Automated Population of Cyc: Extracting Information about Named-entities from the Web†

Purvesh Shah, David Schneider, Cynthia Matuszek, Robert C. Kahlert,
Bjørn Aldag, David Baxter, John Cabral, Michael Witbrock, Jon Curtis

Cycorp, Inc.
3721 Executive Center Drive, Suite 100
Austin, TX 78731, USA
{shah, daves, cynthia, rck, aldag, baxter, jcabral, witbrock, jonc} @cyc.com

Abstract

Populating the Cyc Knowledge Base (KB) has been a manual process until very recently. However, there is currently enough knowledge in Cyc for it to be feasible to attempt to acquire additional knowledge autonomously. This paper describes a system that can collect and validate formally represented, fully-integrated knowledge from the Web or any other electronically available text corpus, about various entities of interest (e.g. famous people, organizations, etc.). Experimental results and lessons learned from their analysis are presented.

Introduction

The purpose of the Cyc project is to provide a *computer understandable* knowledge base—a store of formally represented "common sense," or ubiquitous real world knowledge (Lenat, *et al.* 1983). In the last twenty years, over three million facts and rules have been added to the Cyc Knowledge Base by ontologists fluent in CycL, the formal representation language used in Cyc. The process of adding large numbers of facts by manually formalizing the content of unstructured text is tedious and slow. Despite the implementation of methods for alleviating this bottleneck (such as providing better tools to ontologists (Genarri, *et al.* 2002), providing knowledge authoring tools to subject matter experts (Panton, *et al.* 2002, Witbrock, *et al.*, 2003), creating methods for volunteers to enter and vet various types of knowledge (Witbrock, *et al.* 2005)), the knowledge acquisition process remains time-consuming and labor-intensive. However, as we shall show, Cyc now has sufficient knowledge to allow it to gather knowledge autonomously. Notably, this knowledge is of high enough quality that it can be effectively used in Cyc.

The World Wide Web has emerged as a vast source of electronic data in the form of unstructured text documents that are *information* accessible to computers but *knowledge* accessible only to humans. Various indexing systems, such as Google and Yahoo!, have made this information increasingly accessible (Brin and Page 1998) by supporting ranked document retrieval. However, because computer systems see these documents as more or less opaque strings, they do not extract facts or fuse information from multiple textual sources.

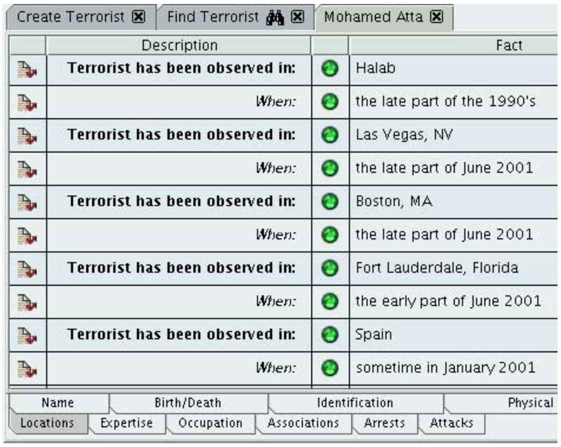
This paper describes a method for finding facts from the World Wide Web and a framework for validating candidate facts. The work presented here differs in both the approach and the goal from others (e.g. Etzioni, *et al.* 2004 and Pasca 2004) who have worked on discovering and extracting information from the unstructured Web. Our approach is similar to these other systems in that we use simple text extraction patterns. Among the differences is the fact that we leverage Cyc's natural language generation and parsing facilities (designed to support learning from English corpora (Witbrock, *et al.* 2004)) through the use of hand-generated extraction templates to generate search-strings and extract and formalize the resulting knowledge. Like the work of Craven, *et al.* (1999), our goal is to populate an existing ontology, seeded by the knowledge and structure already present. Our work differs from that of Craven, *et al.* in that we do unsupervised information extraction. Preliminary work on this system (Schneider, *et al.* 2005, Matuszek, *et al.* 2005) has shown that it is capable of extracting knowledge about famous people from the Web. In this paper, we show that it is capable of acquiring high quality, targeted, contextually relevant knowledge about a large number of relatively obscure organizations.

