences in lexical ambiguity texts for a single trial.

One of the ten simulation pairs was allowed to train for 6000 epochs. This was done to determine whether or not additional training would change the conclusions indicated by the data in Figure 5.1. The results of this extra training are shown in Figure 5.2. (The data points represent the state of training of the network applied to the testing set of texts at forty epoch intervals.) While there is some improvement for this particular simulation pair when context is present, the improved performance does not contradict the conclusions drawn from the data in Figure 5.1. Indeed, it further supports the idea of the usefulness of a global context because of the slightly improved performance with further training. However, there are a few things to note about the data as reported in this figure. First, the overall accuracy when context is present is a bit higher than what is reported in Figure 5.1. The actual accuracy value in this case after training for the 6000 epochs is approximately 98.9%. Also, the data points for the accuracy when context is not present are much more scattered. Both of these are because the data reported in Figure 5.2 are for a single simulation pair whereas the data reported in Figure 5.1 are averaged over ten simulation pairs. It might be useful to explore further why the data is more scattered when context is not used than when it is used. The hypothesis offered here is that this is because the network is unable to settle on a particular interpretation of the ambiguous text; that is, PIP oscillates between interpretations for those units representing the ambiguous words of the final sentences of the texts. Since these data points are forty epochs apart, the network can change considerably during the intervening training epochs.

Consider again text (d) from the beginning of the chapter, repeated here.

(d) *Sande* bought some *knick-knacks*. The *pitcher* was *blue*.

Recall that the final sentence of this text has two plausible interpretations out of context: "the water jug was colored blue" and "the baseball player was in a sour mood." These are represented in PIP by the following role/filler pairs, in a Lisp notation.

```
[(pred color) (object jug-pitcher) (is blue)]
```

and

```
[(pred mood) (actor bb-pitcher) (is -5)].
```

Figure 5.3 shows the output activation levels for these role/filler pairs both with and without context aiding the intra-sentence processor for text (d). The data indicate that PIP is able to determine both an interpretation and the correct interpretation when context is used and is unable to determine any interpretation when

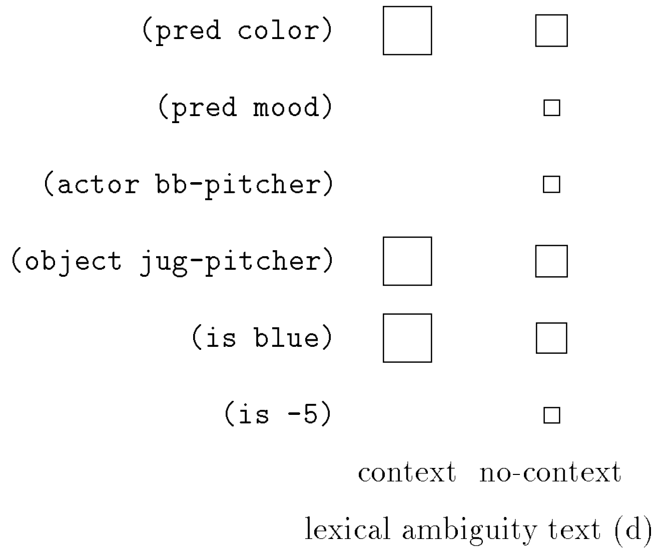

Figure 5.3: Selected output unit activations for the final sentence in text (d): The pitcher was blue.

context is not used. Furthermore, PIP hypothesizes both plausible interpretations by giving some activation to each potential role/filler pair, as is expected, when context is not used for disambiguating. This example illustrates the need for a global context at the intra-sentence processing level.

It is not always the case that, by incorporating context during intra-sentence processing, PIP is able to interpret correctly all texts containing lexically ambiguous words; otherwise, the accuracy plotted in Figure 5.1 would eventually reach 100%. Consider again text (e) from the beginning of the chapter.

(e) The *batter* hit a *line-drive*. *He* made it to the base on time.

For one of the simulation pairs, this text was part of the testing set of texts; thus, it was not used to train the network. Recall that He made it to the base on time has multiple plausible interpretations out of context, depending on the referent of the pronoun He and because of the ambiguous word base. To illustrate two plausible interpretations, assume that He refers either to a baseball batter or a soldier yielding either a "baseball player running to a baseball base" interpretation or a "soldier returning to his military base" interpretation. (Of course, the soldier could be playing a game of baseball, which further complicates matters!) The role/filler representations, in Lisp notation, for these two potential interpretations are then:

```
[(pred ptrans) (actor bb-batter) (object bb-batter) (to bb-base)
                          (status safe)]
```

and

```
[(pred ptrans) (actor soldier) (object soldier) (to mil-base)
                         (status on-time)].
```

For selected output units of the *current proposition* layer, PIP produced, after training for 6000 epochs, the output activations as shown in Figure 5.4 for text (e). Text (e) sets up a context inferring the "baseball player running to a baseball base" interpretation of He made it to the base on time. When context is present, PIP is able to determine correctly all of the role/filler pairs except for the `status` role. (Recall that an output unit is correct if it is within 0.2 of its target value.) However, the `safe` filler has the highest activation of any fillers for that role. Thus, were PIP to use a heuristic which stated "if there is no obvious 'correct' filler for a role, choose the filler having the highest activation," PIP would, in this case, be able to choose correctly all of the role/filler pairs. Unfortunately, this heuristic will not work because not every sentence will have a filler for every possible role; e.g., He made it to the base on time does not have a filler for the `is` role, as does The pitcher is green. Thus, when context is present, PIP is "close" in determining the correct

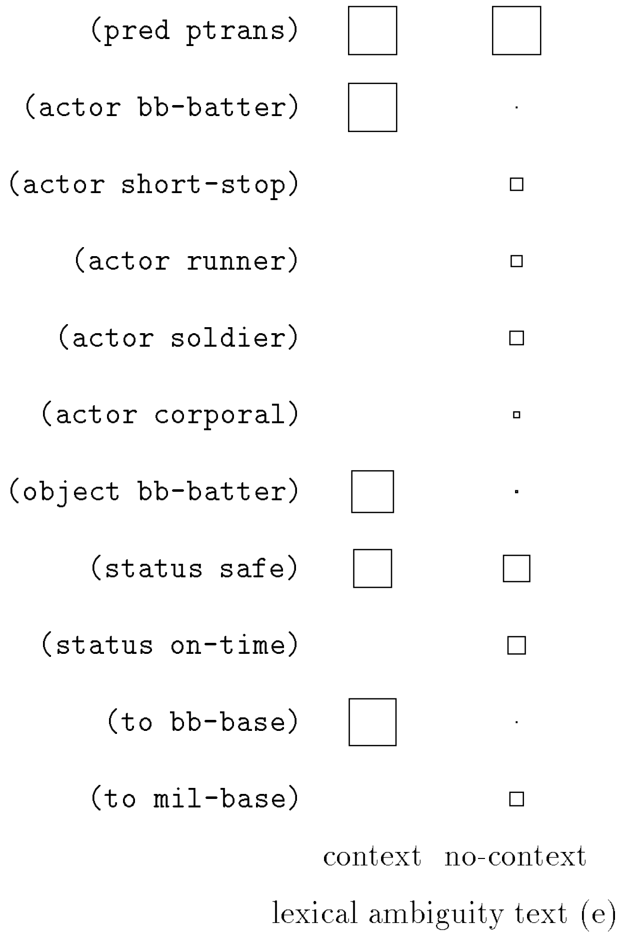

Figure 5.4: Selected output unit activations for the final sentence in text (e): He made it to the base on time.

interpretation; but, it is not always 100% accurate. When context is not present, PIP not only is unable to choose the correct interpretation, it is unable to choose any interpretation, given the definition that an output unit must be within 0.2 of its target value. PIP does, however, hypothesize many different interpretations by partially activating all of the potential fillers for those roles applicable to the final sentence of the text, given any context. This can be observed by looking at the various fillers for the `actor` role with no context in Figure 5.4. Note that the `object` and `to` roles have other fillers which are partially activated in a fashion

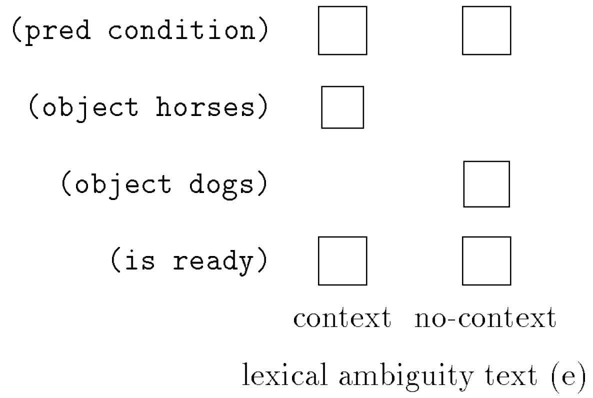

Figure 5.5: Selected output unit activations for the final sentence in text (f): They were ready.

similar to that of the actor role. PIP is able to fill the pred role accurately because both interpretations activate the ptrans filler.

In order to illustrate PIP's ability to resolve the referents of pronouns, again consider text (f). In one of the simulation pairs, this text was part of the testing set of texts; thus, it was not used during training of the network.

(f) The *horses* were in the gates. They were *ready*.

The referent of They in the final sentence is obviously horses yielding the following role/filler pairs.

[(pred condition) (object horses) (is ready)]

Figure 5.5 shows the output activations of the *current proposition* layer for these role/filler pairs plus one other, (object dogs). As the data indicate, PIP is able to resolve the referent of They when context is used; the correct role/filler pair is (object horses). But, when context is not used, PIP activates the wrong filler, dogs, for the object role. This is because there were twice as many texts in the training set where the referent of They was dogs than where the referent was horses; without context, the network learns the more common interpretation as found in the distribution of the training set. Thus, when a global context is present, PIP is able to resolve the pronoun reference; however, when the global context is not present, PIP might not resolve the pronoun reference.

Figures 5.6 and 5.7 present the average errors for all output units of the *current proposition* layer averaged over the ten simulation pairs after the networks had been trained for 2000 epochs for the final sentences of the testing sets of texts. As can be seen, for virtually all of the output units, the average error is higher when context is not present than when context is present during the intra-sentence processing. This is to be expected. The average errors when context is not present are not higher because, for most of the texts, the vast majority of output units should have, and indeed do have, zero activation, for these units apply to role/filler pairs which do not appear in the representations for the final sentences of the texts.

5.3 Conclusion & Transition

The data reported in this chapter support the need for a "global context" during processing of lexically ambiguous words; the "local context" supplied by a sentence is not always enough. This is evident in the example The pitcher was blue. Also of note is PIP's ability to resolve the referents of pronouns; this was shown by the accurate activation of the referent for they in They were ready.

The next two chapters present data showing PIP's ability to utilize the global context in order to interpret correctly so-called "figurative" text, i.e., text whose

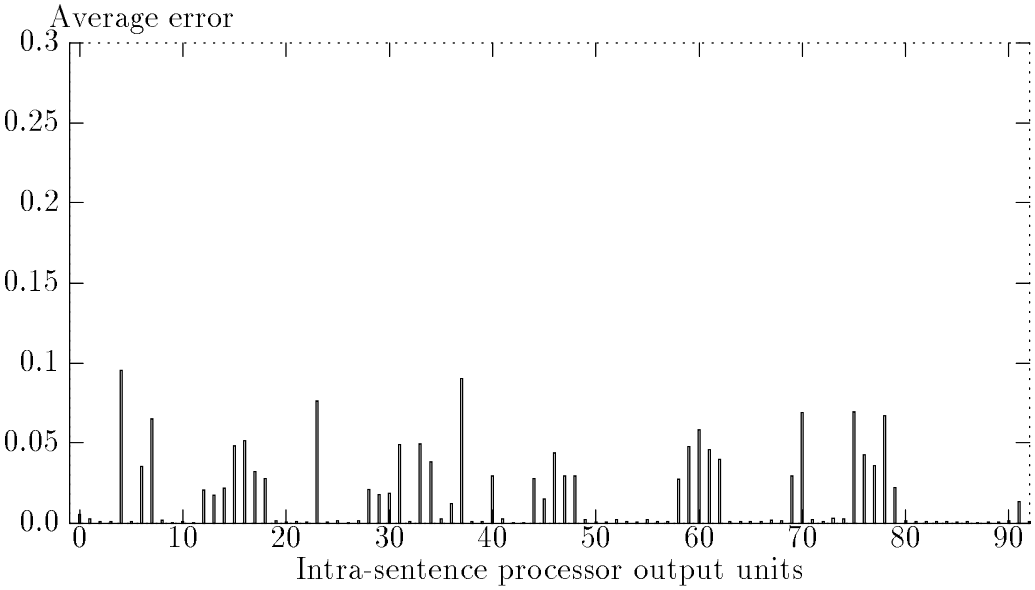

Figure 5.6: Average errors for all output units of intra-sentence processor for all stories when story gestalt context is present.

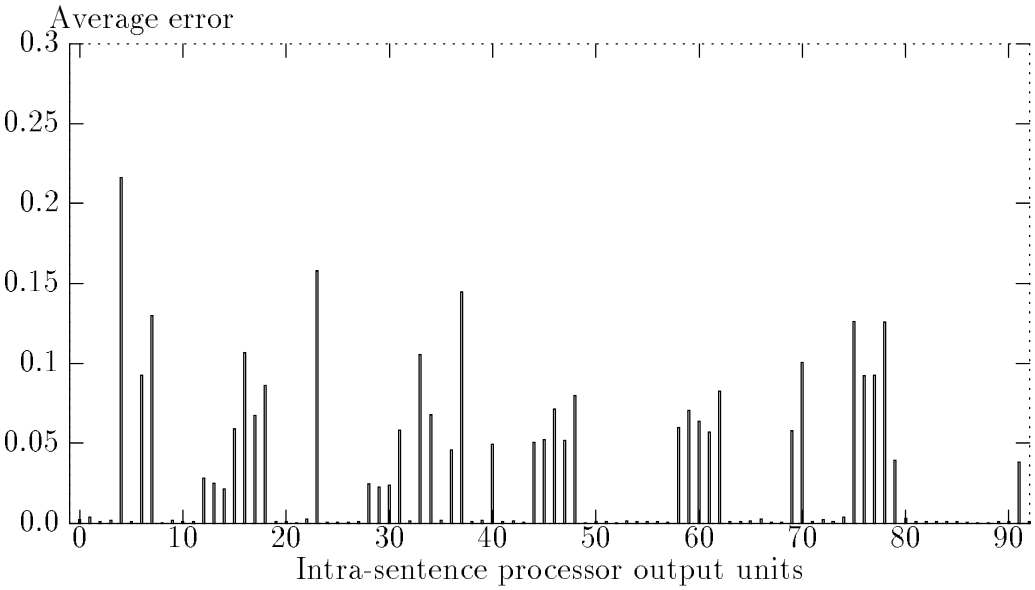

Figure 5.7: Average errors for all output units of intra-sentence processor for all stories when story gestalt context is **not** present.

intended meaning is not the same as the *literal* meaning. The examples applied to PIP include metaphor and indirect speech acts.

Chapter 6

Metaphor Resolution: A Neural Network Simulation

"... He was an old crab, *he* was."

—LEWIS CARROLL, *Alice's Adventures in Wonderland*

This chapter presents the results of running PIP on a set of metaphor texts as generated by the templates found in the Appendix, section A.2. These texts are based on those used by Keysar (1989) in experiments on metaphor. The first few sentences of the text set up a context to determine whether or not the last sentence has a literal or a non-literal metaphorical interpretation. These final sentences are of the form X is a Y where Y represents the metaphor. An example is Terry is a prince; should prince be interpreted literally (as a member of the royal family) or metaphorically (as a "good person")?

PIP is applied to these literal and metaphorical texts both with and without the global story gestalt context aiding the intra-sentence processor. The data reported here demonstrate that there is a difference between PIP's ability to interpret correctly the potentially metaphorical final sentence in each text depending on whether or not the global context is used by the intra-sentence processor. The presence of this difference demonstrates both that the global context is useful and that PIP applies the global context in an appropriate and necessary manner when applied to metaphorical texts.

6.1 Simulation Specifics

For the data reported in this chapter, PIP's network (cf. Figure 4.1) was configured as follows. The *input lexicon* layer had 62 units; the *current proposition* layer (and thus, the *previous proposition* layer) and the *probe answer* layer each had 106 units. There were 122 units in the *probe question* layer. Each *hidden* layer had 20 units and the *story gestalt* layer had 15 units. Initial weights on the connections were randomly generated in the interval $(-0.3, 0.3)$. The learning rate (η) and momentum (α) values were constant throughout training and were 0.03 and 0.7, respectively.

