RICH ENTITY TYPE RECOGNITION IN TEXT

Extended Abstract

Rishav Bhowmick
rishavb@cmu.edu

Advisors

Michael Heilman	Kemal Oflazer	Noah A. Smith
mheilman@cs.cmu.edu	ko@cs.cmu.edu	nasmith@cs.cmu.edu

INTRODUCTION

Many applications in natural language processing (NLP) include summarization of text, classifying documents or automatic answering of questions posed in natural language. Each of these applications require entity type recognition in the text as a pre-processing step. Here, entity refers to concrete and abstract objects identified by proper and common nouns. Entity recognition focuses on detecting instances of types like "person", "location", "organization", "time", "communication", "event", "food", "plant", "animal" and so on. For example, an Entity Recognizer would take the following sentence as input:

George Washington was the first President of the United States of America.

and output:

<noun.person> *George* Washington **</noun.person>** *was the first* **<noun.person>** *President* **</noun.person>** *of* **<noun.location>** *the United States of America* **</noun.Location>**.

Humans generally have no problems finding out what type a noun belongs to. For example, in the example above, a human would look at "President" and know that it is of type *Person*. He/she would also know a location or organization can have President. Additional knowledge about the country, makes him/her think it's a location. Finally, *George Washington* has to be a person as a president can only be a human[1]. The way a human figures out the entity types could be summarized in the following points:

- Recalling what entity type a word most likely belongs to
- Looking at the context the word appears in.
- Looking at features like word capitalization, any punctuation marks. For example, the use of upper-case letter after punctuation marks like period or question mark does not ascertain the fact that the first word of the sentence is a proper noun.

Our task is to use machine learning techniques to train a system that can do entity type recognition with a performance comparable to a human. This problem is hard for a variety of reasons. In general, it is not possible to list all possible instances of a single entity type and feed it to the machine. The lack of large annotated data corpus for training is another major impediment. Due to these reasons, the entity recognizers out there are not very accurate (F-scores (see section EVALUATION METRICS) in 70's and 80's [1]. The obvious task then is to improve the performance of existing machine tagging systems. This

[1] unless it is a line out of a fantasy novel, where an animal (other than a human) presides.

would be achieved by looking for features (new and old) that affect the performance of the tagger the most. Additionally, finding out how much of training data is needed can help solve the problem of lack of large annotated training data corpus.

The goal of this Senior Thesis project is to improve the performance of an existing entity recognizer by figuring out which syntactic and semantic features can boost the performance and whether large training data sets are necessary or not. The outcome of this project will be a step forward in making an enhanced entity recognizer which in turn will benefit other NLP problems stated earlier.

SUPERSENSES

The entity-type tag set we use in this research project contains types that we call *supersense*[1-3]. There are 26 **broad semantic classes**, beyond the usual entity types of *Person, Location* and *Organization* in earlier Named Entity Recognition (NER)[2], used in this project. These are the labels used by lexicographers who developed Wordnet [4], a broad-coverage machine readable database which has proper and common nouns, verbs, adjectives and adverbs interlinked via synonym, antonym, hypernym, hyponym and variety of other semantic relations. **Table 1**[1] shows the supersense labels for nouns and verbs. Not only does the use of this tag set suggest an extended notion of named entity, but it also provides additional training data[3] while tagging words with supersenses. Hence this particular process of recognition is called *supersense tagging*.

NOUNS			
SUPERSENSE	NOUNS DENOTING	SUPERSENSE	NOUNS DENOTING
act	acts or actions	object	natural objects (not man-made)
animal	animals	quantity	quantities and units of measure
artifact	man-made objects	phenomenon	natural phenomena
attribute	attributes of people and objects	plant	plants
body	body parts	possession	possession and transfer of possession
cognition	cognitive processes and contents	process	natural processes
communication	communicative processes and contents	person	people
event	natural events	relation	relations between people or things or ideas
feeling	feelings and emotions	shape	two and three dimensional shapes
food	foods and drinks	state	stable states of affairs
group	groupings of people or objects	substance	substances
location	spatial position	time	time and temporal relations
motive	goals	Tops	abstract terms for unique beginners
VERBS			
SUPERSENSE	VERBS OF	SUPERSENSE	VERBS OF
body	grooming, dressing and bodily care	emotion	feeling
change	size, temperature change, intensifying	motion	walking, flying, swimming
cognition	thinking, judging, analyzing, doubting	perception	seeing, hearing, feeling
communication	telling, asking, ordering, singing	possession	buying, selling, owning
competition	fighting, athletic activities	social	political and social activities and events
consumption	eating and drinking	stative	being, having, spatial relations
contact	touching, hitting, tying, digging	weather	raining, snowing, thawing, thundering
creation	sewing, baking, painting, performing		

TABLE 1 NOUNS AND VERB SUPERSENSE LABELS, AND SHORT DESCRIPTION

[2] See next section RELATED WORK for earlier NER works. Also, *Named Entity* here refers to proper nouns only.