The types of facts targeted in this experiment, and several aspects of the experiment itself, are grounded in a desire to make comprehensive information about terrorist organizations available for formal inference procedures (and for display to users). The information is similar to the sort of information that Dun & Bradstreet[1] provides about companies and Jane's Information Group[2] provides about military equipment.

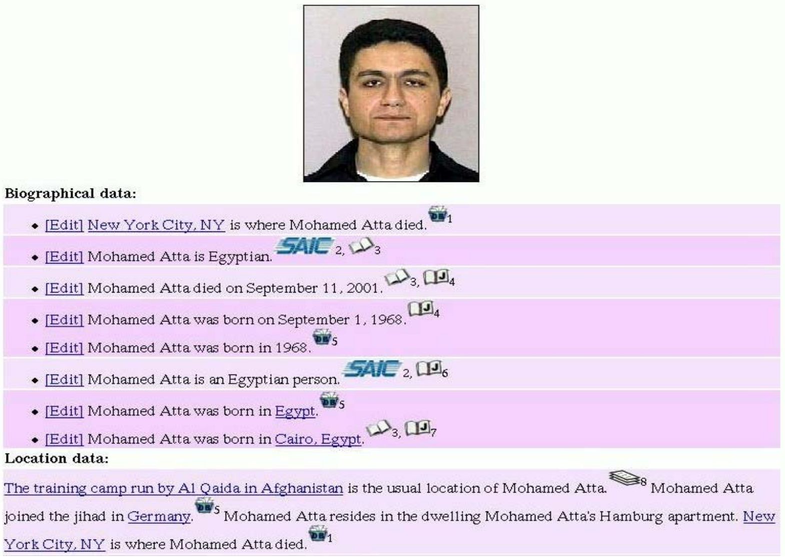
† This material is based upon work funded in whole or in part by the U.S. Government and any opinions, findings, conclusions, or recommendations expressed in this material are those of the author(s) and do not necessarily reflect the views of the U.S. Government.

[1] http://www.dnb.com

[2] http://www.janes.com



Description	Fact
Terrorist has been observed in:	Halab
When:	the late part of the 1990's
Terrorist has been observed in:	Las Vegas, NV
When:	the late part of June 2001
Terrorist has been observed in:	Boston, MA
When:	the late part of June 2001
Terrorist has been observed in:	Fort Lauderdale, Florida
When:	the early part of June 2001
Terrorist has been observed in:	Spain
When:	sometime in January 2001



Biographical data:

- [Edit] New York City, NY is where Mohamed Atta died. 1
- [Edit] Mohamed Atta is Egyptian. SAIC 2, 3
- [Edit] Mohamed Atta died on September 11, 2001. 3, 4
- [Edit] Mohamed Atta was born on September 1, 1968. 4
- [Edit] Mohamed Atta was born in 1968. 5
- [Edit] Mohamed Atta is an Egyptian person. SAIC 2, 6
- [Edit] Mohamed Atta was born in Egypt. 5
- [Edit] Mohamed Atta was born in Cairo, Egypt. 3, 7

Location data:

The training camp run by Al Qaida in Afghanistan is the usual location of Mohamed Atta. 8 Mohamed Atta joined the jihad in Germany. 5 Mohamed Atta resides in the dwelling Mohamed Atta's Hamburg apartment. New York City, NY is where Mohamed Atta died. 1

Figure 1: Fact Sheet (right) and associated fact entry/edit page (above) for Mohamed Atta.

Currently, the Cyc Analytical Environment (CAE) includes both a mechanism for manually adding new facts about an entity to the Cyc KB, as well as *Fact Sheets* (HTML documents) that display the accumulated knowledge about any particular entity (e.g. see Figure 1).

The remainder of this paper is organized as follows. We first introduce the main components of the fact gathering system and then present experimental results and analysis. We conclude with a summary and a discussion of known issues and future work.

Fact Gathering

In order to gather facts about particular named entities, the system uses the Web as its corpus (currently via the Google API). Because the knowledge-gathering goals for any entity are dependent on its type, the system first needs to determine the type of the entity. The system then determines which kinds of facts would be appropriate to gather, gathers as many of them as it can, performs automatic validation, and finally adds them to the Cyc KB.

Entity Typing

The first step is to determine the type of the named entity, e.g. to determine whether the entity is a doctor, a terrorist, a terrorist group, a company, or something else. Because Cyc's type hierarchy supports multiple-inheritance, multiple types for an entity are possible (e.g. an entity might simultaneously be a woman, a doctor and a CEO).

