Human-Computer Interaction for Semantic Disambiguation

Ralf D. Brown
School of Computer Science
Carnegie Mellon University
Pittsburgh, PA 15213
Internet: ralf@cs.cmu.edu

Sergei Nirenburg
Center for Machine Translation
Carnegie Mellon University
Pittsburgh, PA 15213
Internet: sergei@cs.cmu.edu

Topics: User Interaction, Disambiguation

Abstract

We describe a semi-automatic semantic disambiguator integrated in a knowledge-based machine translation system. It is used to bridge the analysis and generation stages in machine translation. The user interface of the disambiguator is built on mouse-based multiple-selection menus.

1. Introduction

Extraction and representation of text meaning is a central concern of natural language application developers. This goal still largely eludes computational linguists. Many problems remain unresolved. They include referential ambiguity resolution [4, 12], determining the nature of semantic dependency relations (as, for instance, in compound nouns in English [8]), treatment of novel language and ill-formed input [21], metaphor and metonymy [6, 7], discourse and pragmatic meanings [11, 14, 17], etc.

Another set of tasks includes work on representation languages both for text meaning proper and for ontological domain models that underlie semantic analysis of texts [1, 7, 13, 15] , problems of acquiring and working with domains and sublanguages of realistic size [15, 16] and taking into account requirements of particular applications, such as machine translation, natural language interfaces to databases and expert systems, etc.

In the partial case of a particular application area, the representation problems are alleviated. However, the treatment of a large number of linguistic phenomena is still a major problem. At this point, the developers of natural language processing (NLP) applications have a choice of

1. not relying on results of semantic and pragmatic analysis;
2. providing semantic analysis for selected phenomena and limited domains only; or
3. using human help in determining facets of text meaning.

In this paper we describe an environment facilitating human involvement in semantic and pragmatic analysis (Figure 1). This environment is applicable to most comprehensive NLP applications and consists of an automatic analyzer of input text, a generator of output text (either in a natural language or in a formal language) and an *augmentor* module that bridges the two and facilitates the involvement of a human in the processing loop. The background knowledge for such a system consists of an ontological domain model, a grammar and a machine-tractable dictionary (MTD)[1] for each natural language involved in either analysis or generation.

We will concentrate on the augmentor module, which consists of a human-computer interface with a dialog manager and a set of automatic semantic analysis components. The composition of the automatic components depends on the capabilities of the particular analyzer with which the augmentor is coupled. We proceed from the assumption that the format and content of the input to generation is fixed. It is this set of knowledge structures that we call the text meaning representation. Therefore, if the automatic analyzer is relatively shallow, the augmentor will have to perform more operations to fill the gaps in this representation. The role of the augmentor will diminish as the sophistication of the automatic analyzers increases. The above means that the environment we suggest is flexible and durable as a software configuration, because new findings and methods of treatment of the various linguistic phenomena will be accommodated in the architecture as they appear.

The concept of the augmentor is also useful from the standpoint of building large software systems. In such applications it is usually desirable to incorporate as many existing software modules as possible, to avoid developing software from scratch. However, many such components expect their inputs and produce their outputs in an idiosyncratic formalism. An augmentor module can include special facilities for reformatting the output of one software module in accordance with the requirements on the input to another module. In the framework of natural language processing, the augmentor will usually reformat the results of the analyzer into the format expected by the generator.

We now describe the augmentor module of the KBMT-89 machine translation system developed at Carnegie Mellon University [10].

In KBMT-89 semantic interpretation occurs partly

[1]This term is due to Yorick Wilks, and is distinct from machine-readable dictionary, which is simply a printed dictionary stored electronically.

