

Empirical Approaches to Multilingual Lexical Acquisition

Lecturer: Timothy Baldwin



THE UNIVERSITY OF
MELBOURNE

Lecture 8

General-purpose Lexical Acquisition

Bootstrapping Deep Lexical Resources: Resources for Courses

(Baldwin 2005)

Target DLR: English Resource Grammar

- Implemented open-source broad-coverage precision Head-driven Phrase Structure Grammar
- 10,500 lexical items (\approx 20,500 word forms)
- Example lexical entry:

```
admission_n1 := n_pp_mc-of_le &  
  [ STEM < "admission" >,  
    SYNSEM [ LKEYS.KEYREL.PRED "_admission_n_rel" ] ] .
```

- ERG associated with Redwoods treebank

Target DLR: English Resource Grammar

- Implemented open-source broad-coverage precision Head-driven Phrase Structure Grammar
- 10,500 lexical items (\approx 20,500 word forms)
- Example lexical entry:

```
admission_n1 := n_pp_mc-of_le &  
  [ STEM < "admission" >,  
    SYNSEM [ LKEYS.KEYREL.PRED "_admission_n_rel" ] ] .
```

- ERG associated with Redwoods treebank

Set of Lexical Types

- Open-class lexical types/items in ERG:

<i>Word class</i>	<i>Lexical types</i>	<i>Lexical items</i>
Noun	28	3,032
Verb	39	1,334
Adjective	17	1,448
Adverb	26	721
Total	110	5,675

- Assume prior knowledge of which POS classes a given word belongs to

Experimental Setup

- Use supervised classifiers to learn LEs for novel words relative to the ERG lexical hierarchy
 - ★ **feature vectors** from a given secondary LR
 - ★ **class labels** from seed data in the ERG lexicon
 - ★ evaluate by 10-fold stratified cross-validation
- Learn a binary classifier for each lexical type (110 binary classifiers for each LR type, with default backoff)
- Learn classifiers using the IB1 k -NN (TiMBL 5.0)

Secondary Language Resources

- Use a range of LRs of varying availability:

<i>Secondary LR type</i>	<i>Preprocessor(s)</i>
Word list***	—
Morphological lexicon*	—
Raw text corpus***	POS tagger** Chunk parser* Dependency parser*
WordNet-style ontology*	—

Predicted availability: * = low; ** = medium; *** = high.

Morphology-based DLA

- Extract features from:
 1. (lemmatised) word lists
 2. morphological lexicon (CATVAR)

Morphology-based DLA: Word Lists

- Convert each lexeme into its component character n -grams ($n \in [1, 6]$)

admission = $a \times 1, ad \times 1, \dots, admiss \times 1, \dots, s \times 2, \dots, sion \times 1,$
 $\dots, n \times 1$

Morphology-based DLA: Derivational

- Identify all sister lexemes from word cluster in CATVAR, and determine if there is an affix-based correspondence for each:

Lexeme	POS	Stem	Conversion rule
admission	N	(admiss)	
admissive	AJ	(admiss)	N -on\$ → AJ +ve\$
admissible	AJ	(admiss)	N -on\$ → AJ +ble\$
admissibility	N	(admiss)	N -on\$ → N +bility\$

Syntax-based DLA

- Extract features from corpus data, based on:
 1. POS tagger output
 2. full text chunk parser output
 3. dependency parser output
- Use range of corpora
 1. Brown corpus (~460K tokens)
 2. WSJ corpus (~1.2M tokens)
 3. British National Corpus (~98M tokens)

Syntax-based DLA: POS Tagging

- Tag and lemmatise raw text corpus, and generate the following features for each word w :

<i>Feature type</i>	<i>Positions</i>	<i>Total</i>
POS tag	$(-4, -3, -2, -1, 0, 1, 2, 3, 4)$	9
Word	$(-4, -3, -2, -1, 1, 2, 3, 4)$	8
POS bi-tag	$((-4, -1), (-4, 0), (-3, -2), \dots$ $\dots, (0, 4), (1, 2), (1, 3), (1, 4), (2, 3))$	16
Bi-word	$((-3, -2), (-3, -1), \dots, (1, 3), (2, 3))$	6

Syntax-based DLA: Chunk Parsing

- Chunk parse and lemmatise raw text corpus, and generate the following features for each word w :

