NL Comprehension via Integrative AI and Human-Computer Interaction

Topics: natural language processing, human-computer interaction, knowledge representation

Ben Goertzel, Michael Ross, Cassio Pennachin, Moshe Looks, Hugo Pinto
Contact: ben@goertzel.org

Abstract

INLINK is an innovative software system enabling an AI program to correctly interpret complex English sentences via interaction with a human user, who reviews and corrects interpretations within a graphical user interface. While not as convenient as unsupervised natural language comprehension, this mode of transforming linguistic knowledge into formal knowledge is drastically more efficient than expert-system-style rule encoding. Furthermore, the system becomes more intelligent with time, via adaptively learning from the feedback humans give it during the interactive interpretation process. The computational language processing inside the system is carried out via a combination of the Sleator and Temperley link parser with a collection of semantic mapping rules and special algorithms for semantic disambiguation, reference resolution and entity extraction. Query processing is carried out via dynamic programming. The system is built within the Novamente integrative AI framework, and a more sophisticated version involving probabilistic inference and replacing the link parser with a more advanced parsing framework is currently under construction.

1 Introduction

INLINK[1] is a software system designed to enable interactive natural language understanding. This refers to a modality in which users enter sentences into a graphical user interface which is connected to an AI system, and then interact with the user interface to ensure that it has come to a correct understanding of their sentences.

The creation of the system has been motivated by the practical need to transform complex knowledge from textual form into abstract, logical, relational form. Currently the standard way of doing this is using expert-system-style knowledge encoding [Jackson, 1998], but the expert system approach is extremely slow and requires highly trained users, so an alternative is badly needed.

Interactive knowledge entry is of course not as good as purely unsupervised AI-based natural language

[1] **I**ntelligent **N**atural **L**anguage **IN**terface for **K**nowledge Entry

understanding would be – if the latter worked fully effectively. But lacking human-level AI software, interactive natural language entry is currently the best we can do.

However, due to the adaptive learning intrinsic in the Novamente integrative AI architecture [Looks et al, 2004] on which the INLINK framework is based, the system learns from experience. Over time, as enough knowledge is entered interactively, the system's capability for accurate unsupervised language understanding will improve. We have already seen this happen, qualitatively, particularly in the area of sense disambiguation.

Here we will discuss INLINK from a user perspective, and report some experiences of actual users entering knowledge into the system; we will then briefly review the computational linguistics and AI underlying the system. The treatment of the latter will naturally be brief, but we will touch on the key ingredients, which include the Novamente AI Engine with its special variety of semantic-network knowledge representation, and the Sleator and Temperley link parser [Sleator and Temperley, 1993].

2 The INLINK Interactive Experience

From a user perspective, the INLINK system consists of a Java user interface that connects to a remote server. Some of the system's linguistic and AI functionalities are explicitly exposed to the user whereas others remain concealed in the "back end" of the software application.

The INLINK interface presents the user with two main functionalities: knowledge entry and querying. To enter knowledge, the user creates a "context" or opens an existing one, and then enters a series of sentences in that context. The entry of each sentence is a multi-step, iterative process; and the system explicitly interprets each sentence in relation to the other ones in the context. Typically a context might refer to a single news article, or a single message.

Querying is quite similar to knowledge entry, in that it also involves interactively entering a sentence within a certain query context -- the difference being that the sentence is a question rather than a statement, and that in the case of querying, the user gets an answer back. Queries may be one-time or persistent; the latter meaning that the user is periodically "pushed back" new answers as new information comes into the system.

The sentence entry process consists of multiple steps. The user types in a sentence and clicks the PROCESS button and the system tries to parse the sentence. Sometimes it fails and the user needs to reformulate. When the parsing process succeeds – which is most of the time, for experienced users – the user is then presented with a selection of several parses to choose from. In the case of complex sentences there could be dozens of parses, but for simple senses there are often 1, 2 or 3 parses. The main reason we chose the link parser for our first version of INLINK is that quite often it finds the correct parse in the first few sentences.