[3] Wordnet is used to lemmatize a word (find the root of the word, for e.g. "ran"->"run") and find the most frequent sense of the word.

RELATED WORK

The entity recognizer, whose performance we are trying to improve, is the Supersense Tagger (SST) [1]. The tagger performs sequence tagging with a perceptron trained Hidden Markov Model (HMM). The performance of perceptron-trained HMMs is very competitive and comparable in performance to that of Conditional Random Field models [1],[5]. Addition of new features such as word/phrase clusters in a more restricted task of NER has shown considerable improvement in performance in the system [6]. The use of word/phrase clusters alleviates the problem of lack of annotated data. So once word clusters with unlabeled data are created, they can be used as features in a supervised training setting. Hence, even when a word is not found in the training data, it may still benefit from the cluster-based features as long as the word belongs to the same cluster with some word in the labeled data.

The baseline tagger for this project is a reimplementation[4] of the SST. It uses the same feature set as that of the SST to tag words which include proper and common nouns and verbs.

EVALUATION METRICS

The following evaluations metrics are used to evaluate the performance of our tagger.

PRECISION:
Precision measures to how many of the entity types the tool recognized are actually correct.

$$precision = \frac{\sum_i number\ of\ correctly\ tagged\ words\ or\ phrases}{\sum_i number\ of\ tagged\ words\ or\ phrases\ by\ the\ tagger}$$

EQUATION 1 PRECISION

In **Equation 1**Equation 1, the numerator and denominator sums over all entity types i. The final output is the overall precision.

RECALL:
Recall measures to how many of the entity types the tool recognized correctly.

$$recall = \frac{\sum_i number\ of\ correctly\ tagged\ words\ or\ phrases}{\sum_i number\ of\ labeled\ words\ or\ phrases\ in\ the\ actual\ test\ data}$$

EQUATION 2 RECALL

In **Equation 2**, the numerator and denominator sums over all entity types i. The final output is the overall recall.

F-SCORE (F1):
F-score (F1) is simply the geometric mean of precision and recall, and combines the two scores. F-score and F1 will be used interchangeably throughout this document.

$$F1 = \frac{2 \times precision \times recall}{precision + recall}$$

EQUATION 3 F-SCORE (F1)

[4]by Michael Heilman (LTI, CMU)

APPROACH

Our approach towards improving the performance of the SST involves two pre-processing steps. As we mentioned earlier, we would like to know how different sizes of training data affects the performance of the tagger. So our first task is to train the system using different size training data and evaluating the trained model with development data.

The next step involves experimenting with existing features and gauging how they affect the performance of the tagger the most. We devised a series of experiments which involves removing one feature at a time and evaluating the tagger output. This task is termed as *feature ablation.* If some features affect the f-score by +/-2 points, we mark them for future experimentation. As for the other features, we group them and check if they collectively affect the f-score. Some of the basic features from SST include most frequent sense (from Wordnet), Part-of-Speech (POS) tags, word shape (upper-case or lower-case, upper-case after period and so on) and label of preceding words.

Context is an essential feature while tagging words. As shown in the example in the INTRODUCTION section, while tagging *George Washington*, the knowledge about a President being of type person helps with tagging *George Washington* as person. The baseline tagger only looks at +/- 2 words around the current word being tagged. We performed additional experiments by reducing the context to not looking at any words (removing the existing context features) and then increasing the context to +/- 4 words (adding new context features).

The results of feature ablation study suggested the addition of word cluster features. In order to increase the overall F1 for the tagger, individual F1 scores need to be high. Supersenses like *noun.process (*F1 = 48.62%*), noun.shape (*F1 = 50.00%*), noun.relation(*F1 = 58.21%*)* and few more brought down the overall F1. Looking deeper, the recall for most of these supersenses were very low (44.53% - 59.99%). In other words, the tagger failed to label these words/phrases. This could be due to the fact that these words were not in the training data or even Wordnet, which in turn led to failure of most-frequent sense retrieval. However, the use of word clusters can solve this problem.

