

Extracting World Knowledge from the Web

➔ Alexander Yates, Temple University



By automatically extracting information from the Web, we can scale up the resulting knowledge bases to much greater sizes than current collections of manually gathered and user-contributed knowledge.

The field of artificial intelligence has made enormous progress in knowledge representation, reasoning, action, and perception in the past decade. However, one long-recognized obstacle to progress in AI keeps surfacing in various ways: Applications lack the world knowledge necessary to behave intelligently in the real world. For instance, natural language processing (NLP) systems need some form of world knowledge to correctly interpret “it” in the sentence “If the cat doesn’t like the milk, try heating it.”

Ambiguities, such as the choice of referent for “it” above, abound in NLP and computer vision. Domain-specific applications in these disciplines have traditionally relied on manually supplied “domain knowledge” or on heuristic methods tied to the domain to help resolve these ambiguities, but they cannot be ported to new domains without significant manual effort.

The latest supervised systems and probabilistic reasoning techniques have made applications highly suc-

cessful in single-domain settings. But broad-coverage world knowledge has become the missing link between existing techniques and successful, portable applications.

Various projects have attempted to collect world knowledge at a large scale for use in AI. Famously, researchers with the Cyc project have painstakingly been entering this knowledge into a database by hand for several decades (D.B. Lenat, “Cyc: A Large-Scale Investment in Knowledge Infrastructure,” *Comm. ACM*, Nov. 1995, pp. 33-38).

Recently, Web 2.0 technology has enabled massive online collaboration efforts for collecting world knowledge. For example, Wikipedia serves user-generated encyclopedia entries, the OpenMind initiative (www.openmind.org) gathers users’ contributions of sentences expressing some piece of common-sense knowledge, and the freebase.com project collects user-contributed databases of information.

In contrast to these manual efforts, recent research, including work at Temple University, has begun to

automate the process of gathering world knowledge from text. The Web and other text repositories provide a massive, readily available collection of world knowledge on all manner of subjects. By extracting information from these sources, we can scale up the resulting knowledge bases to much greater sizes than the knowledge gathered by Cyc and current collections of user-contributed knowledge.

AUTOMATED KNOWLEDGE COLLECTION

Automating the process of knowledge collection for AI involves several key challenges.

Most important among these is that the extraction system must handle arbitrary domains and types of knowledge with little or no human involvement. To gather broad-coverage domain knowledge, the system cannot be tied to any domain.

Unfortunately, this requirement precludes many popular existing techniques for NLP systems, such as supervised learning, which requires new manually supplied labels to

extract knowledge about new classes and domains. For instance, supervised named-entity taggers, which traditionally detect names of people, places, and organizations in text, would fail miserably at the task of detecting names of films unless supplied with labeled examples of the new class.

Moreover, even when detecting people's names, the performance of such systems degrades when the training domain is newswire text and the test domain is, say, e-mails or some other type of text with a significantly different writing style.

A second requirement for knowledge collection systems is that they produce knowledge in a logical (or machine-interpretable) representation. In the extreme, a system would need to translate every text statement into a representation capturing that statement's meaning. This goal has eluded NLP researchers for many decades.

In practice, today's systems work around this requirement by focusing on types of knowledge that are easily represented in logic, such as unary or binary relationships. Moreover, most systems pick out important facts and relationships from "easy-to-understand" text while sacrificing the knowledge contained in complex statements. Fortunately, because of the Web's size, easy-to-understand text constitutes a huge catalog of information.

KNOWITALL

One of the first systems for completely automatic knowledge extraction from text is the KnowItAll system for extracting instances of classes (O. Etzioni et al., "Unsupervised Named-Entity Extraction from the Web: An Experimental Study," *Artificial Intelligence*, June 2005, pp. 91-134).

Two-stage knowledge extraction

KnowItAll features a novel, generate-and-test architecture that extracts information in two stages.

First, the system utilizes a set of eight domain-independent extraction patterns to generate candidate facts. For example, the generic pattern "NP1 such as NPList2" indicates that the head of each simple noun phrase (NP) in NPList2 is a member of the class named in NP1. By instantiating the pattern for the class City, KnowItAll extracts three candidate cities from the sentence "We provide tours to cities such as Paris, London, and Berlin."