In all, 1168 different texts were generated by the metaphor templates. The texts were divided into training and testing sets. Each text was generated with a 0.75 probability of being included in the training set; if a text was not in the training set, it was in the testing set. Thus, the training and testing sets were disjoint. For the data reported, there was an average of 870 texts (or approximately 74.5% of the total) in the training set and 298 texts (or approximately 25.5%) in the testing set.

A series of five simulation pairs were run on this data; each simulation pair consisted of the same training and testing sets with the difference being that one of the pair included the story gestalt context to aid the intra-sentence processing

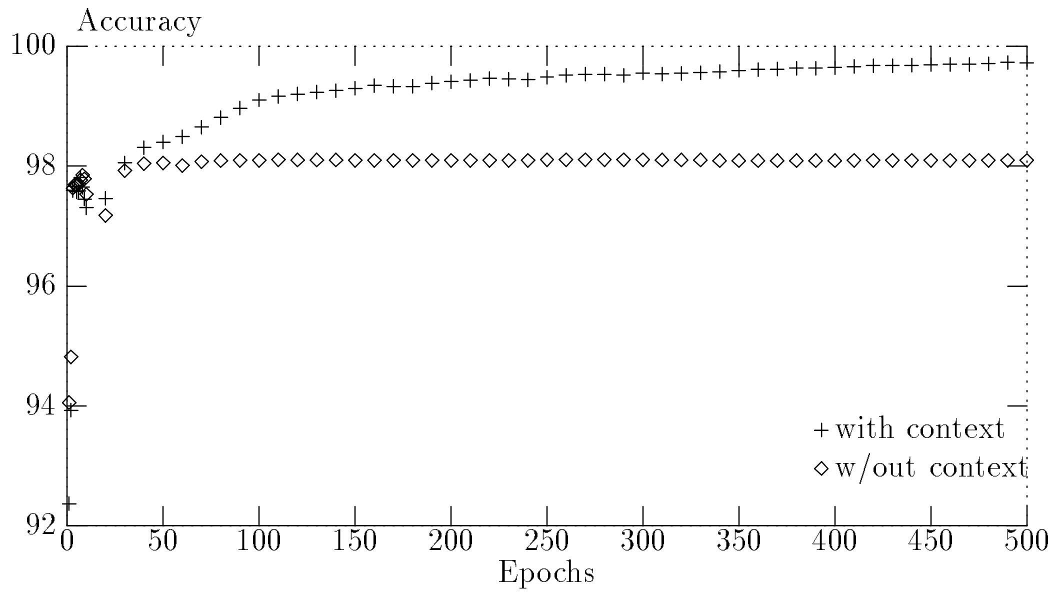

Figure 6.1: Percentage of correct units for final sentences in the metaphor testing texts averaged over five trials.

while the other did not. The data in Figures 6.1, 6.3, and 6.4 are averaged over the five simulation pairs. The data in Figures 6.2, 6.3, and 6.4 are taken after the network was trained for 500 epochs.

6.2 Simulation Results

The graph depicted in Figure 6.1 shows the percentage of correct units (i.e., accuracy) averaged over all of the units in the *current proposition* output layer for the final sentences in the testing sets of texts. (The final sentences of the texts contain the potential metaphors.) Figure 6.1 graphs two curves; one plots the percentage of correct units for those simulations where the story gestalt context aided the intra-sentence processor and the other plots the accuracy where the story gestalt context was absent during intra-sentence processing. At first glance, it appears that there is not much difference between the accuracy shown by the two simulations; this conclusion, however, is misleading. After 500 training epochs (data for both networks seem to have reached an asymptote by this time), the difference between the accuracy for the two simulations reflects the inability of the simulation without context to determine correctly the output activation for the two units representing which interpretation (literal or metaphorical) is to be given to the last sentence of the text. The two units, or approximately 1.9% of all of the output units in the *current proposition* layer, determine which interpretation to give to the potential metaphor. (This is because PIP uses a localist representation.) One of these units is for the literal interpretation while the other is for the metaphorical interpretation. There is no difference in output unit activations for all of the other units, given the same input sentence. When context is not present, PIP is unable to determine the correct activation for either of these units; thus, PIP's interpretation of the potential metaphor is itself ambiguous. When context is present, these units have the correct activation.

Figure 6.1 also indicates that the accuracy of the network topology without the story gestalt context asymptotes earlier and to a lower percentage; this is to be expected because that network topology does not have the information necessary to compute the activation for the ambiguous units correctly. With context present, more training is required to differentiate, by using the context, the ambiguous units. Thus, the story gestalt context is necessary during intra-sentence processing to disambiguate potential metaphors.

Consider the following two example texts as generated by the templates in the Appendix. (Again, italicized words indicate those words which are substitutable.)

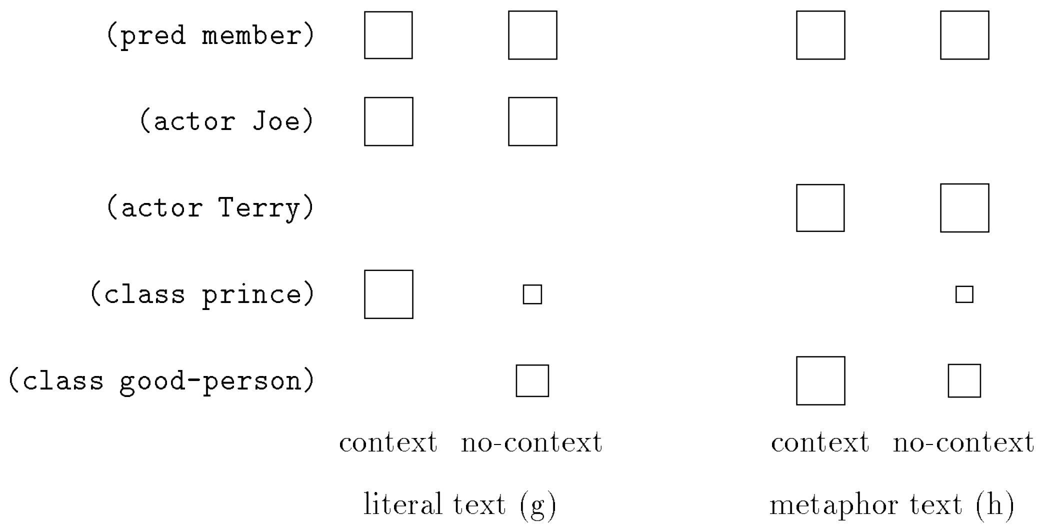

Figure 6.2: Pertinent unit activations for Joe/Terry is a Prince.

(g) *Joe* needed a *sword. Joe ordered Alex* to get the *sword. Joe* is a *prince.*

(h) *Terry* hated *Max. Max* needed a *hat. Terry gave Max* a *hat. Terry* is a *prince.*

When applied to the trained networks from one of the simulation pairs, these texts produce, for the pertinent units from the *current proposition* layer on the final sentences of the texts, the output activations as shown in Figure 6.2. The literal interpretation should activate the `(class prince)` role/filler pair while the metaphorical interpretation should activate the `(class good-person)` role/filler pair. Thus, the last sentence in text (g) would be represented, in Lisp notation, as

```
[(pred member) (actor Joe) (class prince)]
```

As can be seen in Figure 6.2, for the simulation when the story gestalt context is present (first and third columns of activations), the "correct" units are fully activated for all units for both the literal text (`(pred member)`, `(actor Joe)`, and `(class prince)`) and the metaphorical text (`(pred member)`, `(actor Terry)`, and `(class good-person)`) and the "incorrect" units (`(class good-person)` for text (g) and `(class prince)` for text (h)) have no activation, which is as desired. However, when context is absent (second and fourth columns of activations), neither of the units representing the interpretations of the ambiguous word prince (the fillers of the `class` role) are correct for either text. Each plausible filler of the `(class)` role has some activation though, indicating that PIP is unable to disambiguate the interpretation of prince. That is, when there is no global story gestalt context, PIP cannot decide which interpretation to give to prince; it does, however, choose the two plausible interpretations giving each some, but not full, activation. Notice that the non-ambiguous units (representing the fillers of the `(pred)` and `(actor)` roles) are correct, however. The difference in accuracy can be attributed directly to the difference between the two simulations, namely, that one simulation incorporates the story gestalt context during intra-sentence processing and one does not.

Because PIP is unable to disambiguate the `(class)` role, it might be expected that the activation of these units would uniformly be 0.5 when the global context does not aid the intra-sentence processor. That is, when PIP cannot disambiguate between two possible fillers for a particular role, why are the activations for those role/filler pairs not uniformly 0.5? Should they not be because each is a possible filler for the single role and each should have the same "opportunity" to fill that role? The fact that the activations are not uniformly 0.5 is due to the uneven distribution of the different literal and metaphorical texts generated by the templates. That is, there are proportionally more metaphor (for `good-person`) texts than there are literal (for `prince`) texts.

Figures 6.3 and 6.4 present the average error for all 106 output units in the *current proposition* layer (units $0 - 105$); the data in Figure 6.3 are for the simulations where the story gestalt context aids the intra-sentence processor and the data in Figure 6.4 are for the simulations where the story gestalt context does not aid the intra-sentence processor. Of all of these output units, the most interesting are units $16 - 26$, representing the `class` role and its various fillers. (Recall that the `class` role identifies the potential ambiguity in the final sentences of the metaphor texts.) Thus, it is not surprising that both figures indicate that these units, by virture of their having the highest average errors, pose the most problems to the simulations. None of the other units represent any ambiguity in the texts; thus, they have almost no average error. However, when context is present (Figure 6.3), the average errors are much smaller than when context is absent (Figure 6.4). Of these eleven units, numbers 16 and 23 form the basis for the metaphorical interpretations (i.e., their fillers indicate a metaphorical interpretation of a sentence); the other units are for the literal interpretations. Therefore, it is understandable that, in Figure 6.4 when context is not present, these units should have the highest average error. PIP is unable to determine the correct interpretation when the story gestalt context is not present.

6.3 Conclusion

This chapter has presented evidence of the need for a global context during processing of certain types of metaphors. PIP is able to use its global story gestalt context to interpret potential metaphors correctly by producing either a literal

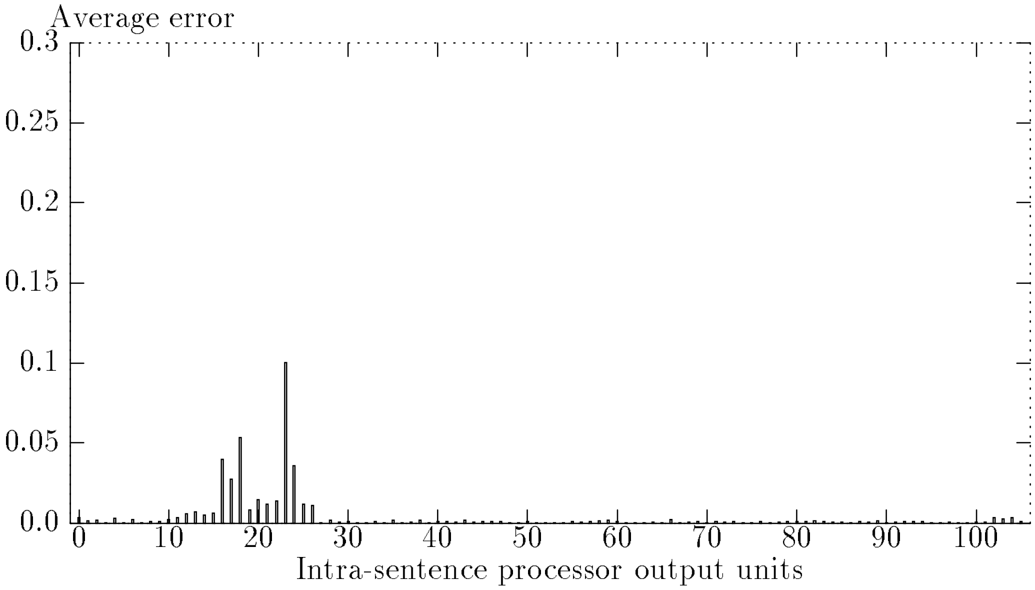

Figure 6.3: Average errors for all output units of intra-sentence processor for all stories when story gestalt context is present.

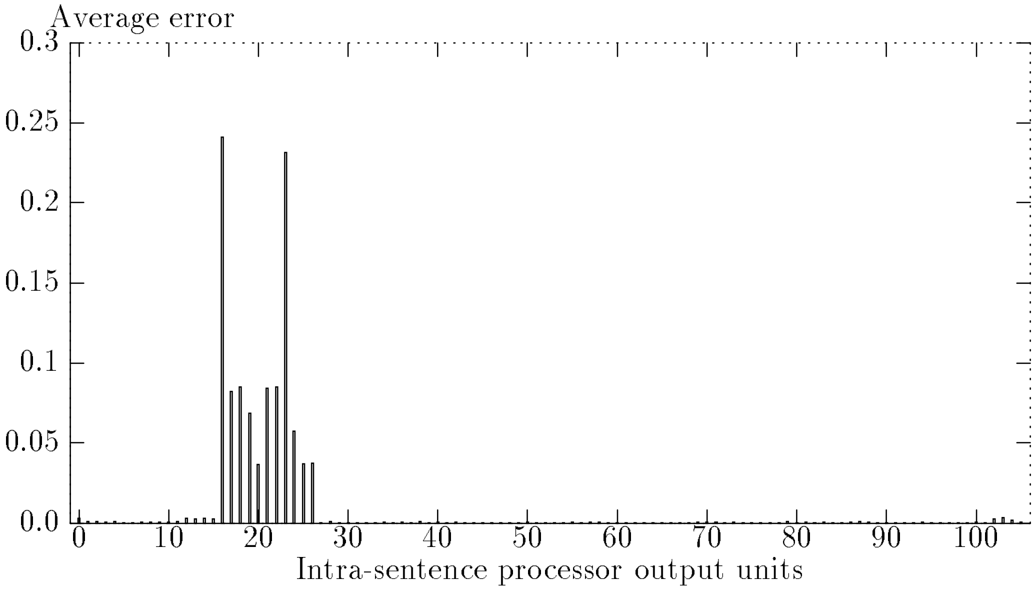

Figure 6.4: Average errors for all output units of intra-sentence processor for all stories when story gestalt context is **not** present.

interpretation when that is warranted or by producing the metaphorical interpretation. When the global context is not present during intra-sentence processing, PIP is unable to produce the desired interpretation of the potential metaphor.

Chapter 7

Indirect Speech Act Interpretation: A Neural Network Simulation

"Then you should say what you mean," the March Hare went on.

"I do," Alice hastily replied;
"at least – at least I mean what I say –
that's the same thing, you know."

"Not the same thing a bit!" said the Hatter.

—LEWIS CARROLL, *Alice's Adventures in Wonderland*

This chapter presents the results of running PIP on the set of texts found in the Appendix, section A.3. The final sentences of these texts are speech acts (Austin, 1962; Searle, 1969) and are of the form "Do you have a X?" where X is some object such as a pencil or notebook. These speech acts can be interpreted either directly (the questioner is asking whether or not the hearer actually possesses the object X) or indirectly (the questioner is asking the hearer to give him the object if the hearer has it). The context prior to the speech act determines the intended interpretation. For example, consider the following two texts generated from the templates (again, the italicized words indicate those which are substitutable).

(i) The *student* went to the *biology* class. The *student* needed a *book* for the *quiz*. The *instructor* asked the *student*, "Do you have a *book*?"

(j) The *student* went to the *biology* class. The *student* needed a *pencil* for the *exam*. The *student* asked the *instructor*, "Do you have a *pencil*?"

In text (i), it is the student who needs the book and not the instructor, the questioner; thus, the speech act here is intended to be interpreted literally. The instructor is actually inquiring whether or not the student has a book. An intended indirect interpretation would imply that the instructor needs the book. The context for text (j), on the other hand, implies that the speech act Do you have a pencil? is intended to be interpreted indirectly; i.e., the student is asking the instructor for the instructor to give a pencil to the student, since it is known that it is the student who needs a pencil.

A subset of all indirect speech act texts generated from the templates was used to train the network topology of PIP as described in Chapter 4. The remaining texts (from those generated by the templates) formed a test set which, when applied to the trained network, demonstrates that the global context is both useful and necessary to determine the intended interpretation of the potential indirect speech acts present in the texts. The results presented below will show a difference between the processing of the literal and indirect speech acts depending on whether or not the global context aids the processing of the speech acts. This difference is attributable directly to the presence or absence of the global context since all else in the network is equal. Thus, the global context is important during processing of (indirect) speech acts.

