MorphAdorner

Morphological Adornment of English Language Literary Texts

Software Development Group
November 1, 2012

Why "Adornment"?

- Terms like annotation, tagging, etc. have too many alternate and confusing meanings
- Adornment harkens back to medieval sense of manuscript adornment or illumination -- attaching pictures, marginal comments, etc. to texts
- Morphological adornment is thus the process of "adorning" words with morphological information such as part of speech, lemma, standardized spelling, semantic category, etc.

Sample Adornment Processes

- Tokenization
- Sentence Boundary Recognition
- Spelling Normalization
- Part of Speech Tagging
- Lemmatization
- Name Extraction

MorphAdorner Pipeline

- MorphAdorner provides "skeleton" for pipelining adornment processes
- Use of Java interfaces for adornment processes allows easy substitution of different implementations into pipeline (e.g., Template method pattern)
- Straightforward to wrap adornment processes as web services using Rest-like interfaces

MorphAdorner Audience

- MorphAdorner intended as a programmer's construction kit, not an end-user program
- MorphAdorner can be used to create customized end-user programs for morphological adornment
- Released to public under open source license in April 2009
- Some continued updates to training data during 2010
- Work on wrapping MorphAdorner facilities as RESTful web services during Fall 2012
- Multiple improvements planned, especially to XML processing, from Fall 2012 through Summer 2013

Tokenization

- Recognizing word boundaries
- Break text on various whitespace and delimiter characters
- Difficulties with badly scanned or transcribed text
- Difficulties with some embedded characters, e.g., quote, hypen
- Difficulties with numbers (Roman numerals, decimals)
- Different part of speech tag sets require different tokenization -- some require compound forms split (don't -> don 't or do n't)

Sentence Boundary Recognition

- Recognize sentence boundaries
- Punctuation not always indicative -- period in abbreviation, etc.
- Need different sentence end detection for prose and poetry
- Deal with existing sentence markup in xml
- Handle tabular input (verticalized text)

Spelling Normalization

- Map variant spellings to standard, hopefully modern, spellings
- Mapping variants to a standard spelling helps with searching, part of speech tagging, etc.
- Mapping done manually and using lists extracted from sources like OED
- Also use variety of rules, heuristics, and algorithms for spelling "correction", phonetic matching. and string distance
- Currently have over 360,000 spellings mapped to standard spellings

The Faerie Queene: Book One Prologue

Lo I the man, whose Muse whilome did maske,
As time her taught in lowly Shepheards weeds,
Am now enforst a far vnfitter taske,
For trumpets sterne to chaunge mine Oaten reeds,
And sing of Knights and Ladies gentle deeds;
Whose prayses hauing slept in silence long,
Me, all too meane, the sacred Muse areeds
To blazon broad emongst her learned throng:
Fierce warres and faithfull loues shall moralize my song.

Lo I the man, whose Muse whilom did mask,
As time her taught in lowly Shepherds weeds,
Am now enforced a far unfitter task,
For trumpets stern to change mine Oaten reeds,
And sing of Knights and Ladies gentle deeds;
Whose praises having slept in silence long,
Me, all too mean, the sacred Muse areads
To blazon broad amongst her learned throng:
Fierce wars and faithful loves shall moralize my song.

Part of Speech Tagging

- Assigns parts of speech to each spelling
- Frequency based, e.g., unigram tagger
- Pattern based, e.g., regular expression tagger, affix tagger
- Rule-based, e.g., modified Hepple tagger
- Statistically based bigram and trigram taggers using hidden Markov models and beam-search variant of Viterbi algorithm
- "Fixup" or second step retagging
- Unknown word recognition using suffix analysis and successive abstraction

Part of Speech Tagging: Tag Sets

- Using Martin Mueller's "nupos" part of speech tag set for early Modern English texts
- Using Penn Treebank set for unrelated contemporary texts project
- Can use arbitrary tag set given appropriate training data

Lemmatization

- Find base form for each spelling
- Lexicon based for large number of spellings
- Standardized mapping helps determine lemmata for variant spellings
- Simple stemming (Porter, Lancaster) -- do not necessarily result in morphologically correct base forms
- Machine learning methods
- Rule and exception list based methods

