A Hybrid Feature Set based Maximum Entropy Hindi Named Entity Recognition

Sujan Kumar Saha
Indian Institute of Technology
Kharagpur, West Bengal
India - 721302
sujan.kr.saha@gmail.com

Sudeshna Sarkar
Indian Institute of Technology
Kharagpur, West Bengal
India - 721302
shudeshna@gmail.com

Pabitra Mitra
Indian Institute of Technology
Kharagpur, West Bengal
India - 721302
pabitra@gmail.com

Abstract

We describe our effort in developing a Named Entity Recognition (NER) system for Hindi using Maximum Entropy (MaxEnt) approach. We developed a NER annotated corpora for the purpose. We have tried to identify the most relevant features for Hindi NER task to enable us to develop an efficient NER from the limited corpora developed. Apart from the orthographic and collocation features, we have experimented on the efficiency of using gazetteer lists as features. We also worked on semi-automatic induction of context patterns and experimented with using these as features of the MaxEnt method. We have evaluated the performance of the system against a blind test set having 4 classes - Person, Organization, Location and Date. Our system achieved a f-value of 81.52%.

1 Introduction

Named Entity Recognition involves locating and classifying the names in text. NER is an important task, having applications in Information Extraction (IE), question answering, machine translation and in most other NLP applications.

NER systems have been developed for English and few other languages with high accuracies. These systems take advantage of large amount of Named Entity (NE) annotated corpora and other NER resources. However when we started working on a NER system for Hindi, we did not have any NER annotated corpora for Hindi, neither did we have access to any comprehensive gazetteer list.

In this work we have identified suitable features for the Hindi NER task. Orthography features, the suffix and prefix information, as well as information about the sorrounding words and their tags are used to develop a Maximum Entropy (MaxEnt) based Hindi NER system. Additionally, we have acquired gazetteer lists for Hindi and used these gazetteers in the Maximum Entropy (MaxEnt) based Hindi NER system. We also worked on semi-automatically learning of context pattern for identifying names. These context pattern rules have been integrated into the MaxEnt based NER system, leading to a high accuracy.

The paper is organized as follows. A brief survey of different techniques used for the NER task in different languages and domains are presented in Section 2. The MaxEnt based NER system is described in Section 3. Various features used in NER are then discussed. Next we present the experimental results and related discussions. Finally Section 8 concludes the paper.

2 Previous Work

A variety of techniques has been used for NER. The two major approaches to NER are:

1. Linguistic approaches.
2. Machine Learning based approaches.

The linguistic approaches typically use rules manually written by linguists. There are several rule-based NER systems, containing mainly lexicalized grammar, gazetteer lists, and list of trigger words, which are capable of providing 88%-92% f-measure accuracy for English (Grishman, 1995; McDonald, 1996; Wakao et al., 1996).

The main disadvantages of these rule-based techniques are that these require huge experience and grammatical knowledge of the particular language or domain and these systems are not transferable to other languages or domains.

Machine Learning (ML) based techniques for NER make use of a large amount of NE annotated training data to acquire high level language knowledge. Several ML techniques have been successfully used for the NER task of which Hidden Markov Model (Bikel et al., 1997), Maximum Entropy (Borthwick, 1999), Conditional Random Field (Li and Mccallum, 2004) are most common. Combinations of different ML approaches are also used. Srihari et al. (2000) combines Maximum Entropy, Hidden Markov Model and handcrafted rules to build an NER system.

NER systems use gazetteer lists for identifying names. Both the linguistic approach (Grishman, 1995; Wakao et al., 1996) and the ML based approach (Borthwick, 1999; Srihari et al., 2000) use gazetteer lists.

The linguistic approach uses hand-crafted rules which needs skilled linguistics. Some recent approaches try to learn context patterns through ML which reduce amount of manual labour. Talukder et al.(2006) combined grammatical and statistical techniques to create high precision patterns specific for NE extraction. An approach to lexical pattern learning for Indian languages is described by Ekbal and Bandopadhyay (2007). They used seed data and annotated corpus to find the patterns for NER.

