Tools, Toys, and Filters

DH from a personal perspective

Bryan Jurish
jurish@bbaw.de

Searching Linguistic Patterns in Large Text Corpora for Digital Humanities Research
ESU Digital Humanities 2016, Universität Leipzig
20th July, 2016
Short Bio

Education
- B.A. Philosophy / Cognitive Science, Northwestern University, 1996
- Diplom Computational Linguistics, Universität Potsdam, 2002
- Dr. phil. Computational Linguistics, Universität Potsdam, 2010

Experience
- Taught various CL courses in e.g. speech synthesis, grammar induction
- Research foci on historical text, integration of rule-based and empirical approaches, linguistic databases and search engines, ...
- Tinkering: various free open-source software packages
  - rats: “musician-friendly” realtime text-to-speech synthesizer
  - moot/WASTE: flexible Hidden Markov Model tagger/tokenizer
  - GFSM: low-level (weighted) finite-state machine utility library
  - DDC²: scalable & efficient corpus search engine
  - DTA::CAB: “cascaded analysis broker” for robust linguistic analysis
  - DiaCollo: diachronic collocation profiler
What is “Digital Humanities” anyways?

“Use filters”
— Brian Eno & Peter Schmidt, *Oblique Strategies*, 1975

- ∃0: mathematical objects are *out there!* (Plato; Gödel; Turing; Chaitin)
- ...but numbers do *not* usually “speak for themselves” (vs. Anderson, 2008)
- DH tools ∼ *filters* for cultural data (e.g. text corpora) (Shannon, 1948)
  - additional *encoding* applied to (already text-encoded) “message”
  - “*lossy*” filters (DH tools) *degrade* messages passed through them
    ...but humans have a whole bevy of lossy filters *already built in!*
    (linguistic, perceptual, cognitive, cultural, ...)
  - “*fast lane*” for *salient* (“interesting”) cultural data
    (∼ movement)
  - “*intuitivity*” ∼ *coherence* of human & software filters
    (∼ mp3, ogg)
- “agile” tool use ∼ *playful interaction* (tools ∼ toys)
- “tools” ⇒ *extrinsic evaluation* (useful for ...?)
  - tinkers & users need to work together!
A Humble Plea

Users, please . . .

- **read the documentation** provided **(and try to understand it!)**
- **don’t be afraid** of error messages **(they’re there to help you!)**
- expect to spend a good deal of **time & energy** acquainting yourself with an unfamiliar tool **(rarely does everything “just work”)**

If at first you don’t succeed . . .

- **read** the error message carefully
- **check** the documentation (again)
- **think** about what might have gone wrong
- **“simplify, simplify”** . . . until something works **(Thoreau)**
- **contact** the author/maintainer, including a **precise** description of:
  - what you wanted
  - what you tried
  - what went wrong