Transformation-Based Learning

and

Part-of-Speech Tagging of Old English

Beni Ruef <bruef@es.unizh.ch>

Why on earth does one choose such an exotic topic?

Mix of personal interests:

- Old English
- Computational and Corpus Linguistics
- TEI/XML

I will *not* talk about:

- (Perl) programming (conversion of corpora, evaluation scripts)
- part-of-speech tagging in general (various approaches)

I *will* talk about:

- transformation-based learning and its application to part-of-speech tagging
- part-of-speech tagging of Old English, its problems and possible optimizations
- the Brooklyn Corpus (if time allows. . .)

Why this enduring interest in part-of-speech tagging?

> The widespread interest in tagging is founded on the belief that many NLP applications will benefit from syntactically disambiguated text. Given this ultimate motivation for part-of-speech tagging, it is surprising that there seem to be more papers on stand-alone tagging than on applying tagging to a task of immediate interest.
>
> (Manning and Schütze 1999: 374–75)

So what's the goal?

Our running example:

Þa	se	cyning	þis	gehyrde	,	þa	ongon
CS	PDN	NNN	PDA	VT	PUN	RT	VT
When	the	king	this	heard	,	then	began

he	lustfullian	þæs	biscopes	wordum	.
PEN	VV	PDG	NNG	NND	PUN
he	enjoy	the	bishop's	words	.

("When the king heard this, he began to rejoice at the words of the bishop.")

What is transformation-based learning?

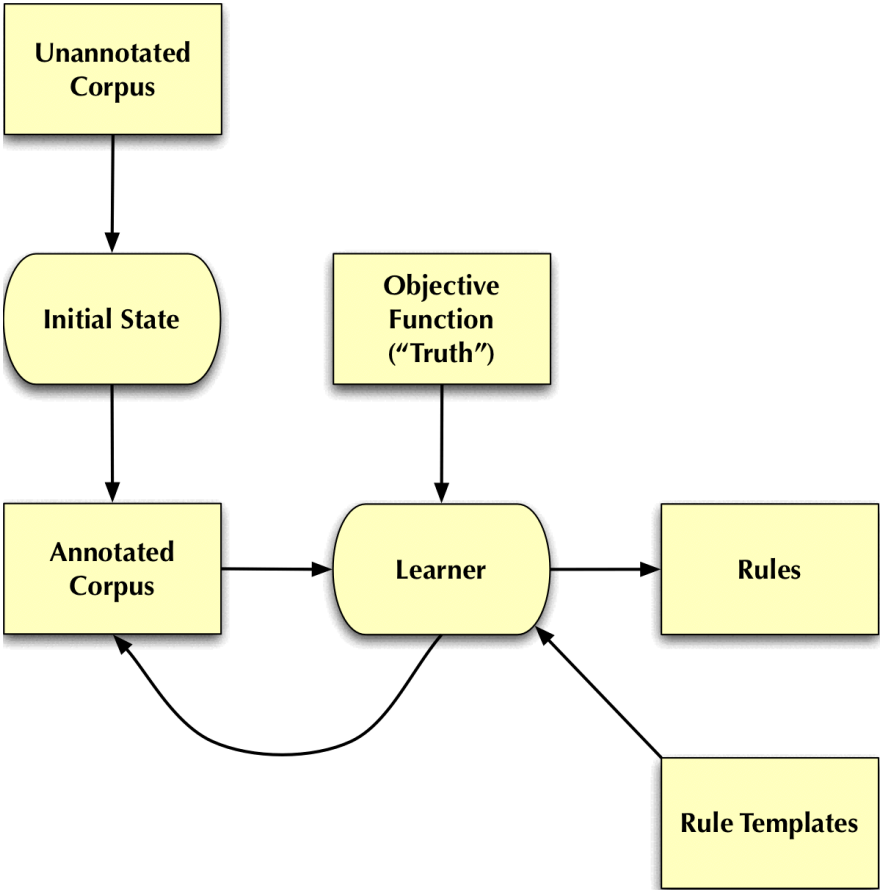
- example of rule-based machine learning
- can be used for many classification tasks
- has been applied to a wide variety of NLP tasks
  - POS tagging
  - PP attachment
  - NP chunking
  - word sense disambiguation
  - etc.

How does transformation-based learning work?

It consists of two phases:

- training phase (typically applied once):
  rules are learnt
- application phase (typically applied many times):
  the rules are applied in the order they were learnt

Training phase in transformation-based learning



(Some) Implementations of transformation-based learning

- original algorithm by Brill (1992): very lengthy training times
- algorithm by Ramshaw and Marcus (1994): faster, but extremely memory-consuming
- algorithm by Ngai and Florian (2001)
  - very fast (typically two orders of magnitude compared to Brill's implementation)
  - more flexible than Brill's implementation (not limited to POS tagging, rule templates are not hard-coded)
  - supports multidimensional learning, i.e. multiple-task classification

POS tagging with transformation-based learning

1. initial state
   - known words — i.e. words found in the lexicon — are tagged with their most frequent tag
   - unknown words are tagged with the most frequent tag in the training corpus (alternatively: most frequent tag of hapax legomena), depending on the first letter (capital or not) of the word in question
2. lexical tagging: the unknown words are tagged in isolation, based on their morphology and their immediate neighbour
3. contextual tagging: all words are tagged in context