7.1 Simulation Specifics

For the data reported in this chapter, PIP's network (cf. Figure 4.1) was configured as follows. The *input lexicon* layer had 33 units; the *current proposition* layer (and thus, the *previous proposition* layer) had 47 units. There were 65 units in the *probe question* layer and 57 units in the *probe answer* layer. The *story gestalt* layer had 15 units and each hidden layer of the network had 20 units. Initial weights on the connections were randomly generated in the interval $(-0.3, 0.3)$. The learning rate (η) and momentum (α) values were constant throughout training and were 0.03 and 0.7, respectively.

The speech act templates generated a total of 2560 texts which were divided into a training set and a testing set. As each text was generated, there was a 0.75 probability that it would be in the training set; if it was not in the training set, it was in the testing set. Thus, the training and testing sets were disjoint and when PIP was applied to the texts in training set, it was seeing "novel" texts. For the data reported below, there was an average of 1924.2 texts (or approximately 75% of the total) in the training set and 635.8 texts (or approximately 25%) in the testing set.

A series of five simulation pairs was run on different training and testing sets of the speech act texts. Each simulation pair consisted of the same text sets with the difference between the simulations being that one included the global story

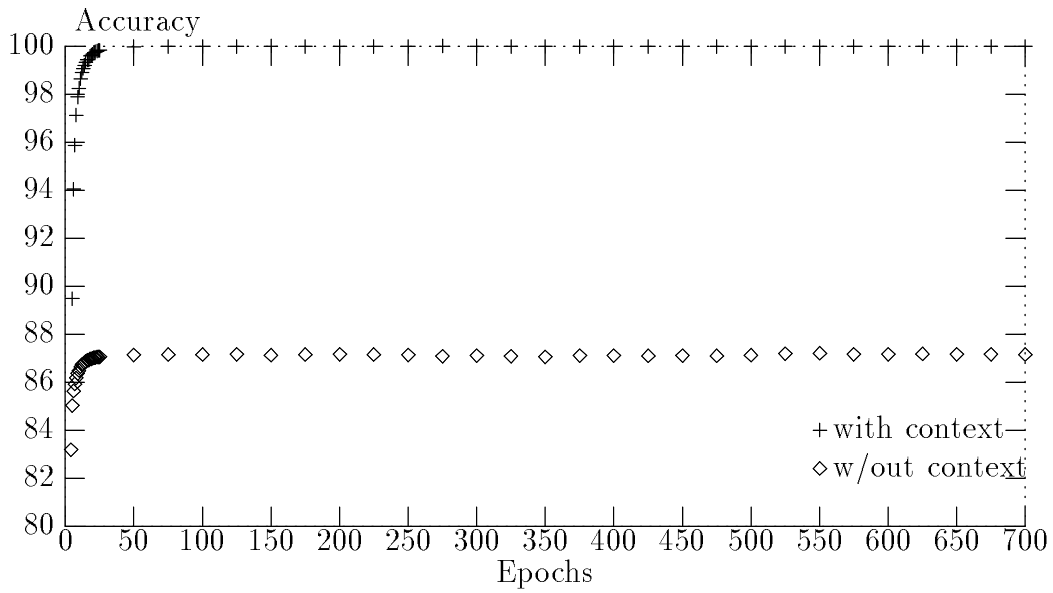

Figure 7.1: Percentage of correct units for final sentences in indirect speech act texts averaged over five trials.

gestalt context during intra-sentence processing and one did not. Data for Figures 7.2 and 7.3 are for a single simulation pair; for the remaining figures, the data are averaged over the five simulation pairs. The data in all figures except Figure 7.1 are taken after the network had been trained for 700 epochs.

7.2 Simulation Results

The graph shown in Figure 7.1 plots the percentage of correct units (i.e., accuracy) in the *current proposition* layer for the last sentence of all texts in the testing set against the number of epochs for which the network had been trained. Two curves are plotted in Figure 7.1: the accuracy when the global story gestalt context is present for the intra-sentence processor and the accuracy when the global context is absent.

The difference in accuracy between these two curves is markedly different from the similar graph in Chapter 6. There are two major reasons for this difference. First, the actual difference between the two curves is greater than the difference depicted in Figure 6.1. This is due to the fact that four of the 47 units (or roughly 12%) in the *current proposition* layer are used to distinguish between the literal and indirect interpretations (two of the four units determine that the interpretation is literal while the other two determine that the interpretation is indirect). This contrasts with the roughly 2% for the metaphor texts. Second, the smaller number of all output units suggests a more homogenous set of texts, which is indeed the case. There were 12 different text templates for the metaphor texts whereas there were only 2 different text templates used for the indirect speech act texts. This fact also accounts for the steeper learning curves in Figure 7.1.

When the global story gestalt context is present during intra-sentence processing, PIP is able to gain virtually 100% accuracy over all output units in the *current proposition* layer. This contrasts with the less than 88% accuracy when the global context is absent. This is to be expected because the global context provides the necessary information to disambiguate the four units which distinguish between the literal and indirect interpretations of the speech acts at the end of the texts.

Careful examination of the curve plotting the average accuracy of the simulation when the global context is not present will show that it asymptotes earlier (and to a lower value) than the other curve, as did the metaphor case reported in Chapter 6. Again, this is due to the fact that, with the added information of the global context, training will take longer because the network is able to disambiguate the ambiguous four output units of the *current proposition* layer. Since

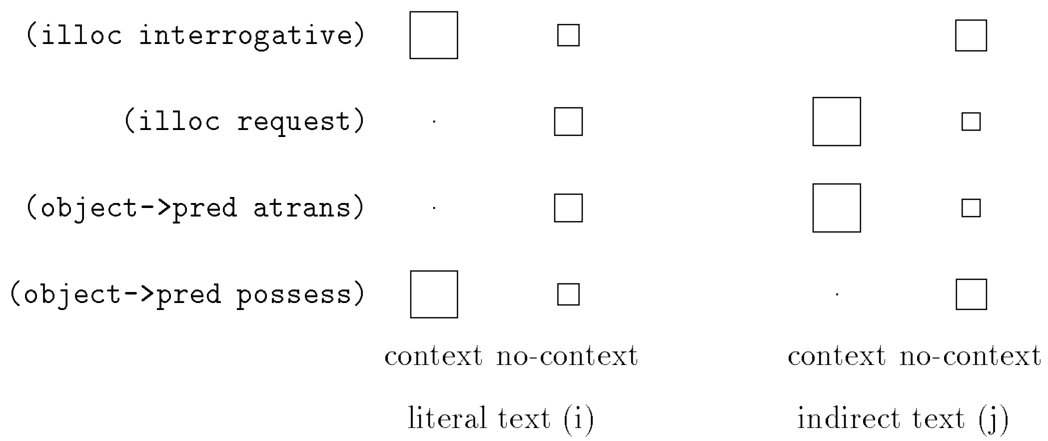

Figure 7.2: Selected *current proposition* layer output unit activations for some role/filler pairs in the literal and indirect speech act texts. Context is necessary for the correct interpretation of these selected role/filler pairs.

the network is unable to disambiguate these four units when the global context is not present (thus, the lower asymptote), it is able to learn faster. Therefore, the story gestalt context is necessary during intra-sentence processing to disambiguate the indirect speech acts.

Consider again the texts (i) and (j) from the beginning of this chapter.

(i) The *student* went to the *biology* class. The *student* needed a *book* for the *quiz*. The *instructor* asked the *student*, "Do you have a *book*?"

(j) The *student* went to the *biology* class. The *student* needed a *pencil* for the *exam*. The *student* asked the *instructor*, "Do you have a *pencil*?"

For one of the five simulation pairs, both of these texts were part of the testing set of texts; thus, they were not used during the training of the network. When subjected to these texts, the trained networks produced the output activations for selected units of the *current proposition* layer as shown in Figures 7.2 and 7.3.

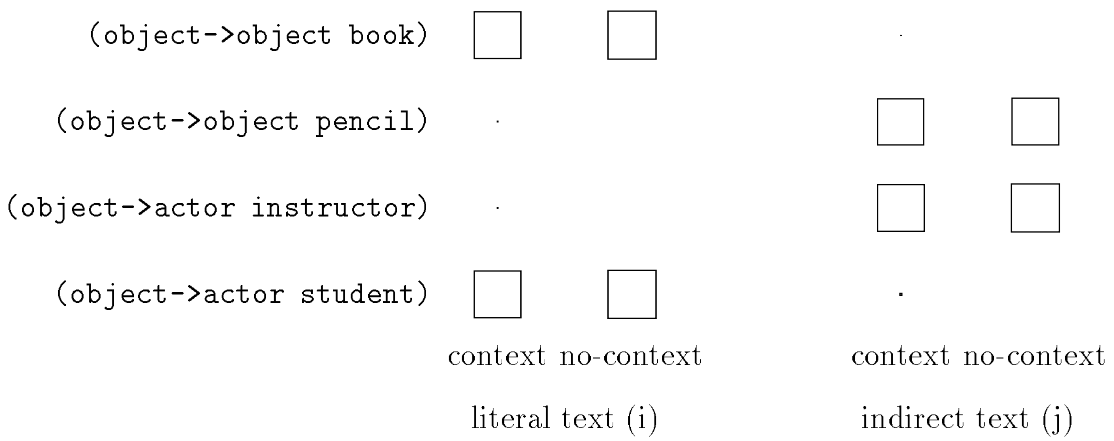

Figure 7.3: Selected *current proposition* layer output unit activations for some role/filler pairs in the literal and indirect speech act texts. Context is **not** necessary for the correct interpretation of these selected role/filler pairs.

The role/filler pairs of Figure 7.2 are those which are used to differentiate between the literal and indirect interpretations of the text ending speech acts. The `(illoc interrogative)` role/filler pair indicates a question is being asked while `(illoc request)` indicates a request. Similarly, `(object->pred atrans)` indicates the request to give while `(object->pred possess)` indicates the question of possession. The role/filler pairs of Figure 7.3 are the `object` and `actor` roles of the speech act itself and do not need context to determine their fillers, as indicated in the figure.

As can be seen in Figures 7.2 and 7.3, when context is present, all output units have the correct activation, for both the literal and indirect texts. However, when context is not present, PIP is unable to determine the correct activation for those four units which distinguish between the literal and indirect interpretation for either the literal or the indirect speech acts (cf. Figure 7.2). Not only is PIP unable to determine the correct activations for these units, it is unable to decide absolutely on which of these units should be fully active[1] and which should not. This implies that PIP cannot disambiguate these roles when context is not present.

Figures 7.4 and 7.5 present the average errors for all 47 output units in the *current proposition* layer for the final sentence in all texts in the testing set when context is present during intra-sentence processing and when it is not, respectively. The four most interesting of the output units are numbers 18, 19, 44, and 45; these are the `atrans/possess` and `request/interrogative` fillers of the `illoc` and `object->pred` roles of the *current proposition* layer. As can be seen in Figure 7.4 when context is present, all units have average error of near zero implying that PIP is able to determine the correct output activations for the speech acts ending the texts. (This is borne out by the near 100% accuracy shown in Figure 7.1 for the case when context is present.)

However, when context is absent, the four units identified above have average error near 0.5. (The 0.5 average error is due to the even distribution between literal and indirect texts. Also contributing to the 0.5 average error is the fact that every literal text has units 19 and 45 activated and no indirect text has these units activated and vice versa for units 18 and 44 and the indirect texts.) Again, PIP is unable to determine the correct activation for these units which distinguish between the literal and indirect interpretations of the speech acts.

Units 30 . . . 39, the other units having less than accurate output activations (cf. Figure 7.5), cover the `(from)` and `(to)` roles of the speech act clause in the representation for the final sentences of the texts. These activation levels can be explained by the fact that, for the literal interpretation, the predicate of the

[1]Full activation indicates a "selected" role/filler pair.

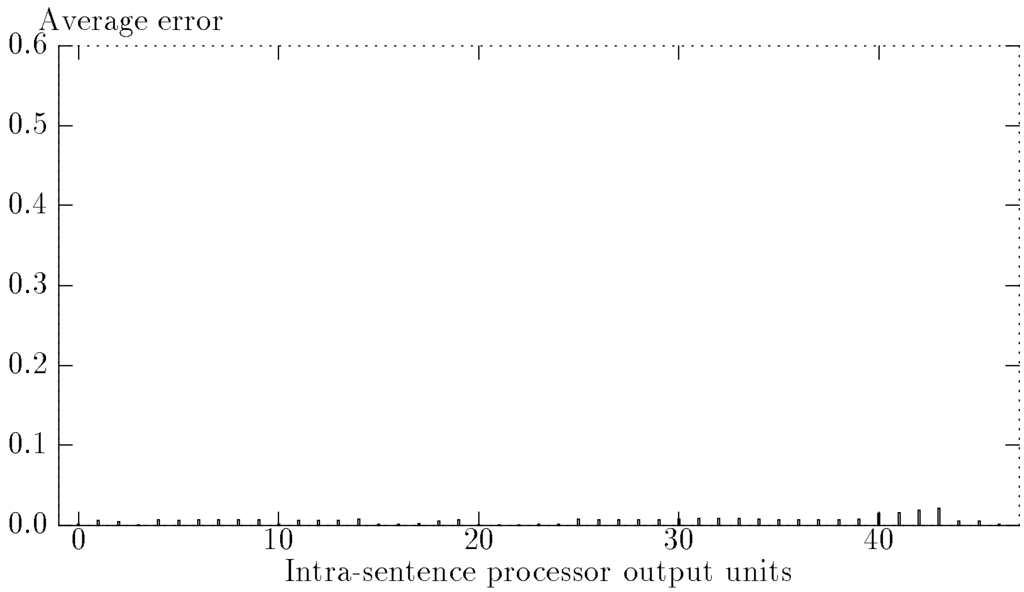

Figure 7.4: Average errors for all output units of intra-sentence processor for all stories when story gestalt context is present.

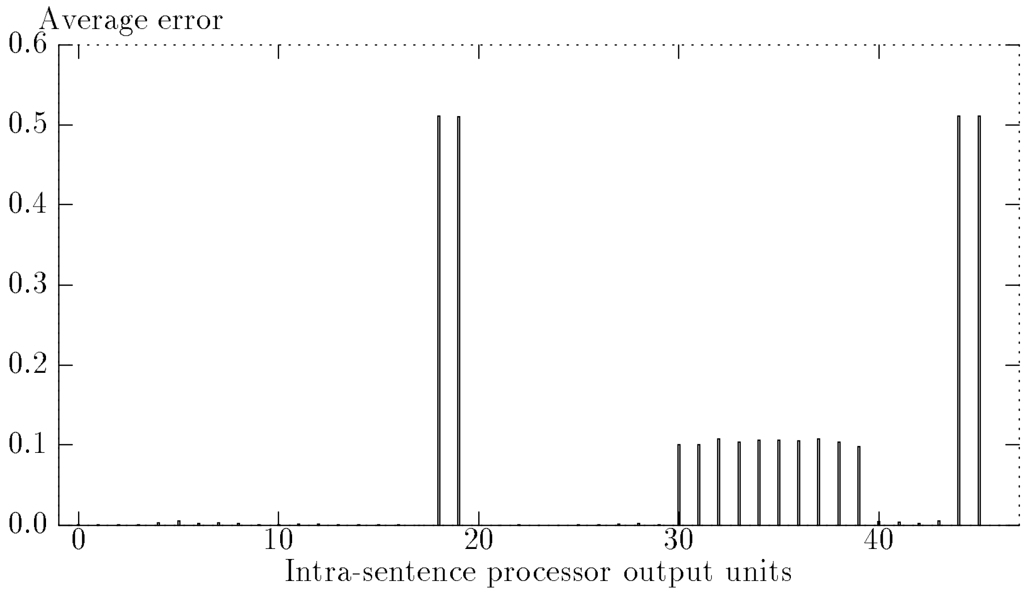

Figure 7.5: Average errors for all output units of intra-sentence processor for all stories when story gestalt context is **not** present.

representation is `possess`, which does not have `from` and `to` roles while the predicate for the indirect interpretation representation (`atrans`) does. Because no literal text has all of these units activated (only two of the units are activated for each literal text) and no indirect text has any of these units activated, error minimization in the back-propagation algorithm is achieved by making a "best guess" according to this distribution of target values for the texts in the training set. That is, minimizing the error over all texts of the training set implies that these units, although they will never be "correct" (because the story gestalt context is not present), will have minimum error. Remember, PIP attempts to minimize the error of all activation units (the error being the difference between the target activation and the actual activation). Thus, these units have a relatively low activation but do not have the "correct activation."

