On the Application of the Cyc Ontology to Word Sense Disambiguation†

Jon Curtis, John Cabral, David Baxter
Cycorp, Inc. 3721 Executive Center Drive, Suite 100, Austin, TX 78731.
{jonc|jcabral|baxter}@cyc.com

Abstract

This paper describes a novel, unsupervised method of word sense disambiguation that is wholly semantic, drawing upon a complex, rich ontology and inference engine (the Cyc system). This method goes beyond more familiar semantic closeness approaches to disambiguation that rely on string co-occurrence or relative location in a taxonomy or concept map by 1) exploiting a rich array of properties, including higher-order properties, not available in merely taxonomic (or other first-order) systems, and 2) appealing to the *semantic contribution* a word sense makes to the content of the target text. Experiments show that this method produces results markedly better than chance when disambiguating word senses in a corpus of topically unrelated documents.

1 Introduction

The problem of word sense disambiguation—the question of how to determine which meaning of a word is the correct meaning for an occurrence of that word in context—has long been considered one of the most difficult yet promising research areas in computational linguistics. Significant headway into its resolution would herald great advances in NLP-based applications, from content-based e-mail filtering and document retrieval, to topic-based document classification, semantic text annotation, and question answering.

Work in the area of word sense disambiguation has typically involved attempts to define and implement algorithms for determining the relative *semantic closeness* of competing word senses to the senses of words that occur in the same context. Such algorithms typically embody or combine one of two approaches: The *lexical* approach identifies the sense of a word with a set of strings, arrived at either by an analysis of a disambiguated training corpus, or by extracting strings that occur in a definition of that sense. The lexical approach to sense disambiguation thus bottoms out in string co-occurrence: Where "bat" occurs ambiguously between *BAT-1*, a set of strings including "wing," "small," "mammal," "nocturnal," and *BAT-2*, a sense that includes "device," "blunt," "strike," then the occurrence of "flapping" in the target text, whose sense *FLAP-1* contains "wings," will count in favor of *BAT-1*.

By contrast, the *taxonomic* approach to the problem of word sense disambiguation identifies the senses of words with nodes in a hierarchy of senses, and uses relationships among those nodes to form measures of semantic closeness. Keeping with our example, analogous to *BAT-1* on the lexical scheme would be a taxonomic entry, *Bat_Chiroptera* that occurs as a more specific node under *Mammal*and *Winged_Animal*, under which *Wing* occurs as a partonomic sub-node. A reference in the target text to the concept designated by *Wing* would serve to validate *Bat_Chiroptera* over the competing *Bat_Device* as the most likely meaning of "bat" in that context, given the relative closeness of *Wing* to *Bat_Chiroptera*.

With the free availability of online dictionaries and thesauri, NLP research in the area of word sense disambiguation has tended generally toward refinements of and moderate extensions to lexical and taxonomic-based semantic closeness approaches. WordNet in particular has become something of a standard tool in this regard, as it supports both lexical approaches (via synonym-sets and glosses) as well as taxonomic approaches (by virtue of organizing word senses (of nouns) into a hierarchy).[1] Relatively little has been done in the way of applying a full-scale *ontology* to the problem of sense disambiguation. The theoretical benefits of doing so include the availability of a much richer set of relations among word senses, enabling the development of more robust mechanisms for determining semantic closeness. Additionally, a rich ontology holds out the possibility of representing some or all of the content of a target text in the formalism used to express that ontology.

In what follows, we describe an application of the Cyc ontology and inference engine to the problem of sense disambiguation that 1) contains a method of determining semantic closeness of word senses that draws upon complex relations among concepts in the Cyc ontology, including

† Distribution Statement A: Approved for Public Release, Distribution Unlimited.

