IJCNLP-08 Workshop

On

NER for South and South East Asian Languages

Proceedings of the Workshop

12 January 2008
IIIT, Hyderabad, India

Introduction

Welcome to the IJCNLP Workshop on Named Entity Recognition for South and South East Asian Languages, a meeting held in conjunction with the Third International Joint Conference on Natural Language Processing at Hyderabad, India. The goal of this workshop is to ascertain the state of the art in Named Entity Recognition (NER) specifically for South and South East Asian (SSEA) languages. This workshop continues the work started in the NLPAI Machine Learning Contest 2007 which was focused on NER for South Asian languages. NER was selected this time for the contest as well as for this workshop because it is one of the fundamental and most important problems in NLP for which systems with good accuracy have not been built so far for SSEA languages. The primary reason for this is that the characteristics of SSEA languages relevant for NER are different in many respects from English, on which a lot of work has been done with a significant amount of success in the last few years. An introductory article further explains the background of and motivation for this workshop. It also presents the results of an experiment on a reasonable baseline and compares the results obtained by the participating teams with the results for this baseline.

The workshop had two tracks: One track for regular research papers on NER for SSEA languages and the second track on the lines of a shared task. The workshop attracted a lot of interest, especially from the South Asian region. Participation from most of the research centers in South Asia working on NER ensured that the workshop met its goal of ascertaining and advancing the state of the art in NER for SSEA languages. Another major achievement was that a good quantity of named entity annotated corpus was created in five South Asian languages. The notable point about this effort was that this was done almost informally on a voluntary basis, without funding. This is an important point in the context of SSEA languages because lack of annotated corpora has held back progress in many areas of NLP so far in this region.

Each paper was reviewed by three reviewers to ensure satisfactory quality of the selected papers. Another major feature of the workshop is that it includes two invited talks by senior researchers working on the NER problem for South Asian languages. The only drawback of the workshop was that there was no paper on any South East Asian language.

We would like to thank the program committee members for all the hard work that they did during the reviewing process. We would also like to thank all the people involved in organizing the IJCNLP conference. We hope that this workshop will help in creating interest in NER for SSEA languages and we will soon be able to achieve results comparable to those for languages like English.

Rajeev Sangal, Dipti Misra Sharma and Anil Kumar Singh (Chairs)

Organizers:

Rajeev Sangal, IIIT, Hyderabad, India
Dipti Misra Sharma, IIIT, Hyderabad, Hyderabad, India
Anil Kumar Singh, IIIT, Hyderabad, India

Program Committee:

Rajeev Sangal, IIIT, Hyderabad, India
Dekai Wu, The Hong Kong University of Science & Technology, Hong Kong
Ted Pedersen, University of Minnesota, USA
Dipti Misra Sharma, IIIT, Hyderabad, Hyderabad, India
Virach Sornlertlamvanich, TCL, NICT, Thailand
Alexander Gelbukh, Center for Computing Research, National Polytechnic Institute, Mexico
M. Sasikumar, CDAC, Mumbai, India
Sudeshna Sarkar, Indian Institute of Technology, Kharagpur, India
Thierry Poibeau, CNRS, France
Sobha L., AU-KBC, Chennai, India
Tzong-Han Tsai, National Taiwan University, Taiwan
Prasad Pingali, IIIT, Hyderabad, India
Canasai Kreungkrai, NICT, Japan
Manabu Sassano, Yahoo Japan Corporation, Japan
Kavi Narayana Murthy, University of Hyderabad, India
Sivaji Bandyopadhyay, Jadavpur University, Kolkata, India
Anil Kumar Singh, IIIT, Hyderabad, Hyderabad, India
Doaa Samy, Universidad Autnoma de Madrid, Spain
Ratna Sanyal, IIIT, Allahabad, India
V. Sriram, IIIT, Hyderabad, Hyderabad, India
Anagha Kulkarni, Carnegie Mellon University, USA
Soma Paul, IIIT, Hyderabad, Hyderabad, India
Sofia Galicia-Haro, National Autonomous University, Mexico
Grigori Sidorov, National Polytechnic Institute, Mexico

Special Acknowledgment:

Samar Husain, IIIT, Hyderabad, India
Harshit Surana, IIIT, Hyderabad, India

Invited Speakers:

Sobha L., AU-KBC, Chennai, India
Sivaji Bandyopadhyay, Jadavpur University, Kolkata, India

Table of Contents

Workshop Program

Saturday, January 12, 2008

Session 1:

09:00-9:30 **Opening Remarks:** *Named Entity Recognition for South and South East Asian Languages: Taking Stock*
Anil Kumar Singh

09:30-10:00 **Invited Talk**: *Named Entity Recognition: Different Approaches*
Sobha L

10:00-10:30 *A Hybrid Named Entity Recognition System for South and South East Asian Languages*
Sujan Kumar Saha, Sanjay Chatterji, Sandipan Dandapat, Sudeshna Sarkar and Pabitra Mitra

10:30-11:00 **Break**

Session 2:

11:00-11:30 **Invited Talk**: *Multilingual Named Entity Recognition*
Sivaji Bandyopadhyay

11:30-12:00 *Aggregating Machine Learning and Rule Based Heuristics for Named Entity Recognition*
Karthik Gali, Harshit Surana, Ashwini Vaidya, Praneeth Shishtla and Dipti Misra Sharma

12:00-12:30 *Language Independent Named Entity Recognition in Indian Languages*
Asif Ekbal, Rejwanul Haque, Amitava Das, Venkateswarlu Poka and Sivaji Bandyopadhyay

12:30-14:00 **Lunch**

Session 3:

14:00-14:30 *Named Entity Recognition for Telugu*
P Srikanth and Kavi Narayana Murthy

14:30-15:00 **Poster Display and Discussion**

An Experiment on Automatic Detection of Named Entities in Bangla
Bidyut Baran Chaudhuri and Suvankar Bhattacharya

Saturday, January 12, 2008 (continued)

Hybrid Named Entity Recognition System for South and South East Asian Languages
Praveen P and Ravi Kiran V

Named Entity Recognition for South Asian Languages
Amit Goyal

Named Entity Recognition for Indian Languages
Animesh Nayan, B. Ravi Kiran Rao, Pawandeep Singh, Sudip Sanyal and Ratna Sanyal

Experiments in Telugu NER: A Conditional Random Field Approach
Praneeth M Shishtla, Karthik Gali, Prasad Pingali and Vasudeva Varma

15:30-16:00 **Break**

Session 4:

16:00-16:30 *Bengali Named Entity Recognition Using Support Vector Machine*
Asif Ekbal and Sivaji Bandyopadhyay

16:30-17:00 *Domain Focused Named Entity Recognizer for Tamil Using Conditional Random Fields*
Vijayakrishna R and Sobha L

17:00-17:30 *A Character n-gram Based Approach for Improved Recall in Indian Language NER*
Praneeth M Shishtla, Prasad Pingali and Vasudeva Varma

17:30-18:00 **Closing Discussion**

Named Entity Recognition: Different Approaches

Sobha, L
AU-KBC Research Centre
MIT Campus of Anna University
Chennai-44
sobha@au-kbc.org

Abstract

The talk deals with different approaches used for Named Entity recognition and how they are used in developing a robust Named Entity Recognizer. The talk includes the development of tagset for NER and manual annotation of text.

Proceedings of the IJCNLP-08 Workshop on NER for South and South East Asian Languages, pages 1–2,
Hyderabad, India, January 2008.

Multilingual Named Entity Recognition

Sivaji Bandyopadhyay
Computer Science & Engineering Department
Jadavpur University
Kolkata, INDIA.
sbandyopadhyay@cse.jdvu.ac.in

Abstract

The computational research aiming at automatically identifying named entities (NE) in texts forms a vast and heterogeneous pool of strategies, techniques and representations from hand-crafted rules towards machine learning approaches. Hand-crafted rule based systems provide good performance at a relatively high system engineering cost. The availability of a large collection of annotated data is the prerequisite for using supervised learning techniques. Semi-supervised and unsupervised learning techniques promise fast deployment for many NE types without the prerequisite of an annotated corpus. The main technique for semi-supervised learning is called bootstrapping and involves a small degree of supervision, such as a set of seeds, for starting the learning process. The typical approach in unsupervised learning is clustering where systems can try to gather NEs from clustered groups based on the similarity of context. The techniques rely on lexical resources (e.g., Wordnet), on lexical patterns and on statistics computed on a large unannotated corpus.

In multilingual named entity recognition (NER), it must be possible to use the same method for many different languages and the extension to new languages must be easy and fast. Person names can be recognized in text through a lookup procedure, by analyzing the local lexical context, by looking at part of a sequence of candidate words that is a known name component etc. Some organization names can be identified by looking at contain organization-specific candidate words. Identification of place names necessarily involves lookup against a gazetteer, as most context markers are too weak and ambiguous.

An important feature in multilingual person name detection is that the same person can be referred to by different name variants. The main reasons for these variations are: the reuse of name parts to avoid repetition, morphological variants such as the added suffixes, spelling mistakes, adaptation of names to local spelling rules, transliteration differences due to different transliteration rules or different target languages etc.. Name variants can be found within the same language documents.

The major challenges for looking up place names in a multilingual gazetteer are the following: place names are frequently homographic with common words or with person names, presence of a number of exonyms (foreign language equivalences), endonyms (local variants) and historical variants for many place names etc..

Application of NER to multilingual document sets helps to find more and more accurate informa-tion on each NE, while at the same time rich in-formation about NEs is helpful and can even be a crucial ingredient for text analysis applications that cross the language barrier.

Proceedings of the IJCNLP-08 Workshop on NER for South and South East Asian Languages, pages 3–4,
Hyderabad, India, January 2008.

Named Entity Recognition for South and South East Asian Languages: Taking Stock

Anil Kumar Singh
Language Technologies Research Centre
IIIT, Hyderabad, India
anil@research.iiit.ac.in

Abstract

In this paper we first present a brief discussion of the problem of Named Entity Recognition (NER) in the context of the IJCNLP workshop on NER for South and South East Asian (SSEA) languages[1]. We also present a short report on the development of a named entity annotated corpus in five South Asian language, namely Hindi, Bengali, Telugu, Oriya and Urdu. We present some details about a new named entity tagset used for this corpus and describe the annotation guidelines. Since the corpus was used for a shared task, we also explain the evaluation measures used for the task. We then present the results of our experiments on a baseline which uses a maximum entropy based approach. Finally, we give an overview of the papers to be presented at the workshop, including those from the shared task track. We discuss the results obtained by teams participating in the task and compare their results with the baseline results.

1 Introduction

One of the motivations for organizing a workshop (NERSSEAL-08) focused on named entities (NEs) was that they have a special status in Natural Language Processing (NLP) because they have some properties which other elements of human languages do not have, e.g. they refer to specific things or concepts in the world and are not listed in the grammars or the lexicons. Identifying and classifying them automatically can help us in processing text because they form a significant portion of the types and tokens occurring in a corpus. Also, because of their very nature, machine learning techniques have been found to be very useful in identifying them. In order to use these machine learning techniques, we need corpus annotated with named entities. In this paper we describe such a corpus developed for five South Asian languages. These languages are Hindi, Bengali, Oriya, Telugu and Urdu.

This paper also presents an overview of the work done for the IJCNLP workshop on NER for SSEA languages. The workshop included two tracks. The first track was for regular research papers, while the second was organized on the lines of a shared task.

Fairly mature named entity recognition systems are now available for European languages (Sang, 2002; Sang and De Meulder, 2003), especially English, and even for East Asian languages (Sassano and Utsuro, 2000). However, for South and South East Asian languages, the problem of NER is still far from being solved. Even though we can gain much insight from the methods used for English, there are many issues which make the nature of the problem different for SSEA languages. For example, these languages do not have capitalization, which is a major feature used by NER systems for European languages.

Another characteristic of these languages is that most of them use scripts of Brahmi origin, which have highly phonetic characteristics that could be utilized for multilingual NER. For some languages, there are additional issues like word segmentation

[1] http://ltrc.iiit.ac.in/ner-ssea-08

Proceedings of the IJCNLP-08 Workshop on NER for South and South East Asian Languages, pages 5–16,
Hyderabad, India, January 2008.

(e.g. for Thai). Large gazetteers are not available for most of these languages. There is also the problem of lack of standardization and spelling variation. The number of frequently used words (common nouns) which can also be used as names (proper nouns) is very large for, unlike for European languages where a larger proportion of the first names are not used as common words. For example, 'Smith', 'John', 'Thomas' and 'George' etc. are almost always used as person names, but 'Anand', 'Vijay', 'Kiran' and even 'Manmohan' can be (more than often) used as common nouns. And the frequency with which they can be used as common nouns as against person names is more or less unpredictable. The context might help in disambiguating, but this issue does make the problem much harder than for English.

Among other problems, one example is that of the various ways of representing abbreviations. Because of the alpha-syllabic nature of the SSEA scripts, abbreviation can be expressed through a sequence of letters or syllables. In the latter case, the syllables are often combined together to form a pseudo-word, e.g. BAjapA (bhaajapaa) for Bharatiya Janata Party or BJP.

But most importantly, there is a serious lack of labeled data for machine learning. As part of this workshop, we have tried to prepare some data but we will need much more data for really accurate NER systems.

Since most of the South and South East Asian languages are scarce in resources as well as tools, it is very important that good systems for NER be available, because many problems in information extraction and machine translation (among others) are dependent on accurate NER.

The need for a workshop specifically for SSEA languages was felt because the South and South East Asian region has many major and numerous minor languages. In terms of the number of speakers there are at least four in any list of top ten languages of the world. For practical reasons, we focus only on the major languages in the workshop (and in this paper). Most of the major languages belong to two families: Indo-European and Dravidian. There are a lot of differences among these languages, but there are a lot of similarities too, even across families (Emeneau, 1956; Emeneau, 1980). For the reasons mentioned above, NER is perhaps more difficult for SSEA languages than for European languages. For better or for worse, there too many languages and too few resources. Moreover, these languages are also comparatively less studied by researchers. However, we can benefit from the similarities across these languages to build multilingual systems so as to reduce the overall cost and effort required.

All the issues mentioned above show that we might need different methods for solving the NER problem for SSEA languages. However, for comparing the results of these different methods, we will need a reasonably good baseline. A mature system tuned for English but trained on SSEA language data can become such a baseline. We will describe such a baseline in a later section. This baseline system has been tested on the data provided for the shared task. We present the results for all five languages under the settings required for the shared task.

2 Related Work

Various techniques have been used for solving the NER problem (Mikheev et al., 1999; Borthwick, 1999; Cucerzan and Yarowsky, 1999; Chieu and Ng, 2003; Klein et al., 2003; Kim and Woodland, 2000) ranging from naively using gazetteers to rules based techniques to purely statistical techniques, even hybrid approaches. Several workshops consisting of shared tasks (Sang, 2002; Sang and De Meulder, 2003) have been held with specific focus on this problem. In this section we will mention some of techniques used previously.

Most of the approaches can be classified based on the features they use, whether they are rule based or machine learning based or hybrid approaches. Some of the commonly used features are:

- Word form and part of speech (POS) tags
- Orthographic features like capitalization, decimal, digits
- Word type patterns
- Conjunction of types like capitalization, quotes, functional words etc.
- Bag of words
- Trigger words like New York *City*

Tag	Name	Description
NEP	Person	Bob Dylan, Mohandas Gandhi
NED	Designation	General Manager, Commissioner
NEO	Organization	Municipal Corporation
NEA	Abbreviation	NLP, B.J.P.
NEB	Brand	Pepsi, Nike (ambiguous)
NETP	Title-Person	Mahatma, Dr., Mr.
NETO	Title-Object	Pride and Prejudice, Othello
NEL	Location	New Delhi, Paris
NETI	Time	3rd September, 1991 (ambiguous)
NEN	Number	3.14, 4,500
NEM	Measure	Rs. 4,500, 5 kg
NETE	Terms	Maximum Entropy, Archeology

Table 1: The named entity tagset used for the shared task

- Affixes like Hydera*bad*, Ram*pur*, Mehdi*patnam*, Lingam*pally*
- Gazetteer features: class in the gazetteer
- Left and right context
- Token length, e.g. the number of letters in a word
- Previous history in the document or the corpus
- Classes of preceding NEs

The machine learning techniques tried for NER include the following:

- Hidden Markov Models or HMM (Zhou and Su, 2001)
- Decision Trees (Isozaki, 2001)
- Maximum Entropy (Borthwick et al., 1998)
- Support Vector Machines or SVM (Takeuchi and Collier, 2002)
- Conditional Random Fields or CRF (Settles, 2004)

Different ways of classifying named entities have been used, i.e., there are more than one tagsets for NER. For example, the CoNLL 2003 shared task[2] had only four tags: persons, locations, organizations and miscellaneous. On the other hand, MUC-6[3] has a near ontology for information extraction purposes. In this (MUC-6) tagset, there are three[4] main kinds of NEs: ENAMEX (persons, locations and organizations), TIMES (time expressions) and NUMEX (number expresssions).

There has been some previous work on NER for SSEA languages (McCallum and Li, 2003; Cucerzan and Yarowsky, 1999), but most of the time such work was an offshoot of the work done for European languages. Even including the current workshop, the work on NER for SSEA languages is still in the initial stages as the results reported by papers in this workshop clearly show.

3 A New Named Entity Tagset

The tagset being used for the NERSSEAL-08 shared task consists of more tags than the four tags used for the CoNLL 2003 shared task. The reason we opted for these tags was that we needed a slightly finer tagset for machine translation (MT). The initial aim was to improve the performance of the MT system.

