EXPERT OPINION

Contact Editor: **Brian Brannon,** bbrannon@computer.org

The Unreasonable Effectiveness of Data

Alon Halevy, Peter Norvig, and Fernando Pereira, *Google*

Eugene Wigner's article "The Unreasonable Effectiveness of Mathematics in the Natural Sciences"[1] examines why so much of physics can be neatly explained with simple mathematical formulas such as $f = ma$ or $e = mc^2$. Meanwhile, sciences that involve human beings rather than elementary particles have proven more resistant to elegant mathematics. Economists suffer from physics envy over their inability to neatly model human behavior. An informal, incomplete grammar of the English language runs over 1,700 pages.[2] Perhaps when it comes to natural language processing and related fields, we're doomed to complex theories that will never have the elegance of physics equations. But if that's so, we should stop acting as if our goal is to author extremely elegant theories, and instead embrace complexity and make use of the best ally we have: the unreasonable effectiveness of data.

One of us, as an undergraduate at Brown University, remembers the excitement of having access to the Brown Corpus, containing one million English words.[3] Since then, our field has seen several notable corpora that are about 100 times larger, and in 2006, Google released a trillion-word corpus with frequency counts for all sequences up to five words long.[4] In some ways this corpus is a step backwards from the Brown Corpus: it's taken from unfiltered Web pages and thus contains incomplete sentences, spelling errors, grammatical errors, and all sorts of other errors. It's not annotated with carefully hand-corrected part-of-speech tags. But the fact that it's a million times larger than the Brown Corpus outweighs these drawbacks. A trillion-word corpus—along with other Web-derived corpora of millions, billions, or trillions of links, videos, images, tables, and user interactions—captures even very rare aspects of human behavior. So, this corpus could serve as the basis of a complete model for certain tasks—if only we knew how to extract the model from the data.

Learning from Text at Web Scale

The biggest successes in natural-language-related machine learning have been statistical speech recognition and statistical machine translation. The reason for these successes is not that these tasks are easier than other tasks; they are in fact much harder than tasks such as document classification that extract just a few bits of information from each document. The reason is that translation is a natural task routinely done every day for a real human need (think of the operations of the European Union or of news agencies). The same is true of speech transcription (think of closed-caption broadcasts). In other words, a large training set of the input-output behavior that we seek to automate is available to us *in the wild*. In contrast, traditional natural language processing problems such as document classification, part-of-speech tagging, named-entity recognition, or parsing are not routine tasks, so they have no large corpus available in the wild. Instead, a corpus for these tasks requires skilled human annotation. Such annotation is not only slow and expensive to acquire but also difficult for experts to agree on, being bedeviled by many of the difficulties we discuss later in relation to the Semantic Web. The first lesson of Web-scale learning is to use available large-scale data rather than hoping for annotated data that isn't available. For instance, we find that useful semantic relationships can be automatically learned from the statistics of search queries and the corresponding results[5] or from the accumulated evidence of Web-based text patterns and formatted tables,[6] in both cases without needing any manually annotated data.

1541-1672/09/$25.00 © 2009 IEEE
Published by the IEEE Computer Society

Another important lesson from statistical methods in speech recognition and machine translation is that memorization is a good policy if you have a lot of training data. The statistical language models that are used in both tasks consist primarily of a huge database of probabilities of short sequences of consecutive words (*n-grams*). These models are built by counting the number of occurrences of each *n*-gram sequence from a corpus of billions or trillions of words. Researchers have done a lot of work in estimating the probabilities of new *n*-grams from the frequencies of observed *n*-grams (using, for example, Good-Turing or Kneser-Ney smoothing), leading to elaborate probabilistic models. But invariably, simple models and a lot of data trump more elaborate models based on less data. Similarly, early work on machine translation relied on elaborate rules for the relationships between syntactic and semantic patterns in the source and target languages. Currently, statistical translation models consist mostly of large memorized *phrase tables* that give candidate mappings between specific source- and target-language phrases.

Instead of assuming that general patterns are more effective than memorizing specific phrases, today's translation models introduce general rules only when they improve translation over just memorizing particular phrases (for instance, in rules for dates and numbers). Similar observations have been made in every other application of machine learning to Web data: simple *n*-gram models or linear classifiers based on millions of specific features perform better than elaborate models that try to discover general rules. In many cases there appears to be a threshold of sufficient data. For example, James Hays and Alexei A. Efros addressed the task of *scene completion:* removing an unwanted, unsightly automobile or ex-spouse from a photograph and filling in the background with pixels taken from a large corpus of other photos.[7] With a corpus of thousands of photos, the results were poor. But once they accumulated millions of photos, the same algorithm performed quite well. We know that the number of grammatical English sentences is theoretically infinite and the number of possible 2-Mbyte photos is $256^{2,000,000}$. However, in practice we humans care to make only a finite number of distinctions. For many tasks, once we have a billion or so examples, we essentially have a closed set that represents (or at least approximates) what we need, without generative rules.