The NER task for Hindi has been explored by Cucerzan and Yarowsky in their language independent NER work which used morphological and contextual evidences (Cucerzan and Yarowsky, 1999). They ran their experiment with 5 languages - Romanian, English, Greek, Turkish and Hindi. Among these the accuracy for Hindi was the worst. For Hindi the system achieved 41.70% f-value with a very low recall of 27.84% and about 85% precision. A more successful Hindi NER system was developed by Wei Li and Andrew Mccallum (2004) using Conditional Random Fields (CRFs) with feature induction. They were able to achieve 71.50% f-value using a training set of size 340k words. In Hindi the maximum accuracy is achieved by (Kumar and Bhattacharyya, 2006). Their Maximum Entropy Markov Model (MEMM) based model gives 79.7% f-value.

3 Maximum Entropy Based Model

We have used a Maximum Entropy model to build the NER in Hindi. MaxEnt is a flexible statistical model which assigns an outcome for each token based on its history and features. MaxEnt computes the probability $p(o|h)$ for any o from the space of all possible outcomes O, and for every h from the space of all possible histories H. A history is all the conditioning data that enables one to assign probabilities to the space of outcomes. In NER, history can be viewed as all information derivable from the training corpus relative to the current token. The computation of $p(o|h)$ in MaxEnt depends on a set of features, which are helpful in making predictions about the outcome. The features may be binary-valued or multi-valued. For instance, one of our features is: the current token is a part of the surname list; how likely is it to be part of a person name. Formally, we can represent this feature as follows:

$$f(h,o) = \begin{cases} 1 & \text{if } w_i \text{ in surname list and } o = \text{person} \\ 0 & \text{otherwise} \end{cases} \tag{1}$$

Given a set of features and a training corpus, the MaxEnt estimation process produces a model in which every feature f_i has a weight α_i. We can compute the conditional probability as (Pietra et al., 1997):

$$p(o|h) = \frac{1}{Z(h)} \prod_i {\alpha_i}^{f_i(h,o)} \tag{2}$$

$$Z(h) = \sum_o \prod_i {\alpha_i}^{f_i(h,o)} \tag{3}$$

So the conditional probability of the outcome is the product of the weights of all active features, normalized over the products of all the features. For our development we have used a Java based open-nlp MaxEnt toolkit[1] to get the probability values of

[1]www.maxent.sourceforge.net.

a word belonging to each class. That is, given a sequence of words, the probability of each class is obtained for each word. To find the most probable tag corresponding to each word of a sequence, we can choose the tag having the highest class conditional probability value. But this method is not good as it might result in an inadmissible output tag.

Some tag sequences should never happen. To eliminate these inadmissible sequences we have made some restrictions. Then we used a beam search algorithm with a beam of length 3 with these restrictions.

The training data for this task is composed of about $243K$ words which is collected from the popular daily Hindi newspaper "Dainik Jagaran". This corpus has been manually annotated and has about 16,482 NEs. In this development we have considered 4 types of NEs, these are $Person$(P), $Location$(L), $Organization$(O) and $Date$(D). To recognize entity boundaries each name class N is subdivided into 4 sub-classes, i.e., N_Begin, $N\_Continue$, N_End, and N_Unique. Hence, there are a total of 17 classes including 1 class for not-name. The corpus contains $6,298$ Person, $4,696$ Location, $3,652$ Organization and $1,845$ Date entities.

4 Features for Hindi NER

Machine learning approaches like MaxEnt, CRF etc. make use of different features for identifying the NEs. Orthographic features (like capitalization, decimal, digits), affixes, left and right context (like previous and next words), NE specific trigger words, gazetteer features, POS and morphological features etc. are generally used for NER. In English and some other languages, capitalization features play an important role as NEs are generally capitalized for these languages. Unfortunately this feature is not applicable for Hindi. Also Indian person names are more diverse, lots of common words having other meanings are also used as person names. These make difficult to develop a NER system on Hindi. Li and Mccallum (2004) used the entire word text, character n-grams (n = 2, 3, 4), word prefix and suffix of lengths 2, 3 and 4, and 24 Hindi gazetteer lists as atomic features in their Hindi NER. Kumar and Bhattacharyya (2006) used word features (suffixes, digits, special characters), context features, dictionary features, NE list features etc. in their MEMM based Hindi NER system. In the following we have discussed about the features we have identified and used to develop the Hindi NER system.

4.1 Feature Description

The features which we have identified for Hindi Named Entity Recognition are:

Static Word Feature: The previous and next words of a particular word are used as features. The previous m words $(w_{i-m}...w_{i-1})$ to next n words $(w_{i+1}...w_{i+n})$ can be treated. During our experiment different combinations of previous 4 to next 4 words are used.