As annotation progressed, we realized that there were some problems that we had not anticipated. Some classes were hard to distinguish in some contexts, making the task hard for annotators and bringing in inconsistencies. For example, it was not always clear whether something should be marked as

[2]http://www.cnts.ua.ac.be/conll2003/ner/

[3]http://cs.nyu.edu/cs/faculty/grishman/muc6.html

[4]http://cs.nyu.edu/cs/faculty/grishman/NEtask20.book_6.html

Number or as Measure. Similarly for Time and Measure. Another difficult class was that of (technical) terms. Is 'agriculture' a term or not? If no (as most people would say), is 'horticulture' a term or not? In fact, Term was the most difficult class to mark.

An option that we explored was to merge the above mentioned confusable classes and ignore the Term class. But we already had a relatively large corpus marked up with these classes. If we merged some classes and ignored the Term class (which had a very large coverage and is definitely going to be useful for MT), we would be throwing away a lot of information. And we also had some corpus annotated by others which was based on a different tagset. So some problems were inevitable. Finally, we decided to keep the original tagset, with one modification. The initial tagset had only eleven tags. The problem was that there was one Title tag but it had two different meanings: 'Mr.' is a Title, but 'The Seven Year Itch' is also a Title. This tag clearly needed to be split into two: Title-Person and Title-Object

We should mention here that we considered using another tagset developed at AUKBC, Chennai. This was based on ENAMEX, TIMEX and NUMEX. The total number of tags in this tagset is more than a hundred and it is meant specifically for MT and only for certain domains (health, tourism). Moreover, this is a tagset for entities in general, not just named entities.

The twelve tags in our tagset are briefly explained in Table-1. In the next section we mention the constraints under which the annotated corpus was created, using this tagset.

4 Annotation Constraints

The annotated corpus was created under severe constraints. The annotation was to be for five languages by different teams, sometimes with very little communication during the process of annotation. As a result, there were many logistical problems.

There were other practical constraints like the fact that this was not a funded project and all the work was mainly voluntary. Another major constraint for all the languages except Hindi was time. There was not enough time for cross validation as the corpus was required by a deadline. To keep annotation reasonably consistent, annotation guidelines were created and a common format was specified.

5 Annotation Guidelines

The annotation guidelines were of two kinds. One was meant for preparing training data through manual annotation. The other one was meant for preparing reference data as well as for automatic annotation. The main guidelines for preparing the training data are as follows:

- *Specificity*: The most important criterion while deciding whether some expression is a named entity or not is to see whether that expression specifies something definite and identifiable as if by a name or not. This decision will have to be based on the context. For example, 'aanand' (in South Asian languages, where there is no capitalization) is not a named entity in 'saba aanand hii aanand hai' ('There is bliss everywhere'). But it is a named entity in 'aanand kaa yaha aakhiri saala hai' ('Anand is in the last year (of his studies)'). Number, Measure and Term may be seen as exceptions (see below).

- *Maximal Entity*: Only the maximal entities have to be annotated for training data. Structure of entities will not be annotated by the annotators, even though it has to be learnt by the NER systems. For example, 'One Hundred Years of Solitude' has to be annotated as one entity. 'One Hundred' is not to be marked as a Number here, nor is 'One Hundred Years' to be made marked as a Measure in this case. The purpose of this guideline is to make the task of annotation for several languages feasible, given the constraints.

- *Ambiguity*: In cases where an entity can have two valid tags, the more appropriate one is to be used. The annotator has to make the decision in such cases. It is recommended that the annotation be validated by another person, or even more preferably, two different annotators have to work on the same data independently and inconsistencies have to be resolved by an adjudicator. Abbreviation is an exception to the Ambiguity guideline (see below).

Some other guidelines for specific tags are listed below:

- *Abbreviations*: All abbreviations have to be marked as Abbreviations, Even though every abbreviation is also some other kind of named entity. For example, APJ is an Abbreviation, but also a Person. IBM is also an Organization. Such ambiguity cannot be resolved from the context because it is due to the (wrong?) assumption that a named entity can have only one tag. Multiple annotations were not allowed. This is an exception to the third guideline above.

- *Designation and Title-Person*: An entity is a Designation if it represents something formal and official status with certain responsibilities. If it is just something honorary, then it is a Title-Object. For example, 'Event Coordinator' or 'Research Assistant' is a Designation, but 'Chakravarti' or 'Mahatma' are Titles.

- *Organization and Brand*: The distinction between these two has to be made based on the context. For example, 'Pepsi' could mean an Organization, but it is more likely to mean a Brand.

- *Time and Location*: Whether something is to be marked as Time or Location or not is to be decided based on the Specificity guideline and the context.

- *Number, Measure and Term*: These three may not be strictly named entities in the way a person name is. However, we have included them because they are different from other words of the language. For problems like machine translation, they can be treated like named entities. For example, a Term is a word which can be directly translated into some language if we have a dictionary of technical terms. Once we know a word is a Term, there is likely to be less ambiguity about the intended sense of the word, unlike for other normal words.

The second set of guidelines are different from the first set mainly in one respect: the corpus has to be annotated with not just the maximal NEs, but with all levels of NEs, i.e., nested NEs also have to be marked.

Nested entities were introduced because one of the requirements was that the corpus be useful for building systems which can become parts of a machine translation (MT) system. Nested entities can be useful for MT systems because, quite often, parts of the entities can need to be translated, while the others can just be transliterated. An example of a nested named entity is 'Mahatma Gandhi International Hindi University'. This would be translated in Hindi as *mahaatmaa gaandhii antarraashtriya hindii vishvavidyaalaya*. Only 'International' and 'University' are to be translated, while the other words are to be transliterated. The nested named entities in this case are: 'Mahatma' (NETO), 'Gandhi' (NEP), 'Mahatma Gandhi' (NEP), and 'Mahatma Gandhi International Hindi University' (NEO).

6 Named Entity Annotated Corpus

For Hindi, Oriya and Telugu, all the annotation was performed at IIIT, Hyderabad. For Bengali, the corpus was developed at IIIT, Hyderabad and Jadavpur University (Ekbal and Bandyopadhyay, 2008b), Calcutta. For Urdu, annotation was performed at CRULP, Lahore (Hussain, 2008) and IIIT, Allahabd. Even though all the annotation was done by native speakers of respective languages, named entity annotation was a new task for everyone involved. This was because of practical constraints as explained in an earlier section.

The corpus was divided into two parts, one for training and one for testing. The testing corpus was annotated with nested named entities, while the training corpus was only annotated with 'maximal' named entities.

Since different teams were working on different languages, in some cases even the same language, and also because most of the corpus was created on short notice, each team made its own decisions regarding the kind of corpus to be annotated. As a result, the characteristics of the corpus differ widely among the five languages. The Hindi and Bengali (partly) text that was annotated was from the multilingual comparable corpus known as the CIIL (Central Institute of Indian Languages) corpus. The Oriya corpus was part of the Gyan Nidhi corpus.

NE	Hindi		Bengali		Oriya		Telugu		Urdu	
	Trn	**Tst**	**Trn**	**Tst**	**Trn**	**Tst**	**Trn**	**Tst**	**Trn**	**Tst**
NEP	4025	199	1299	728	2079	698	1757	330	365	145
NED	935	61	185	11	67	216	87	77	98	41
NEO	1225	44	264	20	87	200	86	12	155	40
NEA	345	7	111	9	8	20	97	112	39	3
NEB	5	0	22	0	11	1	1	6	9	18
NETP	1	5	68	57	54	201	103	2	36	15
NETO	964	88	204	46	37	28	276	118	4	147
NEL	4089	211	634	202	525	564	258	751	1118	468
NETI	1760	50	285	46	102	122	244	982	279	59
NEN	6116	497	407	144	124	232	1444	391	310	47
NEM	1287	17	352	146	280	139	315	53	140	40
NETE	5658	843	1165	314	5	0	3498	138	30	4
NEs	26432	2022	5000	1723	3381	2421	8178	3153	2584	1027
Words	503179	32796	112845	38708	93173	27007	64026	8006	35447	12805
Sentences	19998	2242	6030	1835	1801	452	5285	337	1508	498
Trn: Training Data, **Tst**: Testing Data										

Table 2: Statistics about the corpus: counts of various named entity classes and the size of the corpus as the number of words and the number of sentences. Note that the values for the testing part are of nested NEs. Also, the number of sentences, especially in the case or Oriya is not accurate because the sentences were not correctly segmented as there was no automatic sentence splitter available for these languages and manual splitting would have been too costly: without much benefit for the NER task.

Both of these (CIIL and Gyan Nidhi) corpora consist of text from educational books written on various topics for common readers. The Urdu text was partly news corpus. The same was the case with Telugu, but the text for both these languages included text from other domains too.

Admittedly, the texts selected for annotation were not the ideal ones. For example, many documents had very few named entities. Also, the distribution of domains as well as the classes of NEs was not representative. The size of the annotated corpora for different languages is also widely varying, with Hindi having the largest corpus and Urdu the smallest. However, this corpus is hopefully just a starting point for much more work in the near future.

Some statistics about the annotated corpus are given in Table-2.

7 Shared Task

In the shared task, the contestants having their own NER systems were given some annotated test data. The contestants had the freedom to use any technique for NER, e.g. a purely rule based technique or a purely statistical technique.

The contestants could build NER systems targeted for a specific language, but they were required to report results for their systems on all the languages for which training data had been provided. This condition was meant to provide a somewhat fair ground for comparison of systems, since the amount of training data is different for different languages.

The data released for the shared task has been made accessible to all for non-profit research word, not just for the shared task participants, with the hope others will contribute in improving this data and adding to it.

The task in this contest was different in one important way. The NER systems also had to identify nested named entities. For example, in the sentence The Lal Bahadur Shastri National Academy of Administration is located in Mussoorie, 'Lal Bahadur Shastri' is a Person, but 'Lal Bahadur Shastri National Academy of Administration' is an Organization. In this case, the NER systems had to identify both 'Person' and 'Organization' in the given sentence.

An evaluation script was also provided to evaluate the performance of different systems in a uniform way.

8 Evaluation Measures

As part of the evaluation process for the shared task, precision, recall and F-measure had to be calculated for three cases: maximal named entities, nested named entities and lexical matches. Thus, there were nine measures of performance:

- Maximal Precision: $P_m = \frac{c_m}{r_m}$
- Maximal Recall: $R_m = \frac{c_m}{t_m}$
- Maximal F-Measure: $F_m = \frac{2*P_m*R_m}{P_m+R_m}$
- Nested Precision: $P_n = \frac{c_n}{r_n}$
- Nested Recall: $R_n = \frac{c_n}{t_n}$
- Nested F-Measure: $F_n = \frac{2P_nR_n}{P_n+R_n}$
- Lexical Precision: $P_l = \frac{c_l}{r_l}$
- Lexical Recall: $R_l = \frac{c_l}{t_l}$
- Lexical F-Measure: $F_l = \frac{2P_lR_l}{P_l+R_l}$

where c is the number of correctly retrieved (identified) named entities, r is the total number of named entities retrieved by the system being evaluated (correct plus incorrect) and t is the total number of named entities in the reference data.

The participants were encouraged to report results for specific classes of NEs. Evaluation was automatic and was against the manually prepared reference data given to the participants. An evaluation script for this purpose was also provided. This script assumes that there are single test and reference file and the number and order of sentences is the same in both. The format accepted by the evaluation script (which was also the format used for annotated data) was explained in an online tutorial[5].

9 Experiments on a Baseline

For our baseline experiments, we used an open source implementation of maximum entropy based Natural Languages Processing tools which are part of the OpenNLP[6] package. This package includes a name finder tool.

This name finder was trained for all the twelve classes of NEs and for all the five languages. The test data, which was the same as that given to the shared task participants, was run through this name finder. Note that this NER tool is tuned for English in terms of the features used, even though it was trained on different SSEA languages in our case. Since the goal of the shared task was to encourage investigation of techniques (especially features) specific to the SSEA languages, this fairly mature NER system (for English) could be used as a baseline against which to evaluate systems tuned (or specially designed) for the five South Asian languages.

The overall results of the baseline experiments are shown in Table-3. The performance on specific NE classes is given in Table-4. It can be seen from the tables that the results are drastically low in comparison to the state of the art results reported for English. These results clearly show that even a machine learning based system cannot be directly used for SSEA languages even when it has been trained with annotated data for these languages.

In the next section we present a brief overview of the papers selected for the workshop including the shared task papers.

10 An Overview of the Papers

In all, twelve papers were selected for the workshop, out of which four were in the shared task track. Saha et al., who were able to achieve the best results in the shared task, describe a hybrid system that applies maximum entropy models, language specific rules, and gazetteers. For Hindi, the features they utilized include orthographic features, information about suffixes and prefixes, morphological features, part of speech information, and information about the surrounding words. They used rules for numbers, measures and time classes. For designation, title-person and some terms (NETE), they built lists or gazetteers. They also used gazetteers for person and location. They did not use rules or gazetteers for Oriya, Urdu and Telugu. To identify some kinds of nested entities, they applied a set of rules.

Gali et al. also combined machine learning with language specific heuristics. In a separate section, they discussed at some length the issues relevant to NER for SSEA languages. Some of these have al-

[5] http://ltrc.iiit.ac.in/ner-ssea-08/NER-SAL-TUT.pdf
[6] http://opennlp.sourceforge.net/

Measure →	**Precision**			**Recall**			**F-Measure**		
Language ↓	P_m	P_n	P_l	R_m	R_n	R_l	F_m	F_n	F_l
Bengali	50.00	44.90	52.20	07.14	06.90	06.97	12.50	11.97	12.30
Hindi	75.05	73.61	73.99	18.16	17.66	15.53	29.24	28.48	25.68
Oriya	29.63	27.46	48.25	09.11	07.60	12.18	13.94	11.91	19.44
Telugu	00.89	02.83	22.85	00.20	00.67	5.41	00.32	01.08	08.75
Urdu	47.14	43.50	51.72	18.35	16.94	18.94	26.41	24.39	27.73
m**: Maximal,** n**: Nested,** l**: Lexical**									

Table 3: Results for the experiments on a baseline for the five South Asian languages

	Bengali	**Hindi**	**Oriya**	**Telugu**	**Urdu**
NEP	06.62	26.23	28.48	00.00	04.39
NED	00.00	12.20	00.00	00.00	00.00
NEO	00.00	15.50	03.30	00.00	11.98
NEA	00.00	00.00	00.00	00.00	00.00
NEB	NP	NP	00.00	00.00	00.00
NETP	00.00	NP	11.62	00.00	00.00
NETO	00.00	05.92	04.08	00.00	00.00
NEL	03.03	44.79	25.49	00.00	40.21
NETI	34.00	47.41	22.38	01.51	38.38
NEN	62.63	62.22	10.65	03.51	09.52
NEM	13.61	24.39	08.03	00.71	07.15
NETE	00.00	00.18	00.00	00.00	00.00
NP: Not present in the reference data					

Table 4: Baseline results for specific named entity classes (F-Measures for nested lexical match)

ready been mentioned, but two others are the agglutinative property of these (especially Dravidian) languages and the low accuracy of available part of speech taggers, particularly for nouns. They used a Conditional Random Fields (CRF) based method for machine learning and applied heuristics to take care of the language specific issues. They also point out that a very high percentage of NEs in the Hindi corpus were marked as NETE and machine learning failed to take care of this class of NEs. This has been validated by our results on the baseline too (Table-4) and is understandable because terms are hard to identify even for humans.

Ekbal et al. also used an approach based on CRFs. They also used some language specific features for Hindi and Bengali. Srikanth and Murthy describe the results of their experiments on NER using CRFs for Telugu. They concentrated only on person, place and organization names and used newspaper text as the corpus. In this focused setting, they were able to achieve overall F-measures between 80% and 97% in various experiments. Chaudhuri and Bhattacharya also experimented on a news corpus for Bengali using a three stage NER system. The three stages were based on an NE dictionary, rules and contextual co-occurrence statistics. They only tried to identify the NEs, not classify them. For this task, they were able to achieve an overall F-measure of 89.51%.

Praveen and Ravi Kumar present the results of experiments (as part of the shared task) using two approaches: Hidden Markov Models (HMM) and CRF. Surprisingly, they obtained better results with HMM for all the five languages. Goyal described experiments using a CRF based model. He also used part of speech information. He experimented only on Hindi and was able to achieve results above 60%. One notable fact about this paper is that it also de-

Language ↓	BL	IK	IH1	IH2	JU
Bengali	12.30	65.96	40.63	39.77	59.39
Hindi	25.68	65.13	50.06	46.84	33.12
Oriya	19.44	44.65	39.04	45.84	28.71
Telugu	08.75	18.74	40.94	46.58	04.75
Urdu	27.73	35.47	43.46	44.73	35.52
Average	18.78	45.99	42.83	44.75	32.30

BL: Baseline, **IK**: IIT Kharagpur
JU: Jadavpur University, Calcutta
IH1: Karthik et al., IIIT Hyderabad
IH2: Praveen and Ravi Kiran, IIIT Hyderabad

Table 5: Comparison of NER systems which participated in the NERSSEAL-08 shared task against a baseline that uses maximum entropy based name finder tuned for English but trained on data from five South Asian languages

scribes experiments on the CoNLL 2003 shared task data for English, which shows that the significantly higher results for English are mainly due to the fact that the CoNLL 2003 data is already POS tagged and chunked with high accuracy. Goyal was also able to show that capitalization is a major clue for English, either directly or indirectly (e.g., for accurate POS tagging and chunking). He also indicated that the characteristics of the Hindi annotated corpus were partly responsible for the low results on Hindi.