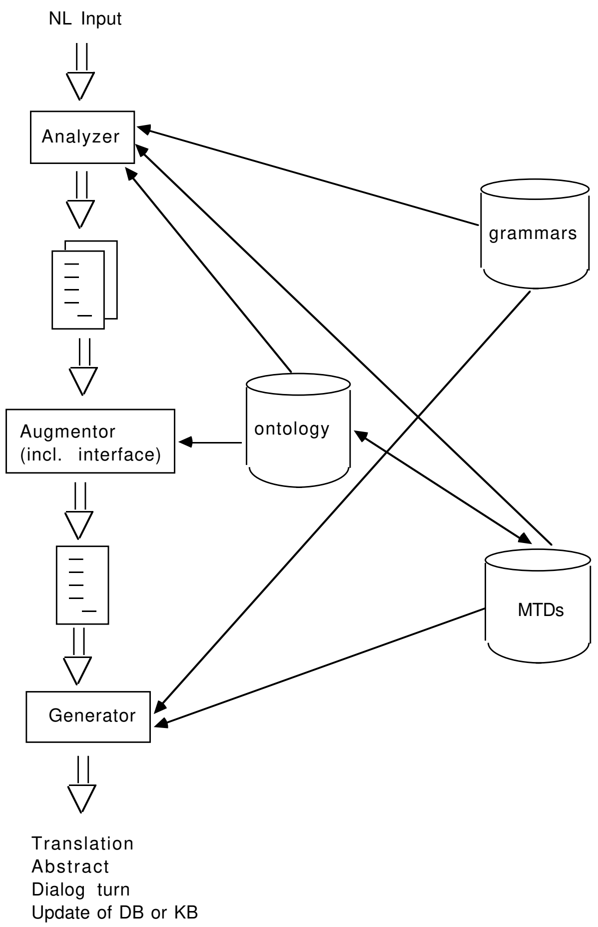

Figure 1: The architecture of an NLP system which facilitates human intervention

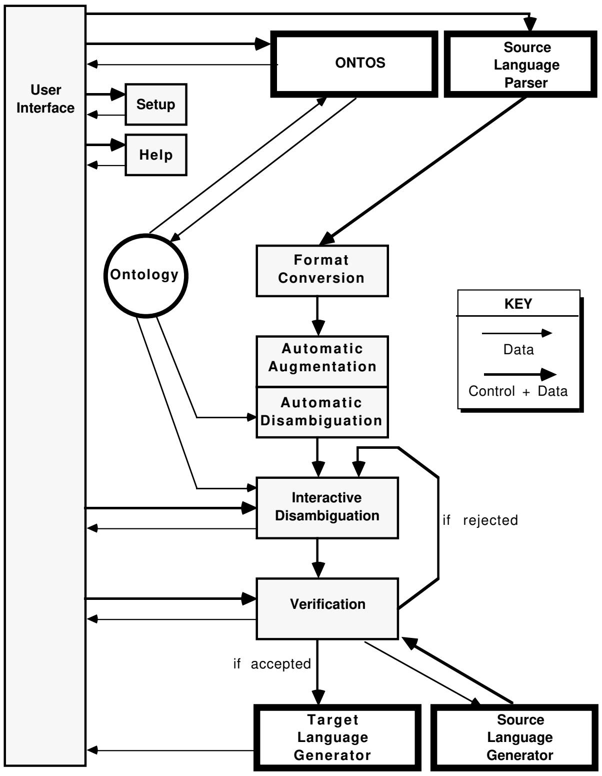

Figure 2: The augmentor, shown as part of KBMT-89. The augmentor components are shaded.

in the analyzer module and partly in the augmentor, a division dictated largely by the requirements of computational efficiency and the reuse of an existing module. The KBMT-89 analyzer was built around a parser developed by Tomita in 1986 [20]. It proved to be essential to apply semantic constraints early in the parsing process to reduce the number of ambiguities; however, the semantic processing integrated into the analyzer was insufficient in many cases. Since the output of the KBMT-89 analyzer had to be reformatted in accordance with the requirements of the generator, additional semantic analysis was to be performed at the augmentation stage. Parts of this analysis and disambiguation can be performed automatically; for the remainder, human interaction is used. The user is asked to supply missing information and to choose among ambiguous alternatives until a single, unambiguous result (called an *interlingua text*, or ILT) is obtained. This is in contrast to other systems such as TEAM, which selects a "best" parse to process based on *a priori* syntactic considerations [19].

The KBMT-89 augmentor was thus designed with three main components to meet the criteria mentioned above: a format converter, an automatic augmentor/disambiguator and an interactive disambiguator (Figure 2, previously published in [2]).

