Weakly supervised supertagging with grammar-informed initialization

Jason Baldridge
Department of Linguistics
The University of Texas at Austin
jbaldrid@mail.utexas.edu

Abstract

Much previous work has investigated weak supervision with HMMs and tag dictionaries for part-of-speech tagging, but there have been no similar investigations for the harder problem of supertagging. Here, I show that weak supervision for supertagging does work, but that it is subject to severe performance degradation when the tag dictionary is highly ambiguous. I show that lexical category complexity and information about how supertags may combine syntactically can be used to initialize the transition distributions of a first-order Hidden Markov Model for weakly supervised learning. This initialization proves more effective than starting with uniform transitions, especially when the tag dictionary is highly ambiguous.

1 Introduction

Supertagging involves assigning words lexical entries based on a lexicalized grammatical theory, such as Combinatory Categorial Grammar (CCG) (Steedman, 2000) Tree-adjoining Grammar (Joshi, 1988), or Head-driven Phrase Structure Grammar (Pollard and Sag, 1994). Supertag sets are larger than part-of-speech (POS) tag sets and their elements are generally far more articulated. For example, the English verb *join* has the POS VB and the CCG category $((S_b\backslash NP)/PP)/NP$ in CCGbank (Hockenmaier and Steedman, 2007). This category indicates that *join* requires a noun phrase to its left, another to its right, and a prepositional phrase to the right of that.

Supertags convey such detailed syntactic subcategorization information that supertag disambiguation is referred to as *almost parsing* (Bangalore and Joshi, 1999). Standard sequence prediction models are highly effective for supertagging, including Hidden Markov Models (Bangalore and Joshi, 1999; Nielsen, 2002), Maximum Entropy Markov Models (Clark, 2002; Hockenmaier et al., 2004; Clark and Curran, 2007), and Conditional Random Fields (Blunsom and Baldwin, 2006). The original motivation for supertags–parse prefiltering for lexicalized grammars–of Bangalore and Joshi (1999) has been realized to good effect: the supertagger of Clark and Curran (2007) provides staged n-best lists of multi-tags that dramatically improve parsing speed and coverage without much loss in accuracy. Espinosa et al. (2008) have shown that hypertagging (predicting the supertag associated with a logical form) can improve both speed and accuracy of wide-coverage sentence realization with CCG. Supertags have gained further relevance as they are increasingly used as features for other tasks, including machine translation (Birch et al., 2007; Hassan et al., 2007).

Supertaggers typically rely on a significant amount of carefully annotated sentences. As with many problems, there is pressing need to find strategies for reducing the amount of supervision required for producing accurate supertaggers, but as yet, no one has explored the use of weak supervision for the task. In particular, there are many dialog systems which rely on hand-crafted lexicons that both provide a starting point for bootstrapping a supertagger and which could benefit greatly from supertag pre-parse filter. For example, the dialog system used by Kruijff et al. (2007) uses a hand-crafted CCG grammar for OpenCCG (White and Baldridge, 2003). It is important to stress that there are many such uses of CCG and related frameworks which do *not* rely on first annotating (even a small number of) sentences in a corpus: these define a lexicon that maps from words to categories (supertags) for a particular domain/application.

This scenario is a natural fit for learning taggers from tag dictionaries using hidden Markov models with Expectation-Maximization (EM). Here, I investigate such weakly supervised learning for supertagging and demonstrate the importance of proper initialization of the tag transition distributions of the HMM. In particular, such initialization can be done using inherent properties of the CCG formalism itself regarding how categories[1] may combine. Informed initialization should help with supertagging for two reasons. First, categories have structure–lacking in POS tags–waiting to be exploited. For example, it is far more likely *a priori* to see the category sequence $(\mathsf{S\backslash NP})/\mathsf{NP}$ $\mathsf{NP/N}$ than the sequence $\mathsf{S/S}$ $\mathsf{NP\backslash NP}$. Given the categories for a word, this information can be used to influence our expectations about categories for adjacent words. Second, this kind of information truly matters for the task: a key aspect of supertagging that differentiates it from POS tagging is that the contextual information is much more important for the former. Lexical probabilities handle most of the ambiguity for POS tagging, but supertags are *inherently* about context and, furthermore, lexical ambiguity is much greater for supertagging, making lexical probabilities less effective.

