All Words Unsupervised Semantic Category Labeling for Hindi

Siva Reddy, Abhilash Inumella, Rajeev Sangal, Soma Paul
Language Technologies Research Center
International Institute of Information Technology,
Gachibowli, Hyderabad, India - 500032
{*gvsreddy, abhilashi*}*@students.iiit.ac.in*, {*sangal, soma*}*@iiit.ac.in*

Abstract

In the task of semantic category labeling, given a text, every word in it has to be assigned a semantic category. Our language of interest is Hindi. We use the ontological categories defined in Hindi Wordnet as semantic category inventories. In this paper we present two unsupervised approaches namely Flat Semantic Category Labeler (FSCL) and Hierarchical Semantic Category Labeler (HSCL). The former method treats semantic categories as a flat list, whereas the latter one exploits the hierarchy among the semantic categories in a top down manner. Further our methods use simple probabilistic models, using which the category labeling becomes a simple table look up with little extra computation and thus opening the possibility of it's use in real-time interactive systems.

Keywords

word sense disambiguation, Semantic category labeling, unsupervised

1 Introduction

Given a word, its admissible semantic categories and its context, the task of semantic category labeling is to assign the most appropriate semantic category to the word. Our language of interest is Hindi.[1] An example is shown in *Table 1*. We use Hindi Wordnet Ontological categories [5] as semantic category inventories rather than Wordnet synsets. We justify the reason behind this later.

1.1 Ontological Categories

Hindi Wordnet's Ontology is a hierarchical organization of concepts. A separate ontological hierarchy exists for each syntactic category (noun, verb, adjective adverb). Number of nodes (categories) in noun, verb, adjective and adverb hierarchy are 101, 31, 25 and 11 respectively. The maximum depth of the hierarchy is 5. Each synset of the wordnet is mapped to some place in the hierarchy. Figure 1 depicts an example showing the mapping from the synsets of the word *billA* to ontological categories.

1.2 Ontological Categories versus Synsets ?

During manual annotation of few hindi sentences, we found that the inter annotator agreements were more when

[1] Hindi is the official language of India. Urdu is a close cousin to Hindi. Hindi and Urdu are spoken by approximately 500 million people in the world.

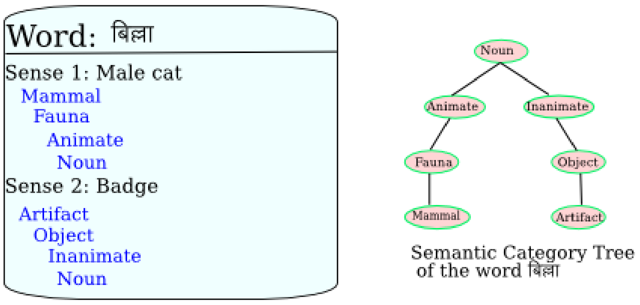

Fig. 1: Hindi wordnet entry of the word *billA*. The word has two senses meaning male cat and badge. Ontological category mappings of the two senses are shown on the left side of the figure. On the right, the semantic category tree(SCT) of the word is shown.

we used ontological categories compared to synsets. This is because various synsets (fine grained) are mappped to the same ontological category (coarse grained). A similar behavior was observed for English in earlier works. In the English Lexical Sample Task [7, 11] of Senseval-2, the inter annotator agreement of verbs rose to 82% using the grouped senses (coarse) from 73% using Wordnet 1.7 ungrouped senses. Ramakrishnan et al. [4] states that, sense disambiguation systems should not commit to a particular sense, but rather, to a set of senses which are not necessarily orthogonal or mutually exclusive. With coarse grained senses, this problem does not arise much. Fine grained senses might be useful for human users but are not necessary for many computer applications (chapter 3 of Agirre et al. [1]). So we use ontological categories of wordnet as semantic category inventories. For a detailed discussion on fine grained vs coarse grained senses, refer to chapter 4 of [1].