Nayan et al. mainly describe how an NER system can benefit from approximate string matching based on phonetic edit distance, both for a single language (to account for spelling variations) and for cross-lingual NER. Shishtla et al. ('Experiments in Telugu NER') experimented only on Telugu and used the CoNLL shared task tagset. Using a CRF based approach, they were able to achieve an F-measure of 44.91%. Ekbal and Bandyopadhyay describe a method based on Support Vector Machines (SVMs) for Bengali NER. On a news corpus and with sixteen NE classes, they were able to achieve an F-measure of 91.8%. Vijayakrishna and Sobha describe a CRF based system for Tamil using 106 NE classes. Their system is a multi-level system which gave an overall F-measure of 80.44%. They also mention that their system achieved this level of performance on a domain focused corpus. Shishtla et al. ('Character n-gram Based Approach') used a character n-gram based method to identify NEs. They experimented on Hindi as well as English and achieved F-measure values up to 45.48% for Hindi and 68.46% for English.

Apart from the paper presentations, the workshop will also have two invited talks. The first one is titled "Named Entity Recognition: Different Approaches" by Sobha L. and the second one is "Multilingual Named Entity Recognition" by Sivaji Bandyopadhyay.

11 Shared Task Results

Five teams participated in the shared task. However, only four submitted papers for the workshop. All the teams tried to combine machine learning with some language specific heuristics, at least for one of the languages. The results obtained by the four teams are summarized in Table-5, which shows only the F-measure for lexical match. It can be seen from the table that all the teams were able to get significantly better results than the baseline. Overall, the performance of the IIT Kharagpur team was the best, followed by the two teams from IIIT Hyderabad.

Even though all the teams obtained results much better than the baseline, it is still quite evident that the state of the art for NER for SSEA languages leaves much to be desired. At around 46% maximum F-measure on lexical matching, the results mean that the NER systems built so far for SSEA languages are not quite practically useful. But, after this workshop, we at least know where we stand and how far we still have to go.

However, it may be noted that the conditions for

the shared task were very stringent compared to the previous shared tasks on NER, e.g. neither the corpus was tagged with parts of speech or chunks, nor were good POS taggers or chunkers available for the languages involved. This indicates that with progress in building better resources and basic tools for these languages, the accuracy of NER systems should also increase. Already, some very high accuracies are being reported under less stringent conditions, e.g. for domain focused NER.

12 Conclusions

We started by discussing the problem of NER for South and South East Asian languages and the motivations for organizing a workshop on this topic. We also described a named entity annotated corpus for five South Asian languages used for this workshop. We presented some statistics about the corpus and also the problems we encountered in getting the corpus annotated by teams located in distant places. We also presented a new named entity tagset that was developed for annotation of this corpus. Then we presented the results for our experiments on a reasonable baseline. Finally we gave an overview of the papers selected for the NERSSEAL-08 workshop and discussed the systems described in these papers and the results obtained, including those for the shared task which was one of the two tracks in the workshop.

References

Sivaji Bandyopadhyay. 2008. Invited talk: Multilingual named entity recognition. In *Proceedings of the IJCNLP-08 Workshop on NER for South and South East Asian Languages*, pages 15–17, Hyderabad, India, January.

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 152–160.

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

Bidyut Baran Chaudhuri and Suvankar Bhattacharya. 2008. An experiment on automatic detection of named entities in bangla. In *Proceedings of the IJCNLP-08 Workshop on NER for South and South East Asian Languages*, pages 51–58, Hyderabad, India, January.

H.L. Chieu and H.T. Ng. 2003. Named entity recognition with a maximum entropy approach. *Proceedings of the seventh conference on Natural language learning at HLT-NAACL 2003-Volume 4*, pages 160–163.

S. Cucerzan and D. Yarowsky. 1999. Language independent named entity recognition combining morphological and contextual evidence. *Proceedings of the Joint SIGDAT Conference on EMNLP and VLC 1999*, pages 90–99.

Asif Ekbal and Sivaji Bandyopadhyay. 2008a. Bengali named entity recognition using support vector machine. In *Proceedings of the IJCNLP-08 Workshop on NER for South and South East Asian Languages*, pages 85–92, Hyderabad, India, January. Association for Computational Linguistics.

Asif Ekbal and Sivaji Bandyopadhyay. 2008b. Development of bengali named entity tagged corpus and its use in ner systems. In *Proceedings of the Sixth Workshop on Asian Language Resources*, Hyderabad, India, January.

Asif Ekbal, Rejwanul Haque, Amitava Das, Venkateswarlu Poka, and Sivaji Bandyopadhyay. 2008. Language independent named entity recognition in indian languages. In *Proceedings of the IJCNLP-08 Workshop on NER for South and South East Asian Languages*, pages 33–40, Hyderabad, India, January.

M. B. Emeneau. 1956. India as a linguistic area. *Linguistics*, 32:3-16.

M. B. Emeneau. 1980. *Language and linguistic area. Essays by Murray B. Emeneau. Selected and introduced by Anwar S. Dil*. Stanford University Press.

Karthik Gali, Harshit Surana, Ashwini Vaidya, Praneeth Shishtla, and Dipti Misra Sharma. 2008. Aggregating machine learning and rule based heuristics for named entity recognition. In *Proceedings of the IJCNLP-08 Workshop on NER for South and South East Asian Languages*, pages 25–32, Hyderabad, India, January.

Amit Goyal. 2008. Named entity recognition for south asian languages. In *Proceedings of the IJCNLP-08 Workshop on NER for South and South East Asian Languages*, pages 63–70, Hyderabad, India, January.

Sarmad Hussain. 2008. Resources for urdu language processing. In *Proceedings of the Sixth Workshop on Asian Language Resources*, Hyderabad, India, January.

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 306–313.

J.H. Kim and PC Woodland. 2000. A rule-based named entity recognition system for speech input. *Proc. of ICSLP*, pages 521–524.

D. Klein, J. Smarr, H. Nguyen, and C.D. Manning. 2003. Named entity recognition with character-level models. *Proceedings of CoNLL*, 3.

Sobha L. 2008. Invited talk: Named entity recognition: Different approaches. In *Proceedings of the IJCNLP-08 Workshop on NER for South and South East Asian Languages*, pages 13–14, Hyderabad, India, January.

A. McCallum and W. Li. 2003. Early results for named entity recognition with conditional random fields, feature induction and web-enhanced lexicons. *Seventh Conference on Natural Language Learning (CoNLL)*.

A. Mikheev, M. Moens, and C. Grover. 1999. Named Entity recognition without gazetteers. *Proceedings of the ninth conference on European chapter of the Association for Computational Linguistics*, pages 1–8.

Animesh Nayan, B. Ravi Kiran Rao, Pawandeep Singh, Sudip Sanyal, and Ratna Sanyal. 2008. Named entity recognition for indian languages. In *Proceedings of the IJCNLP-08 Workshop on NER for South and South East Asian Languages*, pages 71–78, Hyderabad, India, January. Association for Computational Linguistics.

Praveen P and Ravi Kiran V. 2008. Hybrid named entity recognition system for south and south east asian languages. In *Proceedings of the IJCNLP-08 Workshop on NER for South and South East Asian Languages*, pages 59–62, Hyderabad, India, January.

Vijayakrishna R and Sobha L. 2008. Domain focused named entity recognizer for tamil using conditional random fields. In *Proceedings of the IJCNLP-08 Workshop on NER for South and South East Asian Languages*, pages 93–100, Hyderabad, India, January. Association for Computational Linguistics.

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 and South East Asian Languages*, pages 17–24, Hyderabad, India, January.

E.F.T.K. Sang and F. De Meulder. 2003. Introduction to the CoNLL-2003 Shared Task: Language-Independent Named Entity Recognition. *Development*, 922:1341.

Erik F. Tjong Kim Sang. 2002. Introduction to the conll-2002 shared task: Language-independent named entity recognition. In *Proceedings of CoNLL-2002*, pages 155–158. Taipei, Taiwan.

Manabu Sassano and Takehito Utsuro. 2000. Named entity chunking techniques in supervised learning for japanese named entity recognition. In *Proceedings of the 18th conference on Computational linguistics*, pages 705–711, Morristown, NJ, USA. Association for Computational Linguistics.

B. Settles. 2004. Biomedical Named Entity Recognition Using Conditional Random Fields and Rich Feature Sets. *log*, 1:1.

Praneeth M Shishtla, Karthik Gali, Prasad Pingali, and Vasudeva Varma. 2008a. Experiments in telugu ner: A conditional random field approach. In *Proceedings of the IJCNLP-08 Workshop on NER for South and South East Asian Languages*, pages 79–84, Hyderabad, India, January. Association for Computational Linguistics.

Praneeth M Shishtla, Prasad Pingali, and Vasudeva Varma. 2008b. A character n-gram based approach for improved recall in indian language ner. In *Proceedings of the IJCNLP-08 Workshop on NER for South and South East Asian Languages*, pages 101–108, Hyderabad, India, January. Association for Computational Linguistics.

P Srikanth and Kavi Narayana Murthy. 2008. Named entity recognition for telugu. In *Proceedings of the IJCNLP-08 Workshop on NER for South and South East Asian Languages*, pages 41–50, Hyderabad, India, January.

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)*.

G.D. Zhou and J. Su. 2001. Named entity recognition using an HMM-based chunk tagger. *Proceedings of the 40th Annual Meeting on Association for Computational Linguistics*, pages 473–480.

A Hybrid Approach for Named Entity Recognition in Indian Languages

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

Sanjay Chatterji
Indian Institute of Technology
Kharagpur, West Bengal
India - 721302
sanjay_chatter@yahoo.com

Sandipan Dandapat
Indian Institute of Technology
Kharagpur, West Bengal
India - 721302
sandipan@cse.iitkgp.ernet.in

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

In this paper we describe a hybrid system that applies Maximum Entropy model (MaxEnt), language specific rules and gazetteers to the task of Named Entity Recognition (NER) in Indian languages designed for the IJCNLP NERSSEAL shared task. Starting with Named Entity (NE) annotated corpora and a set of features we first build a baseline NER system. Then some language specific rules are added to the system to recognize some specific NE classes. Also we have added some gazetteers and context patterns to the system to increase the performance. As identification of rules and context patterns requires language knowledge, we were able to prepare rules and identify context patterns for Hindi and Bengali only. For the other languages the system uses the MaxEnt model only. After preparing the one-level NER system, we have applied a set of rules to identify the nested entities. The system is able to recognize 12 classes of NEs with 65.13% f-value in Hindi, 65.96% f-value in Bengali and 44.65%, 18.74%, and 35.47% f-value in Oriya, Telugu and Urdu respectively.

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 (QA), Machine Translation (MT) and in most other NLP applications.

This paper presents a Hybrid NER system for Indian languages which is designed for the IJCNLP NERSSEAL shared task competition, the goal of which is to perform NE recognition on 12 types of NEs - person, designation, title-person, organization, abbreviation, brand, title-object, location, time, number, measure and term.

In this work we have identified suitable features for the Hindi NER task. Orthography features, suffix and prefix information, morphology information, part-of-speech information as well as information about the surrounding words and their tags are used to develop a MaxEnt based Hindi NER system. Then we realized that the recognition of some classes will be better if we apply class specific language rules in addition to the MaxEnt model. We have defined rules for time, measure and number classes. We made gazetteers based identification for designation, title-person and some terms. Also we have used person and location gazetteers as features of MaxEnt for better identification of these classes. Finally we have built a module for semi-automatic extraction of context patterns and extracted context patterns for person, location, organization and title-object classes and these are added to the baseline NER system.

The shared task was defined to build the NER systems for 5 Indian languages - Hindi, Bengali, Oriya, Telugu and Urdu for which training data was pro-

Proceedings of the IJCNLP-08 Workshop on NER for South and South East Asian Languages, pages 17–24,
Hyderabad, India, January 2008.

vided. Among these 5 languages only Bengali and Hindi are known to us but we have no knowledge for other 3 languages. So we are unable to build rules and extract context patterns for these languages. The NER systems for these 3 languages contain only the baseline system i.e. the MaxEnt system. Also our baseline MaxEnt NER system uses morphological and parts-of-speech (POS) information as a feature. Due to unavailability of morphological analyzer and POS tagger for these 3 languages, these information are not added to the systems. Among the 3 languages, only for Oriya NER system we have used small gazetteers for person, location and designation extracted from the training data. For Bengali and Hindi the developed systems are complete hybrid systems containing rules, gazetteers, context patterns and the MaxEnt model.

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. Also a brief survey on nested NE recognition system is presented here. A discussion on the training data is given in Section 3. The MaxEnt based NER system is described in Section 4. Various features used in NER are then discussed. Next we present the experimental results and related discussions in Section 8. Finally Section 9 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 (ML) 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.

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 (HMM) (Bikel et al., 1997), Maximum Entropy (MaxEnt) (Borthwick, 1999), Conditional Random Field (CRF) (Li and Mccallum, 2004) are most common. Combinations of different ML approaches are also used. Srihari et al. (2000) combines MaxEnt, Hidden Markov Model (HMM) 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.

Linguistic approach uses handcrafted 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.

All the NER systems described above are able to detect one-level NEs. In recent years, the interest in detection of nested NEs has increased. Here

we mention few attempts for nested NE detection. Zhou et al. (2004) described an approach to identify cascaded NEs from biomedical texts. They detected the innermost NEs first and then they derived rules to find the other NEs containing these as substrings. Another approach, described by McDonald et al. (2005), uses structural multilevel classification to deal with overlapping and discontinuous entities. B. Gu (2006) has treated the task of identifying the nested NEs a binary classification problem and solved it using support vector machines. For each token in nested NEs, they used two schemes to set its class label: labeling as the outermost entity or the inner entities.

3 Training Data

The data used for the training of the systems was provided. The annotated data uses Shakti Standard Format (SSF). For our development we have converted the SSF format data into the IOB formatted text in which a $B-XXX$ tag indicates the first word of an entity type XXX and $I-XXX$ is used for subsequent words of an entity. The tag O indicates the word is outside of a NE. The training data for Hindi contains more than 5 lakh words, for Bengali about 160K words and about 93K, 64K and 36K words for Oriya, Telugu and Urdu respectively.

In time of development we have observed that the training data, provided by the organizers of the shared task, contains several types of errors in NE tagging. These errors in the training corpora affects badly to the machine learning (ML) based models. But we have not made corrections of the errors in the training corpora in time of our development. All the results shown in the paper are obtained using the provided corpora without any modification in NE annotation.

4 Maximum Entropy Based Model

We have used MaxEnt model to build the baseline NER system. MaxEnt is a flexible statistical model which assigns an outcome for each token based on its history and features. Given a set of features and a training corpus, the MaxEnt estimation process produces a model. For our development we have used a Java based open-nlp MaxEnt toolkit[1] to get the probability values of 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 assignment.

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.

4.1 Features

MaxEnt makes 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 the Indian languages. Also Indian person names are more diverse, lots of common words having other meanings are also used as person names. 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 Indian language NER systems.

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 considered. 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 $w_{i-3}...w_{i+3}$

[1] www.maxent.sourceforge.net

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), '$pradhAnama.ntrI$' (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.

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 NEs. Two types of suffix features have been used. Firstly a fixed length word suffix of the current and surrounding words are used 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 binary features corresponding to the lists - whether a given word has a suffix from a particular list.

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

Root Information of Word: Indian languages are morphologically rich. Words are inflected in various forms depending on its number, tense, person, case etc. Identification of NEs becomes difficult for these inflections. The task becomes easier if instead of the inflected words, corresponding root words are checked whether these are NE or not. For that task we have used morphological analyzers for Hindi and Bengali which are developed at IIT kharagpur.

Parts-of-Speech (POS) Information: The POS of the current word and the surrounding words may be useful feature for NER. We have accessed to Hindi and Bengali POS taggers developed at IIT Kharagpur which has 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 for Hindi. 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 Language Specific Rules

After building of the MaxEnt model we have observed that only a small set of rules are able to identify the classes like number, measure, time, more efficiently than the MaxEnt based model. Then we have tried to define the rules for these classes. The rule identification is done manually and requires language knowledge. We have defined the required rules for Bengali and Hindi but we are unable to do the same for other 3 languages as the languages are unknown to us. In the following we have mentioned some example rules which are defined and used in our system.

- IF ((W_i is a number or numeric word) AND (W_{i+1} is an unit))
  THEN (W_i W_{i+1}) bigram is a $measure$ NE.
- IF ((W_i is a number or numeric word) AND (W_{i+1} is a month-name) AND (W_{i+2} is a 4 digit number))
  THEN (W_i W_{i+1} W_{i+2}) trigram is a $time$ NE.
- IF ((W_i denotes a day of a week) AND (W_{i+1} is a number or numeric word) AND (W_{i+2} is a month name))
  THEN (W_i W_{i+1} W_{i+2}) trigram is a $time$ NE.

We have defined 36 rules in total for time, measure and number classes. These rules use some lists

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

which are built. These lists contain corresponding entries both in the target language and in English. For example the months names list contains the names according to the English calender and the names according to the Indian calender. In the following we have mentioned the lists we have prepared for the rule-based module.

- Names of months.
- Names of seasons.
- Days of a week.
- Names of units.
- Numerical words.

5.1 Semi-automatic Extraction of Context Patterns

Similar to the rules defined for time, measure and date classes, if efficient context patterns (CP) can be extracted for a particular class, these can help in identification of NEs of the corresponding class. But extraction of CP requires huge labour if done manually. We have developed a module for semi-automatically extraction of context patterns. This module makes use of the most frequent entities of a particular class as $seed$ for that class and finds the surrounding tokens of the $seed$ to extract effective patterns. We mark a pattern as 'effective' if the precision of the pattern is very high. Precision of a pattern is defined as the ratio of correct identification and the total identification when the pattern is used to identify NEs of a particular type from a text.