[1] Interesting, recent lexical approaches include those described in [Agirre and Lopez de Lacalle, 2003], which involves tying semantic closeness of two words in a given language with their being translated to the same word in a second language, and [Cavaglia, 1999], which uses Dewey Decimal classification to supplement WordNet synsets with strings associated with topics. [Pederson, *et al*., 2005] describes the use of WordNet in a combined lexical/taxonomic approach.

higher order relations not supportable in any taxonomic system; and 2) includes a novel method of gauging whether a word sense makes a *semantic contribution* to the content of the target text, by appealing to partial formalizations of the text. Experiments involving a 13,535-word corpus consisting of 4 topically disparate documents show that this approach performs markedly (69.04%) better than chance. We first turn to a description of Cyc, past uses of it in systems to perform limited sense disambiguation, and the TextLearner prototype, of which the disambiguation methodology described is an integral part. We then describe the algorithms for determining semantic closeness and semantic contribution, and report the results of experimentation. We close with a brief discussion of future directions for improving this approach.

2 Cyc

In development since 1984, the Cyc Knowledge Base (KB) is a general-purpose repository of common sense and specialized knowledge consisting of over 328,000 concepts and over 3,500,000 explicitly declared assertions.[2] Knowledge in Cyc is represented in CycL, a higher-order logical language based on predicate calculus. Every CycL assertion occurs in a context, or *microtheory*, allowing for the representation of competing theories, hypotheses, and fictional claims. Cyc's inference engine combines general theorem proving (rule chaining) with specialized reasoning modules to handle commonly encountered inference tasks, such as transitivity.

The higher order features of CycL support both high expressivity and efficient inferencing [Ramachandran, *et al*., 2005]. In particular, Cyc is able to declaratively represent and reason about the properties of collections, relations, and CycL sentences. As will become clear, this level of expressiveness—unobtainable in a first order or taxonomic representational scheme—underlies both the robust notion of semantic closeness applied here as well as the notion of semantic contribution used to augment it.

2.1 Word Senses in Cyc

The representation of word senses in Cyc is achieved primarily declaratively in the Cyc Knowledge base, using vocabulary for the representation of words, their properties (e.g., part-of-speech and string pairings) and specialized mapping predicates. The predicate #$denotation is the relation most frequently associated with the mapping of concepts of natural languages into Cyc. For example, (#$denotation #$Bat-TheWord #$CountNoun 0 #$Bat-Mammal) means, in effect, that there is a count noun sense of the word "bat" which is represented in the Cyc ontology as #$Bat-Mammal. Semantic ambiguity is recognized by the system when a competing mapping for a word-and-part of speech pair is entered into the Cyc lexicon. So Cyc is in a position to recognize "bat" as semantically ambiguous in "He spotted the bat," if, in addition to the count noun mapping of #$Bat-TheWord to #$Bat-Mammal, there is a second count noun mapping to #$Bat-Device.

Unlike dictionary-based taxonomies such as WordNet, in which every node in the system is identified with a word sense, the Cyc ontology is *not* an ontology of word senses. The concepts represented in Cyc are those needed to support commonsense reasoning, one effect of which is that there is no comprehensive mapping of Cyc concepts into words of a natural language, no sense in which each "node" of the Cyc ontology captures a word sense.[3]

At the same time, there are many word senses of a given language—particularly more obscure word senses or senses that are idiosyncratic to a particular field—that are not explicitly represented in Cyc. Again, this is a consequence of the minimization of concepts to only those needed to represent consensus reality. As WordNet contains an enormous number of fine-grained senses, similarities among which make disambiguation difficult [Deibel, 2004], this makes a direct apples-to-apples comparison with a Cyc-based approach to sense disambiguation difficult. The eXtended WordNet project [Mihalcea and Moldovan, 2001] is an attempt to collapse word senses so as to make a version of WordNet that is less susceptible to the difficulties of distinguishing among fine-grained senses; the resultant product should be more closely aligned with Cyc in terms of the relative number and coarse-grainedness of word senses in Cyc, making apt comparisons to (future) applications based on eXtended WordNet possible.

2.2 Relevant Previous Applications

Cyc has been used to perform or facilitate some level of word sense disambiguation in a number of systems. These include a commercial web-search enhancement system, a commercial question-answering system described in [Curtis, *et al*., 2005] and an interactive knowledge acquisition system described in [Witbrock *et al*., 2003]. In these systems Cyc draws upon mostly taxonomic knowledge (Cyc's #$isa (membership) and #$genls (subset) hierarchies) to generate disambiguating strings used in clarification questions posed to end-users. For example, in response to an ambiguous noun, "turkey," these applications would ask the user whether "turkey (bird)," "turkey (meat)," or "Turkey (country)" was meant.