2. Automatic Augmentation

The KBMT-89 system consists of multiple components which run in separate Lisp processes (usually on separate workstations) in a distributed fashion. The distinct components (Figure 2) are a source-language analyzer, a source-language generator for paraphrasing (used for verification), a target-language generator, the augmentor, and the ONTOS knowledge acquisition tool [18] (used for queries or updates of the ontological domain model).

The analyzer provides the augmentor with a nested list representation (Figure 3) of the meaning and syntax for each of the possible interpretations of the input sentence. The augmentor extracts the semantic information (itself in a nested list format within the `(SEM ...)` sublists), removes any completely duplicated semantic parses, and converts the nested lists to an isomorphic set of trees of linked frames using the FRAMEKIT knowledge representation package [5, 9]. The hierarchies of frames produced by the format conversion form the candidate interlingua texts. At this stage, however, the ILTs are still "bare", containing only that information which appears directly in the analyzer output (Figure 4) and a stub for the speech act information.

The automatic augmentor and disambiguator in KBMT-89 consist of a pattern matcher and a pronominal anaphora resolver described in [4]. The pattern matcher performs a number of structural rearrangements on the trees of linked FRAMEKIT frames, as well as adding information which is readily derivable from other information already present in the parser output.

After the pattern matcher completes its modifications of the interlingua texts, the automatic disambiguation procedures are invoked. Currently, only the pronominal anaphora resolver MARS (Multiple Anaphora Resolution Strategies) is implemented.

```
(((SEM
   (*SEM*
    ((NUMBER-BULLET
      (($IS-TOKEN-OF ANY-NUMBER)($ID (*ID* 1))
       (CARDINALITY 1)
       ($MAP-DATA
        (*MAP* { map-str (ANY-NUMBER-MAP)} ))))
     (CLAUSAL-MARK +) (MOOD IMPERATIVE)
     (TENSE PRESENT)
     (SOURCE
      ((REFERENCE DEFINITE) (NUMBER SINGULAR)
       ($MAP-DATA
        (*MAP* { map-str diskette drive } ))
       ($IS-TOKEN-OF DISKETTE-DRIVE)
       ($ID (*ID* 27))))
     (THEME …)
     (AGENT *READER)
     ($MAP-DATA (*MAP* { map-str remove } ))
     ($IS-TOKEN-OF REMOVE) ($ID (*ID* 5)))))
  (NUMBER-BULLET
   ((ROOT 1) (VALUE 1)
    (SEM …)))
  (OBJ
   ((CASE ACC)
    (SEM
     (*SEM*
      ((REFERENCE DEFINITE) (NUMBER SINGULAR)
       ($MAP-DATA (*MAP* { map-str tape } ))
       ($IS-TOKEN-OF STICKY-TAPE)
       ($ID (*ID* 6)))))
    (REF DEFINITE)
    (DET ((ROOT THE) (REF DEFINITE)))
    (ROOT TAPE) (COUNT NO) (PERSON 3)
    (NUMBER SINGULAR) (MEAS-UNIT NO)
    (PROPER NO)))
  (VALENCY TRANS) (MOOD IMPERATIVE)
  (TENSE PRESENT) (FORM INF)
  (PPADJUNCT …)
  (ROOT REMOVE) (COMP-TYPE NO) (PASSIVE -)))
```

Figure 3: Abbreviated parse of *1. Remove the tape from the diskette drive.*

```
[CLAUSE490
    (SPEECHACTID
        [SPEECH-ACT488
            (TIME TIME489)
            (SPACE)
            (DIRECT?)
            (SPEECH-ACT)])
    (PROPOSITIONID
        [*REMOVE
            (NUMBER-BULLET
                [*ANY-NUMBER
                    (CARDINALITY 1)])
            (MOOD IMPERATIVE)
            (TENSE PRESENT)
            (SOURCE
                [*DISKETTE-DRIVE
                    (REFERENCE DEFINITE)
                    (NUMBER SINGULAR)])
            (THEME
                [*STICKY-TAPE
                    (REFERENCE DEFINITE)
                    (NUMBER SINGULAR)])
            (AGENT *READER)])
]
```

Figure 4: Bare Interlingua Text in a compact display format emphasizing its tree structure

MARS attempts to find the referent for each pronoun and definite noun phrase in the interlingua texts, and adds a link to the referent if found. It is often possible to eliminate a candidate ILT during resolution, particularly if further processing of the parses is delayed until the next several sentences have been processed by the anaphora resolver.

