

Learning Transformation Rules for Semantic Role Labeling

2004 CoNLL Shared Task

A highly non-state-of-the-art approach

Ken Williams, Christopher Dozier, Andrew McCulloh

Thomson Legal & Regulatory

{ken.williams|chris.dozier|andrew.mcculloh}@thomson.com

Motivation

- Need extremely high-precision extraction (~98%)
- Typically ~1 person-year per application spent hand-writing rules
- Need to integrate with human architects, human users, decrease time-to-application

Technique

- Transformation-Based Error-Driven Learning in the style of Eric Brill
- Similar to Derrick Higgins' entry, but with different features
- No more hand-written rules - replaced by hand-written rule *templates*

Templates (verb)

Lengthen [shorten] the end of region V by one token if:

a,b) followed by chunk with tag=X

c,d) followed by token with POS=X

e,f) followed by chunk with tag=X and token with POS=Y

g,h) the verb token's lemma is X

Templates (arguments)

- A,B) If chunk with tag=X is followed [preceded] directly by region with tag=Y, mark chunk as Z
- C,D) If token with POS=X is followed [preceded] directly by region with tag=Y, mark token as Z
- E,F) If chunk with tag=X is followed [preceded] (perhaps indirectly) by region with tag=Y, mark chunk as Z
- G,H) If region with tag=X is followed [preceded] by chunk with tag=PP, which is in turn followed [preceded] by chunk with tag=Y, extend X forward [backward] through Y
- I,J) If verb's first token has POS=X [and is preceded by POS=Y], switch A0 and A1
- K) If region with tag=X is contained in a clause-starting verb phrase, and this is preceded by a clause-starting token with POS=Y , mark token Y as Z

Learning

- Iterative hill-climbing search through large space (optimization is crucial)
- Target measure is F_1
- To alleviate non-optimal dependencies, a “*look-behind*” window is used for re-ordering learned rules
- We trained first on verbs, then A0 & A1, then all argument types
- No lexical information used (for loose definition of “no”)
- PropBank [almost] not used at all

Results

Basically created an automatically-trained baseline system

	Precision	Recall	F_1
Dev	53.37	32.43	40.35
Test	58.08	34.75	43.48

Other Results

- Much better when training separately on common verbs: $F_1 = 73.6\%$ on “say”, 56.6% on “have” (*how does this compare?*)
- Overall dev-set performance increases from 40.35% to 41.18% when isolating “say”; to 41.3% when isolating “say” and “have”
- Probably need more training data to continue this for other verbs – perhaps group verbs by class?

Thank You