The entity-typing process starts with a Web search for sentences that mention the entity. A coarse typing of the entity is done using off-the-shelf named entity recognition systems (Klein, *et al*. 2003). Because the same named entity may appear in multiple documents, it can receive multiple inconsistent typings. In such cases, the system abandons work on that entity and moves on to another. When coarse typing yields consistent results, the sentences mentioning the entity are syntactically analyzed using the Charniak Parser (Charniak 2001) and the Link Parser (Sleator and Temperly 1993) to find descriptive references to the entities, such as the italicized portion of the sentence "*Palestinian militant group* HAMAS says it will not extend a ceasefire that expires at the end of this year." These descriptions are then semantically interpreted as Cyc *Collections* (in this case, (SubcollectionOfWithRelationToFn TerroristGroup hasHeadquartersInRegion Palestine-Region) or "terrorist groups with headquarters in Palestine"). These Collections are categories that are typically more specific (e.g. military group, terrorist group) than the coarse types (e.g. organization) of the entity, and so are compatible with those types. The entity is assumed to be an instance of all such categories that are consistent with the coarse typing.

What types of facts to look for?

Several different methods could be used for determining which facts are relevant for any particular type of entity. These include explicitly represented rules about types of information that should be present for an entity to be fully represented in the KB (e.g. medical professionals commonly have specialties), frequency information (e.g. if the system knows the CEO for 50% of companies represented, it should try to find them for other companies). For the experiment reported here, we chose to focus on the types of facts gathered from CAE fact-entry templates for entering knowledge about terrorist groups.

Finding the Facts

Three different methods are used to extract facts, based on the linguistic characteristics that are commonly associated with the fact-types in text. Each different type of fact is manually classified according to the method believed most likely to retrieve that type of fact. These methods are described below.

Description Strings The descriptions that were used to determine the type of the entity can be used to determine certain facts, such as the ethnicity or religion of a person, the ideology of an organization, etc. These types of facts typically have no consistently recognizable text patterns, but are very often mentioned in the descriptive references to the entity.

For example, the italicized descriptions in the following sentences provide such facts:

Palestinian leader Yasser Arafat was laid to rest after memorials in Cairo and Ramallah.
(ethnicity YassirArafat EthnicGroupOfPalestinians)

Parlak raised funds for *Islamic terrorist group* PKK while living in West Germany in the 1980s.
(hasBeliefSystems KurdistanWorkersParty Islam)

Targeted Search-Strings In this most-frequently used method, a CycL query (a specific knowledge goal) is computed using knowledge about the target entity and its type. Search-strings representing this knowledge goal are generated by instantiating hand-written templates with pre-existing lexical information (names, aliases, synonyms, etc.) about the entity. Thus, even though there are a fixed number of templates (on average, 2) per fact-type, the number of search-strings varies from goal to goal (and averaged 6.3 per goal in this experiment). For example, search strings for the query (terroristOrgPoliticalWing LebaneseHizballah ?X) are:

"*, a | the political wing of Hezbollah"

"* is the political wing of the Lebanese Hizbollah"

After retrieval, the downloaded pages (between ten and twenty per search) are converted into plain text and the sentences that contain the search-string are selected. The system tries to semantically interpret the possible fillers for the knowledge goal, starting from the smallest constituent (as determined by the Charniak parser and the named entity recognizer) that appears immediately adjacent to the search text (i.e. in place of the "*"), until an interpretable filler has been found. Figure 2 illustrates this method.

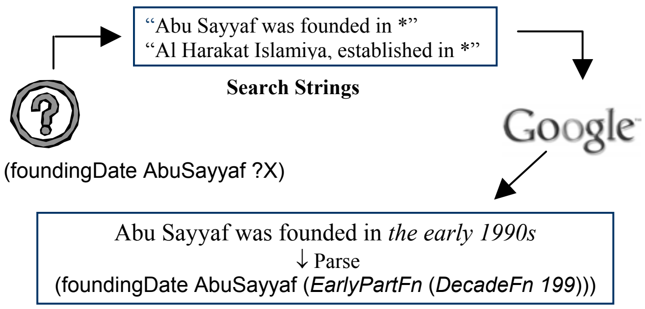

"Abu Sayyaf was founded in *"
"Al Harakat Islamiya, established in *"
Search Strings
(foundingDate AbuSayyaf ?X)
Google
Abu Sayyaf was founded in *the early 1990s*
↓ Parse
(foundingDate AbuSayyaf (*EarlyPartFn* (*DecadeFn 199*)))