MARS employs a set of constraints and preference strategies[2] to determine the referent of a pronoun or definite noun phrase. The constraints are applied first to reduce the set of candidate referents, and then the preference strategies are applied using a voting scheme. The candidate with the largest total weight is considered the desired referent, unless there are other candidates within a predetermined threshold, in which case the anaphor is held to be ambiguous among those candidate referents. Possibly after an interactive disambiguation session (described below), the user is asked to confirm a paraphrase of the input.

3. Interactive Disambiguation

If multiple candidate ILTs remain after automatic disambiguation, a *composite* ILT (as described in detail in [2, 10]) is created by combining all candidate parses, and any parses which are proper subsets of other remaining candidates are removed. The composite ILT is then used to generate a set of multiple-selection menus which will be used in the interactive disambiguation.

A composite ILT retains the tree structure of each candidate interlingua text used to form it. Each frame in the composite contains all of the slots contained in each of the original ILT frames from which it was made. In turn, each slot of a composite frame contains all of the distinct fillers together with pointers to the original ILTs containing each distinct filler.

To begin interactive disambiguation, the augmentor checks the slots of the composite ILT for multiple fillers. If there are multiple fillers, the augmentor builds a set of multiple-choice menus for the user to decide which of these fillers must remain in the final interlingua text. The user interface (Figures 2 and 5) displays as many as four menus at a time during disambiguation, and the user makes his selections on any of them. This puts the user in partial control of the order in which ambiguities are eliminated, allowing him to choose the menu which is simplest or most obvious. By allowing more than one choice from a menu to be selected, some disambiguation can occur even if the user is unsure which value is most appropriate.

After a decision has been made on a menu by clicking the mouse button over the desired choices and then `DONE`, the augmentor examines the composite ILT and determines which of the candidate interlingua texts contain any of the selected values. The ILTs which do not contain any selected values are discarded, and the composite ILT is adjusted by removing the discarded

[2] These currently include local anaphora constraints, case role semantic constraints, pre/postcondition constraints, case role persistence preference, intersentential recency preference, and syntactic topicalization preference.

entries. Finally, the menu contents are adjusted to reflect any possible reduction in choices, and menus with only a single entry are deleted. Because the menus are not entirely independent, it is not unusual for a single selection to cause the removal of multiple menus, even if the menu on which the selection was made still contains more than one choice. Once the menus have been adjusted, another set of menus is displayed, and the cycle of menu display and user input repeats until a single, unambiguous interlingua text remains, which is passed on to the generator.

4. Augmentor Interface Features

The augmentor user interface (Figure 5) consists of an input/output panel in the bottom half of the screen, a main menu to its right, and the query area in the top half of the screen. The input/output panel is further divided (from top to bottom) into the input window, the status line, the paraphrase window, and the translation window. The input window accepts all typed input; the status line informs the user of the progress of a translation or indicates what input the augmentor is expecting; the paraphrase window displays a paraphrase of the input text after all analysis and disambiguation is complete, and the translation window displays the final translation after the paraphrase has been accepted by the user.

The user interface allows the user to consult the ontological domain model or the relevant dictionaries through the knowledge acquisition system ONTOS. The user may query the knowledge base, displaying either a graphical representation of the heirarchy or the actual contents of the frame for a concept. A simpler query is possible even if ONTOS is not loaded; each menu which asks for a selection among ambiguous concepts for a word allows the user to display the synonymous terms which map into each concept rather than the definition of the concept. The augmentor performs the necessary extraction from the ontology itself.

All of the windows in the KBMT-89 augmentor were implemented using the programmable editor HEMLOCK integrated into Carnegie Mellon University's Common Lisp system. As a result, the input, paraphrase, and translation windows are actually editor buffers and each retains the previous output even after it has scrolled out of the window. This makes reviewing earlier work simply a matter of placing the mouse cursor in the appropriate window and issuing editor-movement commands (either from the keyboard or by pressing the mouse buttons). The entire transcript from a given window can also be saved to a file, if desired.