Parses are presented to the user in a simple graphical format inspired by the linguistic notion of "subcategorization frames." For instance, a rough depiction of the subcategorization frames view of the sentence "He was responsible for the bombing" is as follows:

	subjAGENT		*for*
responsible = [	he	responsible	bombing]
	singular	*%past*	*uncountable*

The view in the actual product looks somewhat different due to color coding and clickable hyperlinking. A typical parse of a typical sentence will lead to 3-8 different frames of this form, which the user must read and understand in order to validate or reject the parse.

Having selected a parse, the next step is for the user to check that the system has chosen the right word meanings for the words in the sentence, and the right subject-argument relations for various argument-bearing words in the sentence. WordNet [Fellbaum, 1998] is used as the basic resource for word meanings, but since WordNet doesn't cover prepositions and other ambiguous function words, we have augmented WordNet with our own WordNet-like resource covering these additional words. The user sees the definitions INLINK has selected and if one of these isn't right, the user can select the correct one from a menu, or create a new word sense.

Next comes reference resolution: the system guesses, for each word in the sentence, whether it refers to some other word in the current context or in its general knowledge base. The user may correct its guesses and may also enter in new referents. Here some fairly simple heuristics let the system guess correctly most of the time. For instance, quite often "he" refers to the last male mentioned in the context.

Finally, entity categorization involves placing words in some basic categories such as person, organization, time, place and number. The system tries to identify all these but occasionally it makes errors which the user can correct.

Having gone through all these steps – which can take the user anywhere between 30 seconds and 10 minutes[2] depending on the complexity and novelty of the sentence and the savvy of the user – it's finally time to click SUBMIT and send the correctly interpreted sentence to the server, where it will be entered into Novamente's semantic network in fully logical, relational form.

Practical Experience with INLINK

How well does this somewhat complex interactive process work in practice? In late 2004 two individuals untrained in computer science or linguistics began using the INLINK system to input a body of knowledge regarding real-world events. Feedback from these users led to a number of refinements to the interface and the parser – but

[2] The computational processing is quite rapid; the vast bulk of the time is occupied with the user reviewing the computer's output and making decisions.

overall the reported user experience has been reasonably positive so far.

The most difficult aspect of INLINK, from a user perspective, has proved to be the disambiguation of prepositions and subject-argument relationships. Also, the error rate of users in parse selection has been found to be fairly high for long and complex sentences, but fairly low for short sentences. Thus users are now strongly encouraged to break long sentences down into series of short sentences – a practice that causes INLINK to parse a higher percentage of sentences anyway. The system's reference resolution capability makes it easy to link a series of short sentences together.

Users become competent fairly easily at rephrasing sentences into series of short sentences that INLINK could easily understand. The following example shows a sentence from a test corpus in its original form, and then in the form in which it was rephrased for INLINK comprehension:

> *Original sentence*:
>
> THE SOURCE LATER FOUND OUT THAT ALTHOUGH NO SUCH MEETING HAD EVER TAKEN PLACE, HASSAN AL HURDABI TOLD HER HOW WELL HIS PRESENTATION HAD BEEN RECEIVED BY THE INTERNATIONAL AUDIENCE.
>
> *Rephrased sentence (perfectly comprehended):*
>
> The source later found out that the meeting never occurred.
>
> Hurdabi told her the presentation of Hurdabi had been received well by the international audience.

3 The Novamente AI Engine

INLINK has been constructed within the Novamente integrative AI framework. While the current version of INLINK makes only modest use of Novamente's learning and reasoning capabilities, it makes heavy use of Novamente's knowledge representation, which is a special form of semantic network.

The Novamente AI Engine is a unique, integrative AI architecture with general intelligence ambitions, which bridges the gap between symbolic and subsymbolic AI using a complex systems approach. It is implemented in C++ for Linux for efficiency and scalability, and architected to support distributed computing. AI-wise, Novamente uses two main cognitive tools – Probabilistic Term Logic (PTL) and the Bayesian Optimization Algorithm (BOA) [Pelikan, 2002] -- to generate numerous cognitive processes operative in its multiple functionally specialized lobes.