Figure 2: Finding the founding date of Abu Sayyaf using a targeted search string

Hypothesize and Verify If all the possible fillers for a particular knowledge goal are known, the system generates a list of potential facts for that fact-type and entity. The English representations of each of these potential facts are computed and then searched for on the Web. Verbatim retrieval of one of the search strings for a hypothesized fact is considered evidence for that hypothesized fact.

For example, consider the knowledge goal (maritalStatus YassirArafat ?X), where all possible values of ?X are known (according to the Cyc Knowledge Base, ?X must range over the instances of PersonTypeByMaritalStatus). Some of the generated candidate facts and corresponding search-strings are:

(maritalStatus YassirArafat Married)
Yasser Arafat's husband | wife

(maritalStatus YassirArafat Divorced)
Yasser Arafat divorced

A Web search for the first of these returns the sentence 'A controversy erupted between officials of the Palestinian Authority and Suha Arafat, *Yasser Arafat's wife*'. Thus, there is evidence for the candidate fact (maritalStatus YassirArafat Married).

Validating the Retrieved Facts

All candidate facts retrieved using the above methods are tested for validity in a two-stage process that uses both the Cyc Knowledge Base and the text corpus (the Web).

The Cyc Knowledge Base is assumed to be an Oracle and hence any candidate facts already in the KB are assumed to be true. In such cases, the system records the URL of the source webpage as a new support for the fact. If Cyc can prove the candidate fact false, this is taken as evidence that the system misunderstood the document, and further processing of that candidate fact is abandoned.

Consider the following example:

Knowledge Goal: (northOf ?X Germany)
Search-string: ____ is north of Germany
Candidate facts:
- *Known to Cyc*: (northOf Denmark Germany)
- *Rejected by Cyc*: (northOf Switzerland Germany)

In the case of Switzerland above, a webpage containing the sentence "Switzerland is north of Germany" is retrieved (the sentence is a true-false question on a quiz). Cyc is able to reject it because of pre-existing knowledge that Switzerland is south of Germany, not north.

The second stage of validation involves searching the Web. For each candidate fact, a number of English representations of the fact are generated. The fact is deemed validated if any of these English strings has an exact match in the corpus. If the English representation of a fact contains an acronym, a disambiguating search-term is added to the search. The search-term is the word that results in the fewest Google hits among all the words in all the known phrasings for the entity.

For example, EjercitoPopularDeLiberacion has the acronym "EPL" and the disambiguating string "Liberacion." Thus one validation search-string for (objectFoundInLocation EjercitoPopularDeLiberacion Ontario-CanadaProvince), is '"EPL is located in Ontario" +Liberacion.' This results in no Google hits. However, had the disambiguation string '+Liberacion' been omitted, the search would have resulted in 2 Google hits, incorrectly validating the candidate fact.

Experimental Results

Experiment

The system was used to find facts about terrorist groups (n=413) that are known in the Cyc Terrorism Knowledge Base (Deaton, *et al.* 2005). Because the Cyc Terrorism Knowledge Base already contains a substantial amount of information about terrorist groups, using such groups makes computing precision easier.

Definitions

The following metrics were computed for this experiment:

Retrieval Rate (RR) is the ratio of the number of candidate facts (r) retrieved and interpreted (but not necessarily verified) for a given set E of entities to the number of desired facts d that were searched for using all the methods mentioned above. Note that r is functionally related to d for any given entity, and may exceed 1 if multiple facts are retrieved for a single query.

$$RR = \frac{\sum_{e \in E} |r_e|}{\sum_{e \in E} |d_e|}$$

Parsing Precision (PP) measures the ability of the system to correctly interpret automatically validated facts (and thereby also measures the correctness of automatic validation). PP is the ratio of the number of human-validated facts (hv) to the total number of automatically validated facts (av). A fact is marked as human-validated if the candidate fact is a reasonable interpretation of the source document. We chose this criterion over checking for the truth of the candidate facts because of the difficulty inherent in determining the truth of some of these facts.

$$PP = \frac{\sum_{e \in E} |av(r_e) \cap hv(r_e)|}{\sum_{e \in E} |av(r_e)|}$$

Invalidation Precision (IP) is the ratio of the number of facts that were correctly invalidated to the total number of facts that were automatically invalidated.

$$IP = \frac{\sum_{e \in E} |\{r_e - av(r_e)\} \cap \{r_e - hv(r_e)\}|}{\sum_{e \in E} |\{r_e - av(r_e)\}|}$$

Results

For this experiment, the system tried to find facts for terrorist groups and succeeded in finding at least one fact for 162 terrorist groups.