For our task we have extracted patterns for person, location, organization and title-object classes. These patterns are able to identify the NEs of a specific classes but detection of NE boundary is not done properly by the patterns. For boundary detection we have added some heuristics and used POS information of the surrounding words. The patterns for a particular class may identify the NEs of other classes also. For example the patterns for identifying person names may also identify the designation or title-persons. These need to be handled carefully at the time of using patterns. In the following some example patterns are listed which are able to identify person names for Hindi.

- <PER> ne kahA ki
- <PER> kA kathana he.n
- mukhyama.ntrI <PER> Aja
- <PER> ne apane gra.ntha
- <PER> ke putra <PER>

6 Use of Gazetteer Lists

Lists of names of various types are helpful in name identification. Firstly we have prepared the lists using the training corpus. But these are not sufficient. Then we have compiled some specialized name lists from different web sources. But the names in these lists are in English, not in Indian languages. So we have transliterated these English name lists to make them useful for our NER task.

Using transliteration we have constructed several lists. Which are, month name and days of the week, list of common locations, location names list, first names list, middle names list, surnames list etc.

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. 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. 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.

7 Detection of Nested Entities

The training corpora used for the models, are not annotated as nested. The maximal entities are annotated in the training corpus. For detection of the nested NEs, we have derived some rules. For example, if a particular word is a number or numeric word and is a part of a NE type other than 'number', then we have made the nesting. Again, if any common location identifier word like, $jilA$ (district), $shahara$ (town) etc. is a part of a 'location' entity then we have nested there. During one-level NE identification, we have generated lists for all the identified location and person names. Then we have searched other NEs containing these as substring to make the

nesting. After preparing the one-level NER system, we have applied the derived rules on it to identify the nested entities.

8 Evaluation

The accuracies of the system are measured in terms of the f-measure, which is the weighted harmonic mean of precision and recall. Nested, maximal and lexical accuracies are calculated separately. The test data for all the five languages are provided. The size of the shared task test files are: Hindi - 38,704 words, Bengali - 32,796 words, Oriya - 26,988 words, Telugu - 7,076 words and Urdu - 12,805 words.

We have already mentioned that after preparing a one-level NER system, the rule-based module is used to modify it to a nested one. A number of experiments are conducted considering various combinations of features to identify the best feature set for Indian language NER task. It is very difficult and time consuming to conduct experiments for all the languages. During the development we have conducted all the experiments on Hindi and Bengali. We have prepared a development test data composed of 24,265 words for Hindi and 10,902 word for Bengali and accuracies of the system are tested on the development data. The details of the experiments on Hindi data for the best feature selection is described in the following section.

8.1 Best Feature Set Selection

The performance of the system on the Hindi data using various features 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 smaller 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. 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 64.17%. But when prefix information are added the f-value decreased to 63.73%. Again when the context lists are added to the feature set containing words, previous tags, suffix information, digit information and the NomPSP binary feature, the accuracy has decreased to 67.33% from 68.0%.

Feature	**Overall F-value**
Word, NE Tag	58.92
Word, NE Tag, Suffix (≤ 2)	64.17
Word, NE Tag, Suffix (≤ 2), Prefix	63.73
Word, NE Tag, Digit, Suffix	66.61
Word, NE Tag, Context List	63.57
Word, NE Tag, POS (full)	61.28
Word, NE Tag, Suffix (≤ 2), Digit, NomPSP	68.60
Word, NE Tag, Suffix (≤ 2), Digit, Context List, NomPSP	67.33
Word, NE Tag, Suffix (≤ 2), Digit, NomPSP, Linguistic Rules	73.40
Word, NE Tag, Suffix (≤ 2), Digit, NomPSP, Gazetteers	72.08
Word, NE Tag, Suffix (≤ 2), Digit, NomPSP, Linguistic Rules, Gazetteers	74.53

Table 1: Hindi development set f-values for different features

The feature set containing words, previous tags, suffix information, digit information and the NomPSP binary feature is the identified best feature set without linguistic rules and gazetteer information. Then we have added the linguistic rules, patterns and gazetteer information to the system and the changes in accuracies are shown in the table.

8.2 Results on the Test Data

The best identified feature set is used for the development of the NER systems for all the five languages. We have already mentioned that for only for Bengali and Hindi we have added linguistic rules

and gazetteer lists in the MaxEnt based NER systems. The accuracy of the system on the shared task test data for all the languages are shown in Table 2.

Language	Type	Precision	Recall	F-measure
Bengali	Maximal	52.92	68.07	59.54
	Nested	55.02	68.43	60.99
	Lexical	62.30	70.07	65.96
Hindi	Maximal	75.19	58.94	66.08
	Nested	79.58	58.61	67.50
	Lexical	82.76	53.69	65.13
Oriya	Maximal	21.17	26.92	23.70
	Nested	27.73	28.13	27.93
	Lexical	51.51	39.40	44.65
Telugu	Maximal	10.47	9.64	10.04
	Nested	22.05	13.16	16.48
	Lexical	25.23	14.91	18.74
Urdu	Maximal	26.12	29.69	27.79
	Nested	27.99	29.21	28.59
	Lexical	37.58	33.58	35.47

Table 2: Accuracy of the system for all languages

Class	Hindi		Bengali	
	Maximal	Nested	Maximal	Nested
Person	70.87	71.00	77.45	79.09
Designation	48.98	59.81	26.32	26.32
Organization	47.22	47.22	41.43	71.43
Abbreviation	-	72.73	51.61	51.61
Brand	-	-	-	-
Title-person	-	60.00	5.19	47.61
Title-object	41.32	40.98	72.97	72.97
Location	86.02	87.02	76.27	76.27
Time	67.42	67.42	56.30	56.30
Number	84.59	85.13	40.65	40.65
Measure	59.26	55.17	62.50	62.50
Term	48.91	50.51	43.67	43.67

Table 3: Comparison of maximal and nested f-values for different classes of Hindi and Bengali

The accuracies of Oriya, Telugu and Urdu languages are poor compared to the other two languages. The reasons are POS information, morphological information, language specific rules and gazetteers are not used for these languages. Also the size of training data for these languages are smaller. To mention, for Urdu, size of the training data is only about 36K words which is very small to train a MaxEnt model.

It is mentioned that we have prepared a set of rules which are capable of identifying the nested NEs. Once the one-level NER system has built, we have applied the rules on it. In Table 3 we have shown the f-values of each class after addition of the nested rules. The detailed results for all languages are not shown. In the table we have shown only the results of Bengali and Hindi.

For both the languages 'title-person' and 'designation' classes are suffering from poor accuracies. The reason is, in the training data and also in the annotated test data, these classes contains many annotation errors. Also the classes being closely related to each other, the system fails to distinguish them properly. The detection of the 'term' class is very difficult. In the test files amount of 'term' entity is large, for Bengali - 434 and for Hindi - 1080, so the poor accuracy of the class affects badly to the overall accuracy. We have made rule-based identification for 'number', 'measure' and 'time' classes; the accuracies of these classes proves that the rules need to be modified to achieve better accuracy for these classes. Also the accuracy of the 'organization' class is not high, because amount of organization entities is not sufficient in the training corpus. We have achieved good results for other two main classes - 'person' and 'location'.

8.3 Comparison with Other Shared Task Systems

The comparison of the accuracies of our system and other shared task systems is given in Table 4. From the comparison we can see that our system has achieved the best accuracies for most of the languages.

9 Conclusion

We have prepared a MaxEnt based system for the NER task in Indian languages. We have also added

Language	Our	S2	S6	S7
Bengali	65.96	39.77	40.63	59.39
Hindi	65.13	46.84	50.06	33.12
Oriya	44.65	45.84	39.04	28.71
Telugu	18.74	46.58	40.94	4.75
Urdu	35.47	44.73	43.46	35.52

Table 4: Comparison of our lexical f-measure accuracies with the systems : S2 - Praveen P.(2008), S6 - Gali et al.(2008) and S7 - Ekbal et al.(2008)

rules and gazetteers for Bengali and Hindi. Also our derived rules need to be modified for improvement of the system. We have not made use of rules and gazetteers for Oriya, Telugu and Urdu. As the size of training data is not much for these 3 languages, rules and gazetteers would be effective. We have experimented with MaxEnt model only, other ML methods like HMM, CRF or MEMM may be able to give better accuracy. We have not worked much on the detection of nested NEs. Proper detection of nested entities may lead to further improvement of performance and is under investigation.

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*, 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*, 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*.

Ekbal A., Haque R., Das A., Poka V. and Bandyopadhyay S. 2008. Language Independent Named Entity Recognition in Indian Languages In *Proceedings of IJCNLP workshop on NERSSEAL. (Accepted)*

Gali K., Surana H., Vaidya A., Shishtla P. and Misra Sharma D. 2008. Named Entity Recognition through CRF Based Machine Learning and Language Specific Heuristics In *Proceedings of IJCNLP workshop on NERSSEAL. (Accepted)*

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

Gu B. 2006. Recognizing Nested Named Entities in GENIA corpus. In *Proceedings of the BioNLP Workshop on Linking Natural Language Processing and Biology at HLT-NAACL 06*, pages 112-113.

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). In *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*, 21–39.

McDonald R., Crammer K. and Pereira F. 2005. Flexible text segmentation with structured multilabel classification. In *Proceedings of EMNLP05*.

Praveen P. 2008. Hybrid Named Entity Recognition System for South-South East Indian Languages. In*Proceedings of IJCNLP workshop on NERSSEAL. (Accepted)*

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*.

Zhou G., Zhang J., Su J., Shen D. and Tan C. 2004. Recognizing Names in Biomedical Texts: a Machine Learning Approach. *Bioinformatics*, 20(7):1178-1190.

Aggregating Machine Learning and Rule Based Heuristics for Named Entity Recognition

Karthik Gali, Harshit Surana, Ashwini Vaidya, Praneeth Shishtla and Dipti Misra Sharma
Language Technologies Research Centre,
International Institute of Information Technology,
Hyderabad, India.
karthikg@students.iiit.ac.in, surana.h@gmail.com,
ashwini_vaidya@research.iiit.ac.in, praneethms@students.iiit.ac.in,
dipti@iiit.ac.in

Abstract

This paper, submitted as an entry for the NERSSEAL-2008 shared task, describes a system build for Named Entity Recognition for South and South East Asian Languages. Our paper combines machine learning techniques with language specific heuristics to model the problem of NER for Indian languages. The system has been tested on five languages: Telugu, Hindi, Bengali, Urdu and Oriya. It uses CRF (Conditional Random Fields) based machine learning, followed by post processing which involves using some heuristics or rules. The system is specifically tuned for Hindi and Telugu, we also report the results for the other four languages.

1 Introduction

Named Entity Recognition (NER) is a task that seeks to locate and classify entities ('atomic elements') in a text into predefined categories such as the names of persons, organizations, locations, expressions of times, quantities, etc. It can be viewed as a two stage process:

1. Identification of entity boundaries
2. Classification into the correct category

For example, if "Mahatma Gandhi" is a named entity in the corpus, it is necessary to identify the beginning and the end of this entity in the sentence. Following this step, the entity must be classified into the predefined category, which is NEP (Named Entity Person) in this case.

This task is the precursor for many natural language processing applications. It has been used in Question Answering (Toral et al, 2005) as well as Machine Translation (Babych et al, 2004).

The NERSSEAL contest has used 12 categories of named entities to define a tagset. The data has been manually tagged for training and testing purposes for the contestants.

The task of building a named entity recognizer for South and South East Asian languages presents several problems related to their linguistic characteristics. We will first discuss some of these linguistic issues, followed by a description of the method used. Further, we show some of the heuristics used for post-processing and finally an analysis of the results obtained.

2 Previous Work

The linguistic methods generally 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 upto 92% f-measure accuracy for English (McDonald, 1996; Wakao et al., 1996).

Linguistic approach uses hand-crafted rules which need skilled linguistics. The chief disadvantage of these rule-based techniques is that they require huge experience and grammatical knowledge of the particular language or domain and these systems are not transferable to other languages or domains. However, given the closer nature of many Indian languages, the cost of adaptation of a re-

Proceedings of the IJCNLP-08 Workshop on NER for South and South East Asian Languages, pages 25–32,
Hyderabad, India, January 2008.

source from one language to another could be quite less (Singh and Surana, 2007).

Various machine learning techniques have also been successfully used for the NER task. Generally hidden markov model (Bikel et al.,1997), maximum entropy (Borthwick, 1999), conditional random field (Li and Mccallum, 2004) are more popular machine learning techniques used for the purpose of NER.

Hybrid systems have been generally more effective at the task of NER. Given lesser data and more complex NE classes which were present in NERSSEAL shared task, hybrid systems make more sense. Srihari et al. (2000) combines MaxEnt, hidden markov model (HMM) and handcrafted rules to build an NER system.

Though not much work has been done for other South Asian languages, some previous work focuses on NER for Hindi. It has been previously attempted 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 different languages. Among these the accuracy for Hindi was the worst. For Hindi the system achieved 42% f-value with a recall of 28% and about 85% precision. A result which highlights lack of good training data, and other various issues involved with linguistic handling of Indian languages.

Later approaches have resulted in better results for Hindi. Hindi NER system developed by Wei Li and Andrew Mccallum (2004) using conditional random fields (CRFs) with feature induction have achieved f-value of 71%. (Kumar and Bhattacharyya, 2006) used maximum entropy markov model to achieve f-value of upto 80%.

3 Some Linguistic Issues

3.1 Agglutinative Nature

Some of the SSEA languages have agglutinative properties. For example, a Dravidian language like Telugu has a number of postpositions attached to a stem to form a single word. An example is:

guruvAraMwo = guruvAraM + wo
up to Wednesday = Wednesday + up to

Most of the NERs are suffixed with several different postpositions, which increase the number of distinct words in the corpus. This in turn affects the machine learning process.

3.2 No Capitalization

All the five languages have scripts without graphical cues like capitalization, which could act as an important indicator for NER. For a language like English, the NER system can exploit this feature to its advantage.

3.3 Ambiguity

One of the properties of the named entities in these languages is the high overlap between common names and proper names. For instance *Kamal* (in Hindi) can mean 'lotus', which is not a named entity, but it can also be a person's name, in which case, it is a named entity.

Among the named entities themselves, there is ambiguity between a location name ***Bangalore*** *ek badzA shaher heI* (Bangalore is a big city) or a person's surname '*M.* ***Bangalore*** *shikshak heI*' (M. Bangalore is a teacher).

3.4 Low POS Tagging Accuracy for Nouns

For English, the available tools like POS (Part of Speech) tagger can be used to provide features for machine learning. This is not very helpful for SSEA languages because the accuracy for noun and proper noun tags is quite low (PVS and G., 2006) Hence, features based on POS tags cannot be used for NER for these languages.

To illustrate this difficulty, we conducted the following experiment. A POS tagger (described in PVS & G.,2006) was run on the Hindi test data. The data had 544 tokens with NEL, NEP, NEO tags. The POS tagger should have given the NNP (proper noun) tag for all those named entities. However the tagger was able to tag only 80 tokens accurately. This meant that only 14.7% of the named entities were correctly recognized.

3.5 Spelling Variation

One other important language related issue is the variation in the spellings of proper names. For instance the same name *Shri Ram Dixit* can be written as *Sri. Ram Dixit*, *Shree Ram Dixit*, *Sh. R. Dixit* and so on. This increases the number of tokens to be learnt by the machine and would perhaps also require a higher level task like co-reference resolution.

2.6 Pattern of suffixes

Named entities of Location (NEL) or Person (NEP) will share certain common suffixes, which can be exploited by the learning algorithm. For instance, in Hindi, ***-pur*** (*Rampur*, *Manipur*) or ***-giri*** (*Devgiri*) are suffixes that will appear in the named entities for Location. Similarly, there are suffixes like ***-swamy*** (*Ramaswamy*, *Krishnaswamy*) or ***-deva*** (*Vasudeva*, *Mahadeva*) which can be commonly found in named entities for person. These suffixes are cues for some of the named entities in the SSEA languages.

A NER system can be rule-based, statistical or hybrid. A rule-based NER system uses handwritten rules to tag a corpus with named entities. A statistical NER system learns the probabilities of named entities using training data, whereas hybrid systems use both.

Developing rule-based taggers for NER can be cumbersome as it is a language specific process. Statistical taggers require large amount of annotated data (the more the merrier) to train. Our system is a hybrid NER tagger which first uses Conditional Random Fields (CRF) as a machine learning technique followed by some rule based post-processing.

We treat the named entity recognition problem as a sequential token-based tagging problem.

According to Lafferty et. al. CRF outperforms other Machine Learning algorithms viz., Hidden Markov Models (HMM), Maximum Entropy Markov Model (MEMM) for sequence labeling tasks.

4 Training data

The training data given by the organizers was in SSF format[1]. For example in SSF format, the named entity 'Rabindranath Tagore' will be shown in the following way:

```
0       ((      SSF
1       ((      NP      <ne=NEP>
1.1     Rabindranath
1.2     Tagore
        ))
2       ne
3       kahaa
        ))
```

1 http://shiva.iiit.ac.in/SPSAL2007/ssf-analysis-representation.pdf

We have converted this format into the BIO format as described in Ramshaw et. al. For example, the above format will now be shown as:

Rabindranath	B-NEP
Tagore	I-NEP
ne	O
kahaa	O

The training data set contains (approximately) 400,000 Hindi, 50,000 Telugu, 35,000 Urdu, 93,000 Oriya and 120,000 Bengali words respectively.