The real essence of the Novamente design lies in its learning dynamics, but we will not broach that topic here, though we will mention some relevant uses of Novamente-based probabilistic inference a little later on. The aspect of Novamente most relevant to the current version of INLINK is its knowledge representation, which we will now briefly discuss.

Novamente Knowledge Representation

Knowledge representation in Novamente involves two levels, the explicit and the emergent: we will discuss only the former here, for sake of compactness and simplicity. Emergent knowledge in Novamente has to do with activation patterns emerging from the action of Novamente dynamics on explicitly represented Novamente knowledge.

Explicit knowledge representation in Novamente involves discrete units called Atoms, which are of several types: *nodes*, *links*, and *containers* (the latter two are ordered or unordered collections of atoms). The network of nodes, links and containers can be thought of as a "knowledge network," similar but not identical to traditional semantic networks.

Each Atom is associated with a *truth value*, indicating, roughly, the degree to which it correctly describes the world. Novamente has been designed with several different types of truth values in mind; the simplest of these consists of a pair of values denoting probability and weight of evidence. All Atoms also have an associated *attention value*, indicating how much computational effort should be expended on them. These contain two values, specifying short and long term importance levels.

Novamente node types include

- ConceptNodes, which derive their meaning via interrelationships with other nodes
- PerceptNodes: nodes representing perceptual inputs into the system (e.g., pixels, words, points in time, etc.)
- TimeNodes representing moments and intervals of time
- PredicateNodes representing complex patterns
- SchemaNodes embodying procedures

SchemaNodes and PredicateNodes are nodes containing procedures that output Atoms and truth values, respectively. Procedures in Novamente are objects that produce an output, possibly based on a sequence of atoms as input. These objects contain structures called *generalized combinator trees* -- small computer programs written in a special language utilizing ideas from combinatory logic as originally introduced in [Curry and Feys, 1958].

There are also special-purpose predicates that, instead of containing combinator trees, represent specific queries that report to the Novamente system some fact about its own state – these are called "feeling nodes". And finally, some predicates may also be designated as "goal nodes", in which

case the system's GoalSatisfaction MindAgent allocates effort towards making them true.

Links are Atoms that represent relationships between other Atoms, such as fuzzy set membership, probabilistic logical relationships, implication, hypotheticality, and context. The complete list of (a few dozen) types and subtypes of links used, and the justifications for their inclusion, are omitted here for brevity. However, the most essential links are

- Inheritance links (representing probabilistic logical implication)
- Similarity links (a symmetric version of Inheritance)
- Evaluation links (representing the relation between a predicate and its argument)
- ExecutionOutput links (representing the application of a function to an argument).

INLINK also makes heavy use of some specialized links such as WSLinks (WordSenseLinks), that links WordNodes (a kind of PerceptNode) to ConceptNodes.

4 The INLINK Language Processing Pipeline

We now briefly describe what happens inside the INLINK software system, utilizing the Novamente integrative AI architecture and knowledge representation and enabling the user experience described in Section 2 above.

The current version of the INLINK NL comprehension pipeline consists of three stages. In the first stage, the syntax-processing component uses a lightly customized version of the Sleator and Temperley "link parser," and a collection of software objects called "Semantic Algorithms," to convert an English sentence into a list of possible semantic representations ("synsem parses"). Also, at this stage, a collection of commercial and open-source entity extractors is used to mark up words and phrases in the sentence that are likely to correspond to particular types of entities like people, places and times.

In the second stage, the Novamente AI Engine chooses the best representations from the list and disambiguates them by finding specific nodes that represent the meanings of words and argument relations. As noted above, users are able to manually override Novamente's choices regarding parse selection and disambiguation by selecting alternate choices within the INLINK user interface.

