CpSc810 – Goddard – Notes Chapter 4

Natural Language Processing

Natural language processing is the understanding of human languages by a computer. This means that the computer should be able to use a human language to accept the kind of data it normally processes. Part of the problem is that we still understand very little of how a human learns languages. But that which makes the problem hard is also what makes language so powerful and expressive.

Consider dealing with written language. One of the biggest problems is ambiguity: incomplete information, contextual information, etc. Most sentences are ambiguous. Pioneering work was done by Winograd. Spoken language has all the problems of written language, as well as the processing of the speech sounds; that is complicated by speech sounds overlapping in time and because different people speak differently.

4.1 Written Language Comprehension

There are always levels. But interaction between levels, of course.

1. ***morphological analysis***: Here the task is to analyze individual words and punctuation into tokens and determine the part of speech they are (or could be).

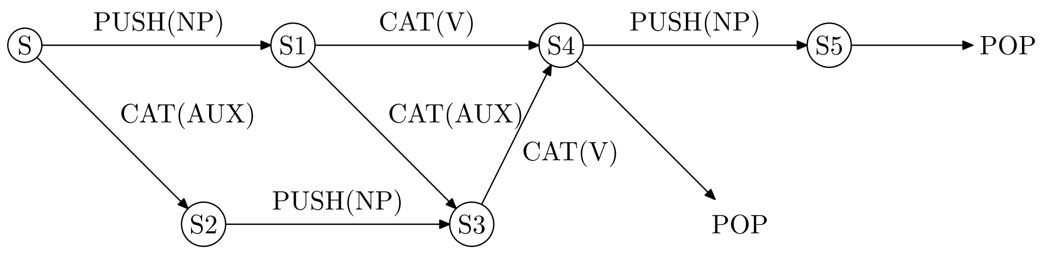
2. ***syntactical analysis or parsing***: Here the goal is to determine the sentence constituents and the hierarchical sentence structure. This uses word order, number agreement and case agreement, and grammars. Some have argued that parsing is logical inference. One may need to determine all possible parsings. (Consider *garden path* sentences: interpret "The horse raced past the barn fell down.")

3. ***semantic analysis***: What does the sentence mean? For example, what does the "with" mean here: "I ate with a friend; I ate with a fork; I ate with a worry". Again there is the recurring ambiguity, specific actions/objects, etc.

4. ***pragmatic analysis***: Determine the actual meaning and intention in context (of speaker, of previous sentence, etc.): One needs to handle pronouns such as "it", implicit meanings. Also, communication is a ***speech act***: it not only informs but implies an action. To handle all this, one needs to keep track of the focus of the dialog, a model of each participant's beliefs, as well as knowing the rules and goals of dialog.

Much of the work on syntactical and semantic analysis centers on grammars: there are extended, augmented, unification, semantic, and case grammars, and more. The idea is that the grammar has built into it more knowledge of language, not just whether a sentence is valid or not.

One popular device was an *augmented transition network* (ATN), used in front-ends. These have syntax, meaning and specific (top-down) parsing instructions. The following example of the topmost level of an ATN is taken from Patterson. NP is noun phrase, V is verb, etc. PUSH means call that subnetwork, CAT means category test.



However, the price is complexity, and some have suggested that one should rather approximate with a regular grammar. One active area (again) is machine translation.

4.2 Speech

For spoken language, the processing tries to work out what word was spoken given the sounds. The key to success turns out to be that words and sounds are spoken in context: when a person says “I like ketchup on my...” the next word is more likely to be “fries” than “flies”.

One of the greatest advances was hidden model Markov chains: HMM. A ***Markov process*** (sometimes called a random walk) is a series of values over time $x_1, x_2, \ldots$. The standard/simplest case is where the process is ***memoryless***: at each point there is a certain probability of each next value which depends only on the current value and is independent of what happened before. So one has conditional probabilities: $\Pr(x_t = j \mid x_{t-1} = i)$.

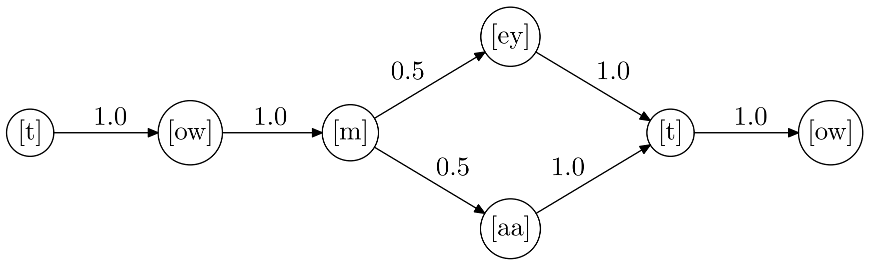
Speech recognition is trying to find the word that maximizes $\Pr(\mathit{word} \mid \mathit{signal})$. By Bayes’ theorem,

$$\Pr(\mathit{word} \mid \mathit{signal}) = \alpha \Pr(\mathit{signal} \mid \mathit{word}) \Pr(\mathit{word})$$

where α is the normalization constant.

The information about $\Pr(\mathit{words})$ is simply the likelihood of words: the ***language model***. In a Markov process, we are interested in the probability of successive words, called ***bigrams***. Fortunately, this can be gleaned by simply digesting many examples of sentences. A more advanced approach is to try to integrate with the fancy grammars mentioned above.

Now, the $\Pr(\mathit{signal} | \mathit{word})$ part is based on knowledge of how people speak (called the ***acoustic model***), homophones (similar words), and the properties of signal-processing equipment. In particular, in trying to develop a model of different dialects, one can break each word up into phonemes, each one a ***phone***: one of the 50 or so fundamental sounds (e.g. the [sh] sound in shoe or the [jh] sound in jet). Then one considers how different languages speak each. Hence the example of “tomato”, taken from Russell & Norvig (the 0.5 is arbitrary).



Now, the next step is to try to model the phones themselves, allowing for the fact that humans are sloppy, different, etc. And there’s segmentation.

Now the use of the Markov chains lies in the observational data. So, for example, one has the rain example. Each day it either rains or it doesn’t. Suppose one has only access to indirect observation: accidents. Rain tends to cause accidents, but one can have accidents when it’s dry and no accidents when it’s wet. The idea is, given a sequence of data (Acc, Acc, Nil, Acc, Acc), what is the most likely original sequence? Now, note that rain one day suggests rain the next day (and the previous day). So the sequence (Rain, Rain, Rain, Rain, Rain) might be more likely than (Rain, Rain, Dry, Rain, Rain). So in speech, we consider the sequence of sounds that we heard, and try to find the most likely original word.

Now, this sounds like an exponential algorithm. But Viterbi showed that the algorithm technique dynamic programming can be exploited to provide the fit fast.