I start by defining a distribution over lexical categories and then use this distribution as part of creating a CCG-informed transition distribution that appropriately breaks the symmetry of uniform HMM initialization. After describing how these components are included in the HMM, I describe experiments with CCGbank varying the ambiguity of the lexicon provided. I show that using knowledge about the formalism consistently improves performance, and is especially important as categorial ambiguity increases.

2 Lexical category distribution

The categories of CCG are an inductively defined set containing elements that are either atomic elements or (curried) functions specifying the canonical linear direction in which they seek their arguments. Some example entries from CCGbank are:

the := $\mathsf{NP_{nb}/N}$
of := $(\mathsf{NP\backslash NP})/\mathsf{NP}$
of := $((\mathsf{S\backslash NP})\backslash(\mathsf{S\backslash NP})/\mathsf{NP}$
were := $(\mathsf{S_{dcl}\backslash NP})/(\mathsf{S_{pss}\backslash NP})$
buy := $(\mathsf{S_{dcl}\backslash NP})/\mathsf{NP}$
buy := $((((\mathsf{S_b\backslash NP})/\mathsf{PP})/\mathsf{PP})/(\mathsf{S_{adj}\backslash NP}))/\mathsf{NP}$

Words can be associated with multiple categories; the distribution over these categories is typically quite skewed. For example, the first entry for *buy* occurs 33 times in CCGbank, compared with just once for the second. That the simpler category is more prevalent is unsurprising: a general strategy when creating CCG lexicons is to use simpler categories whenever possible. This points to the possibility of defining distributions over CCG lexicons based on measures of the complexity of categories. I use a simple distribution here: given a lexicon $\mathcal{L}$, the probability of a category i is inversely proportional to its complexity:

$$\Lambda_i = \frac{\frac{1}{complexity(c_i)}}{\sum_{j\in\mathcal{L}} \frac{1}{complexity(c_j)}} \qquad (1)$$

Here, a very simple complexity measure is assumed: the number of subcategories (tokens) contained in a category.[2] For example, $((\mathsf{S\backslash NP})\backslash(\mathsf{S\backslash NP})/\mathsf{NP}$ contains 9: S (twice), NP (thrice), $\mathsf{S\backslash NP}$ (twice), $(\mathsf{S\backslash NP})\backslash(\mathsf{S\backslash NP})$, and $((\mathsf{S\backslash NP})\backslash(\mathsf{S\backslash NP})/\mathsf{NP}$.

The tag transition distribution defined in the next section uses Λ to bias transitions toward simpler categories, e.g., preferring the first category for *buy* over the second. Performance when using Λ is compared to using a uniform distribution.

Other distributions could be given, e.g., one which gives more mass to adjunct categories such as $(\mathsf{S\backslash NP})\backslash(\mathsf{S\backslash NP})$ than to ones which are otherwise similar but do not display such symmetry, like $(\mathsf{S/NP})\backslash(\mathsf{NP\backslash S})$. However, the most important thing for present purposes is that simpler categories are more likely than more complex ones.

This distribution imposes no internal structure on the likelihood of a lexicon. As far as Λ is concerned, lexicons can as well have the category $(\mathsf{S\backslash NP})\backslash\mathsf{NP}$ for transitive verbs and $((\mathsf{S/NP})/\mathsf{NP})/\mathsf{NP}$ for ditransitive verbs, even though this is a highly unlikely pattern since we

[1]For the rest of the paper, I will refer to *categories* rather *supertags*, but will still refer to the task as supertagging.

[2]This worked better than using category arity or number of unique subcategory types.

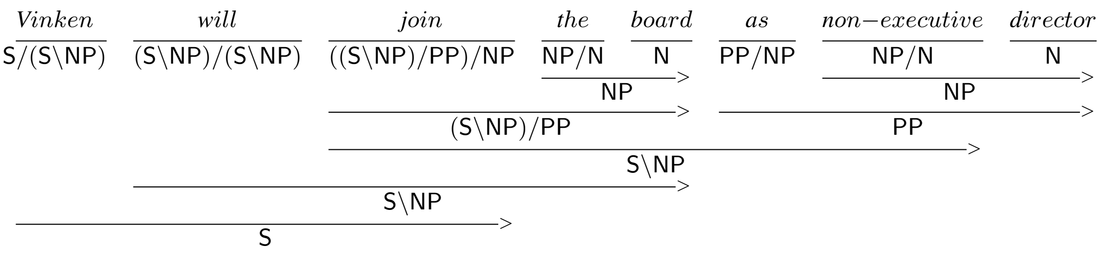

Figure 1: Normal form CCG derivation, using only application rules.

would expect both types of verbs to seek their arguments in the same direction. Languages also tend to prefer lexicons with one or the other slash direction predominating (Villavicencio, 2002). In the future, it would be interesting to consider Bayesian approaches that could encode more complex structure and assign priors over distributions over lexicons, building on these observations.

