Disambiguating Hindi Named Entities: Ensemble learning from multiple learners

(With special emphasis on the role of the Parsarg and Verb Classes)

Shashank Srivastava(Y5827429)

Guided by: Dr. RMK Sinha
Dept of Computer Science and Engineering
IIT Kanpur

April 30, 2009

Contents

1 Introduction & Overview **3**

2 Sources of Data **4**

3 Dataset Pruning and Preparation **4**

4 Approach **5**

4.1 Naive Statistical Approach 5

4.2 POS tagging 6

4.3 Mining Co-occurrence data 7

4.4 Final Feature Set 7

5 Design of Classification Module **7**

5.1 Validation Strategy 8

5.2 Classifiers, and Ensembling by Voting 8

5.3 Cost Sensitive Learning 8

6 Results and Observations **9**

6.1 Using Orthographic Features only 9

6.2 Using POS tags 10

6.3 Using Parsarg annotation with POS tags 11

6.4 Role of the verb 12

7 Conclusion **12**

References

ABSTRACT

We address the problem of disambiguating Hindi names, using knowledge infusion from multiple sources of evidence. The role of parsargs and POS tags is seen to be critical in Hindi NER. Other sources such as semantic word classes and co-occurrence data are also studied as possible discriminatory evidence. Furthermore, an ensemble approach combining votes from two statistical methods (Decision trees and SVM's), and using Cost sensitive learning, is seen to significantly improve both accuracy and recall. A classification accuracy of 77% , and best F-measure of 0.56 is achieved on the provided dataset using the proposed hybrid approach.

1 Introduction & Overview

Named entity recognition is a well studied problem in formal text. The problem has been addressed for English and many European languages with reasonable accuracy(Sassano, Sang and De Meulder), and decent competency has also been achieved in the cases of East Asian Languages. [5][8][9] However, NER for Indian languages is yet largely unstudied. Recently, some attention has been paid in this direction, and a NER task for 5 Indian languages was a part for a workshop at IJCNLP 2008. [4]

NER for Indian languages (and especially Hindi) is a lot more challenging due to mainly two linguistic intricacies, specific to Hindi and Indian languages:

- Indian Languages lack capitalization, the single most important heuristic for NER for European Languages.
- A fair number of common Hindi named entities also confer interpretations as other linguistic abstractions. For example, names like Aarti, Vivek, Kamal, Aadarsh would also be found in any Hindi lexicon.

The present study focuses on the disambiguation of such occurrences of possible Hindi names. The task at hand is therefore to identify specificities of the Hindi language, and accordingly construct features, which when used for training statistical models give better accuracy. The problem appears non-trivial, especially as discrimination of such constructs becomes intractable without context. Intuitively it appears that the task might be beyond purely statistical approaches, Especially due to relatively limited data available for Hindi. Also, the task seems too complex for a manual handcrafting of rules based on linguistic knowledge only. Thus, it would be hard to build a purely rule-based, or purely statistical classifier.

However, we show that by a careful feature selection, and infusion of knowledge from several sources, a hybrid module can achieve a high degree of accuracy. In the process of feature selection, we observe the relative importance of lexical, semantic and syntactic knowledge in this task. We also lay an emphasis of parsargs, which are a construct specific to Indian languages, and intuitively seem to have a lot of discriminatory potential. It is hypothesized, based on linguistic considerations and familiarity with the language, that parsargs, as well as the verb class could have differentiating role between names and non-names.

The hypothesis is tested, and strong evidence is found especially for the role of the associated Parsargs and POS tags of adjoining words.

We also make a comparative study of the performance of two classical learning algorithms Widely used in NLP tasks, and use a novel voting mechanism based on classification confidence (that has a statistical validity) to combine the three classifiers along with preliminary handcrafted rules, into an ensemble that gives better performance on a ten -fold cross validation task, and a superior recall.