Finally, once this is done, the knowledge embodied in the user's sentence is relayed to Novamente, where it can be queried and reasoned about.

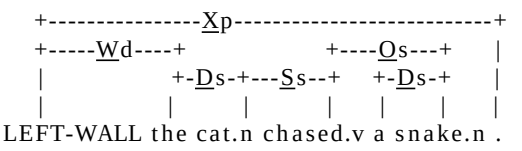
Link Grammar

Link grammar is an unusual NL parsing framework in that it does not involve an explicit notion of phrase structure. This causes it to deal with some linguistic constructs awkwardly, but it also allows the link parser to rapidly find correct parses within its top 1-5 choices in very many cases.

For instance, the link grammar parse structure for the sentence

The cat chased a snake

looks like:



Each of the links shown joins two words and has a particular type which embodies a particular aspect of syntax. For instance, the S link goes between a subject and a verb; the O link goes between an object and a verb.

The link grammar dictionary assigns a collection of link types to each syntactic sense of each word. The parsing process involves drawing links connecting words, consistently with the link parser dictionary and with a set of constraints including the rule that the links in a sentence should form a connected graph, and should not cross.

Mapping Parse Trees into Semantic Structures

The output of the link parser is a collection of syntactic links spanning WordNodes. These must then be turned into Novamente-style semantic nodes and links (e.g. logical Inheritance links, and Evaluation links joining semantically meaningful predicates to their arguments), via the combined action of a pool of Semantic Algorithm objects.

A very simple example of an INLINK Semantic Algorithm is the rule stating that transitive verbs (a syntactic category) map into transitive actions (a semantic concept). Of course, in itself this is not a very useful semantic algorithm, because it's so overly simple. In fact different transitive verbs have different semantics, which leads to a collection of subcategories of TransitiveAction.

Very roughly one might say that one Semantic Algorithm exists corresponding to each link type in the link grammar. But this rule of thumb is broken many times: for instance, ditransitive verbs are not handled very naturally in the link grammar, so the Semantic Algorithms that handle them have to deal with multiple link grammar links. And conjunctions are handled awkwardly by the link grammar, largely in its "postprocessor" phase, so these must be handled somewhat complexly at the Semantic Algorithm stage.

A simple system of semantic "case roles" is used by Semantic Algorithms to mediate the transformation from natural language parses into Novamente nodes and links. For example, the object of a verb may be assigned roles such as objTARGET or objDESCRIPTEE.

INLINK's case roles are partially based on the scheme defined in [Cannesson and Saint-Dizier, 2002] and are also closely related to those used in other knowledge resources such as the SUMO ontology [Niles and Pease, 2001], a fact which facilitates the integration of SUMO knowledge into Novamente to aid with reasoning on nodes and links derived via INLINK.

Semantic Disambiguation and Reference Resolution

Semantic disambiguation, in the current version of INLINK, is carried out partly using WordNet based algorithms loosely similar to those in [Patwardhan et al, 2003]. However, we have found that the subtlest aspects of semantic disambiguation have to do with words and linguistic constructs not covered by WordNet: prepositions and subject-argument relationships, for example.

In practice, in an interactive context, it is not very hard to guess the correct meaning of a noun, verb, adverb or adjective. Such words tend to be used in the same sense repeatedly by the same user, and very strongly tend to be used in the same sense repeatedly within the same context. On the other hand, a word like "by" will generally be used in different senses from sentence to sentence.

In order to handle this difficulty we have implemented an adaptive algorithm, which assigns each preposition or subject-argument relationship a sense based on the senses more recently assigned to the entity when used in combination with words similar or identical to the words it's currently used with. For instance, if user types in the phrase "He walked by the store," and tells INLINK the correct meaning of "by" in this context, then INLINK will make the correct guess for the sense of "by" next time it sees the sentence "She drove by the restaurant."