It might be interesting to look at the average error for the output units of the *current proposition* layer both when context is and is not present for the literal speech act texts independently from the indirect speech act texts. This data is shown in Figures 7.6, 7.7, 7.8, and 7.9. As can be seen, there is no difference when the literal and indirect speech act texts are separated when compared to all texts. This applies both to the case when context is present and when context is absent. Thus, the type of sentence, either literal or indirect, has no effect on the outcome of PIP's ability to interpret correctly the final sentences of the texts.

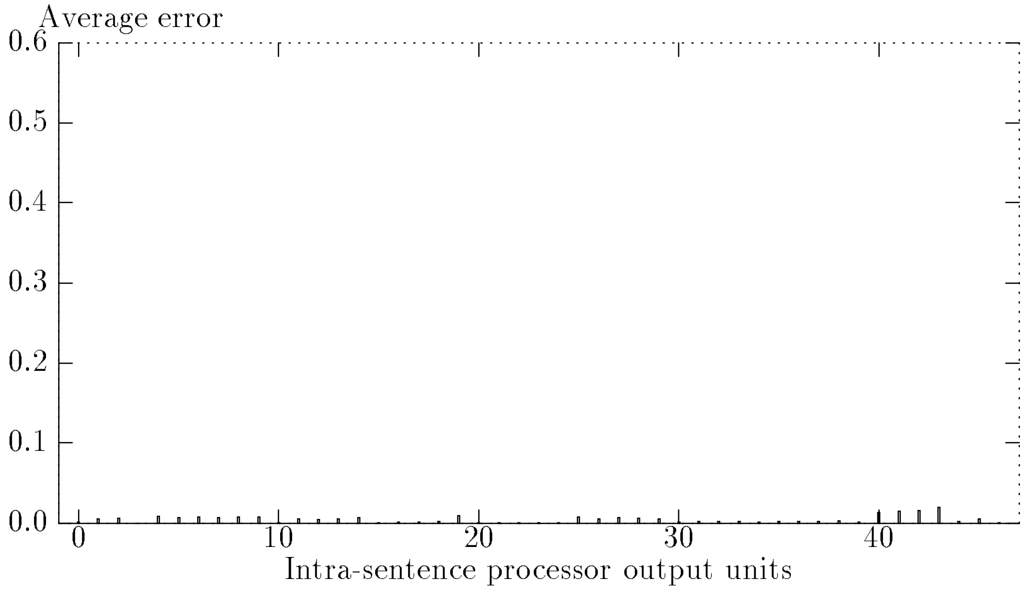

Figure 7.6: Average errors for all output units of the intra-sentence-processor for literal texts when the story gestalt context is present.

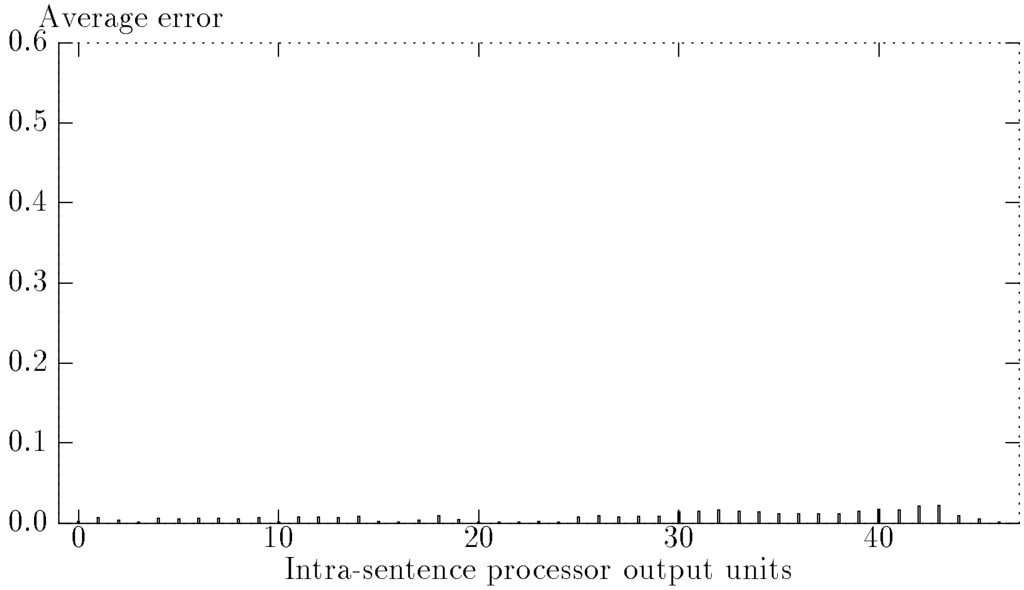

Figure 7.7: Average errors for all output units of the intra-sentence-processor for indirect texts when the story gestalt context is present.

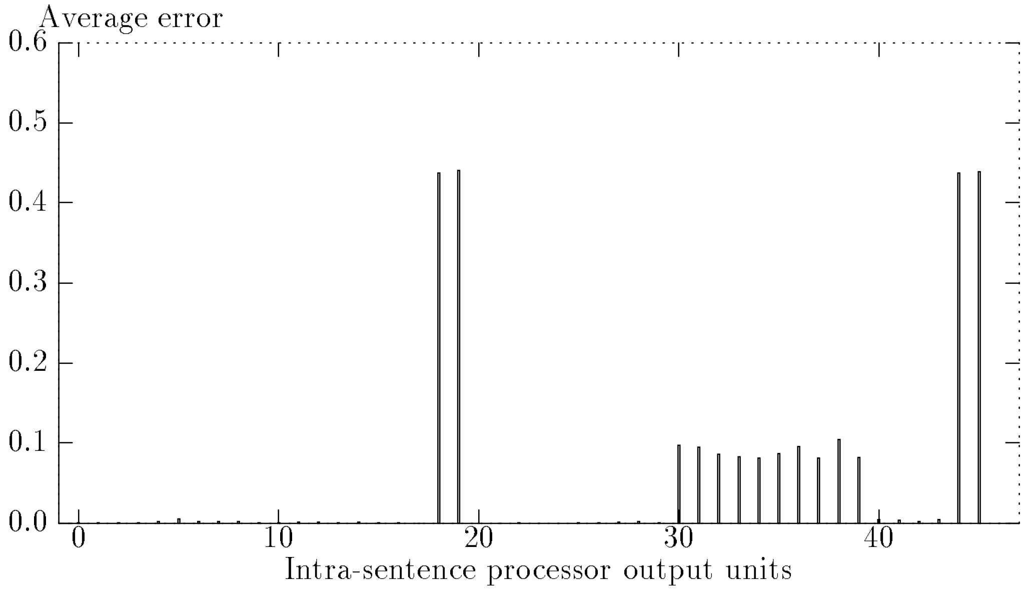

Figure 7.8: Average errors for all output units of the intra-sentence-processor for literal texts when the story gestalt context is **not** present.

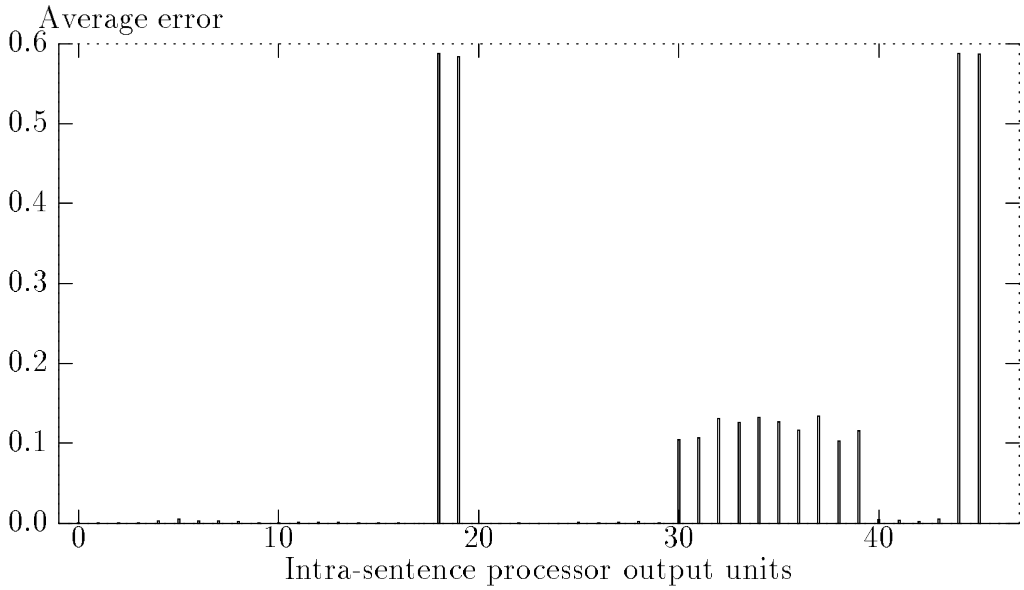

Figure 7.9: Average errors for all output units of the intra-sentence-processor for indirect texts when the story gestalt context is **not** present.

7.3 Conclusion & Transition

This chapter has presented evidence that some form of a global context, such as that used by PIP, is necessary to yield the correct literal or indirect interpretation of potential indirect speech acts. Together with the previous two chapters, PIP presents a unified model of the pragmatics of natural language understanding by incorporating the global story gestalt context during intra-sentence processing.

The next chapter will discuss a selection of other computational natural language processing systems, both symbolic and connectionist, and relate these systems to PIP. Some deal with the issues of ambiguity as described herein and others are related to PIP by virture of the fact that they are connectionist in nature.

Chapter 8

Other Work

Of course the first thing to do
was to make a grand survey
of the country she was going to travel through.

—LEWIS CARROLL, *Through the Looking Glass*

Natural language processing has a long and rich history in computer science and artificial intelligence. Early research on machine translation looked promising (Bar-Hillel, 1960) yet proved more difficult than originally envisioned. Turing (1950) argued that within fifty years, "machines" would be able to pass the imitation game, a game which has since come to be known as the Turing test. It is almost fifty years since Turing first proposed this test and there are no machines poised on the horizon which appear up to the challenge of the Turing test.

This is not to argue that the search for computational approaches to natural language processing is futile. Many impressive systems have been constructed dealing with parts of the problem, e.g., SAM (Cullingford, 1978) and FRUMP (DeJong, 1979). The field has fragmented over the years with research being conducted in specific areas of natural language processing such as knowledge representation (Minsky, 1975; Schank, 1973), syntax analysis (Woods, 1970), and idioms (Zernik, 1987), for example. This chapter presents a discussion of some of the other computational approaches to natural language processing. The first section discusses some symbolic processing models which address issues of ambiguity and compares these models to PIP. The second section discusses other natural language processing systems which use the artificial neural network model.

8.1 Symbolic Models of NLU on Ambiguity

Many theories exist on methods for handling ambiguity during natural language processing. Some of the symbolic computational models implementing these theories are discussed in this section. With the exception of MIDAS, these systems follow the two-stage approach to processing "errors" of understanding: find an initial interpretation; determine that that interpretation is in error (i.e., is inadequate, inappropriate, etc.); submit the initial interpretation to further processing yielding a final and correct interpretation. The differences of these approaches are found in the second stage of the two-stage processing model (cf. Section 2.2): How does one determine that an error has occurred during processing? Once it is determined that an error has occurred, how does one correct that error?

Given that the approach to processing errors of understanding as outlined above is correct, the questions posed would be valid questions to ask. PIP questions the correctness of the general algorithm outlined in Section 2.2 and takes a different approach to processing text; thus, the questions are not meaningful to PIP.

Table 8.1: Some conceptual categories used by QUALM.

conceptual category	explanation/example
verification	yes/no questions Can you go to the store?
concept completion	the concept is not fully specified What are you doing?
instrumental/procedural	How will we eat out tonight?
expectational	Why don't you go home?
specification	further specification is needed – don't answer "chancellor" Who is the chancellor?

8.1.1 Lehnert — QUALM

Lehnert (1978) presents QUALM, a computer model of human cognition as applied to the domain of question answering. QUALM is able to "answer" several different types of questions, including some conveyed by indirect speech acts, about stories that have been previously processed. Story processing is carried out by two other programs: SAM (Cullingford, 1978) and PAM (Wilensky, 1983). Questions are parsed from English using ELI (Riesbeck, 1975) yielding a literal interpretation; ELI, and thus QUALM, use the conceptual dependency (CD) representation of Schank (1973). An inferencing process applies rules to the literal interpretation, sometimes yielding an alternate (or indirect) interpretation of the question. The final process of QUALM is a search through its memory (the story representation as generated by SAM or PAM) for an answer to the question.

As they are initially processed, questions are assigned a *conceptual category* by the parser. Table 8.1 shows those conceptual cagetories related to indirect

Criteria:
1 Conceptual category = Verification
2 Question concept is of the form:
```
(POSSBY ACTOR (?H)
        OBJECT(?X))
```
?H is the person being addressed.
3 ?X is of little value.

Target Interpretation:
Conceptual category ← Request.
Question concept ←
```
(ATRANS ACTOR (?H)
        OBJECT(?X)
        FROM(?Y)
        TO(?Z))
```
?Z is the person asking the question.

ATRANS-Request Conversion Rule

Criteria:
1 Conceptual category = Verification
2 Question concept is of the form:
```
(POSSBY ACTOR (?H)
        OBJECT(?X)
        TIME (*PRESENT*))
```
?H is the person being addressed.
3 The common function associated with ?X can be "easily executed."

Target Interpretation:
Conceptual category ← Request.
Question concept ←
(DO_1 ACTOR (?H))
DO_1 is the instrumental script commonly associated with ?X.

Function-Request Conversion Rule

Figure 8.1: Conversion rule examples of QUALM.

questions. The conceptual categories aid the memory search module in finding the "proper" answer to the question. The conceptual categories are also used by the inferencing mechanism to help determine if a conversion rule should be used to reinterpret the question. The conversion rules contain criteria (patterns to match, including a conceptual category) and a "target" (indirect) interpretation (cf. Figure 8.1). If the criteria for a particular rule are satisfied, the target interpretation is used as the intended interpretation for the question. Conversion rules therefore convert the literal interpretation of a question into the intended, indirect interpretation.

For example, assume QUALM is asked Do you have a match? As part of the initial literal interpretation, this question will be given the conceptual category of *verification* by the parser. This initial interpretation generated by the parser will match the criteria (pattern) for the ATRANS-Request conversion rule (Figure 8.1). The inference process will use this rule to reinterpret the question as an ATRANS request as indicated in the figure. That is, QUALM has created an initial literal interpretation of the question via use of ELI. Then, by using its conversion rules, QUALM has reinterpreted the literal question giving the indirect interpretation.

Lehnert distinguishes between contextual inferences, context-independent inferences, and knowledge-state-assessment inferences. She classifies the ATRANS-Request conversion rule as being a contextual inference. This seems a rather strange classification as it is not context dependent at all; the ATRANS-Request conversion rule is a general rule applicable in any (present time) context with the only constraint being that the object be of "little value." There is no contextual information being used in the rule. Thus, questions such as Do you possess a match? would be interpreted indirectly since they match the criteria for the ATRANS-Request conversion rule. Furthermore, the example given above (Do you have a match?) could have a literal interpretation given an appropriate context or it might be interpreted indirectly as in Please light my cigarette, especially if match is replaced by lighter. The argument about lack of use of context applies not only to the contextual inference rules she identifies, but to all of QUALM's conversion rules. The conversion rules are general default rules that may lead to correct interpretations of indirect questions assuming a proper context.

QUALM thus takes a literal interpretation of a question and applies conversion rules by using, in part, the conceptual category of the question as necessary, to derive a final intended interpretation of the question. There are three major problems with this approach. First, purporting to be a cognitive model, it follows the two-stage model to process ambiguous text; thus, it does not account for the psychological evidence as described in Chapter 2. Second, the rules are fixed and cannot be easily expanded; they are ad hoc and must be hand coded into the system. Third, the conversion rules do not consult any context in order to accomplish their conversions. PIP sufferes from none of these problems.