An aspect of CCGbank that relevant for Λ_i is that some categories actually are not true categories. For example, many punctuation "categories" are given as LRB, ., :, etc. In most grammars, the category of '.' is usually assumed to be S\S. The grammatical behavior of such pseudo-categories is handled via special rules in the parsers of Hockenmaier and Steedman (2007) and Clark and Curran (2007). I relabeled three of these: , to NP\NP, . to S\S and ; to (S\S)/S. A single best change was not clear for others such as LRB and :, so they were left as is.

3 Category transition distribution

CCG analyses of sentences are built up from lexical categories combining to form derived categories, until an entire sentence is reduced to a single derived category with corresponding dependencies. One of CCG's most interesting linguistic properties is it allows alternative constituents. Consider the derivations in Figures 1 and 2, which show a normal form derivation (Eisner, 1996) and fully incremental derivation, respectively. Both produce the same dependencies, guaranteed by the semantic consistency of CCG's rules (Steedman, 2000). This property of CCG of supporting multiple derivations of the same analysis has been termed *spurious ambiguity*. However, the extra constituents are anything but spurious: they are implicated in a range of CCG (along with other forms of categorial grammar) linguistic analyses, including coordination, long-distance extraction, intonation, and incremental processing.

This all boils down to associativity: just as $(1 + (4 + 2)) = ((1 + 4) + 2) = 7$, CCG ensures that (*Ed*·(*saw*·*Ted*)) = ((*Ed*·*saw*)·*Ted*) = S Such multiple derivations arise when adjacent categories can combine through either application or composition. Thus, we would expect that the lexical categories needed to analyze an entire sentence will more often than not be able to combine with their immediate neighbors. For example, six of seven pairs of adjacent lexical categories in the sentence in Figure 1 can combine. Only N PP/NP of *board as* cannot.[3]

This observation can be used in different ways by different models for CCG supertagging. For example, discriminative tagging models could include features that capture whether or not the current supertag can combine with the previous one and possibly via which CCG rule. Here, I show how it can be used to provide a non-uniform starting point for the transition distributions $\theta_{j|i}$ in a first-order Hidden Markov Model. This is similar to how Grenager et al. (2005) use diagonal initialization in an HMM for field segmentation to encourage the model to remain in the same state (and thus predict the same label for adjacent words). For CCG supertagging, the initialization should discourage diagonalization and establish a preference for some transitions over others.

There are many ways to define such a starting point. The simplest would be to reserve a small part of the mass spread uniformly over category pairs which cannot combine and then spread the rest of the mass uniformly over those which can. However, we can provide a more refined distribution, $\Psi_{j|i}$, by incorporating the lexical category distribution Λ_i defined in the previous section to weight these transitions according to this further information. In a similar manner to Grenager et al. (2005), I define Ψ as follows:

[3] I make the standard assumption that type-raising is performed in the lexicon, so the possibility of combining these through type-raising plus composition is not available.