The efficiency of extraction has improved dramatically, making it practical to process and extract knowledge from ever larger corpora.

Second, the system automatically tests the plausibility of the candidate facts it extracts using pointwise mutual information (PMI) statistics. These statistics measure the frequency of a candidate extraction together with words that are highly indicative of the class, relative to the frequency of the candidate extraction on its own.

KnowItAll leverages existing Web search engines to compute these statistics efficiently. Based on the PMI statistics, the system associates a probability with every fact it extracts, enabling it to automatically manage the tradeoff between precision and recall.

While the basic version of KnowItAll extracts knowledge from only those sentences that match its domain-independent patterns, three extensions enable it to extract many times more information.

A *pattern learning* module learns domain-specific extraction rules, which enable additional extractions. A *subclass extraction* module automatically identifies subclasses to boost recall—for example, it iden-

tifies "chemist" and "biologist" as subclasses of "scientist." Finally, a *list extraction* module locates webpages containing lists of class instances, learns a "wrapper" for each list, and extracts elements of each list.

Since each of these modules bootstraps from KnowItAll's domain-independent methods, they also obviate hand-labeled training examples.

In concert, these methods give the system a four- to eightfold increase in recall at a precision of 0.9, and they discover more than 10,000 cities missing from the Tipster Gazetteer, a well-known database of place names.

Beyond KnowItAll

Researchers continue to build upon KnowItAll's successes in several ways to collect larger and more sophisticated bodies of world knowledge.

State-of-the-art systems can now automatically extract instances of binary relations in addition to instances of classes. KnowItAll must be manually supplied with class names, but recent systems instead extract names of relations at the same time that they extract the names of their arguments.

In addition, the efficiency of extraction has improved dramatically, making it practical to process and extract knowledge from ever larger corpora.

TEXTUAL AMBIGUITIES

While current extraction techniques can automatically detect relationships in open text, such as *is capital of*(D.C., United States), this is just part of the answer to extracting world knowledge. These extracted relationships are textual descriptions and, because of the ambiguous nature of language, they lack the precision required by knowledge bases for AI applications.

For example, *is capital city of*(Washington, U.S.) has the same meaning as the extraction above

but it has different names for the relation and for each argument. This type of imprecision in a knowledge base might cause an application to infer that although Washington is the capital city of the US, it is not the capital of the United States, whose capital is instead something called D.C.

Other types of ambiguity in language are equally problematic. *Polysemy* causes a different kind of imprecision because extracted words may have multiple meanings. For instance, extractions concerning “Washington” as a person or a US state can be confused with those concerning Washington, D.C.

Finally, some extracted relationships have ambiguities relating to the scope of the quantifiers in the arguments. For example, *lives in*(a doctor, every city) could mean that there is a single doctor who lives in all cities simultaneously or a different doctor per city. To be precise sources of world knowledge, extracted knowledge bases must handle these types of ambiguities.

RESOLVER

Fortunately, the extracted relationships provide clues for solving some of these thorny representational issues.

The Resolver system uses extracted knowledge to determine when two relation names or two argument names refer to the same thing, in a process called *synonym resolution* (A. Yates and O. Etzioni, “Unsupervised Methods for Determining Object and Relation Synonyms on the Web,” *J. Artificial Intelligence Research*, Jan.-Apr. 2009, pp. 255-296.)

Consider the strings Mars and Red Planet, which appear 659 and 26 times, respectively, in one dataset of extracted relationships. Out of these extractions, four are common to both strings. For example, *lacks*(Mars, ozone layer) and *lacks*(Red Planet, ozone layer) both appear in the extractions.

Resolver determines the probability that Mars and Red Planet are

synonyms after observing k , the number of properties that apply to both, and n_1 and n_2 , the total number of properties for Mars and Red Planet, respectively. It uses a generative probabilistic model of information extraction to predict the probability of observing k shared extractions in $n_1 + n_2$ tries, given that the two strings are synonyms, as well as the probability of observing k when the two strings are not synonyms. Applying Bayes’ rule then yields the desired probability.

Automated solutions for collecting knowledge have made huge strides in recent years in their scale and the sophistication of the extracted knowledge.