8.1.2 Allen and Perrault

A strategy for processing indirect speech acts which utilizes context explicitly is provided by Searle (1975) where the plans and goals of the speaker and hearer are deeply analyzed yielding an interpretation of the indirect speech act. Allen and Perrault (1980; Perrault & Allen, 1980) have implemented this concept in a computer model which acts as an information desk attendant in a train station; that is, the context under which the system operates is that of an information desk attendant which fields "questions" from a user (or client) and attempts to answer these questions with meaningful responses via a deep analysis of the questions.

In analyzing a question, the program employs both a top-down and a bottom-up strategy; thus, it is inherently different from QUALM. Working in both directions and acting as the information desk attendant, it looks for a plan connecting the observed actions (the speech acts or questions) and the inferred goals of the speaker.

If the effect of `act` is `e`
    and `actor` wants `act`
  then `actor` wants `e`.
	[Action-Effect Rule]

If the precondition of `act` is `p`
    and `actor` wants `p`
  then `actor` wants `act`.
	[Precondition-Action Rule]

If `actor` wants to know whether or not `prop` then `actor` wants `prop`.	[Know-Positive Rule]

Figure 8.2: Example rules of Allen & Perrault.

There are two assumed potential goals for the speaker: the speaker seeks information on either train arrivals or train departures. Search takes place through a space of partially completed plans via a best first search strategy; i.e., as the planning process takes place, the "best" current partial plan is chosen for further elaboration. Various heuristics are used during this planning process to limit the search.

Expansion of a partial plan occurs by applying rules which match the "horizon" of the chosen partial plan, that part of the plan which cannot yet be realized (Figure 8.2). These rules are meant to indicate plausible inferences, not logical deductions. Once a (partial) plan has been created, the program seeks to identify *obstacles*; obstacles are defined as parts of the plan which have not yet been completely specified, that is, goals which have not yet been achieved.

For example, assume that the current query (or observed indirect speech act) is: Do you know when the Windsor train leaves? The planning process would find, in the partial plan it creates, an obstacle noting that the client (speaker) lacks information about when the train departs the station. The program would then supply this missing information by applying its planning process. An example of the planning process follows for this example (cf. Allen & Perrault, 1980). The goal derived from the literal interpretation is that the client (C) wants to know whether or not the information attendant (IA) knows the departure time for the Windsor train. Applying the know-positive rule yields the goal that IA knows the departure time. This enables IA to inform C about the departure time via the precondition-action rule; knowing is a precondition to informing. Finally, this achieves the goal that C knows the departure time by the action-effect rule. IA recognizes (or infers) that this final goal has not been achieved and supplies the answer to the indirect request.

This system thus interprets ambiguous indirect speech acts (or questions) by finding *obstacles* (the "error" in processing) and resolves these obstacles by applying a planning process. Once the planning process is complete, the information needed to answer the question is contained in the plan and is extracted therefrom to answer the question. As with QUALM, this reinterpretation scheme does not account for the psychological evidence supporting the one-stage processing model and again, there is no learning mechanism. Schulenburg and Pazzani (1989) present a way in which explanation based learning (DeJong & Mooney 1986; Mitchell, Keller, & Kedar-Cabelli, 1986) methods are applied to the plan processing model of Allen and Perrault to produce QUALM-like rules. This system overcomes the learning problems but not the reinterpretation problem.

8.1.3 Martin — MIDAS

Martin (1990, 1992) describes his approach to processing text involving *conventional* metaphors in the MIDAS system. Unlike other symbolic approaches to processing ambiguous text, Martin applies the results from the psycholinguistic community regarding the amount of time people use to process metaphoric language stating that:

> The basic result that will be used [from the psycholinguistic literature] is that the time needed to process various kinds of non-literal language does not differ significantly from the time taken to interpret direct language in the appropriate context. (Martin, 1990)

As will be discussed, the key phrase here is "in the appropriate context." The other constraint on processing text is that the mechanisms underlying the processing task itself should be similar for both literal and non-literal (i.e., metaphor) text. The interpretation process occurs in "parallel"; MIDAS generates *all* plausible interpretations of the text, be they literal or metaphorical.

MIDAS uses an extremely knowledge-intensive approach to the resolution of conventional metaphors. It explicitly represents the associations between the target concepts (the metaphors) and the source concepts (the intended underlying meaning of the metaphors or those concepts upon which the metaphor is built). Thus, no exhaustive search need be undertaken to do this mapping as in analogical approaches (Gentner, Falkenhainer, & Skorstad, 1988; Holyoak & Thagard, 1989).

Interpretation of text proceeds by first generating a primal representation (Wilensky, 1989). This representation captures the basic syntactic and semantic knowledge of the text and includes no application of constraints. It is this primal representation that is subjected to further processing to gain potential plausible interpretations, be they either literal or metaphorical. Since it is the first step for both literal and non-literal processing, the generation of the primal representation does not violate the constraint that processing mechanisms be similar for both literal and non-literal text and thus, it is not considered to be an initial interpretation. In continuing its interpretation process, MIDAS will consult its hierarchy of concepts according to the data found in the primal representation considering, in parallel, all of those which apply. This deeper conceptual analysis will yield all interpretations given MIDAS's knowledge base of concepts. For potential literal interpretations, a process of *concretion* will eliminate those interpretations which do not satisfy the constraints. A similar process called *unviewing* will do the same thing for the metaphorical concepts yielding all applicable metaphorical interpretations.

MIDAS was designed to process conventional metaphors. Since it considers all potential interpretations, one would expect that general problems of lexical ambiguity would not be that difficult for MIDAS to overcome. Indirect speech acts, on the other hand, would greatly tax MIDAS's processing capabilities; with its described mechanisms, MIDAS would be unable to generate a non-literal interpretation for an indirect speech act such as What are you doing? as seen in Section 1.1.

As presented throughout the discussion, MIDAS does not use any context during the interpretation process. As shown in this dissertation, this, in and of itself, is a potential problem for MIDAS. However, since it generates all plausible interpretations, be they literal or metaphorical, it might appear that MIDAS does not need to reference any context during its interpretation process. Unfortunately, this is not the case, as Martin (1992) states that "ultimately, however, the choice among otherwise coherent interpretations must be left up to a mechanism that has access to the wider discourse context." Thus, MIDAS falls into the trap of requiring a need for an additional processing phase to determine which of the generated plausible interpretations is correct.

8.1.4 Small — WEP

The Word Expert Parser (Small and Rieger, 1982), or WEP, is a model of natural language processing that is "word centric." That is, (almost) all knowledge used during the parsing task is found in "word experts." As Small and Rieger (1982) put it,

> This model's structure rests on our hypotheses that: (1) human knowledge about language is organized primarily as knowledge about words rather than as knowledge about rules; and (2) language understanding is largely the coordination of information exchange among word experts as each diagnoses its own involvement in its linguistic and conceptual environment.

The word experts set up a discrimination-net (Charniak & McDermott, 1985). In the case of the WEP, the discrimination-net takes the form of a set of demons, each being associated with a word expert; thus, word experts can and generally do have multiple demons. The use of the demons categorizes the WEP as an expectation-based parser. As a word expert is activated, its initial demon is activated; this initial demon decides what happens next. The actions of demons can involve such things as activating other demons or word experts, proposing concepts, posting messages onto a "blackboard" (a process known as signaling), looking for messages on the blackboard, looking for specific lexical items, and suspending themselves. Processing continues activating demons and performing the associated actions until the entire sentence has been processed.

Small and Rieger view the WEP as the antithesis of rule-based systems for natural language processing. However, it is a small step to view the word experts in terms of rules which could be fairly easily implemented in a forward chaining production system such as OPS5 (Brownston, Farrell, Kant, & Martin, 1985). Although it is quite rudimentary, the WEP does have a context element. When context is necessary to help disambiguate, the WEP as described will query the user to supply the desired information; however, Small and Rieger do speculate on the automation of this process as part of future work but do not explain how it might be accomplished.

As a model of lexical ambiguity resolution, the WEP can perform quite well. Consider that, because virtually all parsing knowledge is contained within the word experts themselves and that the overall model is expectation based, resolution of lexically ambiguous words can wait until enough information has been obtained to effect the disambiguation. That is, once a word expert is activated, all potential meanings of that word can become activated by activating the demons associated with the different meanings. As these demons fire (or die), the desired meaning manifests itself as a result of the actions of the fired demons. Since all potential meanings can be activated when a word expert is initially fired, the WEP thus follows the Post Decision Hypothesis (see Section 2.1).

How would the WEP be able to handle the ambiguity inherent in metaphors and indirect speech acts? Presumably, all potential meanings, including those metaphorical and indirect, would have to be supplied in the word experts. This is unrealistic due to the novelty found in these lexical constructs. An alternative would be to fall back on the standard approch to processing these ambiguity problems as Small and Rieger (1982) state that "certain information deriving from the meaning of the input sentence must still be inferred in higher-order mechanisms." These higher-order mechanisms would comprise the post-processing of systems such as QUALM.

8.2 Connectionist Approaches

Most of the connectionist approaches to natural language understanding use a spreading activation or marker passing model of processing (e.g., Cottrell & Small, 1983; Eiselt, 1989). Although promising, there have been relatively few natural language processing systems constructed using the feed-forward connection model; this is due mainly to the relatively recent introduction of the generalized delta rule (cf. Section 3.1) allowing the training of non-linear networks. Two of these systems, DISCERN (Miikkulainen, 1990) and the STORY GESTALT (St. John, 1992), are closely related to the current work and are discussed in depth in sections 8.2.1 and 3.3, respectively. Other systems are briefly presented.

8.2.1 Miikkulainen — DISCERN

DISCERN (Miikkulainen, 1990) is a feed-forward connectionist system designed as a model of script processing (Schank & Abelson, 1977) and memory. As a model of script processing, it suffers from the same drawbacks from which SAM

(Cullingford, 1978) suffered; namely, due to its representation limitations, it can process only script type texts. Yet, it is one of the first *text* processing systems (as opposed to sentence processing) to use the artificial neural network model.

DISCERN is able to process script based texts such as

(k) John went to MaMaison. John asked the waiter for lobster. John left a big tip.

In processing such a text, DISCERN uses two separate networks, each configured as a simple recurrent network (Elman, 1990): the *sentence parser* and the *story parser*. The sentence parser (basically, the same as PIP's intra-sentence processor) processes the individual words of a sentence producing a case-role (or role/filler) representation of the sentence as an activation pattern in the output layer of the network. This sentence representation is then passed to the story parser network (serves a similar role to PIP's inter-sentence processor) which builds a "stationary slot-filler representation" of the story (basically, another role/filler representation, this time for a script) of the text, filling in the roles of the script as more information is obtained from the sentence parser. Other than the connection between the output activations of the sentence parser and the input activations of the story parser, there are no other connections between these two networks; they train and process data independently.

DISCERN also has two other capabilities addressing issues of text processing: two separate networks are used to produce a paraphrase of a scriptal text and two separate networks are used to answer questions about a text. Underlying these capabilities is an *episodic memory* component. A *story generator* network can produce a sequence of case-role representations describing a scriptal text given a representation for the entire text which is obtained from *episodic memory.* Each case-role representation can then be given to the *sentence generator* which produces a sequence of words (i.e., a sentence); this process is the opposite of parsing the sentences. Again, these are independent networks with the only connection being that the output of the story generator is fed as input to the sentence generator. Yet two more independent networks are used by DISCERN to answer questions about some text: the *cue former* takes as input the case-role representation of a question (actually, this would be the output of the sentence parser) and produces a representation of the text which is obtained from *episodic memory.* The *answer producer* network uses the output of the *cue former* along with the case-role representation for the question and produces, as its output, a case-role representation of the answer to the question, which is, in turn, passed on to the *sentence generator* to produce the actual answer as English text.

Episodic memory is constructed from the output of the *story parser* and has a different structure from the feed-forward back-propagation model used by the other components of the system. The structure used by DISCERN for episodic memory is based on self-organizing feature maps (Kohonen, 1984). Furthermore, these feature maps are organized into a hierarchy or pyramid like structure. At the top-level is a feature map for the various scripts which DISCERN is able to process; each unit represents one of the script types. As a story is processed by episodic memory, the appropriate unit in this top-level feature map is activated. This, in turn, will limit the choice of which feature map to activate in the next level, depending on the particular script track. As the story representation is processed by episodic memory, the role/filler pairs which are used for selection of the correct unit at each level of the hierarchy are filtered out and not passed along to the next level. For example, at the script-level, before sending a story representation to the track-level, the script role itself is filtered out. At the track-level, the track role is filtered out and the rest of the role/filler pairs are sent to the role-level where the different episodes themselves are represented. Thus, each individual episode has its own representation unit in the role-level.

Although PIP and DISCERN have very similar architectures for processing text, they address different issues. As Miikkulainen (1990) states, "Our goal is to model natural language understanding in the same sense as the symbolic artificial intelligence models do." Thus, DISCERN was designed to mimic the behavior of existing symbolic models of language understanding by using artificial neural networks. PIP, on the other hand, was designed as a uniform approach to processing semantically ambiguous language constructs by the use of a global context. Notwithstanding the limitations of DISCERN's scriptal representation (from which PIP does not suffer due to the use of the STORY GESTALT), it would be unable to process the texts involving ambiguity such as those described herein. There is no context component available to DISCERN's sentence parser. When DISCERN is training its sentence parser, it uses the entire representation of the sentence as the target output during back-propagation. This contrasts with PIP's incremental filling of the target output based on the words of the sentence already processed, a seemingly more plausible approach (cf. Schulenburg, 1992).

8.2.2 Allen — CLUES

Allen (1990) discusses a series of experiments on language use in the CLUES (connectionist language users) system. A few of these experiments are related to

PIP in that they share similar tasks. However, it should be noted that CLUES does not deal with true natural language input; rather, the input is "language-like." Furthermore, with the exception of one case as described below, CLUES does not deal with multiple sentence texts.

In many of the experiments, CLUES deals with a context in order to aid its processing of the language input; however, it is a very different type of context from that used by PIP. CLUES's context is a simulated perceptual view of a micro world which identifies different objects with different features in that micro world. That is, CLUES uses a (localist) representation for different objects that are currently "in view" by activating the assigned units for the objects in its input layer.

One of the tasks of CLUES is to answer questions about the objects perceived in the micro world. The input layer contains a "question" and a description of two objects in the micro world. The question is formed by a command indicating that object in the micro world to be identified. E.g., left indicates *identify the object on the left*. The output layer answers the question by identifying the object. CLUES supplies enough information about the targeted object to distinguish it from the other object according to Grice's (1975) maxims of conversation. For example, if the two objects are of different categories, the category identifying the object is presented; if the objects are of the same category, the object is given; and if the two objects are the same, a distinguishing feature is given. Thus, CLUES is able to exploit the (perceptual) context to answer a question. In contrast, PIP *creates* a context by learning to answer questions correctly.

Another of the experiments addresses issues of anaphora. Before that discussion takes place, however, the general architecture of the CLUES network needs explaining. As with PIP, CLUES uses a hybrid of the Jordan net (Figure 3.2) and the simple recurrent network of Elman (Figure 3.3). The recurrent relation comes from the hidden layer as with the simple recurrent network; yet, that context also maintains and includes the previous state as in the Jordan network. In normal processing of sentences, the state is reset to zero activation before the sentence is processed. Now, back to anaphora. This task uses multiple sentence texts where the last sentence uses a pronoun to refer to some noun found in the earlier sentences. For example, the following two "sentences" are taken from Allen (1990): Is *objectX*$_1$ present? and Is it *fX*$_1$? where *objectX*$_1$ would be replaced by some object from the domain and is the referent of the pronoun it and *fX*$_1$ would be replaced by some feature of the object. A concrete example is Is car present? and Is it blue? where car replaces *objectX*$_1$ and blue replaces *fX*$_1$. The goal is to answer each question correctly and, for the second question, by using the referent of the pronoun it, to determine the correct answer. While CLUES succeeds at this task, there are a couple of points to note. First, the network architecture of CLUES allows for no global context. That is, under normal operation, the state is reset at the completion of sentence processing. This would cause a problem for the current task because CLUES would lose all information regarding the referent of the pronoun. This problem is solved by not resetting the state upon completion of the processing of the first sentence. Thus, CLUES uses a non-uniform solution for processing multi-sentence texts. The second point to make here is that in this experiment, as with all of those described by Allen, the data on which the experiment is run is divided into training and testing sets with the size of these sets heavily favoring the training set. Typical training sets are approximately 97% to 98% of all of the data leaving two to three percent for testing. With this percentage mix, it is difficult to determine just how well CLUES is able to generalize.