Since the KBMT-89 system is modular, changing the direction of translation only requires reconnecting the various modules in different ways. This may be accomplished by executing the setup procedure (which occurs automatically when the augmentor is initially loaded, and may be selected from the main menu) and specifying the source and target languages. A shortcut has been placed on the main menu to switch between English-to-Japanese and Japanese-to-English translation, as those were the languages available to KBMT-89. Regardless of the source and target languages, the augmentor invokes the proper analyzer and generators to accomplish the desired translation.

One of the more interesting features of the KMBT-89 augmentor is that the user interface language has been made completely independent of the source and target languages by passing all messages through a lookup function before displaying them. The language may be selected, during setup, from among those installed, and may either remain fixed or change to the new source language whenever the source language is changed. If the proper set of messages has been installed in the lookup table, it is possible for the user interface to be, for instance, in German while translating from Japanese to English. The main use of this feature, however, is to allow a user to translate from an unfamiliar language into his native language, though not as well as translating from his native language into an unfamiliar one.

The definitions displayed in word-sense disambiguation menus are similarly translatable by placing definitions for the desired languages into the ontology along with the English definition. For both user interface messages and definitions, the augmentor automatically falls back to English if the message or definition is not available in the appropriate language.

5. An Example

We now describe an actual example of the use of the augmentor in the translation of a sentence from English into Japanese. This example begins when the user enters the sentence to be translated (number 19 in the test corpus: *7. Set the power switch on the system unit to On.*)[3]. The augmentor invokes the English analyzer with this sentence as input. Once the candidate parses are obtained, the augmentor converts each of them into a set of FRAMEKIT frames, which it then augments by making a variety of implicit information explicit and performing structural rearrangements. The MARS anaphora resolver does not apply to this sentence, since the latter does not contain pronominal anaphora, and there is no prior context for attempting to determine coreferentiality of definite noun phrases. Therefore, all of the candidate parses remain after the automatic processing.

After augmentation and disambiguation, any remaining ambiguity in the candidate parses invokes an interactive disambiguation session. In this case, four menus appear, indicating that there are at least four points on which the 14 candidate parses differ (Figure 5). We will work with the lower-left menu first, as it has the largest number of entries, which, we hope, will reduce the ambiguity most quickly. After deciding on `DISCRETE-ELECTRONIC-MOVE-LEVER` as the meaning of `SET` and clicking on it and then on `DONE`, the augmentor discards those candidate parses which do not contain the selected value in the appropriate position (we could have selected multiple items if we had been unsure of the correct one). In this case, the number of candidate parses is reduced from fourteen to six, and another menu replaces the one just completed (unfortunately, space constraints prohibit inclusion of further screen images; a complete version of this example will appear in a forthcoming paper [3]).

[3]The domain of KBMT-89 is personal computer installation and maintenance guides.

We now select ON-POSITION in the upper left-hand menu as the meaning of DISCRETE-POSITION (rather than using the more general POSITION)[4], which reduces the number of candidates to two and removes three of the menus, as two of the other menus were not independent of the upper left-hand menu. A new menu appears, and we are left with just two menus. After making a total of three selections, only one candidate parse remains. This is passed on to the English generator for paraphrasing, and the paraphrase is displayed in the center window. The augmentor asks whether the paraphrase properly captures the meaning of the input, and an affirmative response triggers generation in the target language. The translation appears in the bottom-most window. A negative response would have restored all of the candidate parses (including any eliminated automatically) and started another disambiguation session.

6. Future Directions

Knowledge acquisition (KA) is often an integral part of an application which uses natural language. Since the knowledge sources cannot be expected to be adequate in all cases, it will not be unusual for the natural language processing component to require knowledge enhancement. By having a knowledge acquisition component integrated into the NLP application, we may achieve a synergistic effect. The system dictionaries can be updated immediately whenever there is a failure in parsing or generation caused by an inadequate dictionary; similarly for ontologies and grammars. The KA component, in turn, may invoke the natural language analyzer to help automate a part of the knowledge extraction process by processing machine-readable dictionaries and encyclopediae and online corpora, thus easing the knowledge acquisition task. Whether invoked by the application or the knowledge acquisition component, the analyzer may need the augmentor's help in disambiguating the input; the augmentor in turn may determine the need to acquire more knowledge and (re-)invoke the KA component. A proposed knowledge acquisition environment utilizing such an integrated NLP/KA approach, with provisions for use by a team of knowledge-enterers, will be described in [3].