<i>Feature type</i>	<i>Positions/description</i>	<i>Total</i>
Modifier _{head}	Chunk head when w modifier	1
Modifier _{chunk}	Chunk type when w modifier	1
Modifiee _{word}	Modifier when w head	1
Modifiee _{POS}	POS tag of modifier when w head	1
Modifiee _{word+POS}	Word + POS tag of modifier	1
Chunk	$(-4, -3, -2, -1, 0, 1, 2, 3, 4)$	9
Chunk head	$(-3, -2, -1, 1, 2, 3)$	6
Bi-chunk	$((-2, -1), (-2, 0), \dots, (0, 2), (1, 2))$	6
... and POS tag and Word		13

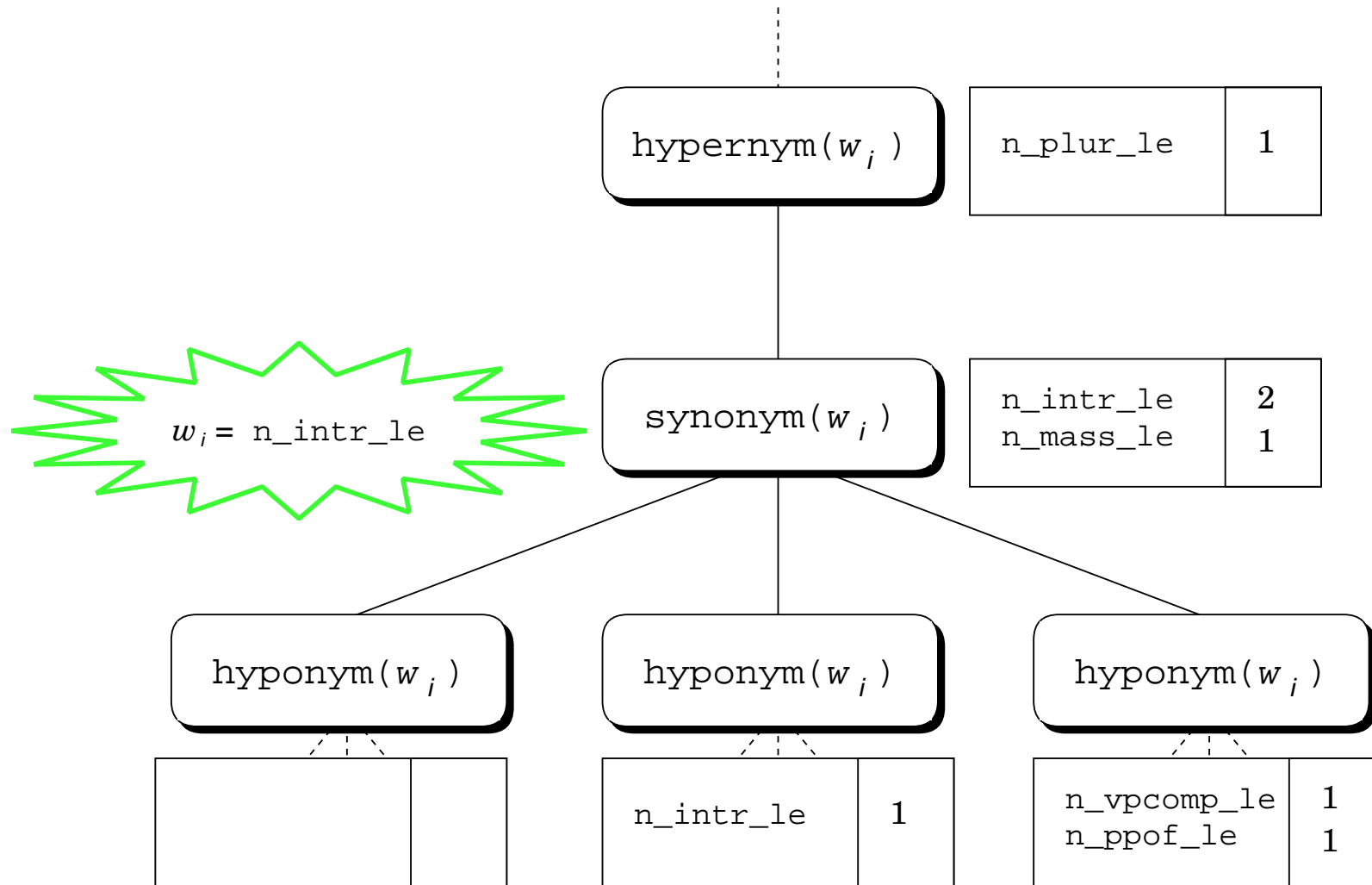
Syntax-based DLA: Dependency Parsing

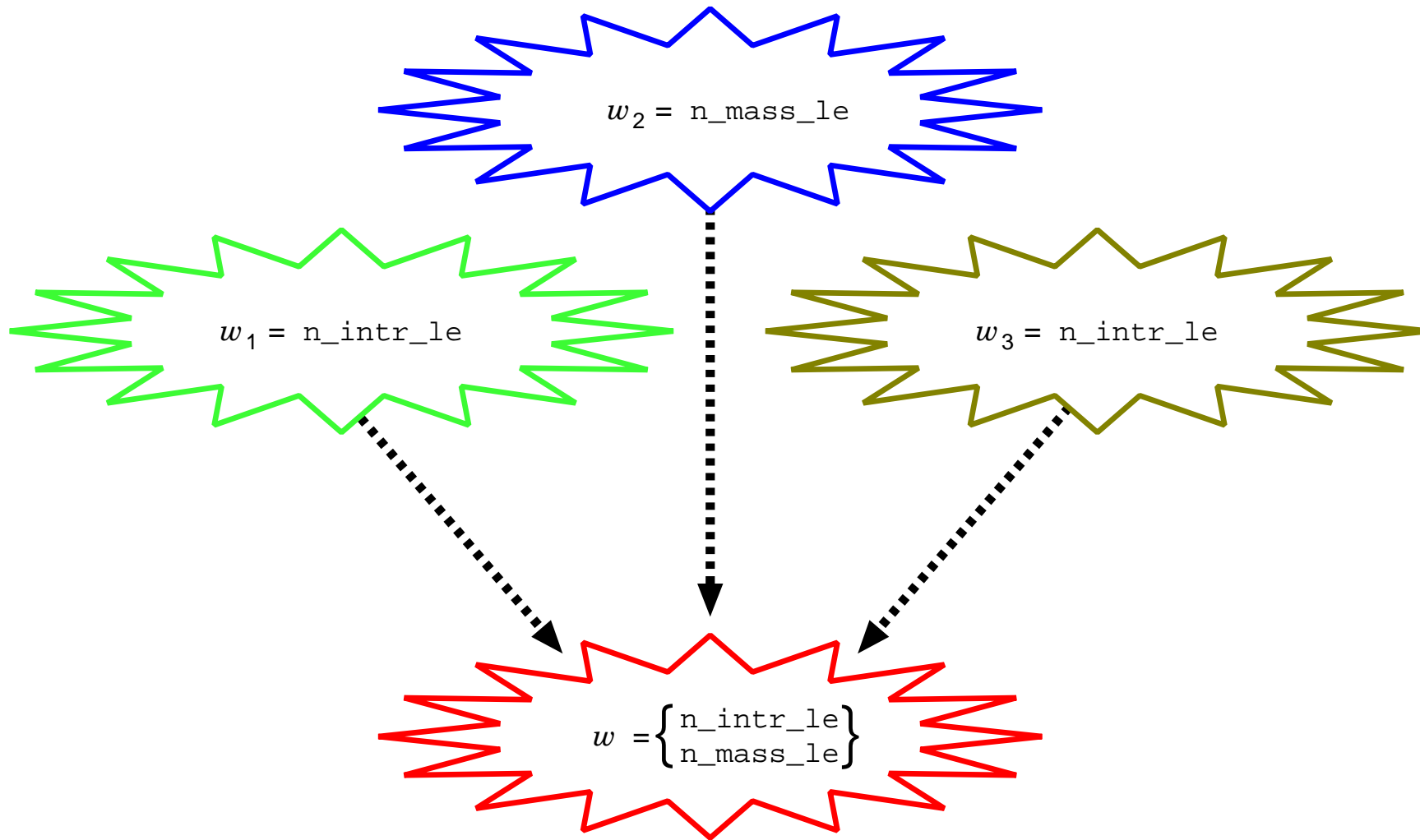
- Dependency parse raw text corpus, and generate the following features for each word w :

<i>Feature type</i>	<i>Positions/description</i>	<i>Total</i>
POS tag	$(-2, -1, 0, 1, 2)$	5
Word	$(-2, -1, 1, 2)$	4
Conj _{word}	Word w coordinates with	1
Conj _{POS}	POS of word w coordinates with	1
Head	Head word when w modifier in rel_i	14
Modifier	Modifier when w head of rel_i	14