1.3 Semantic Category labeling is interesting

Recently, in the work on Hindi dependency parser by Bharati et al.[3], the use of semantic features has been exploited. They used just two features namely, human-nonhuman and animate-inanimate to boost the accuracy of dependency parser. For some labels, the increase is 5-10%. This is an encouraging result which shows the effect of using minimal semantic features on parsing accuracy. Other tasks that can benefit from using our system are Machine translatio, building dictionaries from parallel corpora, named entity recognition, information extraction

1.	*kuwwe/Dog*	ko	*xeKawe/seeing*	hI	**billA/cat**	*pedZa/tree*	para/on	caDZa/climbed	gayA
	Mammal		*NaturalEvent*		*Mammal*	*NaturalObject*		*VerbOfAction*	

2.	*saBA/Meeting*	meM/in	*Aye/came*	saBI/all	*svayaMsevaka/volunteers*	**billA/badge**	lagAye/wear	hue	We
	Event		*VerbOfAction*		*Group*	*Artifact*	*VerbOfState*		

Table 1: Examples showing the task of semantic category labeling.

etc. This motivates us to present, all words unsupervised semantic category labeling using raw or pos tagged text.

The methodology presented here has the capability of performing both unsupervised and supervised (using sense annotated corpora) sense disambiguation. For reasons of clarity and space, we focus in this paper only on the description of unsupervised approach.

Our paper is organized as follows. In section 2 we present the related work. In section 3 we give the definitions. Our developed methods are presented in section 4. Section 5 covers the evaluation aspects. Section 6 has concluding remarks.

2 Related work

Earlier work on Hindi WSD has been done by Sinha et al. [10] using wordnet synsets. They used Lesk like algorithm [8] where the target word's synset which has maximum overlap of its gloss, its hypernymy gloss and its hyponymy gloss with the words in the context of target word is chosen as the sense of the word. Lesk alogrithm cannot be used here since the definition of our semantic (ontological) categories is very general and does not have sufficient gloss to cover all its occurrences. To our knowledge, ours is the first attempt to work on Hindi semantic category labeling using ontological categories.

Patwardhan et al. [12] WSD systems disambiguates a target word by using WordNet-based measures of semantic relatedness to find the sense of the word that is semantically most strongly related to the senses of the words in the context of the target word. Sinha and Mihalcea [13] present Graph based unsupervised word sense disambiguation. Their work combines the word semantic similarity measures and graph centrality measures for sense disambiguation.

Semantic relatedness measures between ontological categories of hindi wordnet have yet to be studied and expored. In this scenario, we present approaches which do not need such semantic relatedness measures.

Yarowsky [14] unsupervised WSD uses Bayesian theoretical framework where words that are indicative to each category are identified and weighed and these words are used in selection of a category. We use a probabilistic model slightly similar to the one used by Yarowsky.

3 Definitions

We first introduce the task formally and define some terms which are used in further discussion. The task of semantic category labeling can be formally defined as follows. Given a sequence of words $W = \{w_1, w_2, \ldots, w_n\}$ with each word w_i having semantic categories $SC_{w_i} = \{{c_{w_i}}^1, {c_{w_i}}^2, \ldots, {c_{w_i}}^{N_{w_i}}\}$, we have to assign a category to each word w_i from the set of it's semantic categories SC_{w_i}.

Definition 3.1. *First order collocational features* of a word *w* are the set of features describing the context of the word *w*. A feature *f* is a tuple which can be defined by one of the following templates (*sw*) or (*sw*, posOf(*sw*)) or (*sw*, posof(*sw*), posOf(*w*)) etc. where *sw* is the surrounding word of *w*, posOf(*w*) is the pos tag of *w*. Consider the following sentences.

- I ate an orange.
- Monkey is eating a banana.
- She eats an apple everyday.

If we define a feature of a word as (*sw*) within a distance of 2 words, then the first order collocational features of *eat, orange, banana* and *apple* are {*I, orange, monkey, banana, she, apple*}, {*eat*}, {*eat*}, and {*eat, everyday*} respectively.

Definition 3.2. *Second order collocates*: A word *x* is said to be second order collocate of *y* with respect to feature *f*, iff the feature *f* is a first order collocational feature of *x* and *y*. In the above example, {*orange, banana, apple*} are second order collocates w.r.t feature *eat* because *eat* is a first order collocational feature of all the three words *orange, banana and apple*

Definition 3.3. *Semantic Category Tree (SCT)*: As already said, hindi wordnet has an ontological hierarchy and each sense of a word is mapped to some place in this hierarchy. The SCT of a word is a sub tree of this hierarchy which is shared with all the senses of this word. For example the SCT of word *billA* is shown in figure 1. If the pos tag of the word is known beforehand, only the subtree corresponding to this pos-tag is considered as SCT.