Some of these applications are able to perform a limited degree of automatic word sense disambiguation by appealing to context, at a sentence-level of granularity. The Recursive Template Parser (RTP) and the Cyclifier are two NL-

[2] These are statistics for the full Cyc KB, which is not (yet) publicly available. A freely available version of Cyc can be downloaded at http://www.opencyc.org/.

[3] For example, the CycL term #$TemporalStuffType is needed to represent a property absolutely essential for understanding the persistence of objects through time (in particular where (#$isa FOO #$TemporalStuffType) and (#$isa BAR FOO), then every time-slice SLICE of BAR is such that (#$isa SLICE FOO) – so that, e.g., every time-slice of an instance of #$Animal is itself an instance of #$Animal). Despite the importance of that the concept represented by #$TemporalStuffType has for temporal projection, it does not correspond to the sense of any word or common-parlance phrase of a natural language.

to-CycL transformation modules that assign CycL semantics to English, based on a string-matching technique in the case of the RTP,[4] and based on a syntactic analysis of the input English and declaratively represented semantic translation templates in the case of the Cyclifier. The applications that use sentence-level NL-to-CycL transformation modules apply Cyc's knowledge of formula operator argument constraints to rule out word senses that violate those constraints. For example, "Gates" as the subject of "Gates founded Microsoft" is ambiguous between *Bill Gates* or *door of a fence*, represented in Cyc as #$BillGates and #$Gate, respectively.[5] Given this sentence as input, the RTP would produce two possible interpretations: (#$foundingAgent #$MicrosoftInc #$BillGates), meaning *Bill Gates founded Microsoft*, and (#$thereExists *?X* (#$and (#$isa *?X* #$Gate) (#$foundingAgent #$MicrosoftInc *?X*))), meaning *some gates founded Microsoft*. However, the second argument-place of #$foundingAgent is constrained to be an *agent*; only #$BillGates meets that criteria, allowing the system to reject the #$Gate interpretation.

This sort of methodology has been used in mechanisms for automatically acquiring knowledge from the web. [Matuszek, *et al*., 2005] describes a knowledge-acquisition tool that generates search strings from open CycL formulae and gathers possible answers from the world wide web, parsing the candidate answer-strings into CycL and substituting the CycL into the open formula. In cases of ambiguity, Cyc uses the argument constraints imposed by the formula operator to reject semantically impossible interpretations.

3 TextLearner

TextLearner is a Cyc-based application designed to acquire knowledge by reading text.[6] Unlike previous Cyc-based programs of the kind described above, TextLearner is designed to maximize context sensitivity. To this end, TextLearner builds an explicit, formal representation of the structure of the documents it reads. Documents are represented in the knowledge base as a series of nested linguistic structures, called "contextualized information structures," or "CISes." CISes are represented from the level of paragraphs down to the level of individual phrases, enabling the system to reason about particular occurrences of a word in relation to particular occurrences of other words, sentences, and paragraphs.

TextLearner approaches the problem of disambiguation by generating all of the possible interpretations that it can, explicitly relating them to CISes as competing hypotheses. For example, if a document contains a string "Washington" that is ambiguous between Washington State, the District of Columbia, and the first US President, TextLearner will explicitly represent three competing (inconsistent) claims as to what this particular occurrence of "Washington" means. These hypotheses are represented as sibling microtheories; their respective content is inaccessible to one another. This allows Cyc to reason *about* the hypotheses, exploring their semantic consequences and how they relate to other hypotheses, without generating contradiction.

The treatment of semantic interpretations as hypotheses enables the system to defer the resolution of ambiguity until "all the evidence is in," so to speak. At any point during the processing of a document, the Cyc inference engine can query the knowledge base for an assessment as to which semantic interpretations the system currently prefers; however, as more information is processed by the system, those preferences can change.

