Open source language technology using statistical methods

Kevin Scannell
Saint Louis University
August 21, 2008

What is Natural Language Processing?

- Defined here in terms of end-user applications:
- Spelling and grammar checking
- Search, information retrieval, question-answering
- Summarization, abstracting
- Speech recognition, synthesis
- Machine translation, translators' aids
- ... and all of the linguistic elements that feed into these applications (morphology, POS tagging, parsing, semantics, word sense disambiguation)

Why is NLP hard?

- Natural languages are ambiguous at many levels
- Lexical categories: “time flies like an arrow” (Marx)
- Lexical semantics: “the pen is in the box”, “the box is in the pen” (Bar-Hillel, 1960)
- Syntax: “I watched a movie with Kevin Scannell”, “I watched a movie with Kevin Costner” (PP attachment)
- Syntax: “old men and women” (coordination ambig.)
- Speech: waveform and several possible “decodings”
- Many others! Easy for humans, hard for computers.

Rule-based approaches

- The classical approach to resolving ambiguities was to construct sets of rules based on contextual clues
- e.g. POS tagging. If a word could be a noun or a verb ("work", "type", "drive" + thousands more), one rule might tag it as a noun if the preceding word is an article. Or if the preceding word is "can" or "should", tag it as a verb. And so on.
- Many rules required. Many exceptions and exceptions to exceptions. Labor intensive. Hard to maintain.

Statistical approaches

- Basic setup: imagine there is an ambiguity (of any of the types mentioned) that can be resolved in one of two ways, A or B (think POS tags or word senses)
- If we could compute the conditional probabilities P(A | context) and P(B | context), we could choose A or B based on which has a higher probability. “context” often means the surrounding words or POS tags
- Could try and estimate these probabilities by looking in a big corpus of texts, but given contexts usually don't recur enough for this to be realistic.

One Trick Pony: Bayes' Law

- P(A | C) = P(C | A)P(A)/P(C)
- P(context) is the same for A and B, so ignore it
- If the context is made up of several “features” (e.g. The three preceding words x,y,z), assume *independence* so P(context | A) = P(x | A)P(y | A)P(z | A)
- So now you can hopefully compute all these terms from a corpus: P(A), P(B), P(x | A), ..., P(x | B), ...
- e.g. “mouse”, A=computer sense, B=zoological sense, and terms like P(optical | A) or P(field | B) will dominate

Problems with statistics

- Need large corpora for training, and according to the description I've given the corpora need to be "tagged" in advance
- Results can depend strongly on the genre of the corpus. A corpus of technical documents will probably resolve the word "mouse" in the computer sense more than the zoological sense (P(A) near 1, P(B) near 0)
- Still not a silver bullet – statistics still can't capture the real-world knowledge humans bring to bear on these disambiguation tasks (e.g. Sample sentences!)

Language survey

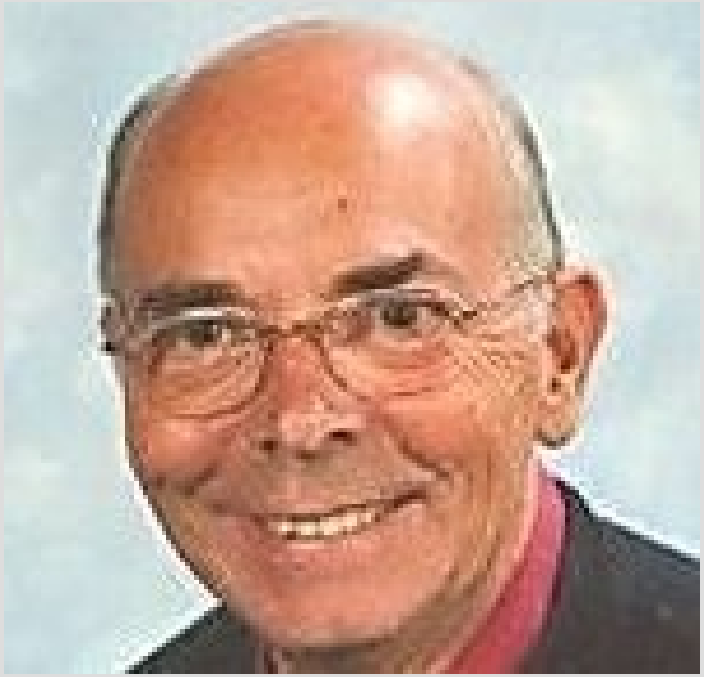
- Almost 7000 spoken languages in the world
- Most have fewer than 10K speakers, and it's expected that at least half will be extinct by 2100
- I am a speaker of one of these endangered languages (Irish, which has less than 20K daily speakers)
- Goal is to develop NLP technology for many of them in the interest of universal accessibility and language preservation
- Statistical techniques are driven by data (corpora and lexicons) - the “data bottleneck” for small languages