5 Conditional Random Fields

Conditional Random Fields (CRFs) are undirected graphical models used to calculate the conditional probability of values on designated output nodes given values assigned to other designated input nodes.

In the special case in which the output nodes of the graphical model are linked by edges in a linear chain, CRFs make a first-order Markov independence assumption, and thus can be understood as conditionally-trained finite state machines (FSMs). Let $o = (o,,o_2,o_3 ,o_4 ,... o_T)$ be some observed input data sequence, such as a sequence of words in text in a document,(the values on n input nodes of the graphical model). Let S be a set of FSM states, each of which is associated with a label, l € £.

Let $s = (s_1,s_2,s_3 ,s_4 ,... s_T)$ be some sequence of states, (the values on T output nodes). By the Hammersley-Clifford theorem, CRFs define the conditional probability of a state sequence given an input sequence to be:

$$P(s|o) = \frac{1}{Z_o} * exp(\sum_{t=1}^{T} \sum_{k} \lambda_k f_k (s_{t-1}, s_t, o, t))$$

where Z_o is a normalization factor over all state sequences is an arbitrary feature function over its arguments, and λ_k is a learned weight for each feature function. A feature function may, for example, be defined to have value 0 or 1. Higher λ weights make their corresponding FSM transitions more likely. CRFs define the conditional probability of a label sequence based on the total probability over the state sequences,

$$P(l|o) = \sum_{s:l(s)=l} P(s|o)$$

	Precision			Recall			F-Measure		
	Pm	**Pn**	**Pl**	**Rm**	**Rn**	**Rl**	**Fm**	**Fn**	**Fl**
Bengali	53.34	49.28	58.27	26.77	25.88	31.19	35.65	33.94	40.63
Hindi	59.53	63.84	64.84	41.21	41.74	40.77	48.71	50.47	50.06
Oriya	39.16	40.38	63.70	23.39	19.24	28.15	29.29	26.06	39.04
Telugu	10.31	71.96	65.45	68.00	30.85	29.78	08.19	43.19	40.94
Urdu	43.63	44.76	48.96	36.69	34.56	39.07	39.86	39.01	43.46

Table 1: Evaluation of the NER System for Five Languages

where *l(s)* is the sequence of labels corresponding to the labels of the states in sequence s.

Note that the normalization factor, Z_o, (also known in statistical physics as the partition function) is the sum of the scores of all possible states.

$$Z_o = \sum_{s \in S^T} \exp\left(\sum_{t=1}^{T} \sum_{k} \lambda_k f_k(s_{t-1}, s_t, o, t) \right),$$

And that the number of state sequences is exponential in the input sequence length, T. In arbitrarily-structured CRFs, calculating the partition function in closed form is intractable, and approximation methods such as Gibbs sampling or loopy belief propagation must be used. In linear-chain structured CRFs (in use here for sequence modeling), the partition function can be calculated efficiently by dynamic programming.

6 CRF Based Machine Learning

We used the CRF model to perform the initial tagging followed by post-processing.

6.1 Statistical Tagging

In the first phase, we have used language independent features to build the model using CRF. Orthographic features (like capitalization, decimals), affixes (suffixes and prefixes), context (previous words and following 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 as explained above this feature is not applicable for the Indian languages.

The exact set of features used are described below.

6.2 Window of the Words

Words preceding or following the target word may be useful for determining its category. Following a few trials we found that a suitable window size is five.

6.3 Suffixes

Statistical suffixes of length 1 to 4 have been considered. These can capture information for named entities having the NEL tag like *Hyderabad*, *Secunderabad*, *Ahmedabad* etc., all of which end in *-bad*. We have collected lists of such suffixes for NEP (Named Entity Person) and NEL (Named Entity Location) for Hindi. In the machine learning model, this resource can be used as a binary feature. A sample of these lists is as follows:

Type of NE	**Example suffixes (Hindi)**
NE- Location	-desa, -vana, -nagara, -garh, -rashtra, -giri
NE – Person	-raja, -natha, -lal, -bhai,-pathi, -krishnan

Table 2: Suffixes for Hindi NER

6.4 Prefixes

Statistical prefixes of length 1 to 4 have been considered. These can take care of the problems associated with a large number of distinct tokens. As mentioned earlier, agglutinative languages can have a number of postpositions. The use of prefixes will increase the probability of *Hyderabad* and *Hyderabadlo* (Telugu for 'in Hyderabad') being treated as the same token.

	Bengali	Hindi	Oriya	Telugu	Urdu
NEP	35.22	54.05	52.22	01.93	31.22
NED	NA	42.47	01.97	NA	21.27
NEO	11.59	45.63	14.50	NA	19.13
NEA	NA	61.53	NA	NA	NA
NEB	NA	NA	NA	NA	NA
NETP	42.30	NA	NA	NA	NA
NETO	33.33	13.77	NA	01.66	NA
NEL	45.27	62.66	48.72	01.49	57.85
NETI	55.85	79.09	40.91	71.35	63.47
NEN	62.67	80.69	24.94	83.17	13.75
NEM	60.51	43.75	19.00	26.66	84.10
NETE	19.17	31.52	NA	08.91	NA

Table 3: F-Measure (Lexical) for NE Tags

6.5 Start of a sentence

There is a possibility of confusing the NEN (Named Entity Number) in a sentence with the number that appears in a numbered list. The numbered list will always have numbers at the beginning of a sentence and hence a feature that checks for this property will resolve the ambiguity with an actual NEN.

6.6 Presence of digits

Usually, the presence of digits indicates that the token is a named entity. For example, the tokens *92, 10.1* will be identified as Named Entity Number based on the binary feature 'contains digits'.

6.7 Presence of four digits

If the token is a four digit number, it is likelier to be a NETI (Named Entity Time). For example, 1857, 2007 etc. are most probably years.

7 Heuristics Based Post Processing

Complex named entities like *fifty five kilograms* contain a Named Entity Number within a Named Entity Measure. We observed that these were not identified accurately enough in the machine learning based system. Hence, instead of applying machine learning to handle nested entities we make use of rule-based post processing.

7.1 Second Best Tag

It was observed that the recall of the CRF model is low. In order to improve recall, we have used the following rule: if the best tag given by the CRF model is O (not a named entity) and the confidence of the second best tag is greater than 0.15, then the second best tag is considered as the correct tag.

We observed an increase of 7% in recall and 3% decrease in precision. This resulted in a 4% increase in the F-measure, which is a significant increase in performance. The decrease in precision is expected as we are taking the second tag.

7.2 Nested Entities

One of the important tasks in the contest was to identify nested named entities. For example if we consider *eka kilo* (Hindi: one kilo) as NEM (Named Entity Measure), it contains a NEN (Named Entity Number) within it.

The CRF model tags *eka kilo* as NEM and in order to tag *eka* as NEN we have made use of other resources like a gazetteer for the list of numbers. We used such lists for four languages.

7.3 Gazetteers

For Hindi, we made use of three different kinds of gazetteers. These consisted of lists for measures (entities like kilogram, millimetre, lakh), numerals and quantifiers (one, first, second) and time expressions (January, minutes, hours) etc. Similar lists were used for all the other languages except Urdu. These gazetteers were effective in identifying this relatively closed class of named entities and showed good results for these languages.

8 Evaluation

The evaluation measures used for all the five languages are precision, recall and F-measure. These measures are calculated in three different ways:

1. **Maximal Matches**: The largest possible named entities are matched with the reference data.
2. **Nested Matches**: The largest possible as well as nested named entities are matched.
3. **Lexical Item Matches**: The lexical items inside largest possible named entities are matched.

9 Results

The results of evaluation as explained in the previous section are shown in the Table-1. The F-measures for nested lexical match are also shown individually for each named entity tag separately in Table-3

10 Unknown Words

Table 4 shows the number of unknown words present in the test data when compared with the training data.

First column shows the number of unique Named entity tags present in the test data for each language. Second column shows the number of unique known named entities present in the test data. Third column shows the percentage of unique unknown words present in the test data of different languages when compared to training data.

11 Error Analysis

We can observe from the results that the maximal F-measure for Telugu is very low when compared to lexical F-measure and nested F-measure. The reason is that the test data of Telugu contains a large number of long named entities (around 6 words), which in turn contain around 4 - 5 nested named entities. Our system was able to tag nested named entities correctly unlike maximal named entity.

We can also observe that the maximal F-measure for Telugu is very low when compared to other languages. This is because Telugu test data has very few known words.

Urdu results are comparatively low chiefly because gazetteers for numbers and measures were unavailable.

The amount of annotated corpus available for Hindi was substantially more. This should have ideally resulted in better results for Hindi with the machine learning approach. But, the results were only marginally better than other languages. A major reason for this was that a very high percentage (44%) of tags in Hindi were NETE. The tagset gives examples like 'Horticulture', 'Conditional Random Fields' for the tag NETE. It has also been mentioned that even manual annotation is harder for NETE as it is domain specific. This affected the overall results for Hindi because the performance for NETE was low (Table 3).

	Num of NE tokens	***Num of known NE***	***% of unknown NE***
Bengali	1185	277	23.37
Hindi	1120	417	37.23
Oriya	1310	563	42.97
Telugu	1150	145	12.60
Urdu	631	179	28.36

Table 4: Unknown Word

Also, the F-measures of NEN, NETI, and NEM could have been higher because they are relatively closed classes. However, certain NEN can be ambiguous (Example: *eka* is a NEN for 'one' in Hindi, but in a different context it can be a non-number. For instance *eka-doosra* is Hindi for 'each other').

In a language like Telugu, NENs will appear as inflected words. For example 2*001lo, guruvaaramto.*

10 Conclusion and Further Work

In this paper we have presented the results of using a two stage hybrid approach for the task of named entity recognition for South and South East Asian Languages. We have achieved decent Lexical F-measures of 40.63, 50.06, 39.04, 40.94, and 43.46 for Bengali, Hindi, Oriya, Telugu and Urdu respectively without using many language specific resources.

We plan to extend our work by applying our method to other South Asian languages, and by using more language specific constraints and resources. We also plan to incorporate semi-supervised extraction of rules for NEs (Saha et. al,

2008) and use transliteration techniques to produce Indian language gazetteers (Surana and Singh, 2008). Use of character models for increasing the lower recalls (Shishtla et. al, 2008) is also underway. We also plan to enrich the Indian dependency tree bank (Begum et. al, 2008) by use of our NER system.

11 Acknowledgments

We would like to thank the organizer Mr. Anil Kumar Singh deeply for his continuous support during the shared task.

References

B. Babych, and A. Hartley, Improving Machine translation Quality with Automatic Named Entity Recognition. www.mt-archive.info/EAMT-2003- Babych.pdf

Rafiya Begum, Samar Husain, Arun Dhwaj, Dipti Misra Sharma, Lakshmi Bai, and Rajeev Sangal. 2008. Dependency annotation scheme for Indian languages. In *Proceedings of IJCNLP-2008*, Hyderabad, India.

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

S. Cucerzan, and D. Yarowsky, 1999. Language independent named entity recognition combining morphological and contextual evidence. *Proceedings of the Joint SIGDAT Conference on EMNLP and VLC.*

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

John Lafferty, Andrew McCallum and Fernando Pereira. 2001. Conditional Random Fields: Probabilistic Models for Segmenting and Labeling Sequence Data. *Proc. 18th International Conf. on Machine Learning*.

D. McDonald 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.*

Avinesh PVS and Karthik G. Part-Of-Speech Tagging and Chunking Using Conditional Random Fields and Transformation Based Learning. *Proceedings of the SPSAL workshop during IJCAI'07*.

Lance Ramshaw and Mitch Marcus. Text Chunking Using Transformation-Based Learning. Proceedings of the Third Workshop on Very Large Corpora.

S.K. Saha , S. Chatterji , S. Dandapat , S. Sarkar and P. Mitra 2008. A Hybrid Approach for Named Entity Recognition in Indian Languages. *In Proceedings of IJCNLP Workshop on NER for South and South East Asian Languages.*

Fei Sha and Fernando Pereira. 2003. Shallow Parsing with Conditional Random Fields. *In the Proceedings of HLT-NAACL*.

P. Shishtla, P. Pingali , V. Varma 2008. A Character n-gram Based Approach for Improved Recall in Indian Language NER. *In Proceedings of IJCNLP Workshop on NER for South and South East Asian Languages.*

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.*

A. K. Singh and H. Surana Can Corpus Based Measures be Used for Comparative Study of Languages? *In Proceedings of Ninth Meeting of the ACL Special Interest Group in Computational Morphology and Phonology*. ACL. 2007.

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

H. Surana and A. K. Singh 2008. A More Discerning and Adaptable Multilingual Transliteration Mechanism for Indian Languages. *In Proceedings of the Third International Joint Conference on Natural Language Processing.*

Charles Sutton, *An Introduction to Conditional Random Fields for Relational Learning*.

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

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.*

CRF++:.Yet another Toolkit.
http://crfpp.sourceforge.net/

Language Independent Named Entity Recognition in Indian Languages

Asif Ekbal, Rejwanul Haque, Amitava Das, Venkateswarlu Poka and Sivaji Bandyopadhyay
Department of Computer Science and Engineering
Jadavpur University
Kolkata-700032, India
asif.ekbal@gmail.com, rejwanul@gmail.com, amit_santu_kuntal@yahoo.com, venkat.ju@gmail.com and sivaji_cse_ju@yahoo.com

Abstract

This paper reports about the development of a Named Entity Recognition (NER) system for South and South East Asian languages, particularly for Bengali, Hindi, Telugu, Oriya and Urdu as part of the IJCNLP-08 NER Shared Task[1]. We have used the statistical Conditional Random Fields (CRFs). The system makes use of the different contextual information of the words along with the variety of features that are helpful in predicting the various named entity (NE) classes. The system uses both the language independent as well as language dependent features. The language independent features are applicable for all the languages. The language dependent features have been used for Bengali and Hindi only. One of the difficult tasks of IJCNLP-08 NER Shared task was to identify the nested named entities (NEs) though only the type of the maximal NEs were given. To identify nested NEs, we have used rules that are applicable for all the five languages. In addition to these rules, gazetteer lists have been used for Bengali and Hindi. The system has been trained with Bengali (122,467 tokens), Hindi (502,974 tokens), Telugu (64,026 tokens), Oriya (93,173 tokens) and Urdu (35,447 tokens) data. The system has been tested with the 30,505 tokens of Bengali, 38,708 tokens of Hindi, 6,356 tokens of Telugu, 24,640 tokens of Oriya and 3,782 tokens of Urdu. Evaluation results have demonstrated the highest maximal F-measure of 53.36%, nested F-measure of 53.46% and lexical F-measure of 59.39% for Bengali.

1 Introduction

Named Entity Recognition (NER) is an important tool in almost all Natural Language Processing (NLP) application areas. Proper identification and classification of named entities are very crucial and pose a very big challenge to the NLP researchers. The level of ambiguity in named entity recognition (NER) makes it difficult to attain human performance.

NER has drawn more and more attention from the named entity (NE) tasks (Chinchor 95; Chinchor 98) in Message Understanding Conferences (MUCs) [MUC6; MUC7]. The problem of correct identification of named entities is specifically addressed and benchmarked by the developers of Information Extraction System, such as the GATE system (Cunningham, 2001). NER also finds application in question-answering systems (Maldovan et al., 2002) and machine translation (Babych and Hartley, 2003).

The current trend in NER is to use the machine-learning approach, which is more attractive in that it is trainable and adoptable and the maintenance of a machine-learning system is much cheaper than that of a rule-based one. The representative machine-learning approaches used in NER are HMM (BBN's IdentiFinder in (Bikel, 1999)), Maximum Entropy

[1]http://ltrc.iiit.ac.in/ner-ssea-08

Proceedings of the IJCNLP-08 Workshop on NER for South and South East Asian Languages, pages 33–40,
Hyderabad, India, January 2008.

(New York University's MENE in (Borthwick, 1999)), Decision Tree (New York University's system in (Sekine 1998), SRA's system in (Bennet, 1997) and Conditional Random Fields (CRFs) (Lafferty et al., 2001; McCallum and Li, 2003).

There is no concept of capitalization in Indian languages (ILs) like English and this fact makes the NER task more difficult and challenging in ILs. There has been very little work in the area of NER in Indian languages. In Indian languages particularly in Bengali, the work in NER can be found in (Ekbal and Bandyopadhyay, 2007a) and (Ekbal and Bandyopadhyay, 2007b). These two systems are based on the pattern directed shallow parsing approach. An HMM-based NER in Bengali can be found in (Ekbal et al., 2007c). Other than Bengali, the work on NER can be found in (Li and McCallum, 2004) for Hindi. This system is based on CRF.