7. References

1. Brachman, R.J. and H.J. Levesque. *Readings in Knowledge Representation.* Morgan Kaufmann, 1985.

2. Brown, R.D. "Augmentation". *Machine Translation (formerly Computers and Translation) 4* (1989), 129-147.

3. Brown, R.D, and S. Nirenburg. Multifunctional Interfaces in NLP (working title). in preparation.

4. Carbonell, J. G. and R. Brown. Anaphora Resolution: A Multi-Strategy Approach. Proceedings of the Twelfth International Joint Conference on Computational Linguistics, COLING '88, 1988.

5. Carbonell, J.G. and R. Joseph. The FrameKit Reference Manual. Carnegie Mellon University Computer Science Department, 1985.

6. Carbonell, J.G. Metaphor: An Inescapable Phenomenon in Natural-Language Comprehension. In *Strategies for Natural Language Processing*, Lawrence Erlbaum Associates, Hillsdale, New Jersey, 1982.

7. Fass, D. Collative Semantics: A Study in the Discrimination of Meaning. Tech. Rept. 88-24, CSS/LCCR, 1988.

8. Finin, T.W. The Semantic Interpretation of Nominal Compounds. Proceedings of the First Annual National Conference on Artificial Intelligence, AAAI-80, 1980, pp. 310-312.

9. Nyberg, E. FrameKit User's Guide. Carnegie Mellon University Center for Machine Translation, 1988.

10. Goodman, K. and S. Nirenburg (Ed.) *KBMT-89 Project Report.* Carnegie Mellon University Center for Machine Translation, 1989.

11. Grosz, B., and C. Sidner. "Attention, Intentions, and the Structure of Discourse". *Computational Linguistics 12*, 3 (1986), 175-204.

12. Hirst, G. *Lecture Notes in Computer Science.* Volume 119:*Anaphora in Natural Language Understanding: A Survey.* Springer Verlag, 1981.

13. Hirst, G. *Semantic Interpretation and the Resolution of Ambiguity.* Cambridge University Press, 1987.

14. Hovy, E.H. *Generating Natural Language Under Pragmatic Constraints*. Ph.D. Th., Yale University, 1987.

15. Lenat, D. and R.V. Guha. The World According to CYC. Tech. Rept. ACA-AI-300-88, Microelectronics and Computer Technology Corporation, 1988.

16. Monarch, I., S. Nirenburg and T. Mitamura. Ontology-Based Lexicon Acquisition for a Machine Translation System. Proceedings of Fourth Workshop on Knowledge Acquisition for Knowledge-Based Systems, Banff, Canada, 1989.

17. Nirenburg, S. and C. Defrise. Aspects of Text Meaning: Using Discourse Connectives and Attitudinals in Natural Language Generation. Carnegie Mellon University Center for Machine Translation, 1989.

18. Nirenburg, S., I. Monarch, T. Kaufmann, I. Nirenburg and J. Carbonell. Acquisition of Very Large Knowledge Bases: Methodology, Tools, and Applications. Tech. Rept. 88-108, Carnegie Mellon University Center for Machine Translation, 1988.

19. Grosz, B.J., D.E. Appelt, P.A. Martin, and F.C.N. Pereira. "TEAM: An Experiment in the Design of Transportable Natural-Language Interfaces". *Artificial Intelligence 32* (1987), 173-243.

20. Tomita, M. *Efficient Parsing for Natural Language: A Fast Algorithm for Practical Systems.* Kluwer Academic Publishers, 1986.

[4]One possible improvement is to detect such cases and automatically discard the more general concept.

21. Weischedel, R.M. and L.A. Ramshaw. Reflections on the Knowledge Necessary to Parse Ill-Formed Input. In S. Nirenburg, Ed., *Machine Translation: Theoretical and Methodological Issues*, Cambridge University Press, Cambridge, England, 1987.

Figure 5: Augmentor Display