2 Sources of Data

Based on available data for training and validation, and eclectic resources for Hindi NLP elsewhere, it was decided to use several knowledge infusions, each giving independent classificatory evidence, to build a classificatory module. This is a statistical paradigm known as 'boosting', which is based on the idea that several weak learners can create a single stronger learner.

In the study, following types of data sources were used:

1. Orthographic features (word length, punctuation, first word, last word mined from dataset)
2. Lexical database (available from Hindi Wordnet)
3. Semantic tags (pre-annotated)
4. Parsarg identification (pre-annotated)
5. POS tagging (from IIIT Shallow parser's source-code, and Language Lab data)
6. Lexicon of Hindi names
7. Co-occurrence data (possibly, proof of negative evidence)
8. Knowledge of Verb classes (pre-annotated, later pruned)

3 Dataset Pruning and Preparation

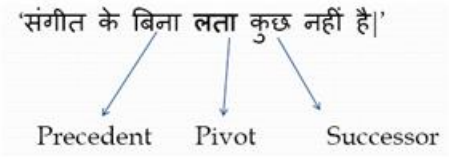
The dataset provided consisted of a text with high occurrences of ambiguous name-words. Some features for each of these name-occurrences were already annotated at Language Lab, IIT Kanpur, and focused on verb and parsarg associations in a window of size three around each data-point. There was a significant number of errata in the provided data, in terms of replication of data, missing annotation, and inconsistency in data format at places. All such instances were pruned, to get a working dataset containing 1265 data-points, with the following break up by class:

Names	347
Non-names	918

Thus we see there is a considerable skew in the data, which might hinder any learning algorithms.

4 Approach

The original dataset was available in Romanized script (IITK Roman) and ARFF extension, with feature vectors in the following format:



Feature Vector:

(BINA, blank, blank, <#LATA>, [climbers], blank, <KUCHH>, [blank], blank, yes)

Nine attributes are available for each of the 1265 name occurrences.

Attribute	*Distinct Values*
Preceding word (P)	692
Semantic tag of P	77
Parsarg association of P	10
The Name occurrence (NO)	200
Semantic tag of NO	53
Parsarg association of NO	10
Next word (N)	711
Semantic tag of N	81
Class of verb (23 classes)	23
Annotation as NE/Not NE	

4.1 Naive Statistical Approach

The dataset, in its native form was not amenable to statistical learning as the feature vectors were too specific (eg. Preceding and next word). Also, even using word classes (semantic tags) instead of single words as level of granularity, the semantic tags were too specific, and their number was too high for learning to be possible with the given size of corpus. Thus a more generic representation underlying the data has to be identified in order to obtain meaningful rules. In its native form, the choice of features would have led to overtraining; and exceedingly complex and unilluminating rules, at best. Thus, the given dataset, in isolation, was insufficient for any meaningful learning of rules, and information mining. Additional linguistics were infused into the system from other data sources.

4.2 POS tagging

POS tags are important features in any NLP task. However, POS data for Hindi is not readily available. Online data for the same was found in UTF format independently on the IJCNLP 2008 site, and LTRC, IIIT Hyderabad. However, conversion to Roman script could not be effected from an online tool for the purpose. The source code of the IIIT Shallow parser for Hindi [3] was also found to contain a small Hindi Lexicon of 7000 common words with their common POS tags. Usable POS data could also not be procured from the Language lab, however some raw data for training of a POS tagger was procured from the lab, consisting of 11897 Hindi sentences, and suggested POS tags. This data was found to contain some errors and omissions. After pruning, the data was aligned to find POS tags for 2,79,113 words (with repetitions).

