A Maximum Entropy Approach To Disambiguating VerbNet Classes

Lin Chen
University of Illinois at Chicago
Chicago, IL, USA
lin@chenlin.net

Barbara Di Eugenio
University of Illinois at Chicago
Chicago, IL, USA
bdieugen@uic.edu

Abstract

This paper focuses on verb sense disambiguation cast as inferring the VerbNet class to which a verb belongs. To train three different supervised learning models –Maximum Entropy (MaxEnt), Naive Bayes and Decision Tree– we used lexical, co-occurrence and typed-dependency features. For each model, we built three classifiers: one single classifier for all verbs, one single classifier for polysemous verbs only, and an ensemble of classifiers, one per each polysemous verb. Among those algorithms, Naive Bayes performs surprisingly badly. In general, MaxEnt models perform better, but Decision Trees models are competitive. Our best results are obtained with classifier ensembles.

1 Introduction

Our research group has long been involved in research on the interpretation and generation of instructional texts. Not only do we believe that verbs provide a crucial component of the semantics of such texts; we also have shown that verb-based semantics helps achieve more accurate discourse parsing (Subba and Di Eugenio, 2009). For our work on discourse parsing, we developed a new resource, the HomeRepair corpus, which contains 176 documents for a total of 53,250 words. It was manually annotated with rhetorical relations and *quasi-automatically* annotated with semantics. It was parsed with LCFLEX (Rosé and Lavie, 2000), which we integrated with VerbNet (Kipper et al., 2008) and with CoreLex, a noun lexicon (Buitelaar, 1998) (see (Subba et al., 2006) for details).

VerbNet (VN) is currently the largest English verb semantics resource. In VN, verbs are grouped in classes and subclasses. Each VN class is completely described by thematic roles, selectional restrictions on the arguments, and frames consisting of a syntactic description and semantic predicates – see the class *remove-10.1* in Figure 1. Our parser was integrated with VerbNet 2.1, which covered 3445 different verbs, for a total of 4656 verb senses, grouped in 191 first level classes. [1]

The *quasi*-automatic quality of the semantic annotation of our corpus is due to manual disambiguation of the correct interpretation among several LCFLEX may return. Some alternative interpretations are due to syntactic ambiguities, but others, to lack of verb sense disambiguation. For example, in the sentence *you may have to cut some tiles*, *cut* is mapped to two distinct VN classes, *BUILD-26.1* and the correct *CUT-21.1.*

Our work builds on much previous work on verb sense disambiguation. Verb sense disambiguation is a subtask within word sense disambiguation, but we do not have room here to review that vast literature. As concerns verb sense disambiguation, a first distinction concerns what counts as a verb sense: some of the work, e.g. (Dang and Palmer, 2005; Dligach and Palmer, 2008; Banerjee and Pedersen, 2010), focuses on verb senses variously derived from WordNet senses, not on VN class disambiguation. Other work, e.g. (Lapata and Brew, 2004), uses Levin's verb class definitions, which in turn are the foundations of VerbNet class definitions, but result in a different classification problem. If we now turn to VN class disambiguation, distinctions in approaches concern the specific models used, the features those models are built from, and / or the corpora that are employed. Previous work on VN class disambiguation (Girju et al., 2005; Abend et al., 2008) has focused almost exclusively on standard corpora such as PropBank; more importantly, it has used no relational information between a verb and its arguments, whereas we use typed dependencies here.

[1] VerbNet 3.1, the latest version, contains 3769 different verbs, for a total of 5257 verb senses, grouped in 274 classes.

CLASS: remove-10.1
PARENT: -
MEMBERS: abstract, cull, delete, disgorge, dislodge, disengage, draw, eject, eliminate, eradicate, remove ...
THEMATIC ROLES: Agent Theme Source
SELECTIONAL RESTRICTIONS: Agent[+int_control OR +organization] Theme[] Source[+location]
FRAMES:

Transitive	Agent V Theme	cause(Agent, E) ∧ ¬location(start(E), Theme, ?Source) ∧ location(end(E), Theme, ?Source)
Transitive (+ Source PP)	Agent V Theme Prep[+src] Source	cause(Agent, E) ∧ ¬location(start(E), Theme, Source) ∧ location(end(E), Theme, Source)

Figure 1: The class remove-10.1 from VerbNet

Name	**Description**	**Sentence**	**Extracted Relation**
dobj	direct object	They win the lottery	dobj(win, lottery)
iobj	indirect object	She gave me a raise	iobj(gave, me)
prep/prepc	prepositional modifier	I saw a cat with a telescope	prep(saw, with)
prt	phrasal verb particle	They shut down the station	prt(shut, down)
nsubj	nominal subject	Clinton defeated Dole	nsubj(defeated, Clinton)
nsubjpass	passive nominal subject	Dole was defeated by Clinton	nsubjpass(defeated, Dole)
xsubj	controlling subject	Tom likes to eat fish	xsubj(eat, Tom)