The general network architecture used by CLUES is very similar to those used by DISCERN, the STORY GESTALT, and PIP, the other connectionist models of language processing discussed here. Allen chose to address issues very different from the text processing issues of the other systems. Yet, his results are germane in that they point out some of the limitations and potential applications of low-level language-like processing. Finally, none of these tasks actually produce a representation of any input text; rather, they all seem to address side issues of language understanding such as question answering. The underlying questions of Allen's work are exactly how well would CLUES be able to generalize given less skewed cardinality of the training and testing sets of data and would the success of the experiments scale when applied to true natural language input?

8.2.3 St. John — STORY GESTALT

Section 3.3 discusses the details of the STORY GESTALT. There are two main differences between the STORY GESTALT model and PIP. The first is the obvious lack of any type of sentence processor in the STORY GESTALT. St. John argues that the SENTENCE GESTALT model could be added as a front end to the STORY GESTALT. While this would expand the direct processing capabilities of the STORY GESTALT, the resulting model would still lack the use of any type of context to aid the SENTENCE GESTALT in its processing task. The second major difference, as noted briefly in Section 3.3, is the form of the "question" as used by the STORY GESTALT. By limiting a "question" to include only the predicate associated with its role/filler representation, the "question" itself is limited in its expressive ability. For example, the two questions Who decided to go to the park? and Where did

Andrew and Roxanne decide to go? would both be respresented by the same "predicate as question" (e.g., `decided-to-go`) even though they are distinct, unique questions. Thus, the STORY GESTALT cannot ask these two distinct questions; rather, it is limited to a single probe representing all possible questions having `decided-to-go` as their predicates. Furthermore, once a specific predicate has been used for one "question," it can not be used again to pose another question with possibly different role fillers. Since these questions are meant to elicit representations for specific sentences from the gestalt, this implies that any text being processed can have only one sentence for each unique predicate. That is, a predicate can be used only once per text. Because PIP uses a representation for the entire question, it suffers from neither of these problems (cf. Section 4.1.2).

8.3 Conclusion & Transition

This chapter has discussed a number of other artificial intelligence systems designed for natural language processing. The symbolic approaches of Section 8.1 address issues of ambiguity in text. Those which handle context do so with a set of *ad hoc* rules which are applied after an initial literal interpretation is deemed inadequate. The exception to this general processing scheme is MIDAS; although it does have its set of rules, it follows a different outline of processing, one similar to that used for text in general. The main capability lacking in the connectionist models of Section 8.2 is that none can deal adequately with the context of the text being processed.

The next, and final chapter, will present an evaluation of PIP, highlighting its accomplishments, and a discussion of ways in which PIP might be extended to deal with yet other problems in the natural language understanding domain.

Chapter 9

The Present and Future of Pip

"What do you mean by that?" said the Caterpillar sternly.
"Explain yourself!"

"I ca'n't explain *myself*, I'm afraid, Sir," said Alice,
"because I'm not myself, you see."

"I don't see," said the Caterpillar.

"I'm afraid I ca'n't put it more clearly," Alice replied very politely,
"for I ca'n't understand it myself to begin with ..."

—LEWIS CARROLL, *Alice's Adventures in Wonderland*

PIP is not the final answer to the problem of creating a computational model of natural language processing. Yet, it is capable of exhibiting behavior of which other such computational models are incapable, behavior which, although not sufficient, is necessary to achieve the final goal of having a true "artificial intelligence" capable of understanding unrestricted natural language input. This behavior is the ability to process ambiguous text such as metaphors and indirect speech acts in a uniform manner and in a way which accounts for some of the psychological data addressing these issues. Thus, PIP is presented as a cognitive model of natural language processing.

9.1 An Evaluation of PIP

Chapter 1 introduced some problems concerning natural language processing, indicating that these are the problems which PIP would address. Has PIP adequately addressed these issues? The answer is both "yes" and "no."

9.1.1 What does PIP do?

The major contributions of PIP are threefold:

- PIP has the ability to incorporate a "global context" during the understanding process of natural language.
- PIP's model of the incorporation of context accounts for some of the psychological data dealing with the processing of metaphors and indirect speech acts.
- PIP presents a unified approach to dealing with the problems of lexical ambiguity, metaphor resolution, and indirect speech act interpretation.

PIP's ability to incorporate a global context (a context developed from previously processed text) directly (without postprocessing) is unique among computational models of natural language understanding. As can be seen from the data presented in Chapters 5, 6, and 7, in order to interpret correctly certain texts exhibiting lexical ambiguity, metaphors, and indirect speech acts, context must be available in some form at some time during the understanding process. PIP takes the view that the incorporation process should be done during an initial (and only) processing phase while most other computational NLU models incorporate context during a postprocessing phase.

PIP's view of a one-stage processing model lends support to the claim that PIP is a cognitive model of language understanding. Chapter 2 discussed some of the psychological evidence regarding the processing of metaphors and indirect speech acts. This evidence supports the notion that people do not process metaphors and indirect speech acts in a multi-stage manner. That is, people apparently do not produce a literal interpretation of some text, determine that that interpretation is "faulty" in the current context, and reprocess that initial literal interpretation to form a final true interpretation. Rather, people seem to be able to determine the final true interpretation directly. PIP basically follows the framework that people use.

Not only does PIP's processing model present a uniform approach to language understanding applied to the problems of lexical ambiguity, metaphor resolution, and indirect speech act interpretation, the same approach is used to process "regular" (or literal) language. PIP does not need ad hoc, special purpose machinery to process language.

PIP has demonstrated that it is possible to create a uniform computational model of language understanding which has the ability to resolve the ambiguities found in texts containing lexical ambiguity, metaphors, and indirect speech acts. Fruthermore, it is evident by the results presented in chapters 5, 6, and 7, that the use of context is necessary to resolve these types of ambiguity.

9.1.2 What does PIP not do?

PIP's ability to understand novel metaphor is less than ideal. The main reason for this is its current inability to draw on the underlying meaning of the metaphor. PIP would have difficulty with the following standalone example (from

Martin, 1990) because of this lack of the underlying meaning and because there is no context supplied by the text.

(1) How can I *kill* this process?

This, of course, refers to terminating a computer program. Resolving the *kill* metaphor here requires both some knowledge of the context in which the query was made and some access to the meaning underlying the word *kill*. PIP lacks in both of these; thus, processing standalone metaphors such as given in example (1) above would prove difficult for PIP.

Resolving pronoun reference in PIP is poorly tested. While the evidence presented in Chapter 5 suggests that PIP has some ability at processing pronouns, more work should be done here.

9.1.3 The Psychology of PIP

The psychological studies presented in Chapter 2 provided much of the motivation for PIP, both in terms of what it does and how it does it. How does PIP reflect the results of these studies?

PIP, as currently implemented, applies its context before lexical access takes place, thus restricting the potential meanings of lexemes that come to play. That is, PIP considers only a single meaning of the words. This is mainly due to the simplicity of PIP's lexical dictionary and not to any belief that the Prior Decision Hypothesis has more merit than the Post Decision Hypothesis. It is recognized that PIP's approach is less than optimal, given the current thought on the issue. In response, section 9.2 discusses the potential for the incorporation of a more sophisticated lexical dictionary model which would result in PIP's better reflecting the Post Decision Hypothesis model.

PIP follows the one-stage processing model for indirect speech act interpretation as supported by the work of and conclusions drawn by Gibbs. There are several reasons for this. First, by using the one-stage processing model, PIP uses the same mechanisms for processing indirect speech acts as it does for processing other types of text, such as those exhibiting metaphor and lexical ambiguity. Second, as the results in chapter 7 showed, there just is no need for special rules as required by the two-stage model. Third, the need for the rules of a two-stage model would make a learning system for natural language processing considerably more difficult.

The underlying theme is that context is necessary for accurate interpretation of text. Furthermore, the applicable question is: "Is a literal interpretation of some text a necessary prerequisite for a final, accurate interpretation?" PIP's answer to this question is "no," as supported by the psychological evidence and by its ability to process ambiguous text without the need to resort to an initial literal interpretation.

9.2 What Next?

There are, of course, many ways in which PIP could be improved. Briefly discussed here are future research directions which could be pursued concerning ways in which PIP could be extended.

Possibly the greatest improvement in PIP could be realized by a reorganization of the dictionary of lexemes and how these are represented. For example, PIP's localist representation of words is less than optimal. At some level, individual words do need to be represented in order to facilitate lookup; yet, in addition to the words themselves, PIP would benefit from a "meaning" of the words, possibly in another network layer placed between the current input layers and the hidden layer of the intra-processor (cf. Figure 4.1).

In conjunction with a more elaborate representation scheme for the dictionary of words, PIP would benefit from some type of prediction mechanism.[1] This would aid PIP in at least two ways: PIP would be able to perform a "sanity check" on the meaning that it chose. That is, if the prediction matched the interpretation given by PIP, more confidence would accrue that this interpretation is "correct." This sanity check could occur during the course of processing as well as after the final interpretation has been decided. And, in the event that PIP were given text with words not in its dictionary, it might be able to use the prediction mechanism to help infer a meaning for the unknown words.

The "local context" supplied by the words surrounding the current word of focus (i.e., partially activating the two words surround the current word) lacks any disciplined principle for its existence (cf. Section 4.1.1). Its incorporation into PIP was basically to provide a *Wickelfeature* mechanism (Wickelgren, 1969) and is similar in function to the decay of activation applied to words previously processed as used by other neural network models of language processing (e.g., Allen, 1990; Hadley & Hayward, 1994).

[1] Elman (1990) shows how the simple recurrent network might be used to predict sequences of characters in repeating strings.

A more "complete" model would have the output of the intra-sentence processor provide the question representations to the the inter-sentence processor, in a fashion similar to that of DISCERN (Miikkulainen, 1990). This would allow PIP actually to process the questions word-by-word, forming a more realistic model. Associated with this idea would be to replace the intra-sentence processor with a model similar to the SENTENCE GESTALT (cf. Section 3.3 and St. John & McClelland, 1990).

Finally, the global context provided by PIP only captures information that has been presented in the text during the understanding process; there is no "extra-textual context" (commonly referred to as *background knowledge*) available. This need relates to the memory component as found in DISCERN. Background knowledge is extremely important during natural language processing; its incorporation into the model would greatly improve PIP's ability to process language. For example, consider the use of kill in McEnroe killed Connors (Martin, 1990). Killed here is clearly intended as a metaphor for "soundly defeated." It is our background knowledge of who McEnroe and Connors are that leads us to this interpretation over the literal interpretation of "causing to die."

9.3 On the Novelty of Sentences

Hadley (1992, 1994) elaborates on objections originally presented by Fodor and Pylyshyn (1988) regarding the ability of connectionist networks to capture the *compositionality* and *systematicity* apparently needed to process natural language computationally. The objections arise from claims attributed to various connectionist natural language processing researchers (Chalmers, 1990; Elman, 1990; McClelland and Kawamoto, 1986; Pollack, 1990; St. John and McClelland, 1990; and Smolensky, 1990) that their systems are able to process *novel* sentences. The question is, just how novel are the testing sentences exactly?

Hadley defines three degrees of systematicity to describe the novelty of the sentences: weak systematicity, quasi-systematicity, and strong systematicity. Weak systematicity allows each word of a testing example to be used in the same syntactic position during training even though the testing example, taken as a whole, was never used during training. Quasi-systematicity requires weak systematicity along with the ability of the system to process sentences containing embedded sentences. Furthermore, a word in the embedded sentence is allowed to exist in some *simple* sentence of the training corpus in the same syntactic position where it is found in the embedded sentence. For example, Mary told Bill that John loves *Alice* and Bill loves *Alice* each have *Alice* as the direct object of the sentence. Strong systematicity requires the testing corpus to contain words in syntactic positions where those words are not found in the same syntactic positions in any sentence of the training corpus. The argument is made that none of the systems mentioned above exhibit strong systematicity and that with the exception of Pollack, they all exhibit weak systematicity at best.

PIP clearly exhibits only weak systematicity; it is able to process novel sentences which were not used during training yet which have shared words in the same syntactic positions with sentences of the training corpus. First, PIP was not designed to deal with embedded clauses; that was not the goal of PIP. Thus, it would certainly fail a quasi-systematicity test. PIP also was not designed to pass a test of strong systematicity. Does this imply that PIP is worthless as a model of computational natural language processing? Two answers suggesting ways in which PIP could be improved to deal effectively with strong systematicity are presented here. Even though Hadley specifically argues against the use of a distributed representation to satisfy strong systematicity (i.e., where words of the testing set share micro-features with words of the training set), this argument is specious. When people process novel sentences (sentences with "words" in syntactic positions never before seen), they bring into play various cues, such as syntactic position and expectations, in their efforts to understand these sentences. These cues could be represented as micro-features. For example, consider the use of the non-word grok by Heinlein (1961) in *Stranger in a Strange Land*. Most people ascribe to grok a meaning synonymous with understand, even though Heinlein explicitly states that grok does not mean understand. How is this meaning acquired? By syntactic, semantic, and contextual cues. Secondly, as discussed in section 9.2, incorporation of a prediction mechanism into connectionist systems dealing with natural language processing might be able to help these systems overcome the problems posed by the requirement for strong systematicity.

9.4 Wrap-Up

As discussed throughout the dissertation, PIP has demonstrated both the need for and ability to incorporate context during the language understanding process. Its approach is both unique and unified in the sense that it is able to process uniformly text with lexical ambiguity, metaphors, and indirect speech acts; no special "hardware" is required to do post processing of these types of text. Use of the basic ideas underlying PIP's architecture may well prove necessary to achieve the goal of a robust computational natural language processor.

References

Ackerman, B. P. (1978). Children's understanding of speech acts in unconventional directive frames. *Child Development*, **49**, 311-318.

Adriaens, G. & S. L. Small (1988). Word expert parsing revisited in a cognitive science perspective. In S. L. Small, G. W. Cottrell, & M. K. Tanenhaus (Eds.), *Lexical Ambiguity Resolution: Perspectives from Psycholinguistics, Neuropsychology, and Artificial Intelligence*. San Mateo, CA: Morgan Kaufman.

Allen, J. & C. Perrault (1980). Analyzing intention in utterances. *Artificial Intelligence*, **15**, 143-178.

Allen, R. B. 1990. Connectionist language users. *Connection Science*, **2**(4), 279-311.

Austin, J. (1962). *How to do things with words*. Cambridge, MA: Harvard University Press.

Bar–Hillel, Y. (1960). The present status of automatic translation of languages. In F. L. Alt (Ed.), *Advances in computers* (Vol. 1). New York: Academic Press, 91-163.

Brownston, L., R. Farrell, E. Kant, & N. Martin (1985). *Programming Expert Systems in* OPS5*: An Introduction to Rule-Based Programming*. Reading, MA: Addison-Wesley.

Carroll, L. (1960). *The annotated Alice: Alice's adventures in Wonderland & Through the looking glass*. New York: Clarkson N. Potter.

Chalmers, D. (1990). Why Fodor and Pylyshyn were wrong: The simplest refutation. *Proceedings of the Twelfth Annual Conference of the Cognitive Science Society*. Hillsdale, NJ: Lawrence Earlbaum Associates.

Charniak, E. & D. McDermott (1985). *Introduction to Artificial Intelligence*. Reading, MA: Addison-Wesley.

Clark, H. H. & P. Lucy (1975). Understanding what is meant from what is said: A study in conversationally conveyed requests. *Journal of Verbal Learning and Verbal Behavior*, **14**, 56-72.

Cottrell, G. W. & A. L. Small (1983). A connectionist scheme for modeling word sense disambiguation. *Cognition and Brain Theory*, **6**, 89-120.

Cullingford, R. E. (1978). *Script application: Computer understanding of newspaper stories.* (Research Report #116). New Haven, Connecticut: Computer Science Department, Yale University.

DeJong, G. F. (1979). *Skimming Stories in Real Time: An Experiment in Integrated Understanding.* PhD dissertation, Department of Computer Science, Yale University, New Haven, CT.

DeJong, G. F. & R. Mooney (1986). Explanation-based learning: An alternate view. *Machine Learning*, **1**, 145-176.

Eiselt, K. P. (1989). *Inference Processing and Error Recovery in Sentence Understanding.* PhD dissertation, Department of Information and Computer Science, University of California, Irvine.

Elman, J. L. (1990). Finding structure in time. *Cognitive Science*, **14**(2), 179-211.

Fodor, J. A., & Z. W. Pylyshyn (1988). Connectionism and cognitive architecture: A critical analysis. *Cognition*, **28**. 3-71.