For many tasks, words and word combinations provide all the representational machinery we need to learn from text.

For those who were hoping that a small number of general rules could explain language, it is worth noting that language is inherently complex, with hundreds of thousands of vocabulary words and a vast variety of grammatical constructions. Every day, new words are coined and old usages are modified. This suggests that we can't reduce what we want to say to the free combination of a few abstract primitives.

For those with experience in small-scale machine learning who are worried about the curse of dimensionality and overfitting of models to data, note that all the experimental evidence from the last decade suggests that throwing away rare events is almost always a bad idea, because much Web data consists of individually rare but collectively frequent events. For many tasks, words and word combinations provide all the representational machinery we need to learn from text. Human language has evolved over millennia to have words for the important concepts; let's use them. Abstract representations (such as clusters from latent analysis) that lack linguistic counterparts are hard to learn or validate and tend to lose information. Relying on overt statistics of words and word co-occurrences has the further advantage that we can estimate models in an amount of time proportional to available data and can often parallelize them easily. So, learning from the Web becomes naturally scalable.

The success of *n*-gram models has unfortunately led to a false dichotomy. Many people now believe there are only two approaches to natural language processing:

- a *deep* approach that relies on hand-coded grammars and ontologies, represented as complex networks of relations; and
- a *statistical* approach that relies on learning *n*-gram statistics from large corpora.

In reality, three orthogonal problems arise:

- choosing a representation language,
- encoding a model in that language, and
- performing inference on the model.

Each problem can be addressed in several ways, resulting in dozens of approaches. The deep approach that was popular in the 1980s used first-order logic (or something similar) as the representation language, encoded a model with the labor of a team of graduate students, and did inference with complex inference rules appropriate to the representation language. In the 1980s and 90s, it became fashionable to use finite state machines as the representation language, use counting and smoothing over a large corpus to encode a model, and use simple Bayesian statistics as the inference method.

But many other combinations are possible, and in the 2000s, many are being tried. For example, Lise Getoor and Ben Taskar collect work on *statistical relational learning*—that is, representation languages that are powerful enough to represent relations between objects (such as first-order logic) but that have a sound, probabilistic definition that allows models to be built by statistical learning.[8] Taskar and his colleagues show how the same kind of maximum-margin classifier used in support vector machines can improve traditional parsing.[9] Stefan Schoenmackers, Oren Etzioni, and Daniel S. Weld show how a relational logic and a 100-million-page corpus can answer questions such as "what vegetables help prevent osteoporosis?" by isolating and combining the relational assertions that "kale is high in calcium" and "calcium helps prevent osteoporosis."[10]

Semantic Web versus Semantic Interpretation

The Semantic Web is a convention for formal representation languages that lets software services interact with each other "without needing artificial intelligence."[11] A software service that enables us to make a hotel reservation is transformed into a Semantic Web service by agreeing to use one of several standards for representing dates, prices, and locations. The service can then interoperate with other services that use either the same standard or a different one with a known translation into the chosen standard. As Tim Berners-Lee, James Hendler, and Ora Lassila write, "The Semantic Web will enable machines to *comprehend* semantic documents and data, not human speech and writings."[11]

The problem of understanding human speech and writing—the *semantic interpretation problem*—is quite different from the problem of software service interoperability. Semantic interpretation deals with imprecise, ambiguous natural languages, whereas service interoperability deals with making data precise enough that the programs operating on the data will function effectively. Unfortunately, the fact that the word "semantic" appears in both "Semantic Web" and "semantic interpretation" means that the two problems have often been conflated, causing needless and endless consternation and confusion. The "semantics" in Semantic Web services is embodied in the code that implements those services in accordance with the specifications expressed by the relevant ontologies and attached informal documentation. The "semantics" in semantic interpretation of natural languages is instead embodied in human cognitive and cultural processes whereby linguistic expression elicits expected responses and expected changes in cognitive state. Because of a huge shared cognitive and cultural context, linguistic expression can be highly ambiguous and still often be understood correctly.

> Because of a huge shared cognitive and cultural context, linguistic expression can be highly ambiguous and still often be understood correctly.