3.1 Determining Semantic Closeness

As noted in the introduction, methods of gauging semantic closeness based on the relationships among concepts in a formalized (typically taxonomic) system are nothing new. TextLearner differentiates itself by bringing the full weight of a rich ontology to bear. This includes an appeal to higher order properties unavailable in a taxonomy, but easily expressible in the language in which the Cyc ontology is described. For example, CycL contains its own semantic closeness vocabulary. Two such relations used by TextLearner in this regard are #$nearestIsa and #$nearestIsaOfType. (#$nearestIsa INSTANCE TYPE) means that of all the collections of which INSTANCE is provably an instance, TYPE subsumes none of them. For example, #$BillGates is an instance of many collections, including #$ComputerScientist and #$Person. However, none of those collections are subsumed by #$ComputerScientist, whereas many of them, including #$ComputerScientist, are subsumed by #$Person. Thus when Cyc is asked to find those collections that are semantically closest to #$BillGates via collection membership, #$ComputerScientist, and not #$Person will be among the answers returned.

Similar to #$nearestIsa, #$nearestIsaOfType allows for more reified "nearness" criteria, by constraining what type of collection the nearest #$isa must belong to. (#$nearestIsaOfType INSTANCE COLLECTION-TYPE TYPE) means that of all the collections that 1) are instances of

[4] The Recursive Template Parser is a top-down string-matching parser driven by thousands of hand-crafted parsing templates. Each template specifies 1) a simple pattern of variables and strings to be matched against parser input, 2) part of speech constraints for unmatched constituents (variables), and 3) an open CycL formula that serves as the basis for target semantics. For example, a template that targets #$foundingAgent might contain the pattern <NP1 "founded" NP2>, where NP1 and NP2 are constrained to be noun phrases, with (#$foundingAgent (#$ResultOfParsing NP1) (#$ResultOfParsing NP2)) as the semantic target. The two #$ResultOfParsing expressions denote the CycL semantics that result from parsing NP1 and NP2 as noun phrases. "Gates founded Microsoft" meets the string-matching pattern of the template, and "Gates" and "Microsoft" are both parsable as NPs, enabling the system to produce CycL interpretations. For more information about the RTP, see [Panton, *et al*., 2002] and [Witbrock, *et al*., 2003].

[5] Note that there are no lexical grounds for interpreting "Gates" as a proper name over a bare plural such as "landowners" as one might see in "Landowners founded this country."

[6] TextLearner is being developed under the Learning Reading Comprehension seedling project funded by DARPA/IPTO.

COLLECTION-TYPE and 2) that have INSTANCE as a member, TYPE is a nearest, or non-subsuming one.[7]

Though #$nearestIsaOfType is useful for constructing semantic closeness measures for any concept in the ontology, it is particularly useful when the closeness of collections is at issue.

For example, the collection #$FacetingInstanceCollection is a collection of collections whose instances are members of a faceting collection—a collection that facets a given collection into a coherent family of specializations. For example, #$PersonTypeByOccupation is a collection that facets #$Person by gathering *occupations*, such as #$Lawyer, #$Doctor, #$ComputerScientist, and the like—into a single collection. As such, the elements of #$PersonTypeByOccupation are all instances of #$FacetingInstanceCollection: They are collections that belong to a faceting of #$Person into occupations. #$FacetingInstanceCollection is a good example of a collection that can be used to articulate a higher-order closeness metric in terms of #$nearestIsaOfType that is much more discriminating than #$nearestIsa: It allows us to reward a possible interpretation of some word or phrase just in case there is another possible interpretation of a (different) word or phrase in context for which TYPE is its nearest instance of #$FacetingInstanceCollection. Thus an article that mentions occupations such as "teacher" and "principal" will, using this metric, reward an interpretation of "coach" *qua* person over "coach" *qua* level of accommodation/service afforded during travel.