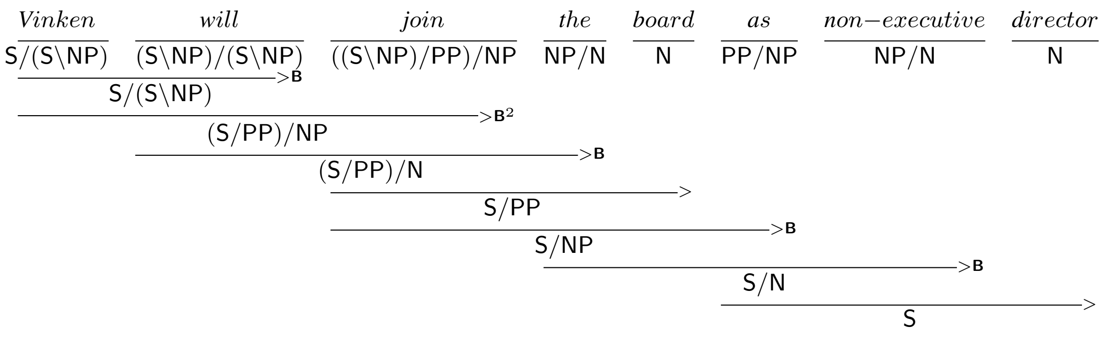

Figure 2: Incremental CCG derivation, using both application and composition (**B**) rules.

$$\Psi_{j|i} = (1-\sigma)\Lambda_j + \sigma \times \kappa(i,j) \times \frac{\Lambda_j}{\sum_{k \in \mathcal{L}|\kappa(i,k)} \Lambda_k} \qquad (2)$$

where $\kappa(i,j)$ is an indicator function that returns 1 if categories c_i and c_j can combine when c_i immediately precedes c_j, σ is a global parameter that specifying the total probability of transitions that are combinable from i. Each j receives a proportion of σ according to its lexical prior probability over the sum of the lexical prior probabilities for all categories that combine with i. For the experiments in this paper, σ was set to .95. For the models referred to as ΨU and ΨU-EM in section 5, the uniform lexical probability $1/|\mathcal{C}|$ is used for Λ_i.

For $\kappa(i,j)$, I use the standard rules assumed for CCGbank parsers: forward and backward application (>, <), order-preserving composition (>**B**, <**B**), and backward crossed composition (<$\mathbf{B}_\times$) for S-rooted categories. Thus, κ(NP, S\NP)=1, κ(S/NP, NP/N)=1, κ((S\NP)/NP, (S\NP)\(S\NP))=1 and κ(S/NP, NP\NP)=0. For application, leftward and rightward arguments are handled separately by assuming that it would be possible to consume all preceding arguments of the first category and all following arguments of the second. So, κ((S/NP)\S, NP/N)=1 and κ(NP, (S\NP)/NP)=1. Unification on categories is standard (so κ(NP[nb], S\NP)=1), except that N unifies with NP only when N is the argument: κ(N, S\NP)=1, but κ(NP/N, NP)=0. This is to deal with the fact that CCGbank represents many words with N (e.g., *Mr.*|N/N *Vinken*|N *is*|(S[dcl]\NP)/NP) and assumes that a parser will include the unary type changing rule N→NP.

The HMM also has initial and final probabilities; distributions can be defined based on which categories are likely to start or end a sentence. For this, I assume only that categories which seek arguments to the left (e.g., S\NP) are less likely at the beginning of a sentence and those which seek rightward arguments are less likely at the end. The initializations for these are defined similarly to the transition distribution, substituting functions $noLeftArgs(i)$ and $noRightArgs(i)$ for $\kappa(i,j)$.

4 Model

A first-order Hidden Markov Model (bitag HMM) is used for bootstrapping a supertagger from a lexicon. See Rabiner (1989) for an extensive introduction to and discussion of HMMs. There are several reasons why this is an attractive tagging model here. First, though extra context in the form of tritag transition distributions or other techniques can improve supervised POS tagging accuracy, the accuracy of bitag HMMs is not far behind. The goal here is to investigate the *relative* gains of using CCG-based information in weakly supervised HMM learning. Second, the expectation-maximization algorithm for bitag HMMs is efficient and has been shown to be quite effective for acquiring accurate POS taggers given only a lexicon (tag dictionary) and certain favorable conditions (Banko and Moore, 2004). Third, the model's simplicity makes it straightforward to test the idea of CCG-initialization on tag transitions.