Resolver can handle polysemous names as well, by determining when two occurrences o_1 and o_2 of the same word should be treated as synonyms or occurrences of the same sense of that word. In this case, the extracted knowledge cannot be used as readily, since each occurrence of a word appears in only one extraction. But if we use the set of named entities appearing around o_1 and o_2 as proxies for extractions and calculate k , n_1 , and n_2 from these sets of named entities, then the system can apply to polysemous words.

To cluster its input strings into sets of synonyms, Resolver must automatically cluster a very large set of extractions. To make this operation scale to enormous input sizes, the system uses a novel clustering algorithm that can operate efficiently on text extractions.

Much of the computational cost of agglomerative clustering involves comparing completely dissimilar strings. To avoid this cost, Resolver builds an index of the extractions and then compares only those strings that

share at least one extraction, such as Mars and Red Planet. Further, if many different strings X share the same extraction Y —for example, *has*(X , a mass)—then the system does not consider Y when trying to decide whether to compare strings.

By making these restrictions, Resolver can reduce the complexity of comparing strings from $O(n_2)$ to $O(n \log n)$, assuming that the extractions observe a Zipf distribution. Moreover, the restrictions prevent comparisons between strings that are unlikely to be synonyms because they share at most only common extractions. Thus the system can perform clustering with much greater efficiency and a minimal cost to accuracy.

Resolver can leverage large bodies of unlabeled text to resolve some of the ambiguities, like synonymy and polysemy for named entities, that affect extracted knowledge bases. As extraction systems scale up to handle more complex sentences and to new domains, they face many of the ambiguity problems that general NLP systems encounter, and the need for fully automated, domain-independent solutions grows. The extracted information itself is an important source of evidence for resolving these ambiguities.

RESEARCH CHALLENGES

Automated solutions for collecting knowledge have made huge strides in recent years in their scale and the sophistication of the extracted knowledge. What began as a process of discovering named entities for arbitrary classes has evolved into methods for collecting relational data; textual relations have become unambiguous relational data, and the whole process of collecting knowledge scales to hundreds of millions or even billions of facts.

As researchers have started to work out the fundamentals of automated knowledge collection, they have begun to address other pressing challenges beyond the automation and scalability of the extraction pro-

cess. One such challenge is whether the accuracy of extraction can be improved with labeled training data.

While supervised techniques are difficult to apply to automated knowledge collection, recent work on *domain adaptation* points toward a way to use a small set of labeled data in one domain, together with a large body of unlabeled data from a new domain, to achieve higher accuracy on the new domain (F. Huang and A. Yates, "Distributional Representations for Handling Sparsity in Supervised Sequence Labeling," *Proc. Ann. Meeting Assoc. for Computational Linguistics, ACL, 2009*; <http://www.cis.temple.edu/~yates/papers/smoothing-acl09.pdf>). This type of system still requires manual effort but is constrained to a manageable amount of a priori labeling.

A second, intriguing problem for researchers is to demonstrate the

extracted knowledge's utility in AI tasks. NLP researchers and linguists have long recognized that world knowledge, or "pragmatics" in linguistics, is essential for many general language-understanding tasks, from parsing to word sense disambiguation to coreference resolution. While research efforts have demonstrated the ability to collect such knowledge, they are only just beginning to connect the extracted knowledge with applications.

Recent efforts have shown how extracted knowledge can be put to use in question answering and parsing, but only in tightly constrained cases. To demonstrate the validity of their extracted knowledge, researchers will soon be putting their extracted pragmatics to use in NLP tasks.

Researchers have also only begun to explore the use of inference

engines to reason over extracted knowledge bases. For instance, with an approximate inference engine, an extraction system can greatly improve its ability to judge the validity of new extractions.

Despite these advances, there are numerous, largely unexplored opportunities concerning the connection between extracted knowledge bases and AI applications. **■**

Alexander Yates is an assistant professor in the Department of Computer and Information Sciences at Temple University. Contact him at yates@temple.edu.

Editor: Naren Ramakrishnan, Dept. of Computer Science, Virginia Tech, Blacksburg, VA; naren@cs.vt.edu