The uses of #$nearestIsa and #$nearestIsaOfType are augmented by a number of other semantic measures, such as the occurrence of word sense-pairs that stand in an #$isa or (proper) #$genls relationship, and an appeal to a particular *hierarchy* of binary predicates.[8] At the top of the relevant hierarchy is #$conceptuallyRelated: (#$conceptuallyRelated THING-1 THING-2) means that there is a *directed semantic closeness*—a path through the ontology from THING-1 to THING-2—that, loosely characterized, makes references to THING-2 more likely given a reference to THING-1. For example, (#$conceptuallyRelated #$FireTruck #$RedColor) means that the concept *fire truck* is conceptually related to the concept *red*, so that we would expect a reference to fire trucks to make a reference to the color red more likely (though the converse is not expected to hold). More often than not, #$conceptuallyRelated links are not declared explicitly in Cyc; rather, they are represented using more specialized relations. For example, the assertion (#$colorOfType #$FireTruck #$RedColor), which means that fire trucks are (typically) red, entails a #$conceptuallyRelated link from "fire truck" to "red," by virtue of the fact that #$colorOfType generalizes to #$conceptuallyRelated.

[7] Though #$nearestIsa is perhaps a more intuitive relation to grasp, it is actually convenient shorthand for the more complex (#$nearestIsaOfType INSTANCE #$Collection TYPE), which merely weakens the constraints on TYPE so as to include any collection.

[8] Note that a hierarchy of relations is a higher order feature of an ontology, and one that can be represented in CycL.

In applying these sorts of semantic closeness metrics, TextLearner draws upon the sentence-level, paragraph-level, and document-level co-occurrence of words to form three classes of weights, values of which are determined by applying an additional distance (offsets) metric between words. Thus a mechanism that rewards the color sense of "red" due to the unambiguous occurrence of "fire truck," would do so more strongly in the sentence "The red fire truck sounded its horn," than in "The fire truck sped toward the red blaze," and even less so in "Where are the fire trucks? Red flames are lighting up the sky." More complex mechanisms for determining semantic contribution are used in conjunction with semantic closeness measures, and it to these that we now turn.

3.2 Determining Semantic Contribution

While lexical and taxonomic approaches to word sense disambiguation both work by forming and applying measures of semantic closeness, the appeal to *semantic contribution* is a novel approach that sets Text-Learner apart. In setting up an explicit model of a document's structure, TextLearner keeps track of individual phrases from the text and records how those phrases relate to their constituents. So, in encountering a phrase, "worn book," Text Learner is able to represent the phrase "worn book," the head, "book," and the fact that "book" occurs as the head of the larger phrase. These larger phrases are sent to NL-to-CycL transformation modules to generate possible CycL semantics. If a word-sense contributes to any CycL interpretation of the larger phrase, this counts in its favor, while failure to contribute to any interpretation counts against it. Keeping with the current example, the word "book" is lexically ambiguous between an *authored work* (as in, "I've read all her books") and a *book copy* ("The book I borrowed is on the table."). However, in the context of the phrase, "worn book," only one interpretation is possible—the physical book copy, which, unlike a conceptual work, can undergo wear and tear.[9]

In some cases, TextLearner is able to gauge the semantic contribution of senses to sentences. Due to the extreme

[9] "worn book" parses compositionally into (#$SubcollectionOfWithRelationToFn #$BookCopy #$physicalStructuralFeatures #$Worn), or the collection of copies of books that have experienced wear-and-tear. This is the only semantics that the CycL-to-NL transformation modules will supply, despite "book" not being the only ambiguous word: other *lexically* possible senses of "worn"—such as *donned, as in garb*, or *fatigued*—are also ruled out on semantic grounds. It is understood that this methodology, as with most any methodology for assigning formal semantics to natural language, fails to account adequately for metaphor. A saying or an idea, for example, might be (though not literally) worn. Except in cases where a metaphor has been lexicalized (e.g., a non-literal use of "worn" might be so common that it warrants its own CycL denotation, in which case it is treated as just another ambiguous word), the use of semantic criteria can, at present, be useful in identifying possible cases of metaphor, but not for semantically interpreting metaphors—a challenge for any non-human system.