The overall retrieval rate (RR) was 28%, the overall parsing precision (PP) was 48%, and the overall invalidation precision (IP) was 87%.

Fact Type	**Retrieved** (RR%)	**Auto-validated** (% of retrieved)	**Human-validated** (PP%)
FoundingDate	459 (111%)	180 (40%)	102 (57%)
foundingAgent	133 (27%)	45 (34%)	16 (36%)
objectFoundIn-Location	225 (54%)	39 (17%)	4 (10%)
hasMembers	83 (20%)	6 (7%)	4 (67%)
residenceOf-Organization	76 (18%)	25 (33%)	9 (36%)
operatesIn-Region	32 (8%)	13 (41%)	11 (78%)
possesses	14 (3%)	6 (43%)	0 (0%)
orgMilitaryWing	13 (3%)	6 (46%)	6 (100%)
terroristOrg-PoliticalWing	10 (2%)	4 (40%)	4 (100%)
Total	1045 (28%)	324 (31%)	156 (48%)

Table 1: The system retrieved 28% of the desired facts, 31% of which were validated by Google. 48% of these auto-validated facts were then judged correct by human reviewers.

Analysis of Results

In general, we see that the system had very high precision for some types of facts, and somewhat lower, though still usefully good, precision for many other types of facts. The high amount of variability seen in both the precision and retrieval numbers is not surprising, given the different types of information that were searched for. Founding dates and founders are generally fairly clear and represented straightforwardly in text documents, and thus we would expect relatively high retrieval (and at least decent precision) for those facts. By contrast, some of the relations (such as terroristOrgPoliticalWing) are only applicable to a relatively small number of terrorist organizations, while other types of facts (such as operatesInRegion) are not typically expressed directly, but rather need to be inferred on the basis of other statements (e.g. "XXX claimed responsibility for the recent bombing in YYY").

A large number of candidate facts were incorrectly interpreted because of just a handful of terrorist groups that have identical or extremely similar names and acronyms. For example, there are two terrorist groups called *National Liberation Army*—one in Bolivia and one in Colombia. So, while searching for a fact about the *National Liberation Army* in Bolivia, the system would find information about the identically named group in Colombia and vice-versa.

This problem (groups with names that are either extremely common or ambiguous between multiple groups) affected five terrorist groups (RISE, Jannubi a.k.a. the South, National Liberation Army, Ejercito de Liberacion

Nacional, and People's Liberation Army) and accounted for 51 auto-validated facts (16% of all auto-validated facts) that were judged incorrect by human reviewers.

Due to an oversight involving objectFoundInLocation and possesses, the system looked not only for a terrorist group's location and possessions, but also searched for what is located in the group, and its owners. Here, it is clear that the system should not have asked what possessed a group or what was located in a group. Indeed, all the answers to those questions were in fact incorrect interpretations of the text. For example, the system used the search string "*, located in Ku Klux Klan" when looking for a filler for (objectFoundInLocation ?X KuKluxKlan), and found "… a well-to-do suburb, Concord, *located in Ku Klux Klan* country …".

Incorrect name chunking by the named entity recognizer led to a large number of incorrect facts. For example, in "Shri Harendra Debbarma, a member of ..." the named-entity recognizer recognizes two separate named entities, *Harendra* and *Debbarma,* as opposed to the (correct) single entity *Harendra Debbarma*. Because of this mistake, the system believed that a number of other people named *Debbarma* (BinoyDebbarma, JagadishDebbarma, KaminiDebbarma, LalitDebbarma, SureshDebbarma) are also members of the organization.