As a large fraction of words had more than one POS tags in different contexts, the tag with the highest frequency was used for each word.

```
se SYM 2 NN 1 RP 1 PSP 680 PRP 2
unheM PRP 84
koI RP 2 NNP 1 PRP 91 QF 6 DEM 5
lABa NN 16
huA VM 70 VAUX 18
xUsare QC 5 NN 2 JJ 11 PRP 4 QO 14
upacAra NN 1
kiye VM 36
gaye VM 12 VAUX 105
unameM PRP 19
BI RP 410 PSP 5
saPalawA NN 13
milI VM 8
aMwa NN 2
meM SYM 2 NN 2 PSP 1225 PRP 7
gAMXIjI NN 6 NNP 285 PRP 1
```

Sample of multiple annotations of POS with corresponding frequencies. The most frequent tag was used in training.

Since the POS data from the IIIT Shallow Hindi parser was more reliable, every word in the provided dataset was first matched in the 7000 words from the IIIT parser. However, it was seen that only about 60% of the words get matched this way, leaving considerable voids in the feature vectors for training. If still unmatched, therefore the data was matched in the larger POS corpus synthesized from raw data. This significantly improved the tagging ratio, and 87% of the words could be tagged with this sequential strategy.

Thus, POS tags could be added as attributes to the feature vectors in the provided dataset.

4.3 Mining Co-occurrence data

If a NO (name occurrence) occurs as a non-name, it often is a representative word of a contextual topic. For example, 'Aarti' when not a name, mostly belongs to a topic of devotion or worship. In this sense, if other words belonging to the same word topic/class are in the vicinity of a NO, it is a negative evidence that favours a non-name classification. We tried to incorporate this paradigm as one of multiple learners:

$$P(Name|ContexualWord) = P(ContextualWord|Name) * \left\{\frac{P(Name)}{P(ContextualWord)}\right\}$$

A text file (720 sentences, many replications) for mining co-occurrence (as a source of negative confidence) was provided from the Language lab. If a co-occurrence of the NO with a word (of length more than 6 to avoid stop words) in the training sentence was also found in the pre-mentioned corpus, it was noted as a binary feature. However, the co-occurrence data was not found to contain instances of most of the 200 NO's, and the attribute vector so obtained was thus both sparse, and unreliable. Hence it was not used for training.

However, it is very reasonable to believe that a larger and more representative corpus with more name -occurrences could further assist classification.

4.4 Final Feature Set

In addition to the 9 attributes in the pruned dataset, features such as word-lengths, position in sentence, and POS tags were added; (and a binary feature from co-occurence data removed) to yield a final feature vector of size 14.

5 Design of Classification Module

Statistical learning algorithms tried for NER tasks have include the following:

- Decision Trees (Isozaki, 2001) [9]
- Maximum Entropy (Borthwick, 1998) [11]
- Support Vector Machines or SVM (Takeuchi and Collier, 2002) [10]
- Conditional Random Fields or CRF (Settles, 2004, Li and McCallum, 2004) [12]
- Hidden Markov Models or HMM (Zhou and Su, 2001) [15]

The last two algorithms work for inherently sequential data. Since we focus only on disambiguating ambiguous NO's, and are not interested in generic recognition of other types of names entities, we compare the training and validation of our feature vectors using Decision trees and SVM's.

5.1 Validation Strategy

10-fold cross validation strategy is adopted for validation. 10 rounds of training and validation are performed, involving partitioning the dataset into exclusive subsets. Each round consists of using one partition as test, and the remaining folds as training data. The average accuracy of predictions over the 10 folds is reported as the accuracy of the classifier.

5.2 Classifiers, and Ensembling by Voting

In our study, we use Decision Trees and SVM's as statistical classifiers.

- **Decision Trees:**We use the popular J48 Decision tree algorithm that uses Information gain to split nodes in a greedy manner. An implementation of this is available in the freely available WEKA toolkit for machine learning. [6]
- **Support Vector Machines [14]:**Proposed by Vapnik, they are a class of large margin classifiers that map points to a higher dimensional hyperspace, and find a separating surface using Kernels. Popular implementations include LibSVM, SVM-light, RapidMiner's SVM and WEKA's implementation based on Sequential Minimum Optimization.