In this paper, we have reported a named entity recognition system for the south and south east Asian languages, particularly for Bengali, Hindi, Telugu, Oriya and Urdu. Bengali is the seventh popular language in the world, second in India and the national language of Bangladesh. Hindi is the third popular language in the world and the national language of India. Telugu is one of the popular languages and predominantly spoken in the southern part of India. Oriya and Urdu are the other two popular languages of India and widely used in the eastern and the northern part, respectively. The statistical Conditional Random Field (CRF) model has been used to develop the system, as it is more efficient than HMM to deal with the non-independent and diverse overlapping features of the highly inflective Indian languages. We have used a fine-grained named entity tagset[2], defined as part of the IJCNLP-08 NER Shared Task for SSEA. The system makes use of the different contextual information of the words along with the variety of orthographic word level features that are helpful in predicting the various named entity classes. In this work, we have considered language independent features as well as the language dependent features. Language independent features include the contextual words, prefix and suffix information of all the words in the training corpus, several digit features depending upon the presence and/or the number of digits in a token and the frequency features of the words. The system considers linguistic features particularly for Bengali and Hindi. Linguistic features of Bengali include the set of known suffixes that may appear with named entities, clue words that help in predicating the location and organization names, words that help to recognize measurement expressions, designation words that help in identifying person names, the various gazetteer lists like the first names, middle names, last names, location names and organization names. As part of linguistic features for Hindi, the system uses only the lists of first names, middle names and last names along with the list of words that helps to recognize measurements. No linguistic features have been considered for Telugu, Oriya and Urdu. It has been observed from the evaluation results that the use of linguistic features improves the performance of the system. A number of experiments have been carried out to find out the best-suited set of features for named entity recognition in Bengali, Hindi, Telugu, Oriya and Urdu.

[2]http://ltrc.iiit.ac.in/ner-ssea-08/index.cgi?topic=3

2 Conditional Random Fields

Conditional Random Fields (CRFs) (Lafferty et al., 2001) are undirected graphical models, a special case of which corresponds to conditionally trained probabilistic finite state automata. Being conditionally trained, these CRFs can easily incorporate a large number of arbitrary, non-independent features while still having efficient procedures for non-greedy finite-state inference and training. CRFs have shown success in various sequence modeling tasks including noun phrase segmentation (Sha and Pereira, 2003) and table extraction (Pinto et al., 2003).

CRFs are used to calculate the conditional probability of values on designated output nodes given values on other designated input nodes. The conditional probability of a state sequence $S = < s_1, s_2, \ldots, s_T >$ given an observation sequence $O = < o_1, o_2, \ldots\ldots, o_T)$ is calculated as:

$P_\Lambda(s \mid o) = \frac{1}{Z_o} \exp(\sum_{t=1}^{T} \sum_{k} \lambda_k f_k(s_{t-1}, s_t, o, t))$, where $f_k(s_{t-1}, s_t, o, t)$ is a feature function whose weight λ_k is to be learned via training. The values of the feature functions may range between $-\propto \ldots\ldots + \propto$, but typically they are binary. To make all

conditional probabilities sum up to 1, we must calculate the normalization factor, $Z_0 = \sum_s \exp(\sum_{t=1}^{T} \sum_k \lambda_k f_k(s_{t-1}, s_t, o, t))$, which, as in HMMs, can be obtained efficiently by dynamic programming.

To train a CRF, the objective function to be maximized is the penalized log-likelihood of the state sequences given observation sequences:

$$L_\wedge = \sum_{i=1}^{N} \log(P_\wedge(s^{(i)} \mid o^{(i)})) - \sum_k \frac{\lambda_k^2}{2\sigma^2},$$

where, $\{ < o^{(i)}, s^{(i)} > \}$ is the labeled training data. The second sum corresponds to a zero-mean, σ^2 -variance Gaussaian prior over parameters, which facilitates optimization by making the likelihood surface strictly convex. Here, we set parameters λ to maximize the penalized log-likelihood using Limited-memory BFGS (Sha and Pereira, 2003), a quasi-Newton method that is significantly more efficient, and which results in only minor changes in accuracy due to changes in σ .

When applying CRFs to the named entity recognition problem, an obsevation sequence is a token of a sentence or document of text and the state sequence is its corresponding label sequence. While CRFs generally can use real-valued functions, in our experiments maximum of the features are binary. A feature function $f_k(s_{t-1}, s_t, o, t)$ has a value of 0 for most cases and is only set to be 1, when s_{t-1}, s_t are certain states and the observation has certain properties. We have used the C^{++} based OpenNLP CRF++ package[3], a simple, customizable, and open source implementation of Conditional Random Fields (CRFs) for segmenting /labeling sequential data.

3 Named Entity Recognition in Indian Languages

Named Entity Recognition in Indian languages (ILs) is difficult and challenging as capitalization is not a clue in ILs. The training data were provided for five different Indian languages, namely Bengali, Hindi, Telugu, Oriya and Urdu in *Shakti Standard Format*[4]. The training data in all the languages were annotated with the twelve NE tags, as defined for the IJCNLP-08 NER shared task taget[5]. Only the maximal named entities and not the internal structures of the entities were annotated in the training data. For example, *mahatma gandhi road* was annotated as location and assigned the tag 'NEL' even if *mahatma* and *gandhi* are named entity title person (NETP) and person name (NEP) respectively, according to the IJCNLP-08 shared task tagset. These internal structures of the entities were to be identified during testing. So, *mahatma gandhi road* will be tagged as *mahatma* /NETP *gandhi*/NEP *road*/NEL. The structure of the tagged element using the *SSF* form will be as follows:

```
1       ((      NP      <ne=NEL>
1.1     ((      NP      <ne=NEP>
1.1.1   ((      NP      <ne=NETP>
1.1.1.1 mahatma
        ))
1.1.2   gandhi
        ))
1.2     road
))
```

3.1 Training Data Preparation for CRF

Training data for all the languages required some preprocessing in order to use in the Conditional Random Field framework. The training data is searched for the multiword NEs. Each component of the multiword NE is searched in the training set to find whether it occurs as a single-word NE. The constituent components are then replaced by their NE tags (NE type of the single-word NE). For example, *mahatma gandhi road*/NEL will be tagged as *mahatma*/NETP *gandhi*/NEP *road*/NEL if the internal components are found to appear with these NE tags in the training set. Each component of a multiword NE is also checked whether the component is made up of digits only. If a component is made up digits only, then it is assigned the tag 'NEN'. Various gazetteers for Bengali and Hindi have been also used in order to identify the internal structure of the NEs properly. The list of gazetteers, which have been used in preparing the training data, is shown in Table 1.

The individual components (not occurring as a single-word NE in the training data) of a multiword NE are searched in the gazetteer lists and

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

[4]http://shiva.iiit.ac.in/SPSAL 2007/ssf.html

[5]http://ltrc.iiit.ac.in/ner-ssea-08/index.cgi?topic=3

assigned the appropriate NE tags. Other than NEs are marked with the NNE tags. The procedure is given below:

Gazetteer list	Number of entries
First person name in Bengali	27,842
Last person name in Bengali	5,288
Middle name in Bengali	1,491
Person name designation in Bengali	947
Location name in Bengali	7,870
First person name in Hindi	1,62,881
Last person name in Hindi	3,573
Middle name in Hindi	450
Cardinals in Bengali, Hindi and Telugu	100
Ordinals in Bengali, Hindi and Telugu	65
Month names in Bengali, Hindi and Telugu	24
Weekdays in Bengali, Hindi and Telugu	14
Words that denote measurement in Bengali, Hindi and Telugu	52

Table 1. Gazetteer lists used during training data preparation

Step 1: Search the multiword NE in the training data
Step 2: Extract each component from the multword NE.
Step 3: Check whether the constituent individual component (except the last one) appears in the training data as a single-word NE.
Step 4: If the constituent NE appears in the training data as a single-word NE then
Step 4.1: Assign the NE tag, extracted from the single-word NE, to the component of the multiword NE.
else
Step 4.2: Search the component in the gazetteer lists and assign the appropriate NE tag.
Step 4.2.1: If the component is not found to appear in the gazetteer list then assign the NE tag of the maximal NE to the individual component.

For example, if *mahatma gandhi road* is tagged as NEL, i.e., *mahatma gandhi road*/NEL then each component except the last one (*road*) of this multword NE is searched in the training set to look for it's appearance (Step 3). Gazetteer lists are searched in case the component is not found in the training set (Step 4.2). If the components are found either in the training set or in the gazetteer list, then *mahatma gandhi road*/NEL will be tagged as: *mahatma*/NETP *gandhi*/NEP *road*/NEL.

3.2 Named Entity Features

Feature selection plays a crucial role in CRF framework. Experiments were carried out to find out most suitable features for NE tagging task. The main features for the NER task have been identified based on the different possible combination of available word and tag context. The features also include prefix and suffix for all words. The term prefix/suffix is a sequence of first/last few characters of a word, which may not be a linguistically meaningful prefix/suffix. The use of prefix/suffix information works well for highly inflected languages like the Indian languages. In addition, various gazetteer lists have been developed to use in the NER task particularly for Bengali and Hindi. We have considered different combination from the following set for inspecting the best feature set for the NER task:

F={ $w_{i-m},...,w_{i-1},w_i,w_{i+1},...,w_{i+n}$, |prefix|≤n, |suffix|≤n, previous NE tag, POS tags, First word, Digit information, Gazetteer lists}

Following is the details of the set of features that were applied to the NER task:

• Context word feature: Previous and next words of a particular word might be used as a feature. We have considered the word window of size five, i.e., previous and next two words from the current word for all the languages.

•Word suffix: Word suffix information is helpful to identify NEs. A fixed length word suffix of the current and surrounding words might be treated as feature. In this work, suffixes of length up to three the current word have been considered for all the languages. More helpful approach is to modify the feature as binary feature. Variable length suffixes of a word can be matched with predefined lists of useful suffixes for different classes of NEs. For Bengali, we have considered the different suffixes that may be particularly helpful in detecting person (e.g., *-babu, -da, -di* etc.).

•Word prefix: Prefix information of a word is also helpful. A fixed length prefix of the current and the surrounding words might be treated as features. Here, the prefixes of length up to three have been considered for all the language.

•Rare word: The lists of most frequently occurring words in the training sets have been calculated for all the five languages. The words that occur more than 10 times are considered as the frequently occurring words in Bengali and Hindi. For Telugu, Oriya and Urdu, the cutoff frequency was chosen to be 5. Now, a binary feature 'RareWord' is defined as: If current word is found to appear in the frequent word list then it is set to 1; otherwise, set to 0.

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

•Contains digit: For a token, if it contains digit(s) then the feature 'ContainsDigit' is set to 1. This feature is helpful for identifying the numbers.

•Made up of four digits: For a token if all the characters are digits and having 4 digits then the feature 'FourDigit' is set to 1. This is helpful in identifying the time (e.g., *2007sal*) and numerical (e.g., *2007*) expressions.

•Made up of two digits: For a token if all the characters are digits and having 2 digits then the feature 'TwoDigit' is set to 1. This is helpful for identifying the time expressions (e.g., *12 ta, 8 am, 9 pm*) in general.

•Contains digits and comma: For a token, if it contains digits and commas then the feature 'ContainsDigitsAndComma' is set to 1. This feature is helpful in identifying named entity measurement expressions (e.g., *120,45,330 taka*) and numerical numbers (e.g., *120,45,330*)

•Contains digits and slash: If the token contains digits and slash then the feature 'ContainsDigitAndslash' is set to 1. This helps in identifying time expressions (e.g., 15/8/2007).

•Contains digits and hyphen: If the token contains digits and hyphen then the feature 'ContainsDigitsAndHyphen' is set to 1. This is helpful for the identification of time expressions (e.g., 15-8-2007).

•Contains digits and period: If the token contains digits and periods then the feature 'ContainsDigitsAndPeriod' is set to 1. This helps to recognize numerical quantities (e.g., *120453.35*) and measurements (e.g., *120453.35 taka*).

•Contains digits and percentage: If the token contains digits and percentage symbol then the feature 'ContainsDigitsAndPercentage' is set to 1. This helps to recognize measurements (e.g., *120%*).

•Named Entity Information: The NE tag of the previous word is also considered as the feature, i.e., the combination of the current and the previous output token has been considered. This is the only dynamic feature in the experiment.

•Gazetteer Lists: Various gazetteer lists have been created from a tagged Bengali news corpus (Ekbal and Bandyopadhyay, 2007d) for Bengali. The first, last and middle names of person for Hindi have been created from the election commission data[6]. The person name collections had to be processed in order to use it in the CRF framework. The simplest approach of using these gazetteers is to compare the current word with the lists and make decisions. But this approach is not good, as it can't resolve ambiguity. So, it is better to use these lists as the features of the CRF. If the current token is in a particular list, then the corresponding feature is set to 1 for the current/previous/next token; otherwise, set to 0. The list of gazetteers is shown in Table 2.

3.3 Nested Named Entity Identification

One of the important tasks of the IJCNLP-NER shared task was to identify the internal named entities within the maximal NEs. In the training data, only the type of the maximal NEs were given. In order to identify the internal NEs during testing, we have defined some rules. After testing the unannotated test data with the CRF based NER system, it is searched to find the sequence of NE tags. The last NE tag in the sequence is assigned as the NE tag of the maximal NE. The NE tags of the constituent NEs may either be changed or may not be changed. The NE tags are changed with the help of rules and various gazetteer lists. We identified NEM (Named entity measurement), NETI (Named entity time expressions), NEO (Named entity organization names), NEP (Named entity person names) and NEL (Named entity locations) to be the potential NE tags, where nesting could occur. A NEM expression may contain NEN, an NETI may contain NEN, an NEO may contain NEP/ NEL, an NEL may contain NEP/NETP/NED and an NEP may contain NEL expressions. The nested

[6] http://www.eci.gov.in/DevForum/Fullname.asp

NEN tags could be identified by simply checking whether it contains digits only and checking the lists of cardinal and ordinal numbers.

Gazetteer	Number of entries	Feature Descriptions
Designation words in Bengali	947	'Designation' set to 1, otherwise 0
Organization names in Bengali	2, 225	'Organization' set to 1, otherwise 0.
Organization suffixes in Bengali	94	'OrgSuffix' set to 1, otherwise 0
Person prefix for Bengali	245	'PersonPrefix' set to 1, otherwise set to 0
First person names in Bengali	27,842	'FirstName' set to 1, otherwise 0
Middle names in Bengali	1,491	'MiddleName' set to 1, otherwise 0
Surnames in Bengali	5,288	'SurName' set to 1, otherwise 0
Common location word in Bengali	75	'CommonLocation' set 1, otherwise 0
Action verb in Bengali	215	'ActionVerb' set to 1, otherwise 0
First person names in Hindi	1,62,881	'FirstName' set to 1, otherwise 0
Middle person names in Hindi	450	'MiddleName' set to 1, otherwise 0
Last person names in Hindi	3,573	'SurName' set to 1, otherwise 0
Location names in Bengali	7,870	'LocationName' set to 1, otherwise 0
Week days in Bengali, Hindi and Telugu	14	'WeekDay' set to 1, otherwise 0
Month names in Bengali, Hindi and Telugu	24	'MonthName' set to 1, otherwise 0
Measurements in Bengali, Hindi and Telugu	52	'Measurement' set to 1, otherwise 0.

Table 2. Named entity gazetteer list

The procedure for identifying the nested NEs are shown below:
Step1: Test the unannotated test set.
Step 2: Look for the sequence of NE tags.
Step 3: All the words in the sequence will belong to a maximal NE.
Step 4: Assign the last NE tag in the sequence to the maximal NE.
Step 5: The test set is searched to look whether each component word appears with a NE tag.
Step 6: Assign the particular NE tag to the component if it appears in the test set with that NE tag. Otherwise, search the gazetteer lists as shown in Tables 1-2 to assign the tag.

4 Evaluation

The evaluation measures used for all the five languages are precision, recall and F-measure. These measures are calculated in three different ways:

(i). Maximal matches: The largest possibles named entities are matched with the reference data.

(ii). Nested matches: The largest possible as well as nested named entities are matched.

(iii). Maximal lexical item matches: The lexical items inside the largest possible named entities are matched.

(iv). Nested lexical item matches: The lexical items inside the largest possible as well as nested named entities are matched.

5 Experimental Results

The CRF based NER system has been trained and tested with five different Indian languages namely, Bengali, Hindi, Telugu, Oriya and Urdu data. The training and test sets statistics are presented in Table 3. Results of evaluation as explained in the previous section are shown in Table 4. The F-measures for the nested lexical match are also shown individually for each named entity tag separately in Table 5.

Experimental results of Table 4 show that the CRF based NER system performs best for Bengali with maximal F-measure of 55.36%, nested F-measure of 61.46% and lexical F-measure 59.39%. The system has demonstrated the F-measures of 35.37%, 36.75% and 33.12%, respectively for maximal, nested and lexical matches. The system has shown promising precision values for Hindi. But due to the low recall values, the F-measures get reduced. The large difference between the recall and precision values in the evaluation results of Hindi indicates that the system is not able to retrieve a significant number of NEs from the test

data. In comparison to Hindi, the precision values are low and the recall values are high for Bengali. It can be decided from the evaluation results that system retrieves more NEs in Bengali than Hindi but involves more errors. The lack of features in Oriya, Telugu and Urdu might be the reason behind their poor performance.

Language	Number of tokens in the training set	Number of tokens in the test set
Bengali	122,467	30,505
Hindi	502,974	38,708
Telugu	64,026	6,356
Oriya	93,173	24,640
Urdu	35,447	3,782

Table 3: Training and Test Sets Statistics

Tag	Bengali	Hindi	Oriya	Telugu	Urdu
NEP	85.68	21.43	43.76	1.9	7.69
NED	35.9	38.70	NF	NF	NF
NEO	52.53	NF	5.60	NF	22.02
NEA	26.92	30.77	NF	NF	NF
NEB	NF	NF	NF	NF	NF
NETP	61.44	NF	12.55	NF	NF
NETO	45.98	NF	NF	NF	NF
NEL	80.00	22.70	31.49	0.73	50.14
NETI	53.43	49.60	27.08	7.64	49.28
NEN	30.12	85.40	9.19	9.16	NF
NEM	79.08	36.64	7.56	NF	79.27
NETE	18.06	1.64	NF	5.74	NF

Table 4. Evaluation for Specific NE Tags (F-Measures for nested lexical match) [NF: Nothing found]

Experimental results of Table 5 show the F-measures for the nested lexical item matches for individual NE tags. For Bengali, the system has shown reasonably high F-measures for NEP, NEL and NEM tags and medium F-measures for NETP, NETI, NEO and NETO tags. The overall F-measures in Bengali might have reduced due to relatively poor F-measures for NETE, NEN, NEA and NED tags. For Hindi, the highest F-measure obtained is 85.4% for NEN tag followed by NETI, NED, NEM, NEA, NEL and NEP tags. In some cases, the system has shown better F-measures for Hindi than Bengali also. The system has performed better for NEN, NED and NEA tags in Hindi than all other languages.