Context Lists: Context words are defined as the frequent words present in a word window for a particular class. We compiled a list of the most frequent words that occur within a window of $w_{i-3}...w_{i+3}$ of every NE class. For example, location context list contains the words like '$jAkara$[2]' (going to), '$desha$' (country), '$rAjadhAnI$' (capital) etc. and person context list contains '$kahA$' (say), '$prdhAnama.ntrI$' (prime minister) etc. For a given word, the value of this feature corresponding to a given NE type is set to 1 if the window $w_{i-3}...w_{i+3}$ around the w_i contains at last one word from this list.

Dynamic NE tag: Named Entity tags of the previous words $(t_{i-m}...t_{i-1})$ are used as features.

First Word: If the token is the first word of a sentence, then this feature is set to 1. Otherwise, it is set to 0.

Contains Digit: If a token 'w' contains digit(s) then the feature $ContainsDigit$ is set to 1. This feature is helpful for identifying company product names (e.g. 06WD1992), house number (e.g. C226) etc.

Numerical Word: For a token 'w' if the word is a numerical word i.e. a word denoting a number (e.g. eka (one), do (two), $tina$ (three) etc.) then the feature $NumWord$ is set to 1.

Word Suffix: Word suffix information is helpful to identify the named NEs. Two types of suffix features have been used. Firstly a fixed length word suffix of the current and surrounding words are used

[2]All Hindi words are written in italics using the 'Itrans' transliteration.

as features. Secondly we compiled lists of common suffixes of person and place names in Hindi. For example, '$pura$', '$bAda$', '$nagara$' etc. are location suffixes. We used two binary features corresponding to the lists - whether a given word has a suffix from the list.

Word Prefix: Prefix information of a word may be also helpful in identifying whether it is a NE. A fixed length word prefix of current and surrounding words are treated as a features.

Parts-of-Speech (POS) Information: The POS of the current word and the surrounding words may be useful feature for NER. We have access to a Hindi POS pagger developed at IIT Kharagpur which has an accuracy about 90%. The tagset of the tagger contains 28 tags. We have used the POS values of the current and surrounding tokens as features.

We realized that the detailed POS tagging is not very relevant. Since NEs are noun phrases, the noun tag is very relevant. Further the postposition following a name may give a clue to the NE type. So we decided to use a coarse-grained tagset with only three tags - nominal (Nom), postposition (PSP) and other (O).

The POS information is also used by defining several binary features. An example is the $NomPSP$ binary feature. The value of this feature is defined to be 1 if the current token is nominal and the next token is a PSP.

5 Enhancement using Gazetteer Feature

Lists of names of various types are helpful in name identification. We have compiled some specialized name lists from different web sources. But the names in these lists are in English, not in Hindi. So we have transliterated these English name lists to make them useful for our Hindi NER task.

For the transliteration we have build a 2-phase transliteration module. We have defined an intermediate alphabet containing 34 characters. English names are transliterated to this intermediate form using a map-table. Hindi strings are also transliterated to the intermediate alphabet form using a different map-table. For a English-Hindi string pair, if transliterations of the both strings are same, then we conclude that one string is the transliteration of the other. This transliteration module works with 91.59% accuracy.

Using the transliteration approach we have constructed 8 lists. Which are, month name and days of the week (*40*)[3], organization end words list (*92*), person prefix words list (*123*), list of common locations (*80*), location names list (*17,600*), first names list (*9722*), middle names list (*35*), surnames list (*1800*).

The lists can be used in name identification in various ways. One way is to check whether a token is in any list. But this approach is not good as it has some limitations. Some words may present in two or more gazetteer lists. For example, '$bangAlora$' is in surnames list and also in location names list. Confusions arise to make decisions for these words. Some words are in gazetteer lists but sometimes these are used in text as not-name entity. For example, '$gayA$' is in location list but sometimes the word is used as verb in text and makes confusion. These limitations might be reduced if the contexts are considered.

We have used these gazetteer lists as features of MaxEnt. We have prepared several binary features which are defined as whether a given word is in a particular list. For example, a binary feature $FirstName$ is 1 for a particular token 't' if 't' is in the first name list.

6 Context Pattern based Features

Context patterns are helpful for identifying NEs. As manual identification of context patterns takes much manual labour and linguistic knowledge, we have developed a module for semi-automatically learning of context pattern. The summary of the context pattern learning module is given follows:

1. Collect some seed entities (E) for each class.

2. For each seed entity e in E, from the corpus find context string(C) comprised of n tokens before e, a placeholder for the class instance and n tokens after e. [We have used $n = 3$] This set of tokens form initial pattern.