The way this works is as follows. Whenever a word is being tagged, word cluster information (the cluster(s) the word belongs to) is extracted from an already provided word cluster input. This cluster (or maybe clusters) has (have) other words which were tagged already from the training data. Weights are attached for each of these labels, making them possible candidates for tagging. This helps in tagging words which are not seen before, even by Wordnet.

EXPERIMENTS AND RESULTS

SETUP

We tested our tagger on the Semcor corpora[7] , containing text from the Brown Corpus that is syntactically and semantically tagged. The Semcor data was split into 3 parts: Training, Development and Testing. The three parts were created by randomly selecting the articles (which came with its sentences). The size of the three parts were as follows:

Number of sentences in training data: 11,973
Number of tokens in training data: 248,581

Number of sentences in development data: 4113
Number of tokens in development data: 92,924

Number of sentences in testing data: 4052
Number of tokens in testing data: 93,269

EXPERIMENT 1

Figure 1 shows the results for the varying f1 score with respect to the amount of training data used. The training data was split into 5%, 10%,...,95%.

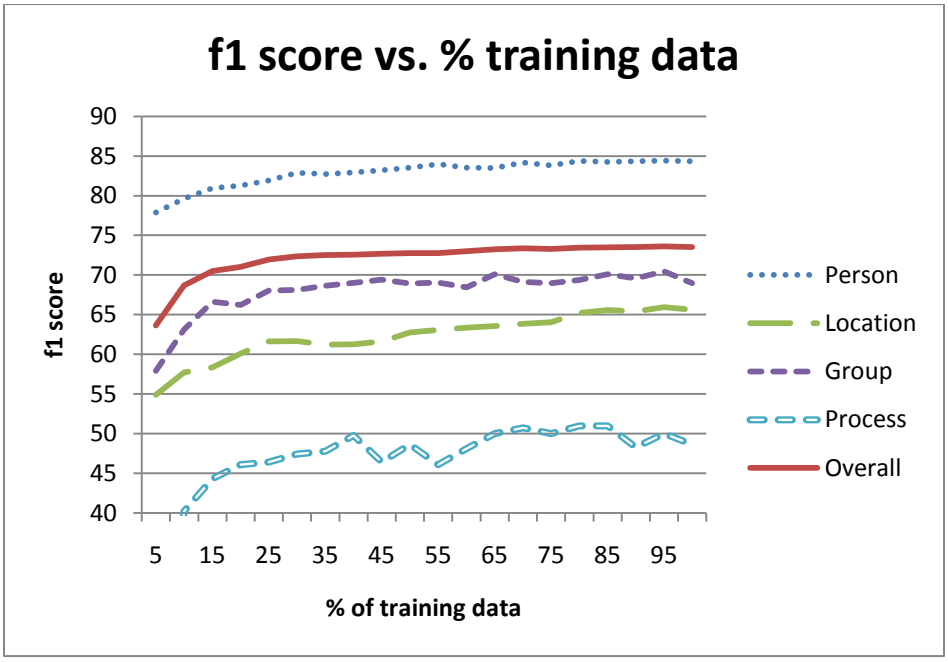

FIGURE 1 F1 VS % TRAINING DATA

After about 1/3rd of the training data is used, the overall f-score does not increase drastically.

Figure 2 Figure 2 shows the f-score for some of the supersenses when 90% of the training data was used.

