Transformation-Based Learning

Christian Siefkes

christian@siefkes.net

Introduction

- An ‘error-driven’ approach for learning an ordered set of rules
- Adds annotations/classifications to each token of the input
- Developed by Brill [1995] for POS tagging
- Also used for other NLP areas, e.g.
  - ➢ text chunking [Ramshaw and Marcus 1995; Florian et al. 2000]
  - ➢ prepositional phrase attachment [Brill and Resnik 1994]
  - ➢ parsing [Brill 1996]
  - ➢ dialogue act tagging [Samuel 1998]
  - ➢ named entity recognition [Day et al. 1997]

Required Input

For application:

- The input to annotate:
  POS: *Recently, there has been a rebirth of empiricism in the field of natural language processing.*

Additionally for training:

- The correctly annotated input (‘truth’):
  POS: *Recently/RB ,/, there/EX has/VBZ been/VBN a/DT rebirth/NN of/IN empiricism/NN in/IN the/DT field/NN of/IN natural/JJ language/NN processing/NN ./.*

Preliminaries

- Templates of admissible transformation rules (triggering environments)
- An initial-state annotator

  POS:

  Known words: Tag each word with its the most frequent tag.

  Unknown words: Tag each capitalized word as proper noun (NNP); each other word as common noun (NP).
- An objective function for learning

  POS: *Minimize the number of tagging errors.*

Transformation Rules

Rewrite rules: what to replace

POS: $t_i \rightarrow t_j$; $* \rightarrow t_j$ *(replace tag t_i / any tag by tag t_j)*

Triggering environment: when to replace

POS:

Non-lexicalized templates:

1. The preceding (following) word is tagged t_a.
2. The word two before (after) is tagged t_a.
3. One of the two preceding (following) words is tagged t_a.
4. One of the three preceding (following) words is tagged t_a.
5. The preceding word is tagged t_a and the following word is tagged t_b.
6. The preceding (following) word is tagged t_a and the word two before (after) is tagged t_b.

Lexicalized templates:

1. The preceding (following) word is w_a.
2. The word two before (after) is w_a.
3. One of the two preceding (following) words is w_a.
4. The current word is w_a and the preceding (following) word is w_b.
5. The current word is w_a and the preceding (following) word is tagged t_a.
6. The current word is w_a.
7. The preceding (following) word is w_a and the preceding (following) tag is t_a.
8. The current word is w_a, the preceding (following) word is w_b and the preceding (following) tag is t_a.

Learning Algorithm

1. Generate all rules that correct at least one error.
2. For each rule:
   (a) Apply to a copy of the most recent state of the training set.
   (b) Score the result using the objective function.
3. Select the rule with the best score.
4. Update the training set by applying the selected rule.
5. Stop if the score is smaller than some pre-set threshold T; otherwise repeat from step 1.

Rules Learnt

The first rules learnt by Brill’s POS tagger (with examples):

#	From	To	If
1	NN	VB	previous tag is TO
	to/TO conflict/NN→ NB		
2	VBP	VB	one of the previous 3 tags is MD
	might/MD vanish/VBP→ VB		
3	NN	VB	one of the previous two tags is MD
	might/MD not reply/NN→ VB		
4	VB	NN	one of the previous two tags is DT
	the/DT amazing play/VB→ NN		

Tagging Unknown Words

Additional rule templates use character-based cues:
Change the tag of an unknown word from X to Y if:

1. Deleting the prefix (suffix) x, $|x| \leq 4$, results in a word.
2. The first (last) 1–4 characters of the word are x.
3. Adding the character string x, $|x| \leq 4$, as a prefix (suffix) results in a word.
4. Word w appears immediately to the left (right) of the word.
5. Character z appears in the word.

Unknown Words: Rules Learnt

#	From	To	If
1	NN	NNS	has suffix **-s**
	rules/NN$\longrightarrow$ NNS		
4	NN	VBN	has suffix **-ed**
	tagged/NN$\longrightarrow$ VBN		
5	NN	VBG	has suffix **-ing**
	applying/NN$\longrightarrow$ VBG		
18	NNS	NN	has suffix **-ss**
	actress/NNS$\longrightarrow$ NN		

Training Speedup: Hepple

Disallows interaction between learnt rules,
by enforcing two assumptions:

- Sample independence: a state change in a sample does not change the context of surrounding samples
- Rule commitment: there will be at most one state change per sample

➔ Impressive reduction in training time, but the quality of the results is reduced (assumptions do not always hold)

‘Lossless’ Speedup: Fast TBL

1. Store for each rule r that corrects at least one error:
   - $good(r)$: the number of errors corrected by r
   - $bad(r)$: the number of errors introduced by r
2. Select the rule b with the best score.
   Stop if the score is smaller than a threshold T.
3. Apply b to each sample s.
4. Considering only samples in the set $\bigcup_{\{s|b\,\text{changes}\,s\}} V(s)$, where $V(s)$ is the set of samples whose tag might depend on s (the ‘vicinity’ of s; $s \in V(s)$):
   - Update $good(r)$ and $bad(r)$ for all stored rules, discarding rules whose $good(r)$ reaches 0.
   - Add rules with a positive $good(r)$ not yet stored.

Repeat from step 2. [Ngai and Florian 2001]

Text Chunking

A robust preparation for / alternative to full parsing.