3. Search the pattern in the corpus and find the coverage and precision.

4. Discard the patterns having low precision.

[3]The italics integers in brackets indicate the size of the lists.

5. Generalize the patterns by dropping one or more tokens to increase coverage.

6. Find best patterns having good precision and coverage.

The quality of a pattern is measured by precision and coverage. Precision is the ratio of correct identification and the total identification, when the particular pattern is used to identify of NEs of a specific type from a raw text. Coverage is the amount of total identification. We have given more importance to precision and we have marked a pattern as $effective$ if the precision is more than 95%. The method is applied on an un-annotated text having 4887011 words collected from "Dainik Jagaran" and context patterns are learned. These context patterns are used as features of MaxEnt in the Hindi NER system. Some example patterns are:

1. mukhyama.ntrI <PER> Aja

2. <PER> ne kahA ki

3. rAjadhAnI <LOC> me

7 Evaluation

We have evaluated the system using a blind test corpus of 25K words, which is distinct from the training corpus. The accuracies are measured in terms of the f-measure, which is the weighted harmonic mean of precision and recall. Here we can mention that we have evaluated the performance of the system on actual NEs. That means the system annotates the test data using 17 tags, similar to the training data. During evaluation we have merged the sub-tags of a particular entity to get a complete NEs and calculated the accuracies. At the end of section 7.1 we have also mentioned the accuracies if evaluated on the tags. A number of experiments are conducted considering various combinations of features to identify the best feature set for the Hindi NER task.

7.1 Baseline

The baseline performance of the system without using gazetteer and context patterns are presented in Table 1. They are summarized below.

While experimenting with static word features, we have observed that a window of previous two words to next two words ($W_{i-2}...W_{i+2}$) gives best results. But when several other features are combined then single word window ($W_{i-1}...W_{i+1}$) performs better. Similarly we have experimented with suffixes of different lengths and observed that the suffixes of length ≤ 2 gives the best result for the Hindi NER task. In using POS information, we have observed that the coarse-grained POS tagger information is more effective than the finer-grained POS values. A feature set, combining finer-grained POS values, surrounding words and previous NE tag, gives a f-value of 70.39%. But when the coarse-grained POS values are used instead of the

Feature	**Class**	**F-value**
f1 = Word, NE Tag	PER	63.33
	LOC	69.56
	ORG	58.58
	DAT	91.76
	TOTAL	69.64
f2 = Word, NE Tag, Suffix (≤ 2)	PER	69.75
	LOC	75.8
	ORG	59.31
	DAT	89.09
	TOTAL	73.42
f3 = Word, NE Tag, Suffix (≤ 2), Prefix	PER	70.61
	LOC	71
	ORG	59.31
	DAT	89.09
	TOTAL	72.5
f4 = Word, NE Tag, Digit, Suffix (≤ 2)	PER	70.61
	LOC	75.8
	ORG	60.54
	DAT	93.8
	TOTAL	74.26
f5 = Word, NE Tag, POS	PER	64.25
	LOC	71
	ORG	60.54
	DAT	89.09
	TOTAL	70.39
f6 = Word, NE Tag, Suffix (≤ 2), Digit, $NomPSP$	PER	72.26
	LOC	78.6
	ORG	51.36
	DAT	92.82
	TOTAL	**75.6**

Table 1: F-values for different features

finer-grained POS values, the f-value is increased to 74.16%. The most interesting fact we have observed that more complex features do not guarantee to achieve better results. For example, a feature set combined with current and surrounding words, previous NE tag and fixed length suffix information, gives a f-value 73.42%. But when prefix information are added the f-value decreased to 72.5%. The highest accuracy achieved by the system is 75.6% f-value without using gazetteer information and context patterns.

The results in Table 1 are obtained by evaluating on the actual NEs. But when the system is evaluated on the tags the f-value increases. For f6, the accuracy achieved on actual NEs is 75.6%, but if evaluated on tags, the value increased to 77.36%. Similarly, for f2, the accuracy increased to 75.91% if evaluated on tags. The reason is the NEs containing 3 or more words, are subdivided to N-begin, N-continue (1 or more) and N-end. So if there is an error in any of the subtags, the total NE becomes an error. We observed many cases where NEs are partially identified by the system, but these are considered as $error$ during evaluation.