Ontology-based DLA

- For each word sense w_i of word w in WordNet 2.0:
 - ★ identify synonym, hypernym and hyponym training words
 - ★ take majority vote over all lexical types of near-neighbour training words
- Take union of lexical types for all w_i





Evaluation

- Evaluate each method via cross-validation over open-class ERG LEs
- **Type precision** = % correct LEs
- **Type recall** = % correctly predicted gold-standard LEs
- **Type F-score** = harmonic mean of precision and recall
- **Token accuracy** = type precision, weighted according to actual token occurrence in Redwoods treebank

Summary of Overall Results

- Syntax-based DLA most successful in terms of overall type F-score
 - ★ chunker-based pre-processing marginally better than tagger or dependency parser
- Ontology-based DLA most successful in terms of overall token accuracy
 - ★ result of overgeneration (low type precision, high type recall)

Results for Individual Word Classes

- Character n -grams effective when learning adjectival lexical items
- Syntax-based methods best-performing method for nouns, verbs and adverbs
- Ontology-based DLA generates high token accuracy over nouns and verbs
- No single method superior to others: “resources for courses”

Impact of Corpus Size on Results

- Increases in corpus size (0.5M–98M tokens) lead to modest gains in type F-score
- Expectation that extra corpus data would boost token accuracy only held true for adverbs (drop in token accuracy for nouns!)
- Small corpus is enough vs. big corpus isn't big enough?

Conclusion

- Morphological, syntactic and ontological DLA methods proposed and evaluated over precision grammar
- Surprising variations in the results for the different methods, with no single standout method

Learning Syntactic Patterns for Automatic Hypernym Discovery

(Snow *et al.* 2005)

Outline

- Much of the targeted DLA research relies on hand-crafted patterns, and makes assumptions about a single lexeme corresponding to a single word class
- Ideally, we want to break down these two assumptions, in:
 - ★ dynamically “learning” patterns;
 - ★ dynamically determining the relative “polysemy” of a given word
- Experiment in the context of learning hypernym (IS-A) relations

Examples of Hand-crafted Patterns

- Hypernym extraction (Hearst 1992):

X and/or other Y

Y such as/including X

- Telic role extraction (Yamada et al. 2007):

N (be | ϕ) (worth | deserving | meriting) (V[+ing] | V[+nom])

N BE worthy of V[+nom]

N (deserves | merits) V[+nom]

Adverb-V[+en] N

Adverb V[+en] N

N BE Adverb-V[ed]

Approach

- Training:
 1. Identify and parse all sentences a given training noun pair occurs in
 2. Extract features from the parse trees
 3. Train a classifier over these features

- Test:

For a given novel pair of nouns, similarly parse all sentences they co-occur in, and use them to classify the noun pair

Parser and Feature Extraction

- Use MINIPAR and (nested) dependency pairs as the basis of feature extraction, e.g.

... such authors as Herrick and Shakespeare



(authors, -N:pre:PreDet, such)

(authors, -N:mod:Prep, as)

(as, -Prep:pcomp-n:N, Herrick)

(as, -Prep:pcomp-n:N, Shakespeare)

(Herrick, -N:punc:U, and)

- Extract out the shortest path connecting the given pair of nouns (incl. satellites), and replace the nouns with variables:

`_Y, N: pcomp-n: Prep, as`

`as, Prep: mod: N, (such, PreDet: pre: N, _X)`

- Prune the space of patterns (to $\approx 70,000$) by excluding those which occurred for 4 or less noun pairs

Experimental Data

- Generate positive and negative instances from WordNet 2.0 for nouns x and y :

$$\text{hype}(x, y) = \begin{cases} 1 & \text{if } \exists x_i, y_j : H_{x_i, y_j} \\ 0 & \text{otherwise} \end{cases}$$

where H_{x_i, y_j} signifies that x_i is an ancestor of y_j

- Bias data such that positive : negative = 1 : 50 (based on hand-coded set of random noun pairs)

Evaluation

- Train classifiers using multinomial naive Bayes, logistic regression, etc.
- Evaluate using 10-fold cross validation and F-score
- Two tasks:
 1. hypernym-only classifier
 2. hypernym classification using coordinate terms:
$$\text{hype}(x, y) \wedge \text{coord}(y, z) \rightarrow \text{hype}(x, z)$$

Results

- Hand-crafted patterns automatically “discovered” by method as being effective, but even better is the accumulation of lots of “OK” (positive and negative) patterns
- Results vastly greater than hand-crafted methods (!)
- Coordinate terms improve hypernym performance

Multilingual Deep Lexical Acquisition for HPSGs via Supertagging

(Blunsom and Baldwin 2006)

Supertagging

- Supertagging = POS tagging with a very fine-grained tagset

*How*_[WRB] *does*_[VBZ] *that*_[DT] *sound*_[VB] *for*_[IN] *you*_[PRP] ?_[.]