Reference resolution is handled in a similar manner: pronouns and other words that explicitly require referents are assigned referents based on analogy to recent history, a simple approach that works quite well. In this case some very simple adaptive learning is able to effectively leverage a modest amount of user feedback. Usually INLINK guesses the referent of "he" correctly, but if it gets it wrong the first time in a particular context and is corrected, it will very rarely get it wrong the second time.

Semantic Normalization of Linguistically-Structured Knowledge

One problem that arose immediately upon trying to apply Novamente's probabilistic inference algorithms to the output of INLINK's semantic algorithms was the wide divergence of representations provided for highly semantically similar sentences. To illustrate this issue, we will show here three representations for the following three almost-semantically-identical sentences, produced by INLINK's semantic analysis component.

For sake of compactness, in these examples we omit WSLinks and the like, and show only semantically meaningful links between ConceptNodes. ConceptNodes and SpecificEntityNodes are denoted by the names of the WordNodes most closely linked to them, and other nodes such as those denoting tense (e.g. %pres_ongoing) are denoted by intuitive shorthand names

Finally, in these examples, links are shown in a relational-logic style, where the notation R(X,Y) is used both for Novamente link types R and for predicates R, i.e. it may mean either

- that a link of type R exists between the node or link X and the node or link Y
- that an Evaluation link exists between the predicate R and the List Atom (X, Y)

For the present purposes this distinction is not an important one (though it is important for Novamente dynamics). Note also that in Novamente links may span links as well as nodes.

Without further ado, the three examples:

Amir is a friend of James.

Inheritance(B,be)
Tense(B,%pres_ongoing)
objTARGET2(B,F)
subjDESCRIPTEE(B,B1)
Inheritance(B1,Amir)
Inheritance(F,friend)
ofDESCRIPTEE(F,O)
Inheritance(O,James)

Amir and James are friends

Inheritance(B,be)
Tense(B,%pres_ongoing)
objTARGET2(B,F)
subjDESCRIPTEE(B,group^777)
Inheritance(B1,Amir)
Inheritance(B1,group^777)
Inheritance(F,friend)
Inheritance(O,James)
Inheritance(O,group^777)

Amir is James's friend

Inheritance(B,be)
Tense(B,%pres_ongoing)
objTARGET2(B,F)
subjDESCRIPTEE(B,O)
Inheritance(B1,Amir)
Inheritance(F,friend)
possFOCUS2(F,B1)
Inheritance(O,James)

This sort of divergence of representation is problematic both for query processing and for inference.

In order to get from the linguistic representation of this knowledge exemplified above to a more inference- and query-processing- friendly representation, a collection of transformation rules must be applied – rules similar in theme but different in detail from the Semantic Algorithm transformations that map syntactic nodes and links into semantic nodes and links.

In general, at this stage, we require roughly one semantic transformation for each subject-argument relationship (e.g. subjAGENT) and each preposition sense (e.g. ofFOCUS) and also for senses of common "glue" verbs such as "be." These transformations are themselves represented as nodes and links and are executed via Novamente inference.

To give examples of these transformations in any detail would take us too far afield as it would require us to enlarge more deeply upon the topic of Novamente knowledge representation. However, a simple example is the transformation for ofDESCRIPTEE, which looks like

ForAll R, X, Y: ImplicationLink(foo1, foo2)
foo2 = (ofIze(R))(X,Y)
foo1 = AND(ofDESCRIPTEE(R,Y), R(X))

where *ofIze* is a Novamente SchemaNode corresponding to the meaning of the relevant sense of the word *of*.

Query Processing

Query processing within the INLINK framework is not a trivial task, because of the above-mentioned phenomenon of semantic diversity. If the user phrases his query significantly differently from the way the knowledge matching the query was entered into the system, then the match between the query and the knowledge will not be all that direct within Novamente's knowledge network – not unless substantial semantic normalization has been done.

