Large-Scale Syntactic Processing: Parsing the Web

James Curran

JHU 2009 Summer Research Workshop
Week 1 Progress
The Team

- Stephen Clark (Cambridge, UK)
- Ann Copestake (Cambridge, UK)
- James Curran (Sydney, Australia)
- Byung-Gyu Ahn (Johns Hopkins, US)
- James Haggerty (Sydney, Australia)
- Aurelie Herbelot (Cambridge, UK)
- Yue Zhang (Oxford, UK)
- Tim Dawborn (Sydney, Australia)
- Jonathan Kummerfeld (Sydney, Australia)
- Jessi Roesner (Texas, US)
- Curt Van Wyk (Northwestern, US)
By 31st July, we aim to produce...

• an end-to-end pipeline
  • from plain text, XML/SGML, HTML, Media wiki, ...
  • right through to logical form

• the parser of choice for the NLP community et al.

• processed and parsed Wikipedia in multiple output formats

• answers to a number of research questions ...
Research Questions

- how does a PTB-trained parser perform on Wikipedia?
- can we improve accuracy on Wikipedia text?
- can we improve parsing efficiency?
- can we improve coordination disambiguation?
- can we relieve the syntactic annotation bottleneck?
- all using large volumes of raw web/Wikipedia text
Pre-Processing (Curt, me)

- important but neglected task in NLP pipelines
- reading from raw text, HTML, wikipedia, etc . . .
- tokenisation and sentence boundary detection

- Boost Xpressive library for regular expressions
- full Unicode support
- integrated handling of document markup
- implemented PTB sed script

- benchmark against (f)lex grammar
- extend to cover web type tokens
Adapting to Wikipedia (Steve, Ann, Laura, Matt)

- approaches to adaptation:
  - manually annotate and re-train at lower levels of representation
  - self-training on parsed Wikipedia data

- create training and test data for Wikipedia:
  - 1,000 sentences of CCG POS-tagged and supertagged data
  - 200 sentences of CCG dependencies for evaluation
  - 300 sentences of grammatical relations for evaluation

- 100’s of sentences POS and supertagged
- 10’s of sentences labelled with grammatical relations

- more annotation!
Supertagger self-training (Jessi, Jonathan, Steve, me)

- the CCG parser has two stages: supertagger and chart parser
- we will investigate the use of self-training to improve:
  - accuracy on Wikipedia data
  - efficiency: train the parser to select the lexical categories
    that the parser will eventually select anyway

- preliminary distributed Maximum Entropy estimation
- added extended context features

- more efficient MPI implementation
- train on enormous Wikipedia parser output
Chart Classification (Byung Gyu, Yue, Steve, me)

- various ways of pruning the chart:
  - predict which chart cells (don't) have constituents
  - rerank the constituents in a cell and prune
  - where will standard beam search work?

- *All of these models trained on parser output*

- more to come from Byung Gyu and Yue shortly
Coordination Disambiguation (Aurelie, Ann)

- parsers typically perform poorly on coordination constructions
- CCG offers elegant analyses of coordination phenomena
- apply coordination reranker to N-best parser output

- baseline system with the following features:
  - head similarity on WordNet
  - distances to 2nd coordinate and between coordinates
  - similar POS sequence of coordinates
  - initial parser ranking in 1000-best parses

- send them to classifier, initially Weka?
Bootstrapping (James, me)

- facts from web/Wikipedia to obtain accurate training data
  - e.g. Mozart was born January 27, 1756
  - short factoid sentences can be reliably parsed
  - sentence with same entities/verbs $\rightarrow$ same dependencies
- Research questions:
  - how to identify the facts and dependencies as constraints
  - how to train the parser model on partial data

- reimplement honours student project with Yahoo! Boss
- explore output with existing factoids

- extract/verify a much larger range of facts
- extract all facts for a given entity in Wikipedia
Pre-built and flat structures (Tim, Steve, me)

- named entities (person, organisation, location, etc)
- fixed expressions (dates, times, phone numbers, etc)

- **Stop press: one parse per n-gram?** (David Yarowsky)
- binary serialisation of derivations – pointer swizzling
- serialisation into fast database – Tokyo Cabinet
- can now load pre-parsed n-gram

- parse Wikipedia and save the yielded n-grams
Self training is the key for efficiency

- **What Would the Parser Do (WWPD)?**
- especially in new domains like Wikipedia
- train the supertagger to emulate the parser’s choices
- train the chart to emulate the parser’s choices
- as much annotated data as we can handle
Oracles give us upper bounds

- what if we had perfect supertags?
- what if we knew part of the gold-standard derivation?
- what if we knew what the gold-standard derivation?
- what if we knew what the parser was going to choose anyway?
Aggressive pruning and backoff

- we have three types of pruning:
  1. supertagging (lexical item pruning)
  2. beam search (constituent pruning) – Byung Gyu
  3. cell classification (chart pruning) – Yue
- C&C parser only does the first at the moment
- C&C parser backs off if this fails
- experimenting with \textit{WSJ} gold-standard in Week 1
- Wikipedia in Week 2 – and d) all of the above
Summary

- parser of choice for the NLP community (especially for researchers doing large-scale web processing)
  - adaptable, efficient, accurate, easy-to-use, multiple outputs, . . .
- moving on from the standard Penn Treebank parsing task
  - larger scale, new domains, different representations