difficulty of mapping arbitrary English into a formal language, this is a process that requires a bit more effort on the part of Cyc's inference. Each CycL sentence produced by NL-to-CycL modules is represented as a hypothesis as to what the English input sentence means. Forward rules query those hypotheses for sub-sentences (often, these sentences are enormous conjunctions of clauses) that are provable by the Cyc inference engine. If such a sub-sentence is found, any CycL term that appears in it as an argument, if it is also a candidate sense for a word or phrase from that same sentence, is rewarded as having contributed semantically to a partial-CycL understanding of the English sentence. Consider, for example, the first sentence of the *Wikipedia* article, "Angola": "Angola is a country in Southwestern Africa bordering Namibia, the Democratic Republic of the Congo, and Zambia, and with a west coast along the Atlantic Ocean." The RTP and Cyclifier modules generate numerous possible interpretations, most of which are large conjunctions. Among the provable conjuncts from some of these interpretations is (#$isa #$Angola #$Country). While "Angola" as a name is ambiguous between the country and Angola Prison (Louisiana State Penitentiary), the fact that #$Angola—the Cyc term for the country Angola—contributes to a sentence known by the system to be true, counts heavily in favor of the country interpretation over the prison interpretation.

By appealing to semantic contribution, TextLearner is imbued with a degree of robustness against the brittleness lurking behind any one measure of semantic closeness. For example, in the sentence, "Their albums broke many sales records," the interpretation of "records" as synonymous with "albums" would be erroneously rewarded by a measure that looks for synonymity, but that reward would be countered by a measure that looks to the semantic contribution "records" makes to the phrase "sales records," which favors the "benchmark" interpretation of "record."

4 Experimental Results

4.1 Experimental Setup

The disambiguation approach was tested on a small but diverse selection of *Wikipedia* articles, namely:

- Angola
- Backgammon
- Biology
- Thomas Jefferson

The choice of *Wikipedia* as a source for documents reflects its suitability for machine learning. First, because of its high popularity with the general public, who are free to edit and contribute to it daily, *Wikipedia* represents a large source of up-to-date knowledge about the world and current affairs. Second, the online publication of *Wikipedia* articles as hypertext files, along with fairly uniformly followed conventions for article structure, make the automatic "chunking" of documents into paragraphs and sentences for use by a text-processing program relatively straightforward. The variety of topics covered in the articles selected reflects the intention of the authors to avoid an accidental skewing of experimental results by processing topically uniform text, which might allow the system to unrepresentatively exploit or suffer from non-uniform levels of KB coverage of that topic. As noted above, the method of disambiguation described here is unsupervised; unlike supervised systems, the target text need not be topically or stylistically similar to a training corpus in order to maximize success.

Ambiguous words were categorized according to whether the correct interpretation (as judged by a human) was among the candidates at all, generating a benchmark against which the automatic disambiguation could be judged. As noted in section 2.1, it is frequently the case that a word sense is unknown to Cyc, creating situations in which it cannot possibly choose the right sense, while evidence nevertheless suggests a preference. Such cases were noted during annotation, and not considered when factoring the system's performance or that of chance.

4.2 Results

The below charts show two types of properties of the TextLearner system as it applies to disambiguation. The first chart reflects Cyc's level of coverage with respect to the concepts referred to in the input articles. The first metric was to evaluate the system's ability to include the right concept as a candidate for an ambiguous term. 71% of the time the correct concept was included in the set of candidate denotations. The average number of candidates in those sets was 3.

Document	Usable Entries	Empty Entries	Percent Usable	Mean Size
Angola	229	48	82.67%	2.96
Backgammon	474	293	61.80%	3.29
Biology	301	79	79.21%	2.62
Jefferson	495	177	73.66%	2.99
Cumulative	1499	597	71.52%	3.00

As shown in the second chart, given the average size of sets containing the correct concept, the chance of randomly picking the correct concept was 33.49%. The disambiguation strategies described in Section 3 enabled the system to pick out the correct concept 56.61% of the time, a 69.04% improvement over chance. The average rank of the correct concept within a set was 1.61 (again, compared to an average size of 3).