Breaking the data bottleneck

- Large corpora already exist for major languages such as English, French, Chinese. Until relatively recently, assembled by scanning or getting texts from publishers
- Rise of statistical NLP coincided with rise of the Web and virtually unlimited amounts of text for training
- I have a web crawler running at SLU that is gathering corpora for 427 languages: http://borel.slu.edu/crubadan/
- Volunteers from around the world are helping edit data extracted from these corpora to create open source spell checkers (more than 20 so far) and lexicons

Case Study: West Frisian

- Germanic language with about 500 000 speakers, most in the Netherlands
- Done over three weeks in Feb. 2007 in collaboration with Eeltje de Vries, a retiree with a background in theoretical physics



Morphological Description

- Root words with one or two prefixes and one or two suffixes
- This simplified description is easily encoded by novices and well-supported in open source tools (OpenOffice.org, Mozilla FF/TB)

```
# Affix file syntax:
# [PS]FX name strip add match

# moai->moaie, kreas->kreaze
SFX S 0 e [^esh]
SFX S ch ge ch
SFX S s ze s

# moai->moaier, kreas->kreazer
SFX T 0 er [^es]
SFX T 0 r e
SFX T s zer s

# moai->moaist, kreas->kreast
SFX U 0 st [^es]
SFX U 0 t s

...
```

Extract root words from corpus

wurdearje/V (5/5): wurdearje(18), wurdearrest(1), wurdearret(1), wurdearre(26), wurdearren(3), wurdearjend(1)

reagearje/V (5/5): reagearje(15), reagearrest(1), reagearret(13), reagearre(17), reagearren(3), reagearjend(1)

ynspirearje/V (4/5): ynspirearje(11), ynspirearrest(0), ynspirearret(2), ynspirearre(23), ynspirearren(1), ynspirearjend(12)

studearje/V (4/5): studearje(27), studearrest(0), studearret(17), studearre(34), studearren(4), studearjend(1)

konsumearje/V (4/5): konsumearje(1), konsumearrest(0), konsumearret(1), konsumearre(2), konsumearren(1), konsumearjend(1)

funksjonearje/V (4/5): funksjonearje(7), funksjonearrest(0), funksjonearret(9), funksjonearre(5), funksjonearren(1), funksjonearjend(1)

tramtearje/V (4/5): tramtearje(2), tramtearrest(0), tramtearret(1), tramtearre(1), tramtearren(1), tramtearjend(1)

Results

- Hand-checked lexicon with 22011 root words and 38677 derived forms
- This approach ensures obscure derived forms are included, unlike a pure corpus approach
- Spell checker recognizes 91% of the words in testing corpus (95% is an approximate expected upper bound for uncleaned corpora from the web)
- Existence of a lexicon with part-of-speech tags enables training of a POS tagger which in turn leads to more advanced tools (grammar checkers, parsers)

NLP Applications for Linux, I

- Spell checkers: Primary language-independent engines are ispell (classic), aspell (fast, good suggestions), and hunspell (support for complex morphology, integrated into OpenOffice, Mozilla). More than 100 dictionaries exist, of varied quality.
- Grammar checkers: Two language-independent rule-based engines: LanguageTool (English, German, Polish...) and An Gramadóir (Irish, Welsh, other small languages). Abiword has (English) Link Grammar parser integrated as a kind of grammar checker.

NLP Applications for Linux, II

- Summarization: MEAD, written in Perl and put into the public domain; see www.summarization.com
- Speech recognition: CMU Sphinx, Julius, voxforge (assembling transcribed speech corpora)
- Speech synthesis: MARY, eSpeak, Festival (can be used with KDE via KTTS daemon)
- Machine Translation: Moses (statistical), Apertium (rule-based, aimed at closely-related language pairs), OpenLogos (open source of an old MT system from the early 1970's)

Semantic Networks and Thesauri

- A semantic network is a database of words and semantic relationships between them, e.g. “tiger” is a kind of “mammal”, “trunk” is part of an “elephant”, ...
- Useful for humans (as a writing aid like a classical thesaurus), but even more useful for computers and NLP tasks like word-sense disambiguation
- First full-scale semantic network was created at Princeton in the 1980's: WordNet. Freely available. Basis for the English thesaurus that can be installed for use with OpenOffice.org.

WordNets for other languages

- WordNets now exist for at least 44 languages (see www.globalwordnet.org) but of these only the Princeton WordNet and my Irish language network are freely available (insert rant here!)
- Good things and unexpected things happen when people make software and data freely available; “mashup culture”
- Demo of aimsigh.com, built using my data and an open source 3D graph browser called Morcego (“bat”)