- Input: *A.P. Green currently has 2,664,098 shares outstanding.*
- Expected output: ***[NP** A.P. Green**] [ADVP** currently**] [VB** has**] [NP** 2,664,098 shares**] [ADJP** outstanding**]**.*
- Alternative representation: *A.P./B-NP Green/I-NP currently/B-ADVP has/B-VP 2,664,098/B-NP shares/I-NP outstanding/B-ADJP ./O*
- Rules: Similar to those used for POS tagging, considering
  - ➢ Words
  - ➢ POS tags
  - ➢ Chunk tags

Prepositional Phrase Attachment

Samples:
1. *I* ***[VB*** *washed****]*** ***[NP*** *the shirt****]*** ***[PP*** *with soap and water****]****.*
2. *I* ***[VB*** *washed****]*** ***[NP*** *the shirt****]*** ***[PP*** *with pockets****]****.*

Task: Is the prepositional phrase attached to the verb (sample 1) or to the noun phrase (sample 2)?

Approach: Apply TBL to 4-tuple of base head words (tag tuple as either *VB* or *NP*):

1. *wash shirt with soap*
2. *wash shirt with pocket*

Rules: Templates consider the words in the tuple and their semantic classes (WordNet hierarchy)

Evaluation

POS tagging:

	Regular TBL	Fast TBL	Hepple
Accuracy	96.61%	96.61%	96.23%
Time	38:06h	17:21min	6:13min

Prepositional Phrase Attachment:

	Regular TBL	Fast TBL	Hepple
Accuracy	81.0%	81.0%	77.8%
Time	3:10h	14:38min	4:01min

Scaling on input data:

Fast TBL: linear

Regular TBL: almost quadratic

Advantages

- Can capture more context than Markov models
- Always learns on the whole data set – no ‘divide and conquer’ ➔ no data sparseness:
  - ➢ Target evaluation criterion can be directly used for training, no need for indirect measures (e.g. entropy)
  - ➢ No overtraining
- Can consider its own (intermediate) results on the whole context ➔ More powerful than other methods like decision trees [Brill 1995, sec. 3]

More Advantages

- Can do any processing, not only classification:
  - ➢ Can change the structure of the input (e.g. parse tree)
  - ➢ Can be used as an postprocessor to any annotation system
- Resulting model is easy to review and understand
- Very fast to apply – rule set can be converted into a finite-state transducer [Roche and Schabes 1995] (for tagging and classification) or finite-state tree automaton [Satta and Brill 1996] (for parsing and other tree transformations)

… and Disadvantages

- Greedy learning so the found rule sequence might not be optimal
- Not a probabilistic method:
  - ➢ Cannot directly return more than one result (*k*-best tagging can be added but is not built-in [Brill 1995, sec. 4.4])
  - ➢ Cannot measure confidence of results (through [Florian et al. 2000] estimate probabilities by converting transformation rule lists to decision trees and computing distributions over equivalence classes)

References

Brill, Eric (1995). Transformation-Based Error-Driven Learning and Natural Language Processing: A Case Study in Part-of-Speech Tagging. *Computational Linguistics*, 21(4):543–565. Online (access date: 2003-01-02): http://citeseer.nj.nec.com/brill95transformationbased.html.

Brill, Eric (1996). *Recent Advances in Parsing Technology*, chap. Learning to Parse with Transformations. Kluwer.

Brill, Eric and Philip Resnik (1994). A Rule-Based Approach To Prepositional Phrase Attachment Disambiguation. In *Proceedings of COLING'94*. Kyoto. Online (access date: 2003-02-04): http://www.cs.jhu.edu/~brill/pp-attachment.ps.

Day, David; John Aberdeen; Lynette Hirschman; Robyn Kozierok; Patricia Robinson; and Marc Vilain (1997). Mixed-Initiative Development of Language Processing Systems. In *Fifth Conference on Applied Natural Language Processing*, pp. 348–355. Association for Computational Linguistics. Online (access date: 2003-02-01): http://www.mitre.org/technology/alembic-workbench/ANLP97-bigger.html.

Florian, Radu; John C. Henderson; and Grace Ngai (2000). Coaxing Confidences from an Old Friend: Probabilistic Classifications from Transformation Rule Lists. In *Proceedings of Joint Sigdat Conference on Empirical Methods in Natural Language Processing and Very Large Corpora (EMNLP/VLC-2000)*. Hong Kong University of Science and Technology. Online (access date: 2003-01-27): http://arXiv.org/ps/cs/0104020.

Ngai, Grace and Radu Florian (2001). Transformation-Based Learning in the Fast Lane. In *Proceedings of the Second Conference of the North American Chapter of the Association for Computational Linguistics (NAACL)*. Pittsburgh, PA. Online (access date: 2003-01-27): http://nlp.cs.jhu.edu/%7Erflorian/papers/naacl01.ps.

Ramshaw, Lance and Mitch Marcus (1995). Text Chunking Using Transformation-Based Learning. In *Proceedings of the Third Workshop on Very Large Corpora*, eds. David Yarovsky and Kenneth Church, pp. 82–94. Association for Computational Linguistics, Somerset, New Jersey. Online (access date: 2003-02-04): http://citeseer.nj.nec.com/ramshaw95text.html.

Roche, Emmanuel and Yves Schabes (1995). Deterministic Part-of-Speech Tagging with Finite-State Transducers. *Computational Linguistics*, 21(2):227–253. Online (access date: 2003-01-31): http://citeseer.nj.nec.com/roche95deterministic.html.

Samuel, Ken (1998). Lazy Transformation-Based Learning. In *Proceedings of the 11th International Florida Artificial Intelligence Research Symposium Conference*. Online (access date: 2003-01-27): http://xxx.lanl.gov/ps/cmp-lg/9806003.

Satta, Giorgio and Eric Brill (1996). Efficient Transformation-Based Parsing. In *Proceedings of ACL 1996*. Online (access date: 2003-01-27): http://www.cs.jhu.edu/~brill/Eff_Pars.ps.