4 Our Approach

Our approach is inspired from the work Lin [9]. He uses syntactic dependency as local context to do word sense disambiguation. His work is based on the intuition that

> Two different words are likely to have similar meanings if they occur in identical local contexts.

Our assumption similar to Lin [9] is

> Two different words are likely to have similar semantic category if they have identical first order collocational features i.e. if they are second order collocates to each other.

In this section, we present the methods Flat Semantic Category Labeler (FSCL) and Hierarchical Semantic Category Labeler (HSCL). FSCL treats semantic categories as a flat list whereas HSCL exploits the hierarchy among the categories. Both these methods take the following steps.

- **Training Phase**
  - **Step 1:** Collect second order collocates w.r.t all the features present in the training corpus.
  - **Step 2:** Build training models from second order collocation sets. Aim of this step is to calculate the likelihood of a category *cat* given a feature.
- **Disambiguation Phase**
  - **Step 3:** Using the above training models for Semantic Category labeling.

Detailed discussion of each step is given below.

Step 1: First order collocational features *F* of all the words present in the training corpus are collected using feature templates. We tried out different feature templates and the best are presented here.

1. (sw_k)
2. (sw_k , posOf(sw_k) , posOf(sw_0))

where sw_k (sw_{-k}) denote the k^{th} surrounding word to the right (left) of word of interest sw_0. k can be only in the range of ($-m$, m) where $2 * m + 1$ is the size of window. posOf(sw) is the part of speech tag of sw.

For every feature f_j in *F*, **S**econd **O**rder **C**ollocate sets w.r.t f_j, SOC_{f_j}, are calculated i.e. all the words which have feature f_j as first order collocational feature are collected.

In the following sections, we discuss two different methods that differ in the way *steps 2,3* are performed.

4.1 Flat Semantic Category labeler (FSCL)

Based on our assumption that second order collocates w.r.t a feature f_j, SOC_{f_j}, are likely to have same semantic category, we calculate the expectation of the occurrence of each semantic category with feature f_j. Only the leaf semantic categories of the all the words in the second order collocate set SOC_{f_j} are considered. Hierarchial information of the semantic categories is not used.

Step 2: Aim of this step is to calculate the expectation of occurrence of category *cat* with feature f_j. To calculate this, we use the following equations.

$$\begin{aligned} Pr(cat|f_j) &= \frac{Count(cat, SOC_{f_j})}{\sum_{cat} Count(cat, SOC_{f_j})} \\ AE(cat|f_j) &= \frac{Pr(cat|f_j)}{Pr(f_j)} \end{aligned} \quad (1)$$

where $Pr(cat|f_j)$ denotes the probability of the occurence of category *cat* with feature f_j. $Count(cat, SOC_{f_j})$ denotes the number of words in SOC_{f_j} which have category *cat* as their leaf semantic category. $AE(cat/f_j)$ is the above expectation measure which gives the expectation of occurrence of *cat* with feature f_j. Some of the most frequent features consisting of function words, occur with almost all the categories. This measure penalizes such words and rewards the salient features of a category. A similar AE measure can be found in Kavalec et al. [6]. In his work, the measure *above expectation* (AE) is employed for non taxonomic relation extraction.

Step 3: This is the disambiguation phase where an utterance of a word w_i with leaf semantic categories SC_{w_i}= $\{c_1, c_2, \ldots\}$ is assigned a category according to the following equation.

$$\underset{c_k \,\in\, SC_{w_i}}{\text{argMax}} \sum_{j=1}^{F} AE(c_k|f_j)$$

where $f_1, f_2, \ldots, f_F$ are the first order collocational features of w_i. $AE(c_k|f_j)$ is the expectation of the occurrence of c_k with feature f_j which is calculated in the previous step (training phase). The expected occurrence of each admissible leaf category of w_i is calculated w.r.t all its first order collocational features and the category with highest score is chosen. We used summation over all features because AE is an expectation measure and not a probability measure.

4.2 Hierarchical Semantic Category labeler (HSCL)

This method uses the hierarchical information of the semantic category tree (FSCL uses only leaf categories). In the training phase, given a feature, the expectation of each category at each level of the semantic category tree are calculated. The disambiguation algorithm runs in a top down fashion and takes a decision at each level based on the available expectation scores at that level. The details are as follows.