Initial state

Þa	se	cyning	þis	gehyrde	,	þa	ongon	
CS	PDN	NNN	PDA	VT	PUN	RT	VT	(correct tags)
RT	PDN	NNN	PDN	VT	PUN	RT	VT	(guessed tags)

he	lustfullian	þæs	biscopes	wordum	.	
PEN	VV	PDG	NNG	NND	PUN	(correct tags)
PEN	VT	PDG	NNG	NND	PUN	(guessed tags)

(*lustfullian* is an unknown word; the tagging of *gehyrde*, *ongon*, *he*, *biscopes*, *wordum*, and the punctuation marks is unambiguous.)

Lexical tagging

Þa	se	cyning	þis	gehyrde	,	þa	ongon	
CS	PDN	NNN	PDA	VT	PUN	RT	VT	(correct tags)
RT	PDN	NNN	PDN	VT	PUN	RT	VT	(guessed tags)

he	lustfullian	þæs	biscopes	wordum	.	
PEN	VV	PDG	NNG	NND	PUN	(correct tags)
PEN	VV	PDG	NNG	NND	PUN	(guessed tags)

(The tagging of *lustfullian* has been corrected.)

Contextual tagging

Þa	se	cyning	þis	gehyrde	,	þa	ongon	
CS	PDN	NNN	PDA	VT	PUN	RT	VT	(correct tags)
CS	PDN	NNN	PDA	VT	PUN	RT	VT	(guessed tags)

he	lustfullian	þæs	biscopes	wordum	.	
PEN	VV	PDG	NNG	NND	PUN	(correct tags)
PEN	VV	PDG	NNG	NND	PUN	(guessed tags)

(All tags have been corrected. . .)

Rules in lexical tagging

Examples of rule templates:

```
pos word::~~2 => pos
pos word^^-1 => pos
```

Examples of corresponding rules:

```
pos=VT word::~~2=~~an => pos=VV
```

(“Change the tagging from *finite verb* to *infinitival verb* if the unknown word ends with *-an*.”)

```
pos=NNG word^^-1=se => pos=JJN
```

(“Change the tagging from *common noun (genitive)* to *adjective (nominative)* if the unknown word’s immediate left neighbour is the word *se*.”)

Rules in contextual tagging

Examples of rule templates:

```
pos_0 pos_1 => pos
pos_0 pos:[-2,-1] => pos
```

Examples of corresponding rules:

```
pos_0=RT pos_1=PEN => pos=CS
```

(“Change the tagging from *temporal adverb* to *subordinating conjunction* if the word’s immediate right neigbour is tagged *personal pronoun (nominative)*.”)

```
pos_0=NNN pos:[-2,-1]=PDA => pos=NNA
```

(“Change the tagging from *common noun (nominative)* to *common noun (accusative)* if one of the two previous words is tagged *demonstrative pronoun (accusative)*.”)

What about the accuracy of the tagging?

- 88.5% total accuracy (79.5% without considering unambiguous tokens, i.e. 44% of the tokens are unambiguous; after the initial state already 80% of the tokens are tagged correctly. . .)
- 91.5% accuracy for known tokens
- (only) 56.5% accuracy for unknown tokens

(training corpus: 108’000 words, test corpus: 12’000 words)

Most frequent tagging errors

1. errors in grammatical case (NNA vs. NNN, JJA vs. JJN)
2. copula ‘be’ vs. auxiliary ‘be’
3. infinitive vs. plural finite verb forms
4. all these notorious small “thorn words”:
   - *þæt* is both a (demonstrative and relative) pronoun and a conjunction (cf. Modern English)
   - *þa* is both a demonstrative pronoun, a conjunction (‘when’), and an adverb (‘then’; cf. running example)
   - *þonne* is both a conjunction (‘when’ and ‘than’) and an adverb (‘then’)

So what are the (main) headaches when applying transformation-based learning to POS tagging of Old English?

- (very) free word order, aggravating the contextual tagging
- non-normalized spelling, resulting in fewer occurrences of identical word forms, i.e. a high ratio of word forms per lemma
- hapax legomena (8.5% of tokens in training corpus, 21% in test corpus!)
- unknown words (8.5% of tokens in test corpus)
- training corpus too small

Some possible optimizations for the improvement of the tagging's accuracy

- enhanced rule templates in the lexical tagging, allowing the matching of longer affixes in unknown words
- bigger (unannotated) word list, helping the tagging of unknown words
- token normalization, starting with lowercasing all words besides proper nouns
- simplified tagset, e.g. by removing grammatical case information

Simplifying the tagset by removing grammatical case information

- both training and testing is carried out with the original tagset (94 different tags occurring in the test corpus)
- the tagged test corpus is matched to a simplified tagset (48 different tags occurring in the test corpus) which lacks grammatical case information for nouns, pronouns, adjectives, and participles
- this results in the following, rather improved accuracy:
  - 92.5% total accuracy (87% without considering unambiguous tokens)
  - 95% accuracy for known tokens
  - 65% accuracy for unknown tokens

Questions?