Dirichlet priors can be used to bias HMMs toward more skewed distributions (Goldwater and Griffiths, 2007; Johnson, 2007), which is especially useful in the weakly supervised setting considered here. Following Johnson (2007), I use variational Bayes EM (Beal, 2003) during the M-step for the transition distribution:

$$\theta^{l+1}_{j|i} = \frac{f(E[n_{i,j}] + \alpha_i)}{f(E[n_i] + |\mathcal{C}| \times \alpha_i)} \qquad (3)$$

$$f(v) = exp(\psi(v)) \qquad (4)$$

$$\psi(v) = \begin{cases} g(v - \frac{1}{2}) & \text{if } v > 7 \\ \psi(v+1) - \frac{1}{v} & \text{o.w.} \end{cases} \quad (5)$$

$$g(x) \approx log(x) + \frac{1}{24x^2} - \frac{7}{960x^4} + \frac{31}{8064x^6} - \frac{127}{30720x^8} \quad (6)$$

where $\mathcal{V}$ is the set of word types, ϕ is the *digamma* function (which is approximated by g), and α_i is the hyperparameter of the Dirichlet priors. In all experiments, the α_i parameters were set symmetrically to .005.

For experiments using the transition prior $\Psi_{j|i}$, the initial expectations of the model were set as $E[n_{i,j}] = |\mathcal{E}_i| \times \Psi_{j|i}$ and $E[n_i] = |\mathcal{E}_i|$, where $\mathcal{E}_i$ is the set of emissions for category c_i. The uniform probability $\frac{1}{|\mathcal{C}|}$ was used in place of $\Psi_{j|i}$ for standard HMM initialization.

The emission distributions use standard EM expectations with more mass reserved for unknowns for tags with more emissions as follows:[4]

$$\phi^{l+1}_{k|i} = \frac{E[n_{i,k}] + |\mathcal{E}_i| \times \frac{1}{|\mathcal{V}|}}{E[n_i] + |\mathcal{E}_i|} \quad (7)$$

The Viterbi algorithm is used for decoding.

5 Experiments

CCGbank (Hockenmaier and Steedman, 2007) is a translation of phrase structure analyses of the Penn Treebank into CCG analyses. Here, I consider only the lexical category annotations and ignore derivations. The standard split used for weakly supervised HMM tagging experiments (Banko and Moore, 2004; Wang and Schuurmans, 2005) is used: sections 0-18 for training (*train*), 19-21 for development (*dev*), and 22-24 for testing (*test*). All parameters and models were developed using *dev*. The test set was used only once to obtain the performance figures reported here.

Counts for word types, word tokens and sentences for each data set are given in Table 1. In *train*, there are 1241 distinct categories, the ambiguity per word *type* is 1.69, and the maximum number of categories for a single word type is 126. This is much greater than for POS tags in CCGbank, for which there are 48 POS tags with an average ambiguity of 1.17 per word and a maximum of 7 tags in *train*.[5]

Dataset	Types	Tokens	Sentences
train	43063	893k	38,015
dev	14961	128k	5484
test	13898	127k	5435

Table 1: Basic statistics for the datasets.

The set of supertags was not reduced: any category found in the data used to initialize a lexicon was considered. This is one of the advantages of the HMM over using discriminative models, where typically only supertags seen at least 10 times in the training material are utilized for efficiency (Clark and Curran, 2007). Ignoring some supertags makes sense when building supervised supertaggers for pre-parse filtering, but not for learning from lexicons, where we cannot assume we have such frequencies.

For supervised training with the HMM on *train*, the performance is 87.6%. This compares to 91.4% for the C&C supertagger. The accuracy of the HMM, though quite a bit lower than that of C&C, is still quite good, indicating that it is an adequate model for the task. Note also that it uses only the words themselves and does not rely on POS tags. The performance of the C&C tagger was obtained by training the C&C POS tagger on the given dataset and tagging the evaluation material with it. Finally, the HMM trains in just a few seconds as opposed to over an hour.[6]

Five different weakly supervised scenarios are evaluated: (1) standard EM with 50 iterations (EM), (2) Ψ initialization with uniform lexical probabilities w/o EM (ΨU), (3) Ψ with Λ probabilities w/o EM ($\Psi\Lambda$), (4) Ψ with uniform lexical probabilities and 10 EM iterations, and (5) Ψ with Λ and 10 EM iterations.[7] These scenarios compare the effectiveness of standard EM with the use of grammar informed transitions; these in turn are of two varieties – one using a uniform lexical prior or one that is biased in favor of less complex categories according to Λ.

As Banko and Moore (2004) discovered when

[4]I also experimented with a Dirichlet prior on the emissions, but it performed worse. Using a symmetric prior was actually detrimental, while performance within a percent of those achieved with the above update was achieved with Dirichlet hyperparameters set relative to $|\mathcal{E}_i|/|\mathcal{V}|$.