Step 2: Given a feature f_j, the aim of this step is to calculate the expectation of each category at each level of the semantic category tree (SCT). Let c_h^i denote i^{th} category at the level h of SCT. This phase is summarized below.

- Aggregate the semantic category trees of all of the words in the set SOC_{f_j}: The semantic category trees of the second order collocates w.r.t feature f_j are obtained. They are aggregated in this step to form the aggregate tree AGT_{f_j}. To perform the aggregation we take the union of semantic category trees of all the words in SOC_{f_i}. Union of two trees is a simple operation by doing which, the nodes common to both the trees get their scores summed up and the for others it remains the same. Initially each tree node carries a score of .

  $$node.score = \frac{1}{|node.siblings| + 1}$$

  Aggregation of trees:

  $$AGT_{f_j} = \{ \bigcup_{w \,\in\, SOC_{f_j}} SCT(w)\}$$

- Normalize the scores of each node in the tree AGT_{f_j} to calculate $Pr(c_h^i|f_j)$: The nodes in of the tree AGT_{f_j} carry the summed up scores as a result of aggregation operation performed in previous step. These scores are normalized according the following equation.

  $$\begin{aligned} Pr(c_h^i|f_j) &\simeq \frac{n_h^i.score}{|SOC_{f_j}|} \\ AE(c_h^i|f_j) &= \frac{Pr(c_h^i|f_j)}{Pr(f_j)} \end{aligned} \quad (2)$$

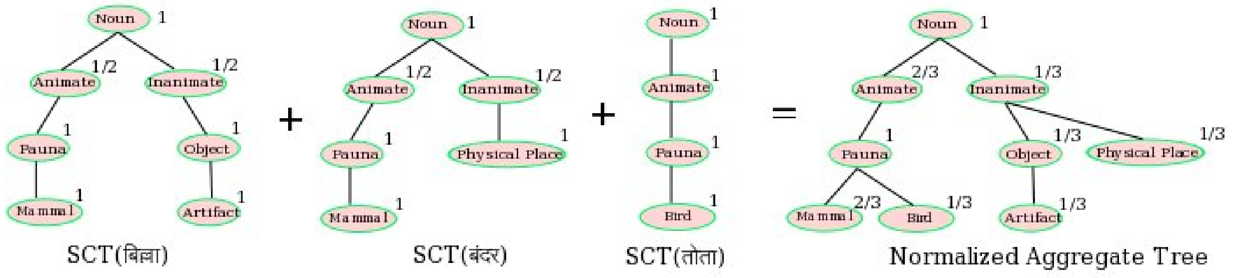

Fig. 2: Aggregation and Normalization of Semantic Category trees

where n_h^i is the node in AGT_{f_j} corresponding to the category c_h^i. $AE(c_h^i|f_j)$ is the above expectation measure which gives the expectation of the occurrence of c_h^i when the feature f_j. Note: $Pr(c_h^i|f_j)$ is not the exact probability. This measure gives more preference to the words in SOC_{f_j} which are less ambiguous.

The example shown in the figure 2 clarifies the aggregation and normalization steps used in this algorithm. The feature used in this example is (*caDZa/climb*). The set $SOC_{\text{(caDZa/climb)}}$ consists of words *billA/Cat, baMxara/Monkey, wowA/Parrot*. The semantic category trees of these words are shown on the left with their initial scores. The right most tree is formed after the normalization of the aggregated tree $AGT_{\text{(caDZa/climb)}}$.

In this paragraph, we discuss an alternative scoring mechanism. The same example in figure 2 is used to explain this mechanism. The probabilities in this are calculated as follows. Take initial $node.score$ to be 1 for each tree. Aggregate all of them to form an aggregate tree. In the example figure, scores on nodes $Noun$, $Animate$, $Inanimate$ of the aggregate tree will be 3, 3, 2 respectively. Normalization is performed using the following equation

$$Pr(c_h^i|f_j) = \frac{n_h^i.score}{\sum_k n_h^k.score}$$

Scores on nodes $Noun$, $Animate$, $Inanimate$ of the aggregate tree after the normalization are $3/3$, $3/5$, $2/5$ respectively. The ratio of the probability of $Animate$ and $Inanimate$ is $(3/5)/(2/5) = 3/2$. Using the former scoring mechanism it is $(2/3)/(1/3) = 2$. The former mechanism accumulates a higher confidence for the category $Animate$ compared to $Inanimate$ because it gives more preference to words with one sense (here wowA/parrot) and the latter model gives equal preference to all the words. To put it in other words, our scoring mechanism gives preference to semantic category of the words with single sense assuming that this semantic category is more likely to occur with the given feature.