Table 1: Typed Dependency Examples

In this paper, in section 2 we describe the three supervised learning approaches we experimented with, using three types of feature sets (section 2.1), and developing three different classifiers per model. We ran experiments on four datasets (section 3), and results can be found in section 4. As discussed in section 5, Naive Bayes performs surprisingly badly in all conditions. MaxEnt models always perform better than Decision Trees on manually built datasets such as VerbNet itself and WordNet; however, on our own HomeRepair corpus, Decision Trees perform better when a single classifier for all verbs is built, most likely because the VN class training data is somewhat noisy. Our best results are obtained with an ensemble of classifiers, one per polysemous verb.

2 Methodology

In this work, we were mainly interested in exploring the purported strength of the MaxEnt classification algorithm, with respect to more traditional models such as Naive Bayes and Decision Tree (DT) classifiers. MaxEnt is a uniform model, which makes no assumptions in addition to what we know from the data. It also has the strong capability to combine multiple and dependent knowledge sources, as opposed to the independence assumption underlying Naive Bayes. MaxEnt has been widely used in NLP and proven to be effective and efficient. Our hypothesis that Naive Bayes would perform poorly was borne out; however, the performance of the DT classifiers is competitive with that of MaxEnt, as we will discuss below.

We recast the VN class disambiguation problem as a classification problem, where a tuple (Sentence, Verb) needs to be assigned to the correct VN class. We devised two different classification models:

- **Single Classifier Model**: Train each classifier on all the verbs in the dataset.
- **Per-verb Classifier Model**: Train one classifier per each verb in the training set. Given a tuple (Sentence, Verb), we only use that verb's classifier to choose its VN class.

2.1 Features

We build our classification models using the following three types of features:

- **Lexical**: The word's base form and its POS tag.
- **Co-occurrence**: The words and POS tags which appear around the target verb in a window size of 5 (the window size of 5 was determined experimentally).
- **Typed Dependency**: All the dependencies where the verb to be disambiguated partic-

ipates, derived by means of the Stanford parser (de Marneffe and Manning, 2008).

We use lexical features and co-occurrence features as in (Girju et al., 2005; Abend et al., 2008); co-occurrence features are used to approximate collocations, since the collocation of a word can help decide its sense (Yarowsky, 1993). We add typed dependencies, since, compared to co-occurrence, grammatical relations capture more specific relations between the verb and other words in the sentence, and encode some of the structure of the sentence. We parse the sentences by means of the Stanford parser (de Marneffe and Manning, 2008) and the dependencies related to the verb to be disambiguated are extracted as part of the feature space. We use all dependencies available, some of which are presented in Table 1 with illustrative examples.

3 Data Sets

Our datasets are composed of all instances defined as follows: (Sentence, Verb, VN Class). We built three data sets according to where the sentences come from, plus a fourth that combines the other three. Whereas the goal of our work is to fully automatize parsing our HomeRepair corpus, other datasets are used to validate our approach. Each sentence is POS-tagged and parsed with the Stanford Dependency parser, to derive all the features we discussed earlier.

- **VerbNet**: For each frame in every VN class, VN provides one example sentence. The sentence is paired with the specific verb it contains. Clearly, this dataset should, and will, give rise to the most accurate results.

- **WordNet**: When VN provides members of a VN class, it also gives WordNet sense mappings when applicable. For example, the verb ”instruct” in the VN class ”advise-37.9” is mapped to WordNet sense ”instruct%2:32:01”. In turn, WordNet provides illustrative sentences as examples for each word sense. We extracted the example sentences to construct a dataset, but we excluded cases where VN provides multiple WordNet sense mappings.

- **HomeRepair**: the portion of the HomeRepair corpus that was used to evaluate the discourse parser in (Subba and Di Eugenio, 2009). As noted earlier, the VN class was obtained via LCFLEX and manual disambiguation of VN classes. However the data was parsed again with the Stanford parser to obtain Typed Dependencies.

Finally, we combined the 3 datasets above to build a larger dataset. The sizes of those datasets are listed in Table 2. In that table, *Instance* gives the number of (Sentence, Verb, VN Class) tuples; *Verb* is the number of distinct verbs in the dataset; *Class* is the total number of distinct VN classes in that dataset. Note that for VerbNet, the number of verbs in Table 2 is much lower than the number of distinct verbs we mentioned above. This is due to the fact that, for each (sub)class, VerbNet uses only one representative of the (sub)class in all the examples for that (sub)class.