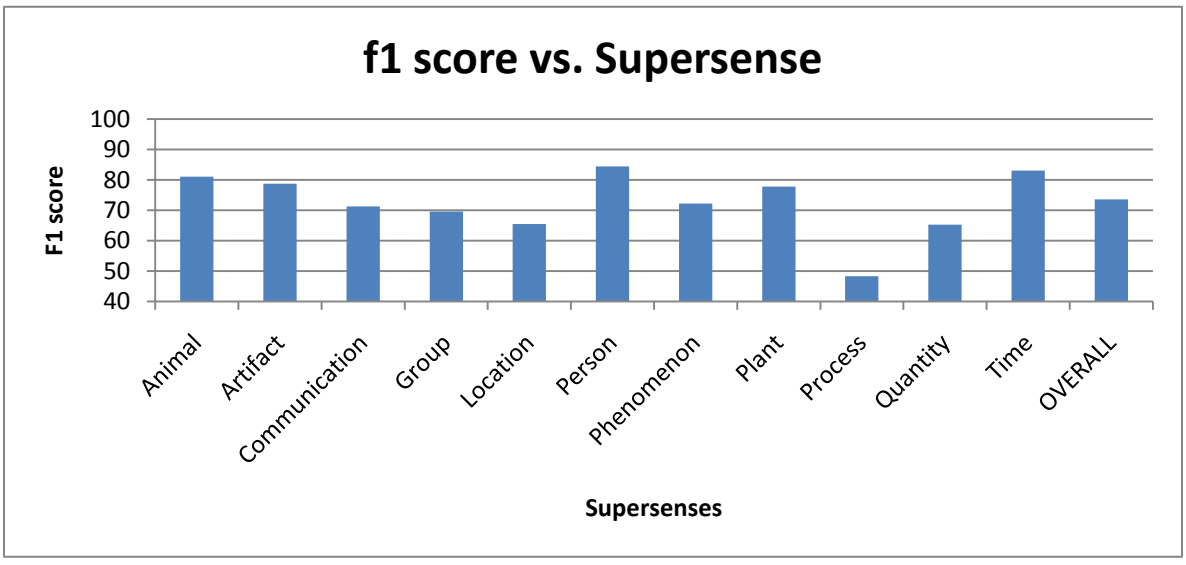

FIGURE 2 F1 VS SUPERSENSES WHEN 90% OF TRAINING DATA WAS USED

EXPERIMENT 2

Feature ablation results. **Table 2**Table 2 shows for each feature removed, the resulting F-score. The baseline F-score is 73.542%

Feature	F-score
First-sense (Most frequent sense)	57.105%
Part-of-Speech	73.110%
Word Shape	73.524%
Label of previous word	73.360%

TABLE 2 FEATURE REMOVED AND RESULTING F-SCORE

EXPERIMENT 3

Experiments on context size are tabulated in table :

Context	F-score
No words	70.187%
Current word	70.943%
Current word +/- 1 word	73.338%
Current word +/- 2 words	73.542% (baseline score)
Current word +/- 3 words	73.501%
Current word +/- 4 words	73.269%

TABLE 3 CONTEXT SIZE AND F-SCORE

Lesser context or more context has reduced the F-score. This is most probably because the further away the word is (for larger context) the less likely there will be any semantic relation.

EXPERIMENT 4

Initial experiments with inclusion of word clusters led to the following results in **Table 4**:

Word Cluster Feature	F-score
Word cluster	73.733%
Word cluster + First Sense	73.823%

TABLE 4 WORD CLUSTER FEATURE AND F-SCORE

CONCLUSION

In this work, we highlighted the importance of syntactic, contextual and word cluster features affect the performance of a system for tagging words with high level sense information. Additionally, we have demonstrated that lack of large annotated data is not a major issue. Nevertheless, the size of training data would be important if the features were more specific[5]. Feature ablation methods like the ones described in the experiments help find out which features are important and hereby suggest areas to work on (e.g.: new features to extend or add). In this project, addition of word cluster features and usage of large context were the outcomes of feature ablation.

[5] Word Sense Disambiguation using un-supervised or semi-supervised learning.[9]

REFERENCE

[1] M. Ciaramita and Y. Altun, "Broad-coverage sense disambiguation and information extraction with a supersense sequence tagger," *Proceedings of the 2006 Conference on Empirical Methods in Natural Language Processing(EMNLP)*, 2006, pp. 594–602.

[2] M. Ciaramita and M. Johnson, "Supersense tagging of unknown nouns in wordnet," *Proceedings of EMNLP*, 2003.

[3] J.R. Curran, "Supersense tagging of unknown nouns using semantic similarity," *Proceedings of the 43rd Annual Meeting of Association for Computational Linguistics(ACL)*, 2005, p. 33.

[4] C. Fellbaum and others, *WordNet: An electronic lexical database*, MIT press Cambridge, MA, 1998.

[5] M. Collins, "Discriminative training methods for hidden markov models: Theory and experiments with perceptron algorithms," *Proceedings of EMNLP*, 2002.

[6] D. Lin and X. Wu, "Phrase clustering for discriminative learning," *Proceedings of the Joint Conference of the 47th Annual Meeting of the ACL and the 4th International Joint Conference on Natural Language Processing of the AFNLP: Volume 2-Volume 2*, 2009, pp. 1030–1038.

[7] G.A. Miller, C. Leacock, R. Tengi, and R.T. Bunker, "A semantic concordance," *Proceedings of the 3rd DARPA workshop on Human Language Technology*, 1993, pp. 303–308.

[8] W.N. Francis and H. Kucera, *Computational analysis of present-day American English*, Brown University Press Providence, 1967.

[9] D. Yarowsky, "Unsupervised word sense disambiguation rivaling supervised methods," *Proceedings of the 33rd annual meeting on ACL*, 1995, pp. 189–196.