Step 3: To disambiguate an occurrence a word w_i with it's collocational features f_j a top down walk is performed on the semantic category tree of w_i. The set of categories at level h (denoted by $SCT^h(w_i)$) are disambiguated first before moving to disambiguate at level $h+1$. Once a category is decided at level h, then the algorithm considers only the children of this category in level $h+1$. This results in reducing the semantic category search space of disambiguation algorithm. For more details, refer to the algorithm below.

Algorithm 1 HSCL Disambiguation phase

```
1: Input: $w_i$ and it's collocational features $f_j$
2: Output: A semantic category path.
3: cur=TOP
4: **for** each level $h \in \{0, 1, \ldots\}$ **do**
5:     $pList = \{c | c \in SCT^h(w_i) \& parent(c) == cur\}$
       //pruning the list of categories at level $h$
6:     $cur = \underset{cat \in pList}{\text{argMax}} \sum_{j=1}^{F} AE(cat|f_j)$
7:     append $cur$ to output
8: **end for**
```

Advantages of HSCL:

- HSCL disambiguates level by level. Number of categories to be disambiguated in the top level are less compared to the number of leaves of the semantic category tree. This reduces the search space while disambiguation and hence it becomes simpler.
- No need of semantic similarity/relatedness measures.
- The nodes at top levels are shared by large number of words. This makes the learning effective for these nodes and hence the method takes better decisions at top levels.
- This can handle unseen category instances because the disambiguation proceeds in top down manner.
- This method can stop at a level which has high confidence score.

5 Evaluation

We trained our methods on a 1.2 million word corpus. We used a separate corpus for evaluating the proposed algorithms. The testing data comprises of 7200 manual annotated sentences which cover 133 semantic categories.

It is desirable to have high precision and low recall systems in certain scenarios. To achieve this, a word is committed to a category only if the confidence score is greater than the set threshold value. The threshold value is chosen to be the k times the average of the set S consisting of all category scores over all the features.

$$\theta = k * averageoftheSet(S)$$

where $\forall cat \forall j Pr(cat|f_j) \in S$. Set S is collected during Training phase. As k is increased, precision increases (with decrease in recall)

5.1 FSCL accuracies

The **baseline** system assigns the semantic category of first sense of the word. The evaluation results of FSCL for **nouns** is shown in table 2.

Model	**P**	**R**
Baseline	85.6	85.6
FSCL trained on raw text	75.6	75.6
FSCL with k=2 trained on raw text	84.7	53.9
FSCL with k=3 trained on raw text	87.8	50.0
FSCL with k=2 trained on pos tagged text	83.2	63.4

Table 2: Accuracies of FSCL and Baseline for **nouns** (P: precision and R:recall)

As discussed in *Section 4*, feature (sw_k) and (sw_j, $pos(sw_j)$, $pos(sw_0)$) are used as features for training on raw and pos-tagged text repectively. Window size of 20 is used in all the models.

As k value is increased, FSCL method performs better than the baseline. We believe that the reason for low recall is because of the size of training corpus. For English, huge corpora above 100 million words are available. But for Hindi, such huge corpora does not exist. Once if our models are built using such huge corpora, recall can also be increased since the number of salient words for each category increases.

We see that the precision of the model trained on pos-tagged text is less compared to others because of the low accuracy of the hindi pos-tagger which is about 78%. Training corpus is pos tagged using [2].

5.2 Level wise accuracies of HSCL

	Baseline		HSCL ($k = 5$)	
Level	P	R	P	R
1	96.9	96.9	99.4	94.0
2	91.5	91.5	96.4	63.8
3	89.8	89.8	95.4	52.0
4	87.7	87.7	94.4	46.4
5	76.8	76.8	83.1	64.4

Table 3: Level wise accuracies of HSCL for **nouns**

For each level, the **baseline** system assigns the semantic category of the first sense of the word corresponding to that level.