Other classifiers such as Neural Networks were also tried for predictive accuracy, but they were found not to do well with the designated task. Furthermore, neural networks don't have a transparent working as is the case with DT's and SVM's which give interpretable rules.

Using the Boosting paradigm as explained before, learning from the two classifiers could be combined provided that the two are (at least) 'weak' classifiers. A weak classifier is one that has a predictive accuracy better than a random guess over many samples. Here we combine the classifiers by taking a vote, i.e. if the confidence of prediction of a classifier is low, while the second has a higher confidence in assignment to one of the two classes, the prediction of the second classifier is given priority. In other words, the confidence of prediction for both classifiers are summed, and the class with a higher cumulative sum is assigned. All classificatory tasks in the study were reported for all three classifiers.

5.3 Cost Sensitive Learning

Secondly, and significantly, the notion of cost sensitive learning was used for training. This is especially relevant in context of NER, which usually suffers from very low recall values. In a normal task, both classes are treated as equal, and hence the classifier tries to minimize the number of misclassifications. In cost sensitive learning, the relevant class is assigned a higher cost on misclassification. Hence, here the misclassification cost gets minimized instead, which can help improve recall, at some tradeoff with precision. In all experiments, costs in ratios 5:4, 3:2 and 2:1 were tried, and the best scores were reported.

This strategy gives us much higher recall rates, and hence higher F measures (at some cost to precision).

6 Results and Observations

6.1 Using Orthographic Features only

Using SVM:

813	105
295	52

Accuracy = 68.4%
Recall = 0.151
Precision = 0.331
F = 0.206

Using Decision Tree:

750	168
238	109

Accuracy = 67.9%
Recall = 0.314
Precision = 0.394
F = 0.349

Using Voting:

759	174
262	90

Accuracy = 67.7%
Recall = 0.256
Precision = 0.338
F = 0.291

In this case, SVM's perform very poorly since we have a very minimal feature space, that doesnt afford mapping to higher spaces. The Voting technique, therefore, does not assist the classification task either.

6.2 Using POS tags

Using SVM:

772	146
225	122

Accuracy = 70.7%
Recall = 0.352
Precision = 0.455
F = 0.397

Using Decision Tree:

632	286
144	203

Accuracy = 66.0%
Recall = 0.585
Precision = 0.415
F = 0.486

Using Voting:

654	264
133	214

Accuracy = 68.6%
Recall = 0.616
Precision = 0.448
F = 0.519

We see that the addition of POS tags to the feature vector significantly improves performance, especially recall for the rare class. Also, the Ensemble technique provides a better classifier than either Decision Trees, or SVMs in isolation.

6.3 Using Parsarg annotation with POS tags

Using SVM:

777	141
166	181

Accuracy = 75.7%
Recall = 0.522
Precision = 0.562
F = 0.541

Using Decision Tree:

664	254
160	187

Accuracy = 67.3%
Recall = 0.540
Precision = 0.424
F = 0.475

Using Voting:

778	140
158	189

Accuracy = 76.5%
Recall = 0.544
Precision = 0.574
F = 0.558

There is a massive improvement in classification of performance on incorporation of POS tags into the feature vector. The Ensemble technique again gives the best performance, with an F measure of 0.56. We also see that this is the best accuracy we get with different combinations of features. Hence, Parsarg annotations along with POS tags, orthographic features, and the Verb class form the optimum feature vector.

Note that due to the low granularity of semantic tags, they are unhelpful in classification in this case, and dont form a part of our feature vector. Also, the word-cooccurence vector is pruned, because it didnt add much to the performance, being exceedingly sparse due to inadequacy of used Co-occurrence data.

6.4 Role of the verb

In the optimal feature set, and using the Voting classifier, Verb class is removed as a feature to illustrate its relative importance. A considerable drop is seen in both Recall and F measure. Verb classes are seen to be still more important when other evidence is sparse. (eg. no POS tags, or Parsarg markings).

