Introduction monologue
CS 562/662: Natural Language Processing
2015-01-06

What is natural language processing (henceforth, NLP)? Let's begin with the idea of *computational linguistics*. Linguistics is, as I think of it, the science of natural—that is, human—language, and *computational linguistics* focuses on problems of computation arising with human languages. The "computations" in question may be those done by computer or by human.

This was, at one time, thought not so different than the study of algorithms for parsing **computer** languages as part of compiler construction. And, languages constructed for use in programming are at least superficially similar to human languages in a number of ways:

- they have a denotation
- the denotation is dependent upon order, so that `x / y` is **not** the same as `y / x`
- intuitively, they can be recognized and interpreted by the intended audience

Early success in compilers led artificial intelligence researchers to believe that human language would yield to their methods, but this was simply not the case. There are at least a few major problems unique to human languages.

First, human language has pervasive structural ambiguity, both local and global, whereas computer languages are, in the general case, not. There are, of course, languages in which certain symbols are *overloaded*—a good example being the asterisk in C, which acts as a binary operator for multiplication and a unary operator for dereferencing—but these can be disambiguated in a computationally simple preprocessing step. This is not the case for structural ambiguity in human language, which often requires extensive world knowledge to disambiguate. There are famous examples, like the following:

(1) Buffalo buffalo Buffalo buffalo buffalo buffalo Buffalo buffalo.

This (patently silly) sentence depends on the fact that *buffalo* has three sentences: the city in New York state, the rare verb meaning 'to bully', and a species of bison; one reading (and there may be others) can be paraphrased as "NY bison that are bullied by other NY bison, they themselves bully NY bison".

But there are many subtler examples, like the following:

(2) Pope Francis on Saturday appointed a victim of sexual abuse and a senior cardinal known for his zero-tolerance approach to a new group charged with advising the Catholic Church on how to respond to the problem of sexual abuse of children. (WSJ; h/t Language Log)

Reading this sentence, you probably didn't notice the many ambiguities present. State of the art natural language parsers get this one way, way wrong. Their representations are such that they think Pope Francis is a new person every day, that the "victim of sexual abuse" and the "senior cardinal" are one and the same person, and that the "senior cardinal" has a "zero-tolerance approach" to groups advising the Catholic church!

But structural ambiguity is not just annoying. It also means that human language is not guaranteed to be efficiently parsable.

Another key problem, which is less widely studied but no less important, concerns the unfortunate reality that human language is not even weakly context-free. We will encounter a proof of this later in the course. Fortunately, as far we know, human languages do seem to be well described by grammar formalisms belonging to a class known as the *mildy context-sensitive grammar formalisms*. A formalism is said to be mildly context-sensitive if and only if there is a known polynomial-time algorithm for parsing all languages in all grammars specified in the formalism. However, such algorithms are currently on the edge of what is feasible for online applications, so this is an active area of research.

Finally, linguistic data is exceptionally sparse, making NLP a major challenge for machine learning approaches. One early statement of this principle is Zipf's Law, named after George Kingsley Zipf, a professor of philology at Harvard University in the 1930s and 1940s. Zipf observed that word frequency and word rank tended to be inversely proportional in corpora. So, when it comes to words, there are *a few giants* (a small set of infrequent words) and *many dwarves* (a large number of rare words). And, worst of all, there are many words that have zero frequency in a corpus but very well **could** have occurred. So, in addition to the infinitude of *true zeros*, there are also an unknown number of *accidental zeros*. It turns out that Zipf's observations are also approximately correct for just about any other linguistic structure you can think of: characters, phones, phonemes, words, phrases, phrase structure rules, and sequences and conjunctions of the above.

One subject we will *not* cover in any great detail is how humans compute—that is, produce and understand—human languages. While there are many important points of overlap between this area and NLP, this class will focus strictly on the **engineering** problems that arise in enabling computers to work with—to process, annotate, and understand—human languages.