Supertagging in PET

Rebecca Dridan

Universität des Saarlandes

DELPH-IN Summit
Barcelona, July 2009

Outline

What is supertagging?

Supertagging is restricting the parser search space by limiting the number of lexical items, in order to increase parser efficiency.

CCG lexicon entries

price	((S[b]\NP)/PP)/NP
price	(S[b]\NP)/NP
price	(S[dcl]\NP)/NP
price	N
price	N/N
price	NP\NP

HPSG lexicon entry

price_n2 := n_pp_c-obl_le &

$$\begin{bmatrix} \text{STEM} & \langle \text{ “price” } \rangle, \\ \text{SYNSEM} & \begin{bmatrix} \text{LKEYS} & \begin{bmatrix} -\text{COMPKEY} & \text{\_of\_p\_sel\_rel} \\ \text{KEYREL.PRED} & \text{“\_price\_n\_of\_rel”} \end{bmatrix}, \\ \text{PHON} & \begin{bmatrix} \text{ONSET} & \text{con} \end{bmatrix} \end{bmatrix} \end{bmatrix}$$

In CCG, all information is in the supertag (category) and the word form $\longrightarrow$ the supertagger produces the lexical items.

In HPSG, this is only true if the supertag is the lexical entry (**price_n2**) $\longrightarrow$ the supertagger is a filter for the lexicon.

Modifying the grammar

Supertagging is facilitated by the new TOKENS feature of word_or_lexrule, added for chart mapping.

Step 1 Add +STAG feature to the **token** type:

token

$$\begin{bmatrix} +\text{FORM} & \text{string} \\ +\text{CLASS} & \text{token\_class} \\ +\text{TRAIT} & \text{token\_trait} \\ +\text{PRED} & \text{predsort} \\ +\text{CARG} & \text{string} \\ +\text{ID} & \text{*diff-list*} \\ +\text{FROM} & \text{string} \\ +\text{TO} & \text{string} \\ +\text{TNT} & \text{tnt}\begin{bmatrix} +\text{TAGS} & \text{*list*} \\ +\text{PRBS} & \text{*list*} \end{bmatrix} \end{bmatrix} + \begin{bmatrix} +\text{STAG} & \text{stag}\begin{bmatrix} +\text{STAGS} & \text{*list*} \\ +\text{SPRBS} & \text{*list*} \end{bmatrix} \end{bmatrix}$$

Modifying the grammar

Step 2 For each lexical type that may be filtered, specify the +STAG feature.

for example:

v_np_le *:+*

$$\left[\text{TOKENS.LIST} \quad \left\langle \left[+\text{STAG.+STAGS} \quad \left\langle \text{“v\_np\_le”}, \ldots \right\rangle\right]\right\rangle\right]$$

This is the first point of supertagger control: if a particular lexical type should never be filtered by the supertagger, leave the +STAG feature unspecified.

Note: the value is of type string and may be set to any string value.

Modifying the grammar

Step 3 Modify token mapping rules

- Make sure +STAG information is copied through where appropriate.
- Expand single token with multiple supertags into multiple tokens each with one supertag?
- Delete +STAG according the tag probabilities?
- Deal with TnT POS tag appropriately.
  - copy?
  - delete?
  - modify?

This is the second point of control that dictates how the supertags will be used in parsing.

How does it work?

lexical entries

$$\begin{bmatrix} \text{STEM} & \langle \text{“ration”} \rangle \\ \text{SYNSEM} & \begin{bmatrix} \text{KEYREL.PRED} & \text{“\_ration\_v\_1\_rel”} \\ \text{PHON.ONSET} & \text{con} \end{bmatrix} \\ \text{TOKENS} & \langle \boxed{1} \begin{bmatrix} \text{+STAG.+STAGS} & \langle \text{“vp\_np\_le”} \rangle \end{bmatrix} \rangle \end{bmatrix} \begin{bmatrix} \text{STEM} & \langle \text{“ration”} \rangle \\ \text{SYNSEM} & \begin{bmatrix} \text{KEYREL.PRED} & \text{“\_ration\_n\_1\_rel”} \\ \text{PHON.ONSET} & \text{con} \end{bmatrix} \\ \text{TOKENS} & \langle \boxed{1} \begin{bmatrix} \text{+STAG.+STAGS} & \langle \text{“n\_-\_c\_le”} \rangle \end{bmatrix} \rangle \end{bmatrix}$$

are unified with

$$\boxed{1} \begin{bmatrix} \text{+FORM} & \text{“rations”} \\ \text{+FROM} & \text{“11”} \\ \text{+TO} & \text{“18”} \\ \text{+TNT.+TAGS} & \langle \text{“NNS”} \rangle \\ \text{+TNT.+PRBS} & \langle \text{“0.97305”} \rangle \\ \text{+STAG.+STAGS} & \langle \text{“n\_-\_c\_le”} \rangle \\ \text{+STAG.+SPRBS} & \langle \text{“0.60377”} \rangle \end{bmatrix}$$

the input token

How does it work?