[5]Note that the POS tag information is not used in these experiments, except for by the C&C tagger.

[6]It should be stressed that the goal of this paper is **not** to compete on supervised performance with C&C; instead, this comparison shows that the HMM supervised performance is reasonable and is thus relevant for bootstrapping.

[7]The number of iterations for standard and grammar informed iteration were determined by performance on *dev*.

reimplementing several previous HMMs for POS tagging, the lexicons had been limited to contain only tags occurring above a particular frequency. For POS tagging, this keeps a cleaner lexicon that avoids errors in annotated data (such as *the* tagged as VB) and rare tags (such as *a* tagged as SYM). When learning from a lexicon alone, such elements receive the same weight as their other (correct or more fundamental) tags in initializing the HMM. The problem of rare tags turns out to be very important for weakly supervised CCG supertagging.[8]

To consider the effect of the CCG-based initialization for lexicons with differing ambiguity, I use tag cutoffs that remove any lexical entry containing a category that appears with a particular word less than X% of the time (Banko and Moore, 2004), as well as using no cutoffs at all. Recall that the goal of these experiments is to investigate the relative difference in performance between using the grammar-based initialization or not, given some (possibly hand-crafted) lexicon. Lexicon cutoffs actually constitute a strong source of supervision because they use tag frequencies (which would not be known for a hand-crafted lexicon), so it should be stressed that they are used here only so that this relative performance can be measured for different ambiguity levels.

Table 2 provides accuracy for **ambiguous** words (and not including punctuation) for the five scenarios, varying the cutoff to measure the effect of progressively allowing more lexical ambiguity (and much rarer categories). The number of ambiguous, non-punctuation tokens is 101,167.

The first thing to note is performance given only the lexicon and the ΨU or $\Psi\Lambda$ initialization of the transitions. These correspond to taggers which have only been given the lexicon and have not utilized any data to improve their estimates of the transition and emission probabilities. Interestingly, both do quite well with a clean lexicon: see the columns under ΨU and $\Psi\Lambda$. These indicate that initializing the transitions based on whether categories can combine does indeed appropriately capture key aspects of category transitions. Furthermore, using the lexical category distribution ($\Psi\Lambda$) to create the transition initialization provides a better starting point than the uniform one (ΨU), especially as lexical ambiguity increases.

Cutoff	EM	ΨU	$\Psi\Lambda$	ΨU-EM	$\Psi\Lambda$-EM
.1	77.4	73.1	74.7	80.0	79.6
.05	69.1	70.6	72.5	79.2	79.2
.01	60.2	62.2	65.0	75.4	76.7
.005	52.2	57.8	59.0	72.5	73.8
.001	41.3	45.5	48.2	63.0	67.6
None	33.0	33.9	37.8	52.9	56.1

Table 2: Performance on *ambiguous* word types of the HMM with standard EM (uniform starting transitions), just the initial Ψ transitions (ΨU and $\Psi\Lambda$), and EM initialized with ΨU and $\Psi\Lambda$, for lexicons with varied cutoffs. Note also that these scores do not include punctuation.

Next, note that both ΨU-EM and $\Psi\Lambda$-EM beat the randomly initialized EM for all cutoff levels. For the 10% tag cutoff (the first row), there is an absolute difference of over 2% for both.[9] As the ambiguity increases, the grammar-informed initialization has a much stronger effect. In the extreme case of using no cutoff at all (the *None* row of Table 2), ΨU-EM and $\Psi\Lambda$-EM beat EM by 19.9% and 23.1%, respectively. Finally, using the lexical category distribution Λ instead of a uniform one is much more effective when there is more lexical ambiguity (e.g., compare the .01 through None rows of the ΨU-EM and $\Psi\Lambda$-EM columns), but has a negligible effect with less ambiguity (rows .05 and .01). This demonstrates that the grammar-based initialization can be effectively exploited – it is in fact crucial for improving performance when we are given much more ambiguous lexicons.

The majority of errors with $\Psi\Lambda$-EM involve marking adjectives (N/N) as nouns (N) or vice versa, and assigning the wrong prepositional category (usually the simpler noun phrase post-modifier (NP\NP)/NP instead of the verb phrase modifier ((S\NP)\(S\NP))/NP. Both of these kinds of errors, and others, could potentially be corrected if the categories proposed by the tagger were further filtered by an attempt to parse each sentence with the categories.