Foss, D. J., & C. M. Jenkins (1973). Some effects of context on the comprehension of ambiguous sentences. *Journal of Verbal Learning and Verbal Behavior*, **12**. 577-589.

Garvey, C. (1975). Requests and responses in children's speech. *Journal of Child Language*, **2**, 41-63.

Gentner, D., B. Falkenhainer, & J. Skorstad. (1988). Viewing metaphor as analogy. In D. H. Helman (Ed.), *Analogical Reasoning.* Boston, MA: Kluwer Academic Publishers.

Gibbs, R. (1983). Do people always process the literal meanings of indirect requests? *Journal of Experimental Psychology*, **9**, 524-533.

Gildea, P. & S. Glucksberg (1983). On understanding metaphor: The role of context. *Journal of Verbal Learning and Verbal Behavior*, **22**, 577-590.

Gordon, D. & G. Lakoff (1971). Conversational postulates. *Papers from the Seventh Regional Meeting, Chicago Linguistic Society*, 1971.

Granger, R. H. (1977). FOUL-UP: A program that figures out meanings of words from context. *Proceedings of the Fifth International Joint Conference on Artificial Intelligence.* Cambridge, MA.

Grice, H. P. (1975). Logic and conversation. In P. Cole & J. L. Morgan (Eds.), *Syntax and Semantics, Vol. 3, Speech Acts.* New York: Academic Press.

Hadley, R. F. (1992). Compositionality and systematicity in connectionist language learning. *Proceedings of the Fourteenth Annual Conference of the Cognitive Science Society.* Hillsdale, NJ: Lawrence Earlbaum Associates.

Hadley, R. F. (1994). Systematicity in connectionist language learning. *Mind and Language*, **9**, 247-272.

Hadley, R. F. & M. Hayward (1994). *Strong Semantic Systematicity from Unsupervised Connectionist Learning.* Technical Report CSS-IS TR94-02, School of Computing Science and Cognitive Science Program, Simon Fraser University, Burnaby, B. C., Canada.

Heinlein, R. A. (1961). *Stranger in a Strange Land.* New York: Putnam.

Holyoak, K. J. & P. Thagard (1989). Analogical mapping by constraint satisfaction. *Cognitive Science*, **13**, 295-355.

Inhoff, A. W., S. D. Lima, & P. J. Carroll (1984). Contextual effects on metaphor comprehension in reading. *Memory & Cognition*, **12** (6), 558-567.

Jordan, M. I. (1986). Attractor dynamics and parallelism in a connectionist sequential machine. *Proceedings of the Eighth Annual Conference of the Cognitive Science Society.* Hillsdale, NJ: Lawrence Earlbaum Associates.

Keysar, B. (1989). On the functional equivalence of literal and metaphorical interpretations in discourse. *Journal of Memory and Language*, **28**, 375-385.

Kohonen, T. (1984). *Self-Organization and Associative Memory.* Springer-Verlag: Berlin.

Lehnert, W. G. (1978). *The process of question answering: A computer simulation of cognition.* Hillsdale, New Jersey: Lawrence Erlbaum Associates.

Martin, J. H. (1990). *A Computational Model of Metaphor Interpretation.* Boston, MA: Academic Press.

Martin, J. H. (1992). Computer understanding of conventional metaphoric language. *Cognitive Science*, **16**, 233-270.

McClelland, J. & A. Kawamoto (1986). Mechanisms of sentence processing: Assigning roles to constituents of sentences. In J. McClelland, D. Rumelhart, and the PDP Research Group (Eds.), *Parallel Distributed Processing: Explorations in the Microstructures of Cognition, Volume 2: Psychological and Biological Models.* Cambridge, MA: MIT Press.

McMahon, E. (1963). *Gramatical Analysis as Part of Understanding.* PhD dissertation, Harvard University, Cambridge, MA.

Miikkulainen, R. (1990). *DISCERN: A Distributed Artificial Neural Network Model of Script Processing and Memory.* PhD dissertation, Computer Science Department, University of California, Los Angeles.

Miikkulainen, R., & M. G. Dyer (1991). Natural language processing with modular PDP networks and distributed lexicon. *Cognitive Science*, **15**, 343-400.

Miller, G. (1990). Special issue, WordNet: An on-line lexical database. *International Journal of Lexicography*, **3**(4).

Minsky, M. (1975). A framework for representing knowledge. In P. H. Winston (Ed.), *The Psychology of Computer Vision.* New York: McGraw-Hill Book Company.

Minsky, M. & S. Papert (1969). *Perceptrons.* Cambridge, MA: MIT Press.

Mitchell, T., R. Keller, & S. Kedar-Cabelli. (1986). Explanation-based learning: A unifying view. *Machine Learning*, **1**, 47-80.

Onifer, W. & D. A. Swinney (1981). Accessing lexical ambiguities during sentence comprehension: Effects of frequency of meaning and contextual bias. *Memory & Cognition.* **9** (3), 225-236.

Ortony, A, D. L. Schallert, R. E. Reynolds, & S. J. Antos (1978). Interpreting metaphors and idioms: Some effects of context on comprehension. *Journal of Verbal Learning and Verbal Behavior*, **17**, 465-477.

Perrault, R. C. & J. F. Allen. (1980). A plan-based analysis of indirect speech acts. *American Journal of Computational Linguistics*, **6**, 167-182.

Pollack, J. B. (1990). Recursive distributed representations. *Artificial Intelligence*, **46**, 77-105.

Riesbeck, C. (1975). Conceptual analysis. In R. C. Schank (Ed.), *Conceptual Information Processing*. Amsterdam: North-Holland.

Rumelhart, D. E. (1979). Some problems with the notion of literal meanings. In A. Ortony (Ed.), *Metaphor and Thought*. Cambridge, UK: Cambridge University Press.

Rumelhart, D. E., G. E. Hinton, & R. J. Williams (1986). Learning internal representations by error propagation. In J. McClelland, D. Rumelhart, and the PDP Research Group (Eds.), *Parallel Distributed Processing: Explorations in the Microstructures of Cognition, Volume 1: Foundations*. Cambridge, MA: MIT Press.

Schank, R. C. (1973). Identification of conceptualizations underlying natural language. In R. C. Schank & K. M. Colby (Eds.), *Computer Models of Language and Thought*. San Francisco: W. H. Freeman and Company.

Schank, R. C. & R. P. Abelson (1977). *Scripts, plans, goals, and understanding.* Hillsdale, NJ: Lawrence Earlbaum Associates.

Schulenburg, D. A. (1992). Sentence processing with realistic feedback. *Proceedings of the International Joint Conference on Neural Networks*. Baltimore, MD: IEEE.

Schulenburg, D. A. & M. J. Pazzani (1989). Explanation-based learning of indirect speech act interpretation rules. *Proceedings of the First International Workshop on Lexical Acquisition*. Detroit, MI.

Searle, J. (1969). *Speech acts*. London: Cambridge University Press.

Searle, J. (1975). Indirect speech acts. In P. Cole & J. L. Morgan (Eds.), *Syntax and Semantics, Vol. 3, Speech Acts*. New York: Academic Press.

Selfridge, M. (1986). A computer model of child language learning. *Artificial Intelligence*, **29**, 171-216.

Shatz, M. (1978). Children's comprehension of their mothers' question-directives. *Journal of Child Language*, **5**, 39-46.

Shinjo, M & J. L. Myers (1987). The role of context in metaphor comprehension. *Journal of Memory and Language*, **26**, 226-241.

Small, S. & C. Rieger (1982). Parsing and comprehending with word experts (A theory and its realization). in W. G. Lehnert & M. H. Ringle (Eds.), *Strategies for Natural Language Processing.* Hillsdale, NJ: Lawrence Erlbaum Associates.

Smolensky, P. (1990). Tensor product variable binding and the representation of symbolic structures in connectionist systems. *Artificial Intelligence*, **46**, 159-216.

St. John, M. F. (1992). The story gestalt: A model of knowledge-intensive processes in text comprehension. *Cognitive Science*, **16**, 271-306.

St. John, M. F. & J. L. McClelland (1988). Applying textual constraints in sentence comprehension. *Proceedings of the Tenth Annual Conference of the Cognitive Science Society.* Hillsdale, NJ: Lawrence Earlbaum Associates.

St. John, M. F. & J. L. McClelland (1990). Learning and applying contextual constraints in sentence comprehension. *Artificial Intelligence*, **46**, 217-257.

Swinney, D. A. (1979). Lexical access during sentence comprehension: (Re)Consideration of context effects. *Journal of Verbal Learning and Verbal Behavior*, **18**, 645-659.

Swinney, D. A., & D. T. Hakes (1976). Effects of prior context upon lexical access during sentence comprehension. *Journal of Verbal Learning and Verbal Behavior*, **15**, 681-689.

Turing, A. M. (1950). Computing machinery and intelligence. *Mind*, **59**, 433-460.

Voorhees, E. M., C. Leacock, & G. Towell (1995). Learning context to disambiguate word senses. in T Petsche, S. J. Hanson, & J. Shavlik (Eds.), *Computation Learning Theory and Natural Learning Systems, Volume III.* Cambridge, MA: MIT Press.

Wickelgren, W. (1969). Context-sensitive coding, associative memory, and serial order in (speech) behavior. *Psychological Review*, **76**, 1-15.

Wilensky, R. (1983). *Planning and understanding: A computational approach to human reasoning.* Readinng, MA: Addison-Wesley.

Wilensky, R. (1989). Primal content and actual content: An antidote to literal meaning. *Journal of Pragmatics*, **13**, 163-186.

Woods, W. A. (1970). Transition network grammars for natural language analysis. *Communications of the ACM*, **13**(10), 591-606.

Zernik, U. (1987). *Strategies in Language Acquisitions: Learning Phrases from Examples in Context.* PhD dissertation, Computer Science Department, University of California, Los Angeles.

Appendix A

Experimental Data

"A likely story indeed!"

—LEWIS CARROLL, *Alice's Adventures in Wonderland*

The data used by PIP to demonstrate its ability at processing text are given below. Each section presents a set of *templates*; each template was used to generate a set of texts. Thus, the union of the sets of texts generated from the templates forms the total set of texts used for that section.

Specified slots in the templates (as indicated by italicized words) were substituted by the corresponding replacements (cf. the slot replacement tables). All combinations of slots and fillers were used to generate the set of texts, subject to certain constraints. One constraint was that $person_1$ could not be the same as $person_2$. For example, consider the first sentence from template_1 from Section A.2: "$Person_1$ hated $person_2$." Using the fillers given in Table A.4, the following set of sentences would be generated.

1. Terry hated Joe.
2. Terry hated Alex.
3. Terry hated Max.
4. Joe hated Terry.

5. Joe hated Alex.
6. Joe hated Max.
7. Alex hated Terry.
8. Alex hated Joe.
9. Alex hated Max.
10. Max hated Terry.
11. Max hated Joe.
12. Max hated Alex.

Furthermore, whenever a "dotted pair" is listed as a replacement, the first element of the pair is used as the replacement for the specified slot while the second element is used as the replacement for the associated *-filler* slot. For example, the last sentence in template$_2$ of Section A.2 has the slot $class_2$. The representation for this sentence references $class_2$*-filler*. Thus, referring to the $class_2$ slot in Table A.4, one set of replacements for this sentence would be `butcher` for $class_2$ and `bad-person` for $class_2$*-filler*.

Each section in the appendix is organized as follows. First, the slot replacement table is presented. These tables, as stated, define the specific replacements for each slot of the templates as explained above. Next, the templates themselves are presented. Each template consists of a set of sentences (the text) followed by a list of representations corresponding to each sentence of the text. Last, a table defining the role/filler pair assignment for each output unit of the *current proposition* layer (cf. Figure 4.1) is given. These units correspond to those given in the accuracy figures of Chapters 5, 6, and 7.

A.1 Lexical Ambiguity Data

Table A.1: Slot replacements for lexical ambiguity data.

slot	replacement
keepers	`crew men grounds-men`
animals	`horses dogs`
$state_2$	`(ready . ready) (green . inexperienced)`
runners	`runners competitors`
$person_3$	`(mom . she) (dad . he) (sande . she) (danny . he)`
$verb_3$	`made baked`
$object_3$	`(cookies . dough) (bread . dough)`
$state_3$	`(ready . condition) (green . color) (blue . color)`
pronoun	`he she`
$state_4$	`(ready . ready) (green . inexperienced)`
$player_5$	`batter catcher short-stop`
hit-ball	`homer single double`
$person_5$	`mom dad sande danny`
$object_5$	`knick-knacks table-ware dishes`
$color_5$	`blue green`
$player_6$	`(batter . bb-batter) (short-stop . short-stop)`
$player_{6c}$	`runner short-stop catcher`
$bb\text{-}base_6$	`home third second`
$soldier_6$	`soldier private corporal`

Lexical Ambiguity Template$_1$: The *keepers* worked all night. The field was ready.

`[(pred work) (actor` *keepers*`) (duration all-night)]`

`[(pred condition) (object field) (is ready)]`

Lexical Ambiguity Template$_2$: The *keepers* worked all night. The field was green.

`[(pred work) (actor` *keepers*`) (duration all-night)]`

`[(pred color) (object field) (is green)]`

Lexical Ambiguity Template$_3$: The *keepers* worked hard. The field was ready.

`[(pred work) (actor` *keepers*`) (effort 8)]`

`[(pred condition) (object field) (is ready)]`

Lexical Ambiguity Template$_4$: The *keepers* worked hard. The field was green.

`[(pred work) (actor` *keepers*`) (effort 8)]`

`[(pred color) (object field) (is green)]`

Lexical Ambiguity Template$_5$: The *animals* were in the gates. The field was $state_2$.

`[(pred loc) (object` *animals*`) (is gates)]`

`[(pred condition) (object` *animals*`) (is` $state_2$*-filler*`)]`

Lexical Ambiguity Template$_6$: The *runners* were in position. The field was $state_2$.

`[(pred loc) (object` *runners*`) (is gates)]`

`[(pred condition) (object` *runners*`) (is` $state_2$*-filler*`)]`

Lexical Ambiguity Template$_7$: The *animals* were in the gates. They were $state_2$.

`[(pred loc) (object` *animals*`) (is gates)]`

`[(pred condition) (object` *animals*`) (is` $state_2$*-filler*`)]`

Lexical Ambiguity Template$_8$: The *runners* were in position. They were $state_2$.

`[(pred loc) (object` *runners*`) (is gates)]`

`[(pred condition) (object` *runners*`) (is` $state_2$*-filler*`)]`

Lexical Ambiguity Template$_9$: *person$_3$ verb$_3$ object$_3$. person$_3$-filler* added food-coloring to the *object$_3$*. The batter was *state$_3$*.

```
[(pred bake) (actor person3) (object object3)]

[(pred add) (actor person3) (object food-coloring) (to object3)]

[(pred state3-filler) (object object3-filler) (is state3)]
```

Lexical Ambiguity Template$_{10}$: The pitcher walked to the mound. *pronoun* was *state$_4$*.

```
[(pred ptrans) (actor bb-pitcher)
               (object bb-pitcher)
               (to pitcher-mound)]

[(pred condition) (actor bb-pitcher) (is state4-filler)]
```

Lexical Ambiguity Template$_{11}$: The pitcher walked to the mound. *pronoun* was blue.

```
[(pred ptrans) (actor bb-pitcher)
               (object bb-pitcher)
               (to pitcher-mound)]

[(pred mood) (actor bb-pitcher) (is -5)]
```

Lexical Ambiguity Template$_{12}$: The pitcher walked to the mound. The batter was *state$_4$*.

```
[(pred ptrans) (actor bb-pitcher)
               (object bb-pitcher)
               (to pitcher-mound)]

[(pred condition) (actor bb-batter) (is state4-filler)]
```

Lexical Ambiguity Template$_{13}$: The pitcher walked to the mound. The batter was blue.

```
[(pred ptrans) (actor bb-pitcher)
               (object bb-pitcher)
               (to pitcher-mound)]

[(pred mood) (actor bb-batter) (is -5)]
```

Lexical Ambiguity Template$_{14}$: The *player$_5$* hit a *hit-ball*. The pitcher was green.

```
[(pred propel) (actor player_5)
               (object bat)
               (to ball)
               (status hit-ball)]

[(pred condition) (actor bb-pitcher) (is inexperienced)]
```

Lexical Ambiguity Template$_{15}$: The *player$_5$* hit a *hit-ball*. The pitcher was blue.

```
[(pred propel) (actor player_5)
               (object bat)
               (to ball)
               (status hit-ball)]

[(pred mood) (actor bb-pitcher) (is -5)]
```

Lexical Ambiguity Template$_{16}$: *person$_5$* bought some *object$_5$*. The pitcher was *color$_5$*.

```
[(pred atrans) (actor person_5)
               (object money)
               (to *unspec*)
               (result [(pred atrans)
                        (actor *unspec*)
                        (object object_5)
                        (to person_5)])]

[(pred color) (object jug-pitcher) (is color_5)]
```

Lexical Ambiguity Template$_{17}$: The *player$_6$* hit the ball. *pronoun* made it to the base on time.

```
[(pred propel) (actor player6-filler)
               (object bat)
               (to bb-ball)]

[(pred ptrans) (actor player6-filler)
               (object player6-filler)
               (to bb-base)
               (status safe)]
```

Lexical Ambiguity Template$_{18}$: The *player$_6$* hit a line-drive. *pronoun* made it to the base on time.

```
[(pred propel) (actor player6-filler)
               (object bat)
               (to bb-ball)]

[(pred ptrans) (actor player6-filler)
               (object player6-filler)
               (to bb-base)
               (status safe)]
```

Lexical Ambiguity Template$_{19}$: The *player$_{6c}$* tried to steal *bb-base$_6$*. *pronoun* made it to the base on time.

```
[(pred attempt) (actor player6c)
                (object [(pred ptrans)
                         (actor player6c)
                         (object player6c)
                         (to bb-base6)])]

[(pred ptrans) (actor player6c)
               (object player6c)
               (to bb-base6)
               (status safe)]
```

Lexical Ambiguity Template_{20}: The $soldier_6$ was late. *pronoun* made it to the base on time.

```
[(pred status) (actor soldier_6) (is late)]

[(pred ptrans) (actor soldier_6)
               (object soldier_6)
               (to mil-base)
               (status on-time)]
```

Table A.2: Output unit definitions for lexical ambiguity data, part 1.