As mentioned above, the system finds the answer-text by using progressively larger constituents that are adjacent to the query text. The first constituent found that can be semantically interpreted as an entity meeting the semantic constraints on the filler is chosen as the filler for the candidate fact. This occasionally results in the system coming up with incorrect interpretations. In the sentence "*KKK is headquartered in* Clark County, Ohio" the italicized portion is the search-string. The system interprets the nearest constituent "Clark County" to get the answer. Thus, it does not try to interpret "Clark County, Ohio" and ends up with an incorrect fact. This method also fails to recognize the correct constituents when the entities are mentioned in lists (e.g. "The MCC is active in Bihar, Jharkhand and northern Chhatisgarh" or "... has locations in Bloomington, Indiana and Chicago").

Conclusions and Future Work

We have shown that this system of gathering facts from the Web results in fairly high quality, formally represented facts. The formal representation of these facts renders them useful not just for human perusal (via Fact Sheets) but also for use in a formal reasoning system. While it is clear that there is still room for improvement, we believe this represents a very promising start to being able to automatically mine the Web for the extension of a formal knowledge base.

Given that the goal for this system is to be able to automatically populate the Cyc Knowledge Base, the precision rates for various types of facts are extremely relevant. For types of facts correlated with high precision, we intend to modify the system to automatically assert those facts, with an appropriate notation, without the need for human review. On the other hand, there are some types of facts that the system does a good job of gathering (i.e. the system achieves a high retrieval rate), but for which precision is still too poor to warrant automatically asserting the results into the Cyc KB. For these types of facts, we will continue with manual review until such time as we are able to raise the precision adequately. Of additional concern is the quality of data found on the Web; a number of pages can be found that report facts inaccurately.

In light of the analysis of the experimental results, a number of avenues for improvement present themselves. One of the main improvements would be doing a better job disambiguating high-frequency ambiguous names and acronyms. For example, we might be able to get better results by adding a term representing the type of the entity (e.g. adding a string like "terrorist" or "revolutionary" to searches for RISE). For organizations, adding the name of a location associated with the group might also improve precision. Alternatively, if the system can determine which particular phrases for an entity are likely to yield ambiguous results, it could simply not use those ambiguous terms in its search strings. This would likely have the effect of marginally lowering the retrieval rate, but might raise the precision quite substantially.

In order to determine what types of facts are being gathered correctly and effectively, and in order to buffer the system against poor-quality Web data, human evaluations of results are currently required. To reduce the burden of this task, an extension is being built that allows untrained volunteers to verify or reject simple facts in a game-like environment. When a sufficient number of independent volunteers reach consensus, a fact can be considered additionally verified. The ultimate goal of this work is to provide a sufficient corpus of reviews for machine learning. We believe that a system could learn to identify quality sources of data as well as which types of facts (knowledge goals) are most effective to gather.

Additional extensions include the ability to target other languages and other search engines. The system has already been extended to use Chinese documents and the Yahoo! search engine API, but those changes have not yet been tested. Another possibility to improve results involves restricting the search to certain corpora or Internet domains. We hypothesize that this optimization would result in much better results for some domains. Likewise, limiting searches to specific, trusted Internet domains may also help avoid "learning" falsehoods.

Another problem faced by the system is not knowing how to weigh the results that it gets back. In the current implementation, a candidate fact found in just one document is treated the same as a fact found in twenty documents. By paying attention to the number of documents a candidate fact is found in, the system could estimate a probability for the fact, and focus its efforts (or the efforts of human adjudicators) on candidate facts that are more commonly represented in the corpus, and thus more likely to be true. Source credibility is another factor that could be used to estimate the probability of a fact. This might also be used to weed out some of the answers for a particular fact-type. If several documents say that Yasser Arafat was born on August 24, 1929 and one document says that he was born on August 4, 1929 (a real ambiguity on the Web), it would not be unreasonable to discard the latter fact. This would make the system robust against typos in text, some false claims (though not widespread, commonly accepted falsehoods) and, in some cases, failure to determine the correct answer-text.

Finally, one of the biggest and most important tasks in extending this work is to use machine learning to automatically determine additional knowledge extraction patterns. Because Cyc already contains examples of the types of knowledge it is looking for (e.g. it already knows the birthdates of some people), it can use that knowledge as a seed for finding other ways of expressing that knowledge in text. Work is currently underway to expand the system so that it can automatically acquire knowledge and knowledge extraction patterns about events.