4 Experiments

We used OpenNLP Tools,[2] an open source Java NLP library that provides a collection of basic text processing tools for tasks like sentence detection, tokenization, part-of-speech tagging, text chunking, named entity recognition, co-reference resolution and tree parsing. OpenNLP also includes the MaxEnt[3] Java package for Maximum Entropy modeling. We employed the data mining tool package Weka (Hall et al., 2009) for Naive Bayes and Decision Tree classifications (Weka’s J48 implementation was used for Decision Trees). The JWNL (Java WordNet Library)[4] was used to extract the WordNet dataset.

For each approach, we conducted three sets of experiments, for each dataset we described in Section 3. In all experiments we used exactly the same features by building a converter to convert the extracted features for MaxEnt to Weka’s Attribute-Relation File Format(.arff) data format. In all experiments, the MaxEnt models were Generalized Iterative Scaling(GIS) models trained through 100 iterations and with no cut off. All the accuracies are calculated with 10-fold cross validation. Our baseline model assigns a (Verb, Sentence) tuple to the majority class of the verb in the training data set.

The first set of experiments used the entire datasets, no matter whether a verb is polysemous or not; results are shown in Table 2. Weka failed to

[2] http://opennlp.sourceforge.net/

[3] http://maxent.sourceforge.net/

[4] http://sourceforge.net/projects/jwordnet/

Dataset	Instance	Verb	Class	Baseline	NaiveBayes	J48	MaxEnt
VerbNet	838	310	265	0.9078	0.3059	0.8091	**0.9558**
WordNet	1586	1108	224	0.8432	0.1883	0.4678	**0.8877**
HomeRepair	2111	293	127	0.8115	0.4424	**0.8423**	0.7335
Combined	5633	1523	329	0.7800	N.A.	0.7264	**0.8528**

Table 2: Single Classifier on All Data

Dataset	Instance	Verb	Class	Baseline	NaiveBayes	J48	MaxEnt
VerbNet	431	151	140	0.8385	0.3364	0.7819	**0.9077**
WordNet	516	257	146	0.5354	0.1686	0.3817	**0.6375**
HomeRepair	1353	154	98	0.5458	0.4804	**0.8064**	0.6776
Combined	2300	418	233	0.5083	0.3561	0.7130	**0.7283**

Table 3: Single Classifier on Polysemous Verbs

generate any result when we ran Naive Bayes on the combined data set, since it ran out of memory even after we assigned to it 2 GB of memory, the maximum amount we had available.

The second set of experiments is restricted to only polysemous verbs (see Table 3). The degree of attested polysemy for the three datasets hovers just above 2, with VerbNet the lowest at 2.08, and the combined set the highest at 2.24 (the same VN class inventory is used in each set).

The third set of experiments uses the per-verb classifier model. In each iteration, for every verb, we select all the instances that include that verb, and split them according to 10-fold validation. For verbs which have less than 10 instances, we randomly choose one of them as testing instance, and use the others for training. We ran this set of experiments only on those polysemous verbs for which there are at least 3 instances of each pair (Verb, VN Class). Results are shown in Table 4.

5 Discussion

Not surprisingly, in all experiments, results on VerbNet are always very high for all classification models. This is due to the consistency of VN data, because VerbNet always uses the same verb to give examples for a VN class.

Baseline gave very high accuracy in experiment set 1 (see Table 2). This is not surprising, since the complete data set contains a large portion of verbs which are not polysemous.

Naive Bayes became the real baseline, since it always performs worst, and by far, in all the experiments. We believe it is because of the nature of posterior probabilities, and lack of independence among features. For co-occurrence features and Typed Dependency features, the feature value space is too big. Additionally, the overlap of values among different co-occurrence features violates the assumption of independence underlying Naive Bayes.

In almost every case, MaxEnt models perform better than the other models. In most cases, χ^2 shows that these differences are significant at the p $\leq$ 0.05 level.

MaxEnt did worse than J48 is the single classifier experiments on the HomeRepair data, both on all verbs and on polysemous verbs only (Tables 2 and 3). As we noted earlier, in HomeRepair the VN class data was obtained by employing the LCFLEX parser, and then manual choice of the correct parse when more than one was returned. Both the parser itself and the manual disambiguation introduce noise: there are similar sentences with the same verb, where the two verb instances are assigned to two different VN classes. Please note that in Table 3, whereas MaxEnt performs better than J48 on the combined set, the difference is not significant. In general, J48 performs poorly on the WordNet set when a single classifier is trained (Tables 2 and 3).

We obtained our most promising result with the per-verb classifier ensemble, and on the HomeRepair / combined corpora (Table 4). Whereas the accuracy drops considerably with respect to VerbNet in the other experiments, it does not here. We note however that in this table, the difference in performance between MaxEnt and baseline on VerbNet is not significant (all other differences are).