Unit	Role/filler	Unit	Role/filler
0	(pred ptrans)	40	(object dough)
1	(pred status)	41	(object food-coloring)
2	(pred attempt)	42	(object bread)
3	(pred propel)	43	(object cookies)
4	(pred color)	44	(object competitors)
5	(pred atrans)	45	(object runners)
6	(pred mood)	46	(object dogs)
7	(pred condition)	47	(object horses)
8	(pred add)	48	(object field)
9	(pred bake)	49	(object->actor catcher)
10	(pred loc)	50	(object->actor short-stop)
11	(pred work)	51	(object->actor runner)
12	(actor corporal)	52	(object->object catcher)
13	(actor private)	53	(object->object short-stop)
14	(actor soldier)	54	(object->object runner)
15	(actor catcher)	55	(object->to second)
16	(actor short-stop)	56	(object->to third)
17	(actor runner)	57	(object->to home)
18	(actor bb-batter)	58	(to mil-base)
19	(actor danny)	59	(to second)
20	(actor sande)	60	(to third)
21	(actor dad)	61	(to home)
22	(actor mom)	62	(to bb-base)
23	(actor bb-pitcher)	63	(to bb-ball)
24	(actor batter)	64	(to *unspec*)
25	(actor men)	65	(to ball)
26	(actor grounds-men)	66	(to pitcher-mound)
27	(actor crew)	67	(to bread)
28	(object corporal)	68	(to cookies)
29	(object private)	69	(status on-time)
30	(object soldier)	70	(status safe)
31	(object catcher)	71	(status double)
32	(object ptrans)	72	(status single)
33	(object short-stop)	73	(status homer)
34	(object runner)	74	(is late)
35	(object bat)	75	(is green)
36	(object bb-batter)	76	(is blue)
37	(object jug-pitcher)	77	(is -5)
38	(object money)	78	(is inexperienced)
39	(object bb-pitcher)	79	(is ready)

Table A.3: Output unit definitions for lexical ambiguity data, part 2.

Unit	Role/filler	Unit	Role/filler
80	`(is gates)`	86	`(result->to danny)`
81	`(result atrans)`	87	`(result->to sande)`
82	`(result->actor *unspec*)`	88	`(result->to dad)`
83	`(result->object dishes)`	89	`(result->to mom)`
84	`(result->object table-ware)`	90	`(effort 8)`
85	`(result->object knick-knacks)`	91	`(duration all-night)`

A.2 Metaphor Data

Table A.4: Slot replacements for metaphor data.

slot	replacement
$person_1$	`terry joe alex max`
$person_2$	`terry joe alex max`
object	`bird mace knife hat`
crime	`murder robbery assault theft`
curse	`curse hex spell`
$place_{12}$	`train theater`
$place_{13}$	`abattoir slaughter-house`
$verb_2$	`(massacred . -10) (maimed . -8) (beat . -5)`
$verb_3$	`gave lent`
$verb_4$	`give lend`
$verb_5$	`put placed`
$verb_8$	`told ordered`
$class_1$	`(witch . bad-person) (dictator . bad-person) (criminal . criminal)`
$class_2$	`(butcher . bad-person) (criminal . criminal)`
$class_3$	`prince princess`
$class_8$	`prince princess dictator`
$class_{12}$	`butcher salesman`

Metaphor Template$_1$: *person$_1$* hated *person$_2$*. *person$_1$* framed *person$_2$* for *crime*. *person$_1$* is a *class$_1$*.

```
[(pred hate) (actor person1) (object person2)]

[(pred do) (actor person1)
           (to person2)
           (result [(pred blame)
                    (actor person2)
                    (object crime)
                    (mode frame)])]

[(pred member) (actor person1) (class class1-filler)]
```

Metaphor Template$_2$: *person$_1$* hated *person$_2$*. *person$_1$* *verb$_2$* *person$_2$*. *person$_1$* is a *class$_2$*.

```
[(pred hate) (actor person1) (object person2)]

[(pred do) (actor person1)
           (to person2)
           (result [(pred phys)
                    (actor person2)
                    (range verb2-filler)])]

[(pred member) (actor person1) (class class2-filler)]
```

Metaphor Template$_3$: *person$_1$* hated *person$_2$*. *person$_1$* *verb$_2$* *person$_2$*. *person$_2$* is a martyr.

```
[(pred hate) (actor person1) (object person2)]

[(pred do) (actor person1)
           (to person2)
           (result [(pred phys)
                    (actor person2)
                    (range verb2-filler)])]

[(pred member) (actor person2) (class martyr)]
```

Metaphor Template$_4$: *person$_1$* hated *person$_2$*. *person$_2$* needed a *object*. *person$_1$* *verb$_3$* *person$_2$* a *object*. *person$_1$* is a *class$_3$*.

```
[(pred hate) (actor person1) (object person2)]

[(pred goal) (actor person2)
             (object [(pred possess)
                      (actor person2)
                      (object object)])]

[(pred atrans) (actor person1)
               (object object)
               (to person2)]

[(pred member) (actor person1) (class good-person)]
```

Metaphor Template$_5$: *person$_1$* hated *person$_2$*. *person$_2$* needed a *object*. *person$_1$* did not *verb$_4$* *person$_2$* a *object*. *person$_1$* is a criminal.

```
[(pred hate) (actor person1) (object person2)]

[(pred goal) (actor person2)
             (object [(pred possess)
                      (actor person2)
                      (object object)])]

[(pred atrans) (actor person1)
               (object object)
               (to person2)
               (mode neg)]

[(pred member) (actor person1) (class bad-person)]
```

Metaphor Template$_6$: *person$_1$* hated *person$_2$*. *person$_1$* *verb$_5$* a *curse* on *person$_2$*. *person$_1$* is a witch.

```
[(pred hate) (actor person1) (object person2)]

[(pred curse) (actor person1) (object person2)]

[(pred member) (actor person1) (class witch)]
```

Metaphor Template$_7$: $person_1$ needed a *object*. $person_1$ $verb_8$ $person_2$ to get the *object*. $person_1$ is a $class_8$.

```
[(pred goal) (actor person1)
             (object [(pred possess)
                      (actor person1)
                      (object object)])]

[(pred speak) (actor person1)
              (to person2)
              (object [(pred atrans)
                       (actor person2)
                       (object object)
                       (to person1)])]

[(pred member) (actor person1) (class class8)]
```

Metaphor Template$_8$: $person_1$ committed *crime*. $person_2$ loved $person_1$. $person_2$ confessed to $person_1$ *crime*. $person_2$ is a martyr.

```
[(pred do) (actor person1)
           (result [(pred crime)
                    (actor person1)])]

[(pred love) (actor person2) (object person1)]

[(pred confess) (actor person2)
                (object [(pred do)
                         (actor person1)
                         (result [(pred crime)
                                  (actor person1)])])]

[(pred member) (actor person2) (class good-person)]
```

Metaphor Template$_9$: *person$_1$* committed *crime*. *person$_2$* loved *person$_1$*. *person$_2$* took the blame for *person$_1$* *crime*. *person$_2$* is a martyr.

```
[(pred do) (actor person1)
           (result [(pred crime)
                    (actor person1)])]

[(pred love) (actor person2) (object person1)]

[(pred blame) (actor person2)
              (object [(pred crime)
                       (actor person1)
                       (mode voluntary)])]

[(pred member) (actor person2) (class good-person)]
```

Metaphor Template$_{10}$: *person$_1$* works at the *place$_{12}$*. *person$_1$* sells *object*. *person$_1$* is a *class$_{12}$*.

```
[(pred job) (actor person1) (loc place12)]

[(pred atrans) (actor person1)
               (object object)
               (result [(pred atrans)
                        (object money)
                        (to person1)])]

[(pred member) (actor person1) (class class12)]
```

Metaphor Template$_{11}$: *person$_1$* works at the *place$_{13}$*. *person$_1$* is a butcher.

```
[(pred job) (actor person1) (loc place13)]

[(pred member) (actor person1) (class butcher)]
```

Metaphor Template$_{12}$: *person$_2$* needed a *object*. *person$_1$* convinced *person$_2$* to buy the *object*. *person$_1$* is a salesman.

```
[(pred goal) (actor person2)
             (object [(pred possess)
                      (actor person2)
                      (object object)])]

[(pred mbuild) (actor person1)
               (to person2)
               (object [(pred atrans)
                        (actor person2)
                        (object money)
                        (result [(pred atrans)
                                 (to person2)
                                 (object object)])])]

[(pred member) (actor person1) (class persuader)]
```

Table A.5: Output unit definitions for metaphor data, part 1.

Unit	Role/filler	Unit	Role/filler
0	(pred member)	40	(object assault)
1	(pred atrans)	41	(object murder)
2	(pred goal)	42	(object->actor alex)
3	(pred hate)	43	(object->actor terry)
4	(pred do)	44	(object->actor max)
5	(pred speak)	45	(object->actor joe)
6	(pred job)	46	(object->object knife)
7	(pred curse)	47	(object->object bird)
8	(pred mbuild)	48	(object->object mace)
9	(pred blame)	49	(object->object hat)
10	(pred love)	50	(object->object money)
11	(pred confess)	51	(object->to max)
12	(actor joe)	52	(object->to joe)
13	(actor alex)	53	(object->to alex)
14	(actor max)	54	(object->to terry)
15	(actor terry)	55	(object->result atrans)
16	(class good-person)	56	(object->result robbery)
17	(class martyr)	57	(object->result theft)
18	(class criminal)	58	(object->result assault)
19	(class dictator)	59	(object->result murder)
20	(class butcher)	60	(object->result->to terry)
21	(class princess)	61	(object->result->to joe)
22	(class prince)	62	(object->result->to alex)
23	(class bad-person)	63	(object->result->to max)
24	(class witch)	64	(object->result->object bird)
25	(class persuader)	65	(object->result->object hat)
26	(class salesman)	66	(object->result->object knife)
27	(object knife)	67	(object->result->object mace)
28	(object possess)	68	(object->result->actor joe)
29	(object alex)	69	(object->result->actor terry)
30	(object bird)	70	(object->result->actor max)
31	(object terry)	71	(object->result->actor alex)
32	(object joe)	72	(to alex)
33	(object atrans)	73	(to terry)
34	(object mace)	74	(to joe)
35	(object hat)	75	(to max)
36	(object max)	76	(result phys)
37	(object theft)	77	(result blame)
38	(object do)	78	(result atrans)
39	(object robbery)	79	(result theft)

Table A.6: Output unit definitions for metaphor data, part 2

Unit	Role/filler	Unit	Role/filler
80	`(result robbery)`	93	`(result->object theft)`
81	`(result assault)`	94	`(result->object assault)`
82	`(result murder)`	95	`(result->mode frame)`
83	`(result->actor alex)`	96	`(result->to alex)`
84	`(result->actor joe)`	97	`(result->to max)`
85	`(result->actor terry)`	98	`(result->to terry)`
86	`(result->actor max)`	99	`(result->to joe)`
87	`(result->range -5)`	100	`(loc theater)`
88	`(result->range -10)`	101	`(loc train)`
89	`(result->range -8)`	102	`(loc abattoir)`
90	`(result->object robbery)`	103	`(loc slaughter-house)`
91	`(result->object money)`	104	`(mode neg)`
92	`(result->object murder)`	105	`(mode voluntary)`

A.3 Indirect Speech Act Data

Table A.7: Slot replacements for indirect speech act data.

slot	replacement
$person_1$	`student t/a professor instructor visitor`
$person_2$	`student t/a professor instructor visitor`
object	`pencil notebook book test-tube`
class	`chemistry physics biology engineering`
event	`exam quiz test experiment`

Indirect Speech Act Template$_1$: The *person$_1$* went to the *class* class. The *person$_1$* needed a *object* for the *event*. The *person$_2$* asked the *person$_1$*, "Do you have a *object*?"

```
[(pred statement) (focus [(pred ptrans)
                          (actor person1)
                          (object person1)
                          (to class)])
                  (illoc inform)]

[(pred statement) (focus [(pred goal)
                          (actor person1)
                          (object [(pred possess)
                                   (actor person1)
                                   (object object)])])]

[(pred statement) (focus [(pred speak)
                          (actor person2)
                          (to person1)
                          (object [(pred possess)
                                   (actor person1)
                                   (object object)])
                          (illoc interrogative)])]
```

Indirect Speech Act Template$_2$: The *person$_1$* went to the *class* class. The *person$_1$* needed a *object* for the *event*. The *person$_1$* asked the *person$_2$*, "Do you have a *object*?"

```
[(pred statement) (focus [(pred ptrans)
                          (actor person1)
                          (object person1)
                          (to class)])
                  (illoc inform)]

[(pred statement) (focus [(pred goal)
                          (actor person1)
                          (object [(pred possess)
                                   (actor person1)
                                   (object object)])])]

[(pred statement) (focus [(pred speak)
                          (actor person2)
                          (to person2)
                          (object [(pred atrans)
                                   (actor person2)
                                   (from person2)
                                   (to person1)
                                   (object object)])
                          (illoc request)])]
```

Table A.8: Output unit definitions for indirect speech act data, part 1.

Unit	Role/filler
0	(pred statement)
1	(focus speak)
2	(focus goal)
3	(focus ptrans)
4	(focus->actor visitor)
5	(focus->actor instructor)
6	(focus->actor professor)
7	(focus->actor t/a)
8	(focus->actor student)
9	(focus->to instructor)
10	(focus->to engineering)
11	(focus->to professor)
12	(focus->to t/a)
13	(focus->to student)
14	(focus->to visitor)
15	(focus->to biology)
16	(focus->to physics)
17	(focus->to chemistry)
18	(focus->object atrans)
19	(focus->object possess)
20	(focus->object visitor)
21	(focus->object instructor)
22	(focus->object professor)
23	(focus->object t/a)
24	(focus->object student)
25	(focus->object->actor instructor)
26	(focus->object->actor visitor)
27	(focus->object->actor professor)
28	(focus->object->actor t/a)
29	(focus->object->actor student)
30	(focus->object->from instructor)
31	(focus->object->from professor)
32	(focus->object->from t/a)
33	(focus->object->from student)
34	(focus->object->from visitor)
35	(focus->object->to visitor)
36	(focus->object->to instructor)
37	(focus->object->to professor)
38	(focus->object->to t/a)
39	(focus->object->to student)

Table A.9: Output unit definitions for indirect speech act data, part 2.

Unit	Role/filler
40	`(focus->object->object test-tube)`
41	`(focus->object->object book)`
42	`(focus->object->object notebook)`
43	`(focus->object->object pencil)`
44	`(focus->illoc request)`
45	`(focus->illoc interrogative)`
46	`(illoc inform)`