The "correct" way to do query processing in INLINK would be using Novamente's probabilistic inference module, and this is intended for the next version of the system. The current INLINK version, however, uses a simplified approach based on dynamic programming, which basically takes the query, parses it into a small semantic network, and then searches Novamente's memory for other sub-networks that closely match the query-network. This approach works quite effectively for small and moderately-sized knowledge bases, but it lacks the capability for generalization and analogical and abductive inference that will be displayed by the inference-based version.

As a trivial but illustrative example of query processing, consider the query

Where does Ben work?
subjDESCRIPTEE(work,Ben).
LOCATION(work,$Y).

as matched against the entered knowledge

Ben works in the USA
subjDECRIPTEE(work,Ben)
objLOCATION(work,USA)

Note how the semantic mapping of a question looks just like the semantic mapping of a statement, except for the presence of a variable $Y (to be filled in via the action of query processing). In this case the matching is trivial as the sentences aren't complex enough for any "divergence of form" subtleties to appear.

5 Future Work

The current version of INLINK has successfully demonstrated the concept of interactive natural language processing. However, many short-cuts were made in its construction, and there are a number of well-understood ways to dramatically improve the system's parsing coverage, as well as its ability to learn through experience and to process queries using analogy and generalization.

In the next version of INLINK, hopefully to be released in late 2005, several important changes will be made.

First, the link parser will be replaced with a new parser that utilizes (a modified version of) the same grammar but operates using Novamente inference and optimization algorithms rather than the link parser's parsing algorithm. This will allow deeper integration between syntactic and semantic analysis, and improve the linguistic coverage of the system.

Second, Novamente's probabilistic reasoning module will be utilized for query processing and also to aid with semantic disambiguation and reference resolution.

Third, the framework will be extended to allow language generation as well as language comprehension. A prototype of this functionality has already been created; while far from simple, this "reversal" is fairly conceptually straightforward within the Novamente/INLINK framework.

And finally, the various processing objects such as Semantic Algorithms and transformation rules will be

encoded in a way that allows Novamente to adapt them via experience – thus providing, in principle, a fully adaptive language processing framework that can modify all aspects of its behavior in accordance with what it learns from its human interactors.

References

[Cannesson and Saint-Dizier, 2002] Emmannuelle Cannesson and Patrick Saint-Dizier. Defining and Representing Preposition Senses – A preliminary Analysis, Proceedings of the SIGLEX/SENSEVAL *Workshop on Word Sense Disambiguation.* Philadelphia, July 2002.

[Fellbaum, 1998] Christiane Fellbaum, Editor. *WordNet: an Electronic Lexical Database*. MIT Press, Cambridge Massachussets, 1998.

[Jackson, 1998] Peter Jackson. *Introduction to Expert Systems.* Addison-Wesley, New York, 1998.

[Looks et al, 2004] Moshe Looks, Ben Goertzel, Cassio Pennachin. Novamente: An Integrative Approach to Artificial General Intelligence. In *Proceedings of the AAAI Symposium on Achieving Human-Level Intelligence through Integrated Systems and Research*, Washington DC, October 2004.

[Niles and Pease, 2001] Ian Niles and Adam Pease. Towards a Standard Upper Ontology. In *Proceedings of the 2nd International Conference on Formal Ontology in Information Systems* (FOIS-2001), Ogunquit, Maine, October 2001

[Patwardhan et al, 2003] Siddharth Patwardhan, Sattanjee Banerjee and Ted Pedersen. Using Measures of Semantic Relatedness for Word Sense Disambiguation. In *Proceedings of the Fourth International Conference on Intelligent Text Processing and Computational Linguistics*, Mexico City, February 2003

[Pelikan, 2002] Martin Pelikan. *Bayesian Optimization Algorithm: From Single-Level to Hierarchy.* PhD Thesis, Department of Computer Science, University of Illinois at Urbana-Champaign, 2002.

[Sleator and Temperley, 1993] Daniel Sleator and Davy Temperley. Parsing English with a Link Grammar. *Third International Workshop on Parsing Technologies*, Tilburg, The Netherlands, 1993