6 Conclusion

We have developed a named entity recognition system using Conditional Random Fields for the five different Indian languages, namely Bengali, Hindi, Telugu, Oriya and Urdu. We have considered the contextual window of size five, prefix and suffix of length upto three of the current word, NE information of the previous word, different digit features and the frequently occurring word lists. The system also uses linguistic features extracted from the various gazetteer lists for Bengali and Hindi. Evaluation results show that the system performs best for Bengali. The performance of the system for Bengali can further be improved by including the part of speech (POS) information of the current and/or the surrounding word(s). The performance of the system for other languages can be improved with the use of different linguistic features as like Bengali.

The system did not perform as expected due to the problems faced during evaluation regarding the tokenization. We have tested the system for Bengali with 10-fold cross validation and obtained impressive results.

References

Babych, Bogdan, A. Hartley. Improving machine translation quality with automatic named entity recognition. In *Proceedings of EAMT/EACL 2003 Workshop on MT and other language technology tools*, 1-8, Hungary.

Bennet, Scott W.; C. Aone; C. Lovell. 1997. Learning to Tag Multilingual Texts Through Observation. In *Proceedings of EMNLP*, 109-116, Rhode Island.

Bikel, Daniel M., R. Schwartz, Ralph M. Weischedel. 1999. An Algorithm that Learns What's in Name. *Machine Learning (Special Issue on NLP)*, 1-20.

Bothwick, Andrew. 1999. A Maximum Entropy Approach to Named Entity Recognition. *Ph.D. Thesis*, New York University.

Chinchor, Nancy. 1995. MUC-6 Named Entity Task Definition (Version 2.1). *MUC-6*, Columbia, Maryland.

Measure→	Precision			Recall			F-measure		
Language ↓	P_m	P_n	P_l	R_m	R_n	R_l	F_m	F_n	F_l
Bengali	51.63	47.74	52.90	59.60	61.46	67.71	55.36	61.46	59.39
Hindi	71.05	76.08	80.59	23.54	24.23	20.84	35.37	36.75	33.12
Oriya	27.12	27.18	50.40	12.88	10.53	20.07	17.47	15.18	28.71
Telugu	1.70	2.70	8.10	0.538	0.539	3.34	0.827	0.902	4.749
Urdu	49.16	48.93	54.45	21.95	20.15	26.36	30.35	28.55	35.52
M: Maximal, *n*: Nested, *l*: Lexical									

Table 5. Evaluation of the Five Languages

Chinchor, Nancy. 1998. MUC-7 Named Entity Task Definition (Version 3.5). *MUC-7*. Fairfax, Virginia.

Cunningham, H. 2001. GATE: A general architecture for text engineering. *Comput. Humanit*. (36), 223-254.

Ekbal, Asif, and S. Bandyopadhyay. 2007a. Pattern Based Bootstrapping Method for Named Entity Recognition. In *Proceedings of 6th International Conference on Advances in Pattern Recognition*, Kolkata, India, 349-355.

Ekbal, Asif, and S. Bandyopadhyay. 2007b. Lexical Pattern Learning from Corpus Data for Named Entity Recognition. In *Proceedings of the 5th International Conference on Natural Language Processing*, Hyderabad, India, 123-128.

Ekbal, Asif, Naskar, Sudip and S. Bandyopadhyay. 2007c. Named Entity Recognition and Transliteration in Bengali. *Named Entities: Recognition, Classification and Use, Special Issue of Lingvisticae Investigationes Journal*, 30:1 (2007), 95-114.

Ekbal, Asif, and S. Bandyopadhyay. 2007d. A Web-based Bengali News Corpus for Named Entity Recognition. *Language Resources and Evaluation Journal (Accepted)*

Lafferty, J., McCallum, A., and Pereira, F. 2001. Conditional random fields: Probabilistic models for segmenting and labeling sequence data. In *Proc. of 18th International Conference on Machine learning*.

Li, Wei and Andrew McCallum. 2003. Rapid Development of Hindi Named Entity Recognition Using Conditional Random Fields and Feature Inductions, *ACM TALIP*, 2(3), (2003), 290-294.

McCallum, A.; W. Li. 2003. Early Results for Named Entity Recognition with Conditional Random Fields, Feature Induction and Web-Enhanced Lexicons. In *Proceedings CoNLL-03*, Edmanton, Canada.

Moldovan, Dan I., Sanda M. Harabagiu, Roxana Girju, P. Morarescu, V. F. Lacatusu, A. Novischi, A. Badulescu, O. Bolohan. 2002. LCC Tools for Question Answering. In *Proceedings of the TREC*, Maryland, 1-10.

Pinto, D., McCallum, A., Wei, X., and Croft, W. B. 2003. Table extraction using conditional random fields. In *Proceedings of SIGIR 03 Conference*, Toronto, Canada.

Sekine, Satoshi. 1998. Description of the Japanese NE System Used for MET-2, *MUC-7*, Fairfax, Virginia.

Sha, F. and Pereira, F. 2003. Shallow parsing with conditional random fields. In *Proceedings of Human Language Technology*, NAACL.

Named Entity Recognition for Telugu

P Srikanth and Kavi Narayana Murthy
Department of Computer and Information Sciences,
University of Hyderabad,
Hyderabad, 500 046,
email: patilsrik@yahoo.co.in, knmuh@yahoo.com

Abstract

This paper is about Named Entity Recognition (NER) for Telugu. Not much work has been done in NER for Indian languages in general and Telugu in particular. Adequate annotated corpora are not yet available in Telugu. We recognize that named entities are usually nouns. In this paper we therefore start with our experiments in building a CRF (Conditional Random Fields) based Noun Tagger. Trained on a manually tagged data of 13,425 words and tested on a test data set of 6,223 words, this Noun Tagger has given an F-Measure of about 92%. We then develop a rule based NER system for Telugu. Our focus is mainly on identifying person, place and organization names. A manually checked Named Entity tagged corpus of 72,157 words has been developed using this rule based tagger through bootstrapping. We have then developed a CRF based NER system for Telugu and tested it on several data sets from the Eenaadu and Andhra Prabha newspaper corpora developed by us here. Good performance has been obtained using the majority tag concept. We have obtained overall F-measures between 80% and 97% in various experiments.

Keywords: Noun Tagger, NER for Telugu, CRF, Majority Tag.

1 Introduction

NER involves the identification of named entities such as person names, location names, names of organizations, monetary expressions, dates, numerical expressions etc. In the taxonomy of Computational Linguistics, NER falls within the category of Information Extraction which deals with the extraction of specific information from given documents. NER emerged as one of the sub-tasks of the DARPA-sponsored Message Understanding Conference (MUCs). The task has important significance in the Internet search engines and is an important task in many of the Language Engineering applications such as Machine Translation, Question-Answering systems, Indexing for Information Retrieval and Automatic Summarization.

2 Approaches to NER

There has been a considerable amount of work on NER in English (Isozaki and Kazawa, 2002; Zhang and Johnson, 2003; Petasis et al., 2001; Mikheev et al., 1999). Much of the previous work on name finding is based on one of the following approaches: (1) hand-crafted or automatically acquired rules or finite state patterns (2) look up from large name lists or other specialized resources (3) data driven approaches exploiting the statistical properties of the language (statistical models).

The earliest work in named-entity recognition involved hand-crafted rules based on pattern matching (Appelt et al., 1993). For instance, a sequence of capitalized words ending in "Inc." is typically the name of an organization in the US, so one could implement a rule to that effect. Another example of such a rule is: Title Capitalized_word → Title Person_name. Developing and maintaining rules and dictionaries is a costly affair and adaptation to different domains is difficult.

Proceedings of the IJCNLP-08 Workshop on NER for South and South East Asian Languages, pages 41–50,
Hyderabad, India, January 2008.

In the second approach, the NER system recognizes only the named entities stored in its lists, also called gazetteers. This approach is simple, fast, language independent and easy to re-target - just recreate the lists. However, named entities are too numerous and are constantly evolving. Even when named entities are listed in the dictionaries, it is not always easy to decide their senses. There can be semantic ambiguities. For example, "Washington" refers to both person name as well as place name.

Statistical models have proved to be quite effective. Such models typically treat named-entity recognition as a sequence tagging problem, where each word is tagged with its entity type if it is part of an entity. Machine learning techniques are relatively independent of language and domain and no expert knowledge is needed. There has been a lot of work on NER for English employing the machine learning techniques, using both supervised learning and unsupervised learning. Unsupervised learning approaches do not require labelled training data - training requires only very few seed lists and large unannotated corpora (Collins and Singer, 1999). Supervised approaches can achieve good performance when large amounts of high quality training data is available. Statistical methods such as HMM (Bikel et al., 1997; Zhou and Su, 2001), Decision tree model (Baluja et al., 2000; Isozaki, 2001), and conditional random fields (McCallum, 2003) have been used. Generative models such as Hidden Markov Models (Bikel et al., 1997; Zhou and Su, 2001) have shown excellent performance on the Message Understanding Conference (MUC) data-set (Chinchor, 1997). However, developing large scale, high quality training data is itself a costly affair.

3 NER for Indian languages

NLP research around the world has taken giant leaps in the last decade with the advent of effective machine learning algorithms and the creation of large annotated corpora for various languages. However, annotated corpora and other lexical resources have started appearing only very recently in India. Not much work has been done in NER in Indian languages in general and Telugu in particular. Here we include a brief survey.

In (Eqbal, 2006), a supervised learning system based on pattern directed shallow parsing has been used to identify the named entities in a Bengali corpus. Here the training corpus is initially tagged against different seed data sets and a lexical contextual pattern is generated for each tag. The entire training corpus is shallow parsed to identify the occurrence of these initial seed patterns. In a position where the seed pattern matches wholly or in part, the system predicts the boundary of a named entity and further patterns are generated through bootstrapping. Patterns that occur in the entire training corpus above a certain threshold frequency are considered as the final set of patterns learned from the training corpus.

In (Li and McCallum, 2003), the authors have used conditional random fields with feature induction to the Hindi NER task. The authors have identified those feature conjunctions that will significantly improve the performance. Features considered here include word features, character n-grams (n = 2,3,4), word prefix and suffix (length - 2,3,4) and 24 gazetteers.

4 NER for Telugu

Telugu, a language of the Dravidian family, is spoken mainly in southern part of India and ranks second among Indian languages in terms of number of speakers. Telugu is a highly inflectional and agglutinating language providing one of the richest and most challenging sets of linguistic and statistical features resulting in long and complex word forms (Kumar et al., June 2007). Each word in Telugu is inflected for a very large number of word forms. Telugu is primarily a suffixing Language - an inflected word starts with a root and may have several suffixes added to the right. Suffixation is not a simple concatenation and morphology of the language is very complex. Telugu is also a free word order Language.

Telugu, like other Indian languages, is a resource poor language - annotated corpora, name dictionaries, good morphological analyzers, POS taggers etc. are not yet available in the required measure. Although Indian languages have a very old and rich literary history, technological developments are of recent origin. Web sources for name lists are available in English, but such lists are not available in Telugu forcing the use of transliteration.

In English and many other languages, named entities are signalled by capitalization. Indian scripts do not show upper-case - lower-case distinction. The concept of capitalization does not exist. Many names are also common nouns. Indian names are also more diverse i.e there are lot of variations for a given named entity. For example “telugude:s’aM” is written as Ti.Di.pi, TiDipi, te.de.pa:, de:s’aM etc. Developing NER systems is thus both challenging and rewarding. In the next section we describe our work on NER for Telugu.

5 Experiments and Results

5.1 Corpus

In this work we have used part of the LERC-UoH Telugu corpus, developed at the Language Engineering Research Centre at the Department of Computer and Information Sciences, University of Hyderabad. LERC-UoH corpus includes a wide variety of books and articles, and adds up to nearly 40 Million words. Here we have used only a part of this corpus including news articles from two of the popular newspapers in the region. The Andhra Prabha (AP) corpus consists of 1.3 Million words, out of which there are approximately 200,000 unique word forms. The Eenaadu (EE) corpus consists of 26 Million words in all.

5.2 Evaluation Metrics

We use two standard measures, *Precision*, *Recall*. Here precision (P) measures the number of correct NEs in the answer file (Machine tagged data) over the total number of NEs in the answer file and recall (R) measures the number of correct NEs in the answer file over the total number of NEs in the key file (gold standard). F-measure (F) is the harmonic mean of precision and recall: $F = \frac{(\beta^2+1)PR}{\beta^2 R+P}$ when β^2 = 1. The current NER system does not handle multi-word expressions - only individual words are recognized. Partial matches are also considered as correct in our analyses here. Nested entities are not yet handled.

5.3 Noun Identification

Named entities are generally nouns and it is therefore useful to build a noun identifier. Nouns can be recognized by eliminating verbs, adjectives and closed class words. We have built a CRF based binary classifier for noun identification. Training data of 13,425 words has been developed manually by annotating each word as noun or not-noun. Next we have extracted the following features for each word of annotated corpus:

- **Morphological features**: Morphological analyzer developed at University of Hyderabad over the last many years has been used to obtain the root word and the POS category for the given word. A morphological analyzer is useful in two ways. Firstly, it helps us to recognize inflected forms (which will not be listed in the dictionary) as not named entities. Secondly, word forms not recognized by morphology are likely to be named entities.

- **Length**: This is a binary feature whose value is 1 if length of the given word is less than or equal 3 characters, otherwise 0. This is based on the observation that very short words are rarely nouns.

- **Stop words**: A stop word list including function words has been collected from existing bi-lingual dictionaries. Bi-lingual dictionaries used for our experiments include C P Brown’s English-Telugu dictionary (Brown, 1997), Telugu-Hindi dictionary developed at University of Hyderabad and the Telugu-English dictionary developed by V Rao Vemuri. We have also extracted high frequency words from our corpora. Initially words which have occurred 1000 times or more were selected, hand filtered and added to the stop word list. Then, words which have occurred 500 to 1000 times were looked at, hand filtered and added to the stop word list. The list now has 1731 words. If the given word belongs to this list, the feature value is 1 otherwise 0.

- **Affixes**: Here, we use the terms prefix/suffix to mean any sequence of first/last few characters of a word, not necessarily a linguistically meaningful morpheme. The use of prefix and suffix information is very useful for highly inflected languages. Here we calculate suffixes of length from 4 characters down to 1 character and prefixes of length from 7 characters

down to 1 character. Thus the total number of prefix/suffix features are 11. For example, for the word "virigiMdi" (broke), the suffixes are "iMdi, Mdi, di, i" and the prefixes are "virigiM, virigi, virig, viri, vir, vi, v". The feature values are not defined (ND) in the following cases:

  - If length of a word is less than or equal to 3 characters, all the affix values are ND.
  - If length of a word is from 4 to 6 characters, initial prefixes will be ND.
  - If the word contains special symbols or digits, both the suffix and prefix values are ND.

- **Position**: This is a binary feature, whose value is 1 if the given word occurs at the end of the sentence, otherwise 0. Telugu is a verb final language and this feature is therefore significant.

- **POS**: A single dictionary file is compiled from the existing bi-lingual dictionaries. This file includes the head word and its Part of Speech. If a given word is available in this file, then its POS tag is taken as feature otherwise feature value is 0.

- **Orthographic information** This is a binary feature whose value is 1 if a given word contains digits or special symbols otherwise the feature value is 0.

- **Suffixes** A list of linguistic suffixes of verbs, adjectives and adverbs were compiled from (Murthy and J.P.L.Gywnn, 1985) to recognize not-nouns in a given sentence. This feature value is 1 if the suffix of the given word belongs to this list, otherwise it is 0.

A feature vector consisting of the above features is extracted for each word in the annotated corpus. Now we have training data in the form of $(\mathbf{W}_i, T_i)$, where $\mathbf{W}_i$ is the i^{th} word and its feature vector, and T_i is its tag - NOUN or NOT-NOUN. The feature template used for training CRF is shown in Table-1, where w_i is the current word, w_{i-1} is previous word, w_{i-2} is previous to previous word, w_{i+1} is next word and w_{i+2} is next to next word.

w_{i-2}
w_{i-1}
w_i
w_{i+1}
w_{i+2}
combination of w_{i-1}, w_i
combination of w_i, w_{i+1}
feature vector of w_i
morph tags of w_{i-2}, w_{i-1}, w_i, w_{i+1} and w_{i+2}
output tag of current and previous word (t_i,t_{i-1})

Table 1: Feature Template used for Training CRF based Noun Tagger

The inputs for training CRF consists of the training data and the feature template. The model learned during training is used for testing. Apart from the basic features described above, we have also experimented by including varying amounts of contextual information in the form of neighbouring words and their morph features. Let us define:

- F1: [(w_i), feature vector of w_i, t_i, t_{i-1}].
- F2 : [w_{i-1}, w_{i+1}, (w_{i-1}, w_i), (w_i, w_{i+1}) and the morph tags of w_{i-1} and w_{i+1}].
- F3 : [w_{i-2}, w_{i+2}, morph tags of w_{i-2} and w_{i+2}]

The CRF trained with the basic template F1, which consists of the current word, the feature vector of the current word and the output tag of the previous word as the features, was tested on a test data of 6,223 words and an F-measure of 91.95% was obtained. Next, we trained the CRF by taking the combination of F1 and F2. We also trained using combination of F1, F2 and F3. The performances of all 3 combinations are shown in Table-2. It may be seen that performance of the system is reducing as we increase the number of neighbouring words as features. Adding contextual features does not help.