Given these challenges, building Semantic Web services is an engineering and sociological challenge. So, even though we understand the required technology, we must deal with significant hurdles:

- *Ontology writing.* The important easy cases have been done. For example, the Dublin Core defines dates, locations, publishers, and other concepts that are sufficient for card catalog entries. Bioformats.org defines chromosomes, species, and gene sequences. Other organizations provide ontologies for their specific fields. But there's a long tail of rarely used concepts that are too expensive to formalize with current technology. Project Halo did an excellent job of encoding and reasoning with knowledge from a chemistry textbook, but the cost was US$10,000 per page.[12] Obviously we can't afford that cost for a trillion Web pages.
- *Difficulty of implementation.* Publishing a static Web page written in natural language is easy; anyone with a keyboard and Web connection can do it. Creating a database-backed Web service is substantially harder, requiring specialized skills. Making that service compliant with Semantic Web protocols is harder still. Major sites with competent technology experts will find the extra effort worthwhile, but the vast majority of small sites and individuals will find it too difficult, at least with current tools.
- *Competition.* In some domains, competing factions each want to promote their own ontology. In other domains, the entrenched leaders of the field oppose any ontology because it would level the playing field for their competitors. This is a problem in diplomacy, not technology. As Tom Gruber says, "Every ontology is a treaty—a social agreement—among people with some common motive in sharing."[13] When a motive for sharing is lacking, so are common ontologies.
- *Inaccuracy and deception.* We

know how to build sound inference mechanisms that take true premises and infer true conclusions. But we don't have an established methodology to deal with mistaken premises or with actors who lie, cheat, or otherwise deceive. Some work in reputation management and trust exists, but for the time being we can expect Semantic Web technology to work best where an honest, self-correcting group of cooperative users exists and not as well where competition and deception exist.

The challenges for achieving accurate semantic interpretation are different. We've already solved the sociological problem of building a network infrastructure that has encouraged hundreds of millions of authors to share a trillion pages of content. We've solved the technological problem of aggregating and indexing all this content. But we're left with a scientific problem of interpreting the content, which is mainly that of learning as much as possible about the context of the content to correctly disambiguate it. The semantic interpretation problem remains regardless of whether or not we're using a Semantic Web framework. The same meaning can be expressed in many different ways, and the same expression can express many different meanings. For example, a table of company information might be expressed in ad hoc HTML with column headers called "Company," "Location," and so on. Or it could be expressed in a Semantic Web format, with standard identifiers for "Company Name" and "Location," using the Dublin Core Metadata Initiative point-encoding scheme. But even if we have a formal Semantic Web "Company Name" attribute, we can't expect to have an ontology for every possible value of this attribute. For example, we can't know for sure what company the string "Joe's Pizza" refers to because hundreds of businesses have that name and new ones are being added all the time. We also can't always tell which business is meant by the string "HP." It could refer to Helmerich & Payne Corp. when the column is populated by stock ticker symbols but probably refers to Hewlett-Packard when the column is populated by names of large technology companies. The problem of semantic interpretation remains; using a Semantic Web formalism just means that semantic interpretation must be done on shorter strings that fall between angle brackets.

The same meaning can be expressed in many different ways, and the same expression can express many different meanings.

What we need are methods to infer relationships between column headers or mentions of entities in the world. These inferences may be incorrect at times, but if they're done well enough we can connect disparate data collections and thereby substantially enhance our interaction with Web data. Interestingly, here too Web-scale data might be an important part of the solution. The Web contains hundreds of millions of independently created tables and possibly a similar number of lists that can be transformed into tables.[14] These tables represent structured data in myriad domains. They also represent how different people organize data—the choices they make for which columns to include and the names given to the columns. The tables also provide a rich collection of column values, and values that they decided belong in the same column of a table. We've never before had such a vast collection of tables (and their schemata) at our disposal to help us resolve semantic heterogeneity. Using such a corpus, we hope to be able to accomplish tasks such as deciding when "Company" and "Company Name" are synonyms, deciding when "HP" means Helmerich & Payne or Hewlett-Packard, and determining that an object with attributes "passengers" and "cruising altitude" is probably an aircraft.

Examples

How can we use such a corpus of tables? Suppose we want to find synonyms for attribute names—for example, when "Company Name" could be equivalent to "Company" and "price" could be equivalent to "discount"). Such synonyms differ from those in a thesaurus because here, they are highly context dependent (both in tables and in natural language). Given the corpus, we can extract a set of schemata from the tables' column labels; for example, researchers reliably extracted 2.5 million distinct schemata from a collection of 150 million tables, not all of which had schema.[14] We can now examine the co-occurrences of attribute names in these schemata. If we see a pair of attributes A and B that rarely occur together but always occur with the same other attribute names, this might mean that A and B are synonyms. We can further justify this hypothesis if we see that data elements have a significant overlap or are of the same data type. Similarly, we can also offer a *schema autocomplete* feature for database designers. For example, by analyzing such a large corpus of schemata, we can discover that schemata that have the attributes Make and Model also tend to have the attributes Year, Color, and Mileage. Providing such feedback to schemata creators can save them time but can also help them use more common attribute names, thereby decreasing a possible source of heterogeneity in Web-based data. Of course, we'll find immense opportunities to create interesting data sets if we can automatically combine data from multiple tables in this collection. This is an area of active research.