Sample Adorned Text Fragment

<p>
<hi rend="sc(1)">
<w eos="0" lem="Emma" pos="np1" reg="Emma" spe="Emma" tok="Emma" xml:id="ncf0204-0000550" ord="40" part="N">Emma</w>
</hi>
<c> </c>
<w eos="0" lem="Woodhouse" pos="np1" reg="Woodhouse" spe="Woodhouse" tok="Woodhouse" xml:id="ncf0204-0000560" ord="41" part="N">Woodhouse</w>
<pc eos="0" lem="," pos="," reg="," spe="," tok="," xml:id="ncf0204-0000570" ord="42" part="N">,</pc>
<c> </c>
<w eos="0" lem="handsome" pos="j" reg="handsome" spe="handsome" tok="handsome" xml:id="ncf0204-0000580" ord="43" part="N">handsome</w>
<pc eos="0" lem="," pos="," reg="," spe="," tok="," xml:id="ncf0204-0000590" ord="44" part="N">,</pc>
<c> </c>

Sample web service example: Lemmatizer

Find lemma form of early modern English spelling "strykynge"

http://localhost:8182/lemmatizer?spelling=strykynge&standardize=true&wordClass=verb&wordClass2=&corpusConfig=eme

Lemmatizer example (cont.)

XML result:

<LemmatizerResult>
<spelling>strykynge</spelling><standardSpelling>striking</standard
Spelling><corpusConfig>eme</corpusConfig>
<wordClass>verb</wordClass><wordClass2/><lemma>strike</lemm
a>
<standardize>true</standardize><lancasterStem>strik</lancasterStem
><porterStem>strike</porterStem>
</LemmatizerResult>

Lemmatizer example (cont.)

JSON result:

{"spelling":"strykynge",
"standardSpelling":"striking",
"corpusConfig":"eme",
"wordClass":"verb",
"wordClass2":"",
"lemma":"strike",
"standardize":true,
"lancasterStem":"strik",
"porterStem":"strike"}

Pos Tagging Example

Text to adorn: Mary had a little lamb.

http://localhost:8182/partofspeechtagger?text=Mary+had+a+little+lamb.&corpusConfig=ncf

(In practice we would use HTTP post to allow for long texts.)

Pos Tagging Example (cont.)

<PartOfSpeechTaggerResult>
<text>Mary had a little lamb.
</text><corpusConfig>ncf</corpusConfig>
<sentences>
<sentence>
<token>Mary</token>
<token>had</token>
<token>a</token>
<token>little</token>
<token>lamb</token>
<token>.</token>
</sentence>
</sentences>

Pos Tagging Example (cont.)

<adornedSentences>
<adornedSentence>
<AdornedWord>
<token>Mary</token>
<spelling>Mary</spelling><standardSpelling>Mary</standardSpelling><lemmata>Mary</lemmata>
<partsOfSpeech>np1</partsOfSpeech>
</AdornedWord>
...

Other MorphAdorner Facilities

- Language Recognition
- Name Standardization
- Parser
- Pluralizer
- Statistics (Dunning's Log Likelihood and others)
- Stemming
- Syllabification
- Text Segmenter
- Text Summarization
- Thesaurus (synonyms and antonyms)
- Verb Conjugator

Other hooks

- Custom output adapters for generating input to Xaira, the Corpus Workbench (CWB), Lucene, and word lists in a variety of formats.
- MorphAdorner can also be integrated with Gate (and therefore UIMA).
- Improvements to Gate and UIMA integration planned for 2012/2013.

Personal Goal: Comprehensive Lexicon

- Spelling and variants with date information
- Frequencies of occurrence across centuries
- Frequencies of occurrence for specific genres
- Frequencies of occurrences by part of speech
- Lemmata by part of speech
- Allows morphological adornment processes to use a standardized lexicon ID

MorphAdorner and Other Projects

- Designed to work with several of the corpora already designated for use in the initial phase of Project Bamboo
- More aware of potential problems and pitfalls than other existing software for adornment of texts
- NUPos tag set allows adornment of English texts from Middle English to present (diachronic corpora)
- Licensed under a very non-restrictive NCSA style open source license
- Already integrated with Abbot.
- Integration planned for Philologic.
- Early planning for integration with NINES, UWisc projects, F21, ESTC, etc.

Summary

- MorphAdorner started providing basic morphological adornment in December 2006
- Used in WordHoard, Monk, and Virtual Orthographic Normalization projects
- First public release in April 2009 under NCSA style open source license
- Updated training data during 2010 and 2011
- Added initial RESTful interfaces in October 2012
- Many improvements scheduled for 2012/2013 academic year
- Looking to integrate with existing/forthcoming web services from other projects