References

Brin, S., and Page, L. 1998. Anatomy of a Large-scale Hypertextual Search Engine. In *Proceedings of the 7th International World Wide Web Conference*, 108-117. Brisbane, Australia.

Charniak, E. 2001. A Maximum-Entropy-Inspired Parser. In *Proceedings of the 1st Conference of the North American chapter of the Association for Computational Linguistics*. Seattle, Washington, 132-139.

Craven, M., Dipasquo, D., Freitag, D., McCallum, A., Mitchell, T., Nigam, K., and Slattery, S. 1999. Learning to Construct Knowledge Bases from the World Wide Web. *Artificial Intelligence* 118: 69-113.

Deaton, C., Shepard, B., Klein, C., Mayans, C., Summers, B., Brusseau, A., and Witbrock, M. 2005. The Comprehensive Terrorism Knowledge Base in Cyc. In *Proceedings of the 2005 International Conference on Intelligence Analysis*. McLean, Virginia.

Etzioni, O., Cafarella, M., Downey, D., Kok, S., Popescu, A., Shaked, T., Soderland, S., Weld, S., and Yates, A. 2004. Web-Scale Information Extraction in KnowItAll. In *Proceedings of the 13th International Conference on World Wide Web*, 100-110. New York, New York: ACM Press.

Gennari, J., Musen, M. A., Fergerson, R. W., Grosso, W. E., Crubezy, M., Eriksson, H., Noy, N. F., and Tu, S. W. 2002. The Evolution of Protégé: An Environment for Knowledge-Based Systems Development. Available at http://smi-web.stanford.edu/auslese/smi-web/reports/SMI-2002-0943.pdf.

Klein, D., Smarr, J., Nguyen, H., and Manning, C. 2003. Named Entity Recognition with Character-Level models. In *Proceeedings of the Seventh Conference on Natural Language Learning*, 180-183.

Lenat D. B., Borning, A., McDonald, D., Taylor, C., and Weyer, S. 1983. Knoesphere: Building Expert Systems with Encyclopedic Knowledge. In *Proceedings of the 8th International Joint Conference on Artificial Intelligence*, 167-169. Karlsruhe, Germany: William Kauffmann.

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

Panton, K., Miraglia, P., Salay, N., Kahlert, R. C., Baxter, D., and Reagan, R. 2002. Knowledge Formation and Dialogue Using the KRAKEN Toolset. In *Eighteenth National Conference on Artificial Intelligence*, 900-905. Edmonton, Canada.

Pasca, M. 2004. Acquisition of categorized named entities for web search. In *Proceedings of the 13th ACM Conference on Information and Knowledge Management*. 137-145. Washington, D. C.: ACM Press.

Schneider, D., Matuszek, C., Shah, P., Kahlert, R., Baxter, D., Cabral, J., Witbrock, M., and Lenat, D. 2005. Gathering and Managing Facts for Intelligence Analysis. In *Proceedings of the 2005 International Conference on Intelligence Analysis*. McLean, Virginia.

Sleator, D. and Temperly, D. 1993. Parsing English with a Link Grammar. In *Third International Workshop on Parsing Technologies*. Tilburg, Germany.

Witbrock, M., Baxter, D., Curtis, J., Schneider, D., Kahlert, R.C., Miraglia, P., Wagner, P., Panton, K., Matthews, G., Vizedom, A. 2003. An Interactive Dialogue System for Knowledge Acquisition in Cyc. In *Proceedings of the Workshop on Mixed-Initiative Intelligent Systems.* 138–145, Acapulco, Mexico.

Witbrock, M., Panton, K., Reed, S. L., Schneider, D., Aldag, B., Reimers, M., and Bertolo, S. 2004. Automated OWL Annotation Assisted by a Large Knowledge Base. In *Workshop Notes of the 2004 Workshop on Knowledge Markup and Semantic Annotation at the 3rd International Semantic Web Conference*. 71-80. Hiroshima, Japan.

Witbrock, M., Matuszek, C., Brusseau, A., Kahlert, R. C., Fraser, C. B., and Lenat, D. 2005. Knowledge Begets Knowledge: Steps towards Assisted Knowledge Acquisition in Cyc. In *Papers from the 2005 AAAI Spring Symposium on Knowledge Collection from Volunteer Contributors* (*KCVC*), 99-105. Stanford, California.