Another opportunity is to combine data from multiple tables with data from other sources, such as unstructured Web pages or Web search queries. For example, Marius Paşca also considered the task of identifying attributes of classes.[15] That is, his system first identifies classes such as "Company," then finds examples such as "Adobe Systems," "Macromedia," "Apple Computer," "Target," and so on, and finally identifies class attributes such as "location," "CEO," "headquarters," "stock price," and "company profile." Michael Cafarella and his colleagues showed this can be gleaned from tables, but Paşca showed it can also be extracted from plain text on Web pages and from user queries in search logs. That is, from the user query "Apple Computer stock price" and from the other information we know about existing classes and attributes, we can confirm that "stock price" is an attribute of the "Company" class. Moreover, the technique works not just for a few dozen of the most popular classes but for thousands of classes and tens of thousands of attributes, including classes like "Aircraft Model," which has attributes "weight," "length," "fuel consumption," "interior photos," "specifications," and "seating arrangement." Paşca shows that including query logs can lead to excellent performance, with 90 percent precision over the top 10 attributes per class.

So, follow the data. Choose a representation that can use unsupervised learning on unlabeled data, which is so much more plentiful than labeled data. Represent all the data with a nonparametric model rather than trying to summarize it with a parametric model, because with very large data sources, the data holds a lot of detail. For natural language applications, trust that human language has already evolved words for the important concepts. See how far you can go by tying together the words that are already there, rather than by inventing new concepts with clusters of words. Now go out and gather some data, and see what it can do.

> Choose a representation that can use unsupervised learning on unlabeled data, which is so much more plentiful than labeled data.

References

1. E. Wigner, "The Unreasonable Effectiveness of Mathematics in the Natural Sciences," *Comm. Pure and Applied Mathematics*, vol. 13, no. 1, 1960, pp. 1–14.
2. R. Quirk et al., *A Comprehensive Grammar of the English Language,* Longman, 1985.
3. H. Kucera, W.N. Francis, and J.B. Carroll, *Computational Analysis of Present-Day American English*, Brown Univ. Press, 1967.
4. T. Brants and A. Franz, *Web 1T 5-Gram Version 1,* Linguistic Data Consortium, 2006.
5. S. Riezler, Y. Liu, and A. Vasserman, "Translating Queries into Snippets for Improved Query Expansion," *Proc. 22nd Int'l Conf. Computational Linguistics* (Coling 08), Assoc. Computational Linguistics, 2008, pp. 737–744.
6. P.P. Talukdar et al., "Learning to Create Data-Integrating Queries," *Proc. 34th Int'l Conf. Very Large Databases* (VLDB 08), Very Large Database Endowment, 2008, pp. 785–796.
7. J. Hays and A.A. Efros, "Scene Completion Using Millions of Photographs," *Comm. ACM*, vol. 51, no. 10, 2008, pp. 87–94.
8. L. Getoor and B. Taskar, *Introduction to Statistical Relational Learning*, MIT Press, 2007.
9. B. Taskar et al., "Max-Margin Parsing," *Proc. Conf. Empirical Methods in Natural Language Processing* (EMNLP 04), Assoc. for Computational Linguistics, 2004, pp. 1–8.
10. S. Schoenmackers, O. Etzioni, and D.S. Weld, "Scaling Textual Inference to the Web," *Proc. 2008 Conf. Empirical Methods in Natural Language Processing* (EMNLP 08), Assoc. for Computational Linguistics, 2008, pp. 79–88.
11. T. Berners-Lee, J. Hendler, and O. Lassila, "The Semantic Web," *Scientific Am.*, 17 May 2001.
12. P. Friedland et al., "Towards a Quantitative, Platform-Independent Analysis of Knowledge Systems," *Proc. Int'l Conf. Principles of Knowledge Representation*, AAAI Press, 2004, pp. 507–514.
13. "Interview of Tom Gruber," *AIS SIGSEMIS Bull.*, vol. 1, no. 3, 2004.
14. M.J. Cafarella et al., "WebTables: Exploring the Power of Tables on the Web," *Proc. Very Large Data Base Endowment* (VLDB 08), ACM Press, 2008, pp. 538–549.
15. M. Paşca, "Organizing and Searching the World Wide Web of Facts. Step Two: Harnessing the Wisdom of the Crowds," *Proc. 16th Int'l World Wide Web Conf.*, ACM Press, 2007, pp. 101–110.

Alon Halevy is a research scientist at Google. Contact him at halevy@google.com.

Peter Norvig is a research director at Google. Contact him at pnorvig@google.com.

Fernando Pereira is a research director at Google. Contact him at pereira@google.com.