It is not possible to draw a real comparison

Dataset	Instance	Verb	Class	Baseline	NaiveBayes	J48	MaxEnt
VerbNet	277	49	52	0.9429	0.3704	0.9259	**0.9667**
WordNet	158	30	33	0.6750	0.3333	0.7095	**0.8378**
HomeRepair	1221	81	59	0.7764	0.4194	0.8065	**0.8956**
Combined	1858	164	145	0.7113	0.2752	0.6833	**0.8986**

Table 4: Per-Verb Classifier on Polysemous Verbs

with work in the literature because we use different datasets. However, at a high level we note that (Girju et al., 2005) uses data derived from PropBank, but finds that only about 4% of verbs are polysemous, and on this set their best model achieves around 80% accuracy. (Abend et al., 2008) performs at around 92% when tested on the Wall Street Journal, but when the model is applied to medical tests, it falls to 55%. Because VerbNet is domain independent, we expect our per-verb classifier trained on the combined datasets to be accurate on other domains as well. This claim clearly needs to be tested on other datasets.

6 Future Work

As just stated, we intend to explore the applicability of our models, specifically the ensemble of classifiers trained on the whole dataset, to other corpora.

One of the remaining issues is handling unseen verbs in the training data. We believe our single classifier model will be able to handle it, but we need to design experiments to evaluate the performances.

Another issue is how to generalize the Typed Dependency features we employ. Because the dependency arguments we extracted are not generalized, when the pre-labeled training data set is small, the extracted features will be hard to match incoming examples. One promising approach is to generalize the arguments to the dependencies. For example, we could use CoreLex (Buitelaar, 1998) to generalize nouns to CoreLex classes. Another way to generalize Typed Dependencies is to use Dynamic Dependency Neighbors as employed by (Dligach and Palmer, 2008).

Acknowledgments

This work is supported by award IIS-0905593, National Science Foundation. Thanks to three anonymous reviewers for their suggestions.

References

Omri Abend, Roi Reichart, and Ari Rappoport. 2008. A supervised algorithm for verb disambiguation into verbnet classes. In *Proceedings of the 22nd International Conference on Computational Linguistics*, volume 1. Association for Computational Linguistics.

Satanjeev Banerjee and Ted Pedersen. 2010. An Adapted Lesk Algorithm for Word Sense Disambiguation Using WordNet. In *Computational Linguistics and Intelligent Text Processing*, volume 2276 of *Lecture Notes in Computer Science*, pages 117–171. Springer Verlag.

Paul Buitelaar. 1998. *CoreLex: Systematic Polysemy and Underspecification*. Ph.D. thesis, Computer Science, Brandeis University, February.

Hoa T. Dang and Martha Palmer. 2005. The Role of Semantic Roles in Disambiguating Verb Senses. In *Proceedings of the 43rd Annual Meeting of the Association f or Computational Linguistics*, pages 42–49, Ann Arbor MI.

Marie-Catherine de Marneffe and Christopher D. Manning. 2008. Stanford typed dependencies manual. Technical report, Stanford University.

D. Dligach and M. Palmer. 2008. Novel semantic features for verb sense disambiguation. In *Proceedings of the 46th Annual Meeting of the Association for Computational Linguistics: Short Papers*, pages 29–32. Association for Computational Linguistics.

Roxana Girju, Dan Roth, and Mark Sammons. 2005. Token-level disambiguation of verbnet classes. In *Proceedings of The Interdisciplinary Workshop on Verb Features and Verb Classes*.

Mark Hall, Eibe Frank, Geoffrey Holmes, Bernhard Pfahringer, Peter Reutemann, and Ian H. Witten. 2009. The WEKA data mining software: An update. *ACM SIGKDD Explorations Newsletter*, 11(1):10–18.

Karin Kipper, Anna Korhonen, Neville Ryant, and Martha Palmer. 2008. A Large-Scale Classification of English Verbs. *Journal of Language Resources and Evaluation*, 42(1):21–40.

Mirella Lapata and Chris Brew. 2004. Verb class disambiguation using informative priors. *Computational Linguistics*, 30(1):45–73.

Carolyn Penstein Rosé and Alon Lavie. 2000. Balancing robustness and efficiency in unification-augmented context-free parsers for large practical applications. In J.-C. Junqua and G. van Noord, editors, *Robustness in Language and Speech Technology*. Kluwer Academic Press.

Rajen Subba and Barbara Di Eugenio. 2009. An effective discourse parser that uses rich linguistic information. In *North American Chapter of the Association for Computational Linguistics - Human Language Technologies (NAACL HLT)*, Boulder, CO, June.

Rajen Subba, Barbara Di Eugenio, and Elena Terenzi. 2006. Building lexical resources for PrincPar, a large coverage parser that generates principled semantic representations. In *LREC06, the fifth International Conference on Language Resources and Evaluation*, pages 327–332, Genova, Italy, May.

David Yarowsky. 1993. One sense per collocation. In *Proceedings of the workshop on Human Language Technology*, pages 266–271. Association for Computational Linguistics.