6 Related work

The idea of using knowledge from the formalism for constraining supertagging originates with Ban-

[8]CCGbank actually corrects many errors in the Penn Treebank, and does not suffer as much from mistagged examples. However, there were two instances of an ill-formed category ((S[b]\NP)/NP)/ in wsj_0595 for the words *own* and *keep*. These were corrected to (S[b]\NP)/NP.

[9]For comparison with the performance of 87.6% for the fully supervised HMM on **all** tokens, Ψ-EM achieves 82.1% and 58.9% using a cutoff of .1 or no cutoff, respectively.

galore and Joshi (1999). They used constraints based on how elementary trees of Tree-Adjoining Grammar could or could not combine as filters to block out tags that do not fit in certain locations in the string. My approach is different is several ways. First, they dealt with fully supervised supertagging; here I show that using this knowledge is important for weakly supervised supertagging where we are given only a tag dictionary (lexicon). Second, my approach encodes grammar-based cues only as an initial bias, so categories are never explicitly filtered. Finally, I use CCG rather than TAG, which makes it possible to exploit a much higher degree of associativity in derivations. This in turn makes it easier to utilize prior knowledge about adjacent contexts – precisely what is needed for using the grammar to influence the transition probabilities of a bigram HMM.

On the other hand, Bangalore and Joshi (1999) use constraints that act at greater distances than I have considered here. For example, if one wishes to provide a word with the category ((S\NP)/PP)/NP, then there should be a word with a category which results in a PP two or more words to its right – this is something which the bigram transitions considered here cannot capture. An interesting way to extend the present approach would be to enforce such patterns as posterior constraints during EM (Graca et al., 2007).

Recent work considers a damaged tag dictionary by assuming that tags are known only for words that occur more than once or twice (Toutanova and Johnson, 2007). A very interesting aspect of this work is that they explicitly model ambiguity classes to exploit commonality in the lexicon between different word forms, which could be even more useful for supertagging.

In a grammar development context, it is often the case that only some of the categories for a word have been assigned. This is the scenario considered by Haghighi and Klein (2006) for POS tagging: how to construct an accurate tagger given a set of tags and a few example words for each of those tags. They use distributional similarity of words to define features for tagging that effectively allow such prototype words to stand in for others. This idea could be used with my approach as well; the most obvious way would be to use prototype words to suggest extra categories (beyond the tag dictionary) for known words and a reduced set of categories for unknown words.

Other work aims to do truly unsupervised learning of taggers, such as Goldwater and Griffiths (2007) and Johnson (2007). No tag dictionaries are assumed, and the models are parametrized with Dirichlet priors. The states of these models implicitly represent tags; however, it actually is not clear what the states in such models truly represent: they are (probably interesting) clusters that may or may not correspond to what we normally think of as parts-of-speech. POS tags are relatively inert, passive elements in a grammar, whereas CCG categories are the very drivers of grammatical analysis. That is, syntax is *projected*, quite locally, by lexical categories. It would thus be interesting to consider the induction of categories with grammar-based priors with such models.

7 Conclusion

I have shown that weakly supervised learning can indeed be used to induce supertaggers from a lexicon mapping words to their possible categories, but that the extra ambiguity in the supertagging task over that of POS tagging makes performance much more sensitive to rare categories that occur in larger, more ambiguous lexicons. However, I have also shown that the CCG formalism itself can provide the basis for useful distributions over lexical categories and tag transitions in a bitag HMM. By using these distributions to initialize the HMM, it is possible to improve performance regardless of the underlying ambiguity. This is especially important for reducing error when the lexicon used for bootstrapping is highly ambiguous and contains very rare categories.

Acknowledgments

Thanks to the UT Austin Natural Language Learning group and three anonymous reviewers for useful comments and feedback. This work was supported by NSF grant BCS-0651988 and a Faculty Research Assignment from UT Austin.

References

Bangalore, Srinivas and Aravind K. Joshi. 1999. Supertagging: an approach to almost parsing. *Computational Linguistics*, 25(2):237–265.

Banko, Michele and Robert C. Moore. 2004. Part-of-speech tagging in context. In *Proceedings of COLING*.