*How*_[adj-wh-le] *does*_[va-does-le] *that*_[n--pr-dei-sg-le] *sound*_[v-pp-pp-seq-le]
*for*_[p-le] *you*_[n--pr-you-le] ?_[.]

Applications of Supertagging

- Deep lexical acquisition
- Means of pruning parser search space (cf. Bangalore and Joshi (1999); Clark and Curran (2004))
- Source of linguistic features (e.g. for word alignment)

Vital Statistics of Target DLRs

	ERG	JACY
GRAMMAR		
Language	English	Japanese
Lexemes	16,498	41,559
Lexical items	26,297	47,997
Lexical types	915	484
Strictly continuous MWEs	2,581	422
Optionally discontinuous MWEs	699	0
Average lexical items per lexeme	1.59	1.16
TREEBANK		
Training sentences	20,000	40,000
Training words	215,015	393,668
Test sentences	1,013	1,095
Test words	10,781	10,669

Vital Statistics of Target DLRs

	ERG	JACY
GRAMMAR		
Language	English	Japanese
Lexemes	16,498	41,559
Lexical items	26,297	47,997
Lexical types	915	484
Strictly continuous MWEs	2,581	422
Optionally discontinuous MWEs	699	0
Average lexical items per lexeme	1.59	1.16
TREEBANK		
Training sentences	20,000	40,000
Training words	215,015	393,668
Test sentences	1,013	1,095
Test words	10,781	10,669

Supertagging: A Shopping List

- We desire a method that:
 - ★ works across different languages with a minimum of fuss
 - ★ can be trained directly from treebank data
 - ★ scales to a large tagset (100s of tags)
 - ★ achieves state-of-the-art accuracy
 - ★ is probabilistic

Proposed Supertagger Model

- Pseudo-likelihood CRF model, where $p_{\Lambda}(\mathbf{a}|\mathbf{s})$ is approximated by p_{Λ}^{PL} :

$$p_{\Lambda}^{PL}(\mathbf{a}|\mathbf{s}) = \prod_t \frac{\exp(U_{\Lambda}^{PL}(\mathbf{a}_t, \mathbf{s}, t))}{\sum_l \exp(U_{\Lambda}^{PL}(l, \mathbf{s}, t))}$$

$$U_{\Lambda}^{PL}(i, \mathbf{s}, t) = \sum_k \lambda_k (h_k(t, \hat{a}_{t-1}, i, \mathbf{s}) + h_k(t, i, \hat{a}_{t+1}, \mathbf{s}))$$

- Smooth with a zero-mean Gaussian prior
- Calculate the most probable labelling \mathbf{a}^* for a test sentence via Viterbi as:

$$\mathbf{a}^* = \arg \max_{\mathbf{a}} p_{\Lambda}(\mathbf{a}|\mathbf{s})$$

Features

- Supertagging based on a combination of **word context** and (existence-based) **lexical** features
- Lexical features based on n -gram prefixes & suffixes, and basic character sets in the given language
 - ★ English = 5 character sets (upper case, lower case, numbers, punctuation and hyphens)
 - ★ Japanese = 6 character sets (Roman letters, hiragana, katakana, kanji, (Arabic) numerals and punctuation)