7.2 Using Gazetteer Lists and Context Patterns

Next we add gazetteer and context patterns as features in our MaxEnt based NER system. In Table 2 we have compared the results after addition of gazetteer information and context patterns with previous results. While experimenting we have observed that gazetteer lists and context patterns are capable of increasing the performance of our baseline system. That is tested on all the baseline feature sets. In Table 2 the comparison is shown for only two features - f2 and f6 which are defined in Table 1. It may be observed that the relative advantage of using both gazetteer and context patterns together over using them individually is not much. For example, when gazetteer information are added with f2, the f-value is increased by 6.38%, when context patterns are added the f-value is increased by 6.64%., but when both are added the increment is 7.27%. This may be due to the fact that both gazetteer and context patterns lead to the same identifications. Using the comprehensive feature set (using gazetteer information and context patterns) the MaxEnt based NER system achieves the maximum f-value of 81.52%.

	F-value				
Feature	**Class**	**No Gaz or Pat**	**With Gaz**	**With Pat**	**With Gaz and Pat**
f2	PER	69.75	74.2	75.61	76.03
	LOC	75.8	82.02	79.94	82.02
	ORG	59.31	72.61	73.4	74.63
	DAT	89.09	94.29	95.32	95.32
	TOTAL	73.42	79.8	80.06	80.69
f6	PER	72.26	76.03	75.61	78.41
	LOC	78.6	82.02	80.49	83.26
	ORG	51.36	72.61	74.1	75.43
	DAT	92.82	94.28	95.87	96.5
	TOTAL	75.6	80.24	80.37	**81.52**

Table 2: F-values for different features with gazetteers and context patterns

8 Conclusion

We have shown that our MaxEnt based NER system is able to achieve a f-value of 81.52%, using a hybrid set of features including traditional NER features augmented with gazetteer lists and extracted context patterns. The system outperforms the existing NER systems in Hindi.

Feature selection and feature clustering might lead to further improvement of performance and is under investigation.

9 Acknowledgement

The work is partially funded by Microsoft Research India.

References

Bikel Daniel M., Miller Scott, Schwartz Richard and Weischedel Ralph. 1997. Nymble: A High Performance Learning Name-finder. In *Proceedings of the Fifth Conference on Applied Natural Language Processing*, pages 194–201.

Borthwick Andrew. 1999. A Maximum Entropy Approach to Named Entity Recognition. *Ph.D. thesis, Computer Science Department, New York University.*

Cucerzan Silviu and Yarowsky David. 1999. Language Independent Named Entity Recognition Combining

Morphological and Contextual Evidence. In *Proceedings of the Joint SIGDAT Conference on EMNLP and VLC 1999*, pages 90–99.

Ekbal A. and Bandyopadhyay S. 2007. Lexical Pattern Learning from Corpus Data for Named Entity Recognition. In *Proceedings of International Conference on Natural Language Processing (ICON), 2007.*

Grishman Ralph. 1995. The New York University System MUC-6 or Where's the syntax? In *Proceedings of the Sixth Message Understanding Conference*.

Kumar N. and Bhattacharyya Pushpak. 2006. Named Entity Recognition in Hindi using MEMM. In *Technical Report, IIT Bombay, India.*.

Li Wei and McCallum Andrew. 2004. Rapid Development of Hindi Named Entity Recognition using Conditional Random Fields and Feature Induction (Short Paper). *ACM Transactions on Computational Logic*.

McDonald D. 1996. Internal and external evidence in the identification and semantic categorization of proper names. In *B. Boguraev and J. Pustejovsky, editors, Corpus Processing for Lexical Acquisition*, pages 21–39.

Pietra Stephen Della, Pietra Vincent Della and Lafferty John. 1997. Inducing features of random fields. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 19(4):380–393.

Srihari R., Niu C. and Li W. 2000. A Hybrid Approach for Named Entity and Sub-Type Tagging. In *Proceedings of the sixth conference on Applied natural language processing*.

Talukdar Pratim P., Brants T., Liberman M., and Pereira F. 2006. A context pattern induction method for named entity extraction. In *Proceedings of the Tenth Conference on Computational Natural Language Learning (CoNLL-X)*.

Wakao T., Gaizauskas R. and Wilks Y. 1996. Evaluation of an algorithm for the recognition and classification of proper names. In *Proceedings of COLING-96*.