Beal, Matthew. 2003. *Variational Algorithms for Approximate Inference*. Ph.D. thesis, University of Cambridge.

Birch, Alexandra, Miles Osborne, and Philipp Koehn. 2007. CCG supertags in factored statistical machine translation. In *Proceedings of the 2nd Workshop on Statistical Machine Translation*.

Blunsom, Phil and Timothy Baldwin. 2006. Multilingual deep lexical acquisition for HPSGs via supertagging. In *Proceedings of EMNLP 06*, pages 164–171.

Clark, Stephen and James Curran. 2007. Wide-coverage efficient statistical parsing with CCG and log-linear models. *Computational Linguistics*, 33(4).

Clark, Stephen. 2002. Supertagging for combinatory categorial grammar. In *Proceedings of TAG+6*, pages 19–24, Venice, Italy.

Eisner, Jason. 1996. Efficient normal-form parsing for combinatory categorial grammars. In *Proceedings of the 35th ACL*.

Espinosa, Dominic, Michael White, and Dennis Mehay. 2008. Hypertagging: Supertagging for surface realization with CCG. In *Proceedings of ACL-08: HLT*, pages 183–191, Columbus, Ohio, June.

Goldwater, Sharon and Tom Griffiths. 2007. A fully Bayesian approach to unsupervised part-of-speech tagging. In *Proceedings of the 45th ACL*.

Graca, Joao, Kuzman Ganchev, and Ben Taskar. 2007. Expectation maximization, posterior constraints, and statistical alignment. In *Proceedings of NIPS07*.

Grenager, Trond, Dan Klein, and Christopher D. Manning. 2005. Unsupervised learning of field segmentation models for information extraction. In *Proceedings of the 43rd ACL*, pages 371–378.

Haghighi, Aria and Dan Klein. 2006. Prototype-driven learning for sequence models. In *Proceedings of HLT-NAACL 2006*.

Hassan, Hany, Khalil Sima'an, and Andy Way. 2007. Supertagged phrase-based statistical machine translation. In *Proceedings of the 45th ACL*.

Hockenmaier, Julia and Mark Steedman. 2007. CCG-bank: A corpus of CCG derivations and dependency structures extracted from the Penn Treebank. *Computational Linguistics*, 33(3):355–396.

Hockenmaier, Julia, Gann Bierner, and Jason Baldridge. 2004. Extending the coverage of a CCG system. *Research in Language and Computation*, 2:165–208.

Johnson, Mark. 2007. Why doesn't EM find good HMM POS-taggers? In *Proceedings of the EMNLP-CoNLL 2007*.

Joshi, Aravind. 1988. Tree Adjoining Grammars. In Dowty, David, Lauri Karttunen, and Arnold Zwicky, editors, *Natural Language Parsing*, pages 206–250. Cambridge University Press, Cambridge.

Kruijff, Geert-Jan M., Hendrik Zender, Patric Jensfelt, and Henrik I. Christensen. 2007. Situated dialogue and spacial organization: What, where,...and why? *International Journal of Advanced Robotic Systems*, 4(1):125–138.

Nielsen, Leif. 2002. Supertagging with combinatory categorial grammar. In *Proceedings of the Seventh ESSLLI Student Session*, pages 209–220.

Pollard, Carl and Ivan Sag. 1994. *Head Driven Phrase Structure Grammar*. CSLI/Chicago University Press, Chicago.

Rabiner, Lawrence. 1989. A tutorial on Hidden Markov Models and selected applications in speech recognition. *Proceedings of the IEEE*, 77(2):257–286.

Steedman, Mark. 2000. *The Syntactic Process*. The MIT Press, Cambridge Mass.

Toutanova, Kristina and Mark Johnson. 2007. A Bayesian LDA-based model for semi-supervised part-of-speech tagging. In *Proceedings of NIPS 20*.

Villavicencio, Aline. 2002. *The Acquisition of a Unification-Based Generalised Categorial Grammar*. Ph.D. thesis, University of Cambridge.

Wang, Qin Iris and Dal Schuurmans. 2005. Improved estimation for unsupervised part-of-speech tagging. In *EEE International Conference on Natural Language Processing and Knowledge Engineering*.

White, Michael and Jason Baldridge. 2003. Adapting chart realization to CCG. In *Proceedings of ENLG*.