INVALID

$$\begin{bmatrix} \text{STEM} & \langle\ \text{“ration”}\ \rangle \\ \text{SYNSEM} & \begin{bmatrix} \text{KEYREL.PRED} & \text{“\_ration\_n\_1\_rel”} \\ \text{PHON.ONSET} & \text{con} \end{bmatrix} \\ \text{TOKENS} & \langle \begin{bmatrix} +\text{FORM} & \text{“rations”} \\ +\text{FROM} & \text{“11”} \\ +\text{TO} & \text{“18”} \\ +\text{TNT.+TAGS} & \langle\ \text{“NNS”}\ \rangle \\ +\text{TNT.+PRBS} & \langle\ \text{“0.97305”}\ \rangle \\ +\text{STAG.+STAGS} & \langle\ \text{“n\_-\_c\_le”}\ \rangle \\ +\text{STAG.+SPRBS} & \langle\ \text{“0.60377”}\ \rangle \end{bmatrix} \rangle \end{bmatrix}$$

Configuration options

Handling lexical gaps

When the supertagger filters out every lexical entry, do we

- **fail to parse?**
  Supertagging is often a trade-off between robustness and efficiency. An option when the supertagger is very good, or efficiency has much higher priority than robustness.
- **create a generic item from the supertag?**
  Similar effect to CCG supertagging.
- **use the TnT POS tag to generate a generic lexical item?**
  This is the standard method of handling any lexical gap. Should compatibility between supertag and POS tag be enforced?

Configuration options

Assigning supertags

Which input tokens should be restricted, and how?

- **single tag all tokens**
  Simple, but brutal. Best sentence tagging accuracy approximately 50% $\rightarrow$ low coverage, high efficiency.
- **multitag all tokens**
  Optimal β value must be determined. Sentence tagging accuracy of 91% possible assigning an average of 2.3 tags per token.
- **tag only some tokens, according to tagger confidence**
  Optimal confidence threshold must be determined. Sentence tagging accuracy of 98% possible when single tagging half of the tokens.

What are the supertags?

In CCG, supertags are the lexical entries.
But

n_pp_c-obl_le &

$$\begin{bmatrix} \text{STEM} & \langle \text{ “price” } \rangle, \\ \text{SYNSEM} & \begin{bmatrix} \text{LKEYS} & \begin{bmatrix} -\text{COMPKEY} & \text{\_of\_p\_sel\_rel} \\ \text{KEYREL.PRED} & \text{“\_price\_n\_of\_rel”} \end{bmatrix}, \\ \text{PHON} & \begin{bmatrix} \text{ONSET} & \text{con} \end{bmatrix} \end{bmatrix} \end{bmatrix}$$

is not a good supertag.

We can use **letype**: n_pp_c-obl_le

But also:

subcat: n_pp
pos: n

What are the supertags?

	Coverage	Precision	F-score	Lexical Items	Lexical Ambiguity	Time (sec.)
Baseline	1.00	0.778	0.777	106.04	7.57	0.77

Table: Parser performance over *jhpstg-test*

What are the supertags?

	Coverage	Precision	F-score	Lexical Items	Lexical Ambiguity	Time (sec.)
Baseline	1.00	0.778	0.777	106.04	7.57	0.77
Gold **letype**	1.00	0.808	0.808	20.05	1.43	0.11
Gold **subcat**	1.00	0.805	0.804	28.46	2.03	0.13
Gold **pos**	1.00	0.800	0.795	55.56	3.97	0.26

Table: Parser performance over *jhpstg-test*

- Upper bound speed increase seven-fold
- Gold **subcat** is almost as fast as gold **letype**

What are the supertags?

	Coverage	Precision	F-score	Lexical Items	Lexical Ambiguity	Time (sec.)
Baseline	1.00	0.778	0.777	106.04	7.57	0.77
Gold **letype**	1.00	0.808	0.808	20.05	1.43	0.11
Single	0.67	0.765	0.550	19.36	1.38	0.09
Beta 0.001	0.71	0.767	0.589	33.16	2.37	0.11
Thresh 0.99	1.00	0.777	0.778	83.07	5.93	0.53

Table: Parser performance over *jhpstg-test*

What are the supertags?

	Coverage	Precision	F-score	Lexical Items	Lexical Ambiguity	Time (sec.)
Baseline	1.00	0.778	0.777	106.04	7.57	0.77
Gold **subcat**	1.00	0.805	0.804	28.46	2.03	0.13
Single	0.78	0.770	0.639	27.78	1.98	0.12
Beta 0.001	0.82	0.762	0.656	46.75	3.34	0.11
Thresh 0.99	1.00	0.777	0.777	84.73	6.05	0.56

Table: Parser performance over *jhpstg-test*

Thank You!