Document	Chance	Success Rate	Percent Better than Chance
Angola	33.83%	53.64%	58.56%
Backgammon	30.43%	53.80%	76.80%
Biology	38.20%	68.85%	80.23%
Jefferson	33.40%	53.23%	59.37%
Cumulative	33.49%	56.61%	69.04%

5 Future Directions

As mentioned above, the sense disambiguation methods described here form a part of the TextLearner system for acquiring knowledge by reading text. Among the types of knowledge TextLearner is designed to acquire are rules that it can apply to improve its performance. The strategies described above are all parameterized, allowing the system to run repeatedly against the same documents and learn what changes lead to improvements or regressions. This parameterization extends beyond the weighting of existing strategies, and allows the system to develop and experiment with new criteria for semantic closeness. For example, the family of strategies that look for a shared minimal #$isa all involve querying the Cyc KB with a CycL formula of the form (#$nearestIsaOfType *?SENSE* TYPE *?WHAT*), where TYPE is the only variant among them. This makes it possible to for the system to experiment with new possible values for TYPE as it runs against previously disambiguated text, enabling the system to adopt, reject, or weight these new strategies based on how the system performs.

Acknowledgments

This research was done as part of the Reading Learning Comprehension project, sponsored by DARPA/IPTO. The authors would like to thank those colleagues who have contributed to various aspects of the TextLearner program: Keith Goolsbey, Ben Gottesman, Robert Kahlert, Kevin Knight, Cynthia Matuszek, Pace Reagan, Dave Schneider, Brett Summers, Peter Wagner, and Michael Witbrock.

References

[Agirre and Lopez de Lacalle, 2003] Eneko Agirre and Oier Lopez de Lacalle. Clustering WordNet Word Senses. In *Proceedings of the Conferece on Recent Advances on Natural Language Processing*, pages 121-130, Bulgary, 2003.

[Cavaglia, 1999] Gabriela Cavaglia. The Development of Lexical Resources for Information Extraction from Text Combining WordNet and Dewey Decimal Classification. In *Proceedings of the 9th Conference of the European Chapter of the Association for Computational Linguistics*, pages 225-228, Bergen, Norway, 1999.

[Curtis *et al*., 2005] Jon Curtis, Gavin Matthews, David Baxter. On the Effective Use of Cyc in a Question Answering System. In *Proceedings of the Knowledge and Reasoning for Answering Questions Workshop, IJCAI 2005*, pages 61–70, Edinburgh, Scotland, July 30 2005.

[Deibel, 2004] Katherine Deibel. Current Approaches for Unrestricted Word Sense Disambiguation, 2004.

[Mandala, *et al*., 1999] Rila Mandala, Takenobu Tokunaga, Hozumi Tanaka. Complementing WordNet with Roget and Thesauri for Information Retrieval. In *Proceedings of the 9th Conference of the European Chapter of the Association for Computational Linguistics*, pages 94-101, Bergen, Norway, 1999.

[Matuszek, *et al*., 2005] Matuszek, Cynthia, M. Witbrock, R. Kahlert, J. Cabral, D. Schneider, P. Shah and D. Lenat. Searching for Common Sense: Populating Cyc from the Web. In *Proceedings of the Twentieth National Conference on Artificial Intelligence,* Pittsburgh, Pennsylvania, July 2005.

[Mihalcea and Moldovan, 2001] Rada Mihalcea and Dan Moldovan. EZ.WordNet: Principles for Automatic Generation of a Coarse Grained WordNet. In *Proceedings of the Fourteenth International Florida Artificial Intelligence Research Society*, pages 454-458, Key West, Florida, May 21-23, 2001.

[Panton, *et al*., 2002] Kathy Panton, Pierluigi Miraglia, Nancy Salay, Robert C. Kahlert, David Baxter, Roland Reagan. Knowledge Formation and Dialogue Using the KRAKEN Toolset. In *Proceedings of the Fourteenth Conference on Innovative Applications of Artificial Intelligence*, pages 900-905. Edmonton, Canada, July 28-August 1, 2002.