The results obtained using HSCL method with $k = 5$ are shown in the table 3. Window size of 20 is taken. Raw text is used for training. We see that HSCL outperforms the baseline (first sense) in terms of precision. The recall values of HSCL are low compared to baseline.

Comparing HSCL with FSCL, precision values of HSCL are very high and the recall values of HSCL are comparable with FSCL. This shows us that high precision values can be achieved with HSCL compared to FSCL for the same recall values.

6 Conclusion

In this paper we have introduced the problem of semantic category labeling and also presented two unsupervised methods for performing this task. These methods do not use semantic similarity measures. To label an utterance of size n, an efficient implementation of our disambiguation procedure takes a time $O(n * s)$, where s is the maximum number of senses of a word in this utterance. Besides presenting the evaluations of our algorithms, we also presented a simple parameter tuning procedure to obtain a precision recall tradeoff.

In the near future, we are integrating our system with Hindi dependency parser and study the effect of semantic features on parsing accuracies. Also, we are interested to apply the methods discussed here to English language using the synset hierarchy of English Wordnet.

References

[1] E. Agirre and P. Edmonds. *Word Sense Disambiguation: Algorithms and Applications (Text, Speech and Language Technology).* Springer-Verlag New York, Inc., Secaucus, NJ, USA, 2006.

[2] Avinesh.PVS and K. Gali. Part-of-speech tagging and chunking using conditional random fields and transformation based learning. In *IJCAI-07 Workshop on "Shallow Parsing in South Asian Languages"*, 2007.

[3] A. Bharati, S. Husain, B. Ambati, S. Jain, D. M. Sharma, and R. Sangal. Two semantic features make all the difference in parsing accuracy. In *International Conference on Natural Language Processing (ICON-08), CDAC, Pune, India*, 2008.

[4] A. D. P. B. Ganesh Ramakrishnan, B. P. Prithviraj and S. Chakrabarti. Soft word sense disambiguation. In *The Second Global Wordnet Conference, Masaryk University Brno, Czech Republic*, pages 33–64, 2004.

[5] S. Jha, D. Narayan, P. Pande, and P. Bhattacharyya. A wordnet for hindi. In *International Workshop on Lexical Resources in Natural Language Processing, Hyderabad, India, January*, 2001.

[6] M. Kavalec, E. Maedche, and V. Svtek. Discovery of lexical entries for non–taxonomic. In *In: Proceedings of SOFSEM 2004: Theory and Practice of Computer Science, LNCS 2932*, pages 249–256, 2004.

[7] A. Kilgarriff. English lexical sample task description. In *In Proceedings of the SENSEVAL -2 workshop, ACL Workshop.*, 2001.

[8] M. Lesk. Automatic sense disambiguation using machine readable dictionaries: how to tell a pine cone from an ice cream cone. In *SIGDOC '86: Proceedings of the 5th annual international conference on Systems documentation*, pages 24–26, New York, NY, USA, 1986. ACM.

[9] D. Lin. Using syntactic dependency as local context to resolve word sense ambiguity. In *In Proceedings of ACL/EACL-97*, pages 64–71, 1997.

[10] P. P. L. K. Manish Sinha, Mahesh Kumar and P. Bhattacharyya. Hindi word sense disambiguation. In *In: International Symposium on Machine Translation, Natural Language Processing and Translation Support Systems, Delhi, India, November, 2004.*, 2004.

[11] S. C. L. D. Martha Palmer, Christiane Fellbaum and H. T. Dang. English tasks: All-words and verb lexical sample. In *In Proceedings of the SENSEVAL -2 workshop, ACL Workshop.*, 2001.

[12] S. Patwardhan, S. Banerjee, and T. Pedersen. Using measures of semantic relatedness for word sense disambiguation. In *Proceedings of the Fourth International Conference on Intelligent Text Processing and Computational Linguistics*, pages 241–257, Mexico City, Mexico, February 2003.

[13] R. Sinha and R. Mihalcea. Unsupervised graph-basedword sense disambiguation using measures of word semantic similarity. In *ICSC '07: Proceedings of the International Conference on Semantic Computing*, pages 363–369, Washington, DC, USA, 2007. IEEE Computer Society.

[14] D. Yarowsky. Word-sense disambiguation using statistical models of roget's categories trained on large corpora. In *In Proceedings of COLING-92*, pages 454–460, 1992.