Without using Verb Class:

778	140
185	162

Accuracy = 74.3%
Recall = 0.467
Precision = 0.536
F = 0.499

7 Conclusion

The contribution of the study is in terms of identification of the primary role of the parsarg (and secondly, POS tags) for the task of name disambiguation. Statistically, the paradigm of ensemble learning is seen to show promise in NLP tasks. Also, it is seen that cost sensitive training in statistical algorithms can partially alleviate the problem of low recall values.

The task of mining data-cooccurence didnt yield much dividend in this case due to inadequate data, but needs to be explored further using a larger, more representative corpus.

References

[1] Sujan Kumar Saha, Sanjay Chatterji, Sandipan Dandapat, Sudeshna Sarkar, and Pabitra Mitra. 2008. A hybrid named entity recognition system for south and south east asian languages. In Proceedings of the IJCNLP- 08 Workshop on NER for South East Asian Languages, pages 1724, Hyderabad.

[2] Sudeshna Sarkar,Sujan Saha and Prthasarthi Ghosh,"Named Entity Recognition for Hindi",Microsoft Research India Summer School talk, p.21-30, May2007.

[3] "Shallow Parser for Hindi", LTRC, IIIT Hyderabad. http://ltrc.iiit.net/showfile.php?filename=downloads/

[4] IJCNLP 2008 workshop for NER in Indian languages, "http://ltrc.iiit.net/ner-ssea-08/index.cgi?topic=5"

[5] Manabu Sassano and Takehito Utsuro. 2000. Named entity chunking techniques in supervised learning for japanese named entity recognition. In Proceedings of the 18th conference onComputational linguistics, pages 705711,Morristown, NJ, USA. Association for Computational Linguistics.

[6] WEKA toolkit for learning: www.cs.waikato.ac.nz/ml/weka/

[7] RapidMiner: http://rapid-i.com/content/blogcategory/38/69/

[8] Erik F. Tjong Kim Sang. 2002. Introduction to the CoNLL 2002 shared task: Language Independent Named entity recognition. In Proceedings of CoNLL-2002,pages 155158. Taipei, Taiwan.

[9] Erik F. Tjong Kim Sang and Fien De Meulder, Introduction to the CoNLL-2003 Shared Task: Language-Independent Named Entity Recognition. In: Proceedings of CoNLL-2003, Edmonton, Canada, 2003, pp. 142-147.

[10] Hideki Isozaki. 2001. Japanese named entity recognition based on a simple rule generator and decision tree learning. In Meeting of the Association for Computational Linguistics, pages 306313.

[11] K. Takeuchi and N. Collier. 2002. Use of Support Vector Machines in extended named entity recognition. In Proceedings of the sixth Conference on Natural Language Learning (CoNLL-2002).

[12] A. Borthwick, J. Sterling, E. Agichtein, and R.Grishman. 1998. Exploiting diverse knowledge sources via maximum entropy in named entity recognition. Proceedings of the Sixth Workshop on Very Large Corpora, pages 152160.

[13] B. Settles. 2004. Biomedical Named Entity Recognition Using Conditional Random Fields and Rich Feature Sets.

[14] Li Wei and McCallum Andrew. 2004. Rapid Development of Hindi Named Entity Recognition using Conditional Random Fields and Feature Induction. In ACM Transactions on Computational Logic.

[15] Corinna Cortes and Vladimir Vapnik. Support Vector Network. Machine Learning, 20:273-297,1995.

[16] G.D. Zhou and J. Su. 2001. Named entity recognition using an HMM-based chunk tagger. Proceedings of the 40th AnnualMeeting on Association for Computational Linguistics, pages 473480.

[17] Hindi Wordnet, Source:"http://www.clt.iitb.ac.in/wordnet/webhwn/" (for lexicon, not used eventually)