Feature Types

FEATURE	DESCRIPTION
WORD CONTEXT FEATURES	
$lexeme(\mathbf{s}_t) = x \ \& \ \mathbf{a}_t = l$	lexeme + label
$\mathbf{s}_t = w \ \& \ \mathbf{a}_t = l$	word unigram + label
$\mathbf{s}_{t-1} = w \ \& \ \mathbf{a}_t = l$	previous word unigram + label
$\mathbf{s}_{t+1} = w \ \& \ \mathbf{a}_t = l$	next word unigram + label
$\mathbf{s}_t = w \ \& \ \mathbf{s}_{t-1} = y \ \& \ \mathbf{a}_t = l$	previous word bigram + label
$\mathbf{s}_t = w \ \& \ \mathbf{s}_{t+1} = y \ \& \ \mathbf{a}_t = l$	next word bigram + label
$\mathbf{a}_{t-1} = l \ \& \ \mathbf{a}_t = m$	clique label pair
LEXICAL FEATURES	
$prefix_n(\mathbf{s}_t) \ \& \ \mathbf{a}_t = l$	n -gram prefix + label
$suffix_n(\mathbf{s}_t) = x \ \& \ \mathbf{a}_t = l$	n -gram suffix + label
$contains(\mathbf{s}_t, C_i) \ \& \ \mathbf{a}_t = l$	word contains element of character set C_i + label

Experimental Setup

- Train supertagger over Redwoods (EN) AND Hinoki (JP) treebank data
- Evaluate relative to a held-out set of ~ 1000 sentences
- Main interest in **unknown lexical entries** = LEs not in training data
- Baseline = unigram supertagger
- Benchmark (EN) = FNTBL 1.1

Evaluation Metrics

- **Token accuracy** $[ACC]$ = overall token-level accuracy
- **Unknown token accuracy** $[ACC_U]$ = token-level accuracy over unknown lexemes
- **Type precision** $[PREC]$ = % correct unknown LEs
- **Type recall** $[REC]$ = % unknown gold-standard LEs correctly predicted
- **Type F-score** $[F-SCORE]$

Results: Reflections

- Token accuracy very high (incl. MWEs)
- Type precision of unknown LEs also respectably high (> 0.50), suggesting possibilities of (semi-)automating DLA
- Type recall highly variable (esp. low for EN)
- Remarkably good results given lack of feature engineering
- Hardest lexical types to learn: V and Adj (EN) / Adj + Adv (JP)

Conclusion

- New method for learning new lexical items for HPSG-based precision grammars through supertagging, using a pseudo-likelihood CRF
- State-of-the-art results achieved for English and Japanese with language-independent feature set

OVERALL SUMMARY

What you have (hopefully) Learned

- What are the basic contrasts between methods in DLA?
- What are some of the tasks in DLA?
- What are the basic empirical methods used in DLA?
- How do you design and evaluate DLA tasks?

Open Questions

- What gains do we get from specialist linguistic knowledge? (template development, feature engineering, etc.)
- What gains do we get from different preprocessors?
- How good is good enough for DLA? (precision vs. recall)
- How can we calibrate expectations for a particular task over a given array of resources?

Where to from here?

- More domains
- More languages
- Less annotation
- More tasks
- More extrinsic evaluation of the results of DLA

References

- BALDWIN, TIMOTHY. 2005. Bootstrapping deep lexical resources: Resources for courses. In *Proc. of the ACL-SIGLEX 2005 Workshop on Deep Lexical Acquisition*, 67–76, Ann Arbor, USA.
- BANGALORE, SRINIVAS, and ARAVIND K. JOSHI. 1999. Supertagging: An approach to almost parsing. *Computational Linguistics* 25.237–65.
- BLUNSOM, PHIL, and TIMOTHY BALDWIN. 2006. Multilingual deep lexical acquisition for HPSGs via supertagging. In *Proc. of the 2006 Conference on Empirical Methods in Natural Language Processing (EMNLP 2006)*, 164–71, Sydney, Australia.
- CLARK, STEPHEN, and JAMES R. CURRAN. 2004. The importance of supertagging for wide-coverage CCG parsing. In *Proc. of the 20th International Conference on Computational Linguistics (COLING 2004)*, 282–8, Geneva, Switzerland.
- HEARST, MARTI. 1992. Automatic acquisition of hyponyms from large text corpora. In *Proc. of the 14th International Conference on Computational Linguistics (COLING '92)*, Nantes, France.
- SNOW, RION, DANIEL JURAFSKY, and ANDREW Y. NG. 2005. Learning syntactic patterns for automatic hypernym discovery. In *Advances in Neural Information Processing Systems 17*, 1297–1304, Vancouver, Canada.

YAMADA, ICHIRO, TIMOTHY BALDWIN, HIDEKI SUMIYOSHI, and NOBUYUKI YAGI. 2007. Automatic acquisition of qualia structure from corpus data. *IEICE Transactions on Information and Systems* E90-D.1534–41.