[Pederson, *et al*., 2005] Ted Pederson, Satanjeev Banerjee, Siddharth Patwardhan, Maximizing Semantic Relatedness to Perform Word Sense Disambiguation. University of Minnesota Supercomputing Institute Research Report, University of Minnesota, 2005.

[Ramachandran, *et al*., 2005] Ramachandran, Deepak, P. Reagan, K. Goolsbey. First-Orderized ResearchCyc: Expressivity and Efficiency in a Common-Sense Ontology. In *Papers from the AAAI Workshop on Contexts and Ontologies: Theory, Practice and Applications.* Pittsburgh, Pennsylvania, July 2005.

[Wikipedia, 2005]

http://www.en.wikipedia.org/wiki/Angola

http://www.en.wikipedia.org/wiki/Backgammon

http://www.en.wikipedia.org/wiki/Biology

http://www.en.wikipedia.org/wiki/Thomas_Jefferson

[Witbrock *et al*., 2003] Michael Witbrock, David Baxter, Jon Curtis, Dave Schneider, Robert Kahlert, Pierluigi Miraglia, Peter Wagner, Kathy Panton, Gavin Matthews, Amanda Vizedom. An Interactive Dialogue System for Knowledge Acquisition in Cyc. In *Proceedings of the Workshop on Mixed-Initiative Intelligent Systems*, pages 138–145, Acapulco, Mexico, August 2003.

â€¢ Word Sense Disambiguation: choosing correct sense in context â€¢ Applications: MT, QA, etc. â€¢ Three classes of Methods. â€¢ Supervised Machine Learning: Naive Bayes classifier â€¢ Thesaurus/Dictionary Methods â€¢ Semi-Ââ€Supervised Learning. â€¢ Main intuition. â€¢ There is lots of information in a wordâ€™s context 44 â€¢ Simple algorithms based just on word counts c an be surprisingly good. nlp ontology wordnet word-sense-disambiguation. share | improve this question |. follow. | edited Aug 13 '15 at 11:41. alvas.Â On the other direction, there's. ontology creation (Cyc, Yago, Freebase, etc.) semantic web (https://en.wikipedia.org/wiki/Semantic_Web). semantic lexical resources (WordNet, Open Multilingual WordNet, etc.) Knowledge base population (http://www.nist.gov/tac/2014/KBP/). There's also a recent task on ontology induction / expansion: http://alt.qcri.org/semeval2015/task17/.Â You can also try Babelfy, which provides Word Sense Disambiguation and Named Entity Disambiguation. Demo: http://babelfy.org/. API: http://babelfy.org/guide. share | improve this answer |. follow. | answered Jun 6 '16 at 8:49. Anonim33Anonim33. The problem of word sense disambiguationâ€”the question of how to determine which meaning of a word is the correct meaning for an occurrence of that word in contextâ€”has long been considered one of the most difficult yet promising research areas in computational linguistics.Â In what follows, we describe an application of the Cyc ontology and inference engine to the problem of sense disambiguation that 1) contains a method of determining semantic closeness of word senses that draws upon complex relations among concepts in the Cyc ontology, including. â€ Distribution Statement A: Approved for Public Release, Dis-tribution Unlimited. NLP - Word Sense Disambiguation - We understand that words have different meanings based on the context of its usage in the sentence. If we talk about human languages, then they are ambiguous to.Â Word sense disambiguation (WSD) is applied in almost every application of language technology. Let us now see the scope of WSD â^'. Machine Translation. Machine translation or MT is the most obvious application of WSD.Â The major problem of WSD is to decide the sense of the word because different senses can be very closely related. Even different dictionaries and thesauruses can provide different divisions of words into senses. Different algorithms for different applications. Word Sense Disambiguation - Algorithms and Applications. Springer, New York (2006) Google Scholar. 2. Aumueller, D., Do, H.H., Massmann, S., Rahm, E.: Schema and Ontology Matching with COMA++. In: Ã–zcan, F. (ed.) Proc.Â Doan, A., Madhavan, J., Domingos, P., Halevy, A.: Learning to Map between Ontologies on the Semantic Web. In: Proc. 11th Int.