5.4 Heuristic based NER system

Nouns which have already been identified in the noun identification phase are now checked for named entities. In this work, our main focus is on identifying person, place and organization names. Indian place names and person names often

Feature combinations	Precision	Recall	F-measure
F1	91.64	92.28	91.95
F1+F2	91.46	92.28	91.86
F1+F2+F3	91.17	91.99	91.57

Table 2: Performance of the CRF based Noun tagger with different feature combinations

have some suffix or prefix clues. For example ”na:yuDu” is a person suffix clue for identifying “ra:ma:na:yuDu” as a person entity and ”ba:d” is a location suffix clue for identifying “haidara:ba:d”, “adila:ba:d” etc as place entities. We have manually prepared a list of such suffixes for both persons and locations as also a list of prefixes for person names. List of organization names is also prepared manually. We have also prepared a gazetteer consisting of location names and a gazetteer of person name contexts since context lists are also very useful in identifying person names. For example, it has been observed that whenever a context word such as “maMtri” appears, a person name is likely to follow. Regular expressions are used to identify person entities like “en.rame:S” and organization entities which are in acronym form such as “Ti.Di.pi”, “bi.je.pi” etc. Initially one file of the corpus is tagged using these seed lists and patterns. Then we manually check and tag the unidentified named entities. These new named entities are also added to the corresponding gazetteers and the relevant contexts are added to their corresponding lists. Some new rules are also observed during manual tagging of unidentified names. Here is an example of a rule:

”**if** word[i] is NOUN and word[i-1] belongs to the person context list **then** word[i] is person name”.

Currently the gazetteers include 1346 location names, 221 organization names, and small lists of prefixes, suffixes and other contextual cues that signal the presence of named entities, their types, or their beginning or ending. Using these lists and rules, we then tag another file from the remaining corpus. This process of semi-automatic tagging is continued for several iterations. This way we have developed a named entity annotated database of 72,157 words, including 6,268 named entities (1,852 place names, 3,201 person names and 1,215 organization names).

5.4.1 Issues in Heuristic NER

There are ambiguities. For example, ”ko:Tla” is a person first name in ”ko:Tla vijaybha:skar” and it is also a common word that exists in a phrase such as ”padi ko:Tla rupa:yalu” (10 crore rupees). There also exists ambiguity between a person entity and place entity. For example, ”siMha:calaM” and ”raMga:reDDi” are both person names as well as place names. There are also some problems while matching prefixes and suffixes of named entities. For example ”na:Du” is a useful suffix for matching place names and the same suffix occurs with time entities such as ”so:mava:raMna:Du”. Prefixes like ”ra:j” can be used for identifying person entities such as ”ra:jkiran”, ”ra:jgo:pa:l”,”ra:js’e:khar” etc. but the same prefix also occurs with common words like ”ra:jaki:ya:lu”. Thus these heuristics are not fool proof. We give below the results of our experiments using our heuristic based NER system for Telugu.

5.4.2 Experiment 1

Here, we have presented the performance of the heuristic-based NER system over two test data sets (AP-1 and AP-2). These test data sets are from the AP corpus. Total number of words (NoW) and number of named entities in the test data sets AP-1 and AP-2 are given in Table-3. Performance of the system is measured in terms of F-measure. The recognized named entity must be of the correct type (person, place or organization) for it to be counted as correct. A confusion matrix is also given. The notation used is as follows: PER - person; LOC - location; ORG - organization; NN - not-name. The results are depicted in Tables 4, 5 and 6.

	AP-1			AP-2		
	PER	LOC	ORG	PER	LOC	ORG
P (%)	83.44	97.5	97.40	60.57	87.93	87.5
R (%)	84.84	96.29	87.20	72.83	86.56	77.77
F (%)	84.13	96.89	92.01	66.13	87.23	82.34

Table 4: Performance of Heuristic based NER System

AP Corpus	PER	LOC	ORG	NoW
AP-1	296	81	86	3,537
AP-2	173	321	63	7,032

Table 3: Number of Entities in Test Data Sets

Actual/Obtained	PER	LOC	ORG	NN
PER	285	0	0	12
LOC	0	81	0	0
ORG	6	0	75	5
NN	63	3	3	3004

Table 5: Confusion Matrix for the Heuristic based System on AP-1

Actual/Obtained	PER	LOC	ORG	NN
PER	126	0	0	47
LOC	2	277	0	41
ORG	0	0	49	14
NN	80	38	7	6351

Table 6: Confusion matrix of heuristic based system on AP-2

5.5 CRF based NER system

Now that we have developed a substantial amount of training data, we have also attempted supervised machine learning techniques for NER. In particular, we have used CRFs. For the CRF based NER system, the following features are extracted for each word of the labelled training data built using the heuristic based NER system.

- **Class Suffixes/Prefixes** This includes the following three features:
  - Location suffix: If the given word contains a location suffix, feature value is 1 otherwise 0.
  - Person suffix: If the given word contains a person suffix, feature value is 1 otherwise it is 0.
  - Person prefix: If the given word contains a person prefix, feature value is 1 otherwise it is 0.
- **Gazetteers** Five different gazetteers have been used. If the word belongs to the person first name list, feature value is 1 else if the word belongs to person middle name list, feature value is 2 else if the word belongs to person last name list, feature value is 3 else if the word belongs to location list, feature value is 4 else if the word belongs to organization list, feature value is 5 else feature value is 0.
- **Context** If the word belongs to person context list, feature value is 1 else if the word belongs to location context list, feature value is 2 else if the word belongs to organization context list, feature value is 3 else the feature value is 0.
- **Regular Expression** This includes two features as follows:
  - REP: This is regular expression used to identify person names. The feature value is 1 if the given word matches.

    ```
    /([a-zA-Z:~]{1,3})\.(
    [a-zA-Z:~]{1,3})?\.?(
    [a-zA-Z:~]{1,3})?\.?
    [a-zA-Z:~']{4,}/
    ```

  - REO: This is regular expression used to identify organization names mentioned in acronym format like “bi.je.pi”, “e.ai.Di.eM.ke”. etc. This feature value is 1, if the given word matches

    ```
    /(.{1,3})\.(.{1,3})\.
    (.{1,3})\.(.{1,3})?\.?
    (.{1,3})?\.?/)/
    ```

- **Noun tagger** Noun tagger output is also used as a feature value.

- **Orthographic Information**, **Affixes**, **Morphological feature**, **Position feature**, **Length** are directly extracted from "Noun Identification" process.

The training data used for training CRFs consists of words, the corresponding feature vectors and the corresponding name tags. We have used "CRF++: Yet another CRF toolkit" (Taku,) for our experiments. Models are built based on training data and the feature template. Results are given in the next subsection. These models are used to tag the test data. The feature template used in these experiments is as follows:

w_{i-3}
w_{i-2}
w_{i-1}
w_i
w_{i+1}
w_{i+2}
combination of w_{i-1}, w_i
combination of w_i, w_{i+1}
feature vector of w_i
morph tags of w_{i-2}, w_{i-1}, w_i, w_{i+1} and w_{i+2}
output tag of the previous word t_{i-1}
context information of the neighbour words

Table 7: Feature Template used for Training CRF

5.5.1 Experiment 2

In this experiment, we took 19,912 words of training data (TR-1) and trained the CRF engine with different feature combinations of the feature template. Details of the training data (TR-1 $\subset$ TR-2 $\subset$ TR-3) and test data sets used in these experiments are given in Tables 8 and 9. Here the experiments are performed by varying the number of neighbouring words in the feature template. In the first case, feature template consists of current word (w_i), feature vector of the current word, two neighbours of the current word (w_{i-1}, w_{i+1}), morph tags of the neighbour words, context information of the neighbour words, combination of current word and its neighbours and the output tag of the previous word. A model is built by training the CRF engine using this template. The model built is used in testing data sets (AP-1 and AP-2). Similarly, we repeated the same experiment by considering 4 and 6 neighbouring words of the current word in the feature template. The results are shown in Table-9 with varying number of neighbour words represented as window-size. It is observed that there is not much improvement in the performance of the system by including more of the neighbouring words as features.

Performance of the system without taking gazetteer features is shown in Table-11. We see that the performance of the system reduces when we have not considered morph features and Noun tagger output in the feature template as can be seen from Table-12.

Finally, we have tested the performance of the system on two new test data sets (EE-1 and EE-2) from the EE corpus with varying amounts of training data. Total number of words (NoW) and the number of named entities in the test data sets EE-1 and EE-2 are depicted in Table-8. Performance of the system in terms of F-measure is shown in table 13.

EE Corpus	PER	LOC	ORG	NoW
EE-1	321	177	235	6,411
EE-2	325	144	187	5221

Table 8: Number of Entities in Test Data Sets

AP corpus	PER	LOC	ORG	NoW
TR-1	804	433	175	19,912
TR-2	1372	832	388	34,116
TR-3	2555	1511	793	60,525

Table 9: Number of Entities in Training Data Sets

Gazetteers have a major role in performance while morph is adding a bit. F-Measures of 74% to 93%

	AP-1	AP-2	EE-1	EE-2
PER	93.76	79.36	70.91	69.84
LOC	96.81	89.78	81.84	70.91
ORG	80.27	91.66	71.73	80.75

Table 12: Performance of the CRF based NER System without Morph and Noun Tagger Features

	Win-Size	AP-1			AP-2		
		PER	LOC	ORG	PER	LOC	ORG
P	2	99.62	100	98.41	90.07	93.55	98.21
	4	99.62	100	96.96	89.36	93.53	98.21
	6	99.62	100	96.96	90.71	93.55	98.21
R	2	89.86	93.82	72.09	72.15	85.98	87.30
	4	89.86	93.82	74.41	71.59	85.66	87.30
	6	89.52	93.82	74.41	72.15	85.98	87.30
F	2	94.49	96.81	83.22	80.12	89.61	92.43
	4	**94.49**	**96.81**	**84.21**	79.49	89.43	92.43
	6	94.30	96.81	84.21	**80.37**	**89.61**	**92.43**

Table 10: Performance of CRF based NER system with different window sizes

	AP-1			AP-2		
	PER	LOC	ORG	PER	LOC	ORG
P	90.86	97.95	97.91	89.05	96.88	96.15
R	57.09	59.25	54.65	69.31	67.91	79.36
F	70.12	73.84	70.14	77.95	79.85	86.95

Table 11: Performance of the CRF based NER system without Gazetteers

Test Data	CLASS	TR-1	TR-2	TR-3
EE-1	PER	75.14	79.70	81.58
	LOC	81.84	80.66	81.45
	ORG	76.76	78.46	79.89
EE-2	PER	69.98	74.47	79.70
	LOC	70.91	70.96	71.2
	ORG	82.13	82.82	83.69

Table 13: Performance of CRF based NER system with varying amounts of Training Data on EE Test Data

have been obtained. Effect of training corpus size has been checked by using 19,912 words, 34,116 words and 60,525 words training corpora built from the AP newspaper corpus. Test data was from EE newspaper. It is clearly seen that larger the training data, better is the performance. See table 13.

5.5.2 Experiment 3: Majority Tag as an Additional Feature

There are some names like "kRSNa:", which can refer to either person name, place name or a river name depending up on the context in which they are used. Hence, if the majority tag is incorporated as a feature, a classifier can be trained to take into account the context in which the named entity is used, as well as frequency information. In this experiment, we have used an unlabelled data set as an additional resource from the EE news corpus. The unlabelled data set consists of 11,789 words.

Initially, a supervised classifier h_1 is trained on the labelled data (TR-3) of 60,525 words. Then this classifier labels the unlabelled data set (U) (11,789 words) and produces a machine tagged data set U'. Although our NER system is not so robust, useful information can still be gathered as we shall see below.

Next, a majority tag list (L) is produced by extracting the list of named entities with their associated majority tags from the machine tagged data set U'. The process of extracting majority tag list (L) is simple: We first identify possible name classes assigned for the named entities in U' and we assign the class that has occurred most frequently. Next, in order to recover unidentified named entities (inflections of named entities already identified), we compare the root words of those words whose class is assigned neither to person, place or organization with the named entities already identified. If there is any match with any of the named entities, the tag of the identified named entity is assigned to the unidenti-

EE Corpus	Without Majority Tag			With Majority Tag		
	PER	LOC	ORG	PER	LOC	ORG
P	96.99	98.4	99.36	97.02	98.38	98.78
R	70.40	69.49	66.80	71.02	68.92	68.93
F	81.58	81.45	79.89	82.01	81.06	81.20

Table 14: Performance of CRF based NER using Maj-tag on EE-1

EE Corpus	Without Majority Tag			With Majority Tag		
	PER	LOC	ORG	PER	LOC	ORG
P	98.18	83.96	98.55	98.22	84.11	97.88
R	67.07	61.80	72.72	68	62.5	74.31
F	79.70	71.2	83.69	80.36	71.71	84.49

Table 15: Performance of CRF based NER using Maj-tag on EE-2

fied named entity. L thus consists of (NE, Maj-tag) pairs, where Maj-tag is the name class that occurs most frequently for the named entity (NE) in the machine tagged data set U'.

Now, we add this Maj-tag as an additional feature to labelled data (TR-3): if a word in labelled data matches with a named entity in the majority tag list (L), then the corresponding Maj-tag (name class) is assigned as a feature value to that word in the labelled data. Finally, a classifier h_2 is trained on the labelled data (TR-3). We use this classifier (h_2) to tag the test data sets (EE-1 and EE-2). It can be observed from tables 14 and 15 that including the majority tag feature improves the performance a bit.

6 Conclusions

Not much work has been done in NER in Telugu and other Indian languages so far. In this paper, we have reported our work on Named Entity Recognition for Telugu. We have developed a CRF based noun tagger, whose output is used as one of the feature for the CRF based NER system. We have also described how we have developed a substantial training data using a heuristic based system through boot-strapping. The CRF based system performs better when compared with the initial heuristic based system. We have also shown that performance of the system can be improved by adding gazetteers as features. Morphological analyser has shown a small contribution to the performance of the system. It is also observed that there is some increase in performance of the system by using majority tag concept. We have obtained F-measures between 80% and 97% in various experiments. It may be observed that we have not used any POS tagger or parser or annotated corpora tagged with POS or syntactic information. Once adequate POS taggers and chunkers are developed, we may be able to do better. The current work is limited to recognizing single word NEs. We plan to consider multi-token named entities and nested structures in our future work.

References

D. Appelt, J. Hobbs, J. Bear, D. Israel, M. Kameyama, A.Kehler, D. Martin, K.Meyers, and M. Tyson. 1993. SRI international FASTUS system: MUC-6 test results and analysis.

Shumeet Baluja, Vibhu O. Mittal, and Rahul Sukthankar. 2000. Applying Machine Learning for High-Performance Named-Entity Extraction. *Computational Intelligence*, 16(4):586–596.

Daniel M. Bikel, Scott Miller, Richard Schwartz, and Ralph Weischedel. 1997. Nymble: a high-performance learning name-finder. In *Proceedings of the fifth conference on Applied natural language processing*, pages 194–201, San Francisco, CA, USA. Morgan Kaufmann Publishers Inc.

Charles Philip Brown. 1997. *Telugu-English dictionary*. New Delhi Asian Educational Services.

Nancy Chinchor. 1997. MUC-7 Named Entity Task Definition (version 3.0). In *Proceedings of the 7th Message Understanding Conference (MUC-7)*.

M. Collins and Y. Singer. 1999. Unsupervised models for named entity classification. Proceedings of the Joint SIGDAT Conference on Empirical Methods in Natural Language Processing and Very Large Corpora.

Asif Eqbal. 2006. Named Entity Recognition for Bengali. Satellite Workshop on Language, Artificial Intelligence and Computer Science for Natural Language Applications (LAICS-NLP), Department of Computer Engineering Faculty of Engineering Kasetsart University, Bangkok, Thailand.

Hideki Isozaki and Hideto Kazawa. 2002. Efficient support vector classifiers for named entity recognition. In *Proceedings of the 19th international conference on Computational linguistics*, pages 1–7, Morristown, NJ, USA. Association for Computational Linguistics.

Hideki Isozaki. 2001. Japanese named entity recognition based on a simple rule generator and decision tree learning. In *ACL '01: Proceedings of the 39th Annual Meeting on Association for Computational Linguistics*, pages 314–321, Morristown, NJ, USA. Association for Computational Linguistics.

G. Bharadwaja Kumar, Kavi Narayana Murthy, and B.B.Chaudhari. June 2007. Statistical Analysis of Telugu Text Corpora. *IJDL,Vol 36, No 2*, pages 71–99.

Wei Li and Andrew McCallum. 2003. Rapid development of Hindi named entity recognition using conditional random fields and feature induction. *ACM Transactions on Asian Language Information Processing (TALIP)*, 2(3):290–294.

McCallum. 2003. Early results for Named Entity Recognition with Conditional Random Fields, feature induction and web-enhanced lexicons. In *Proceedings of the seventh conference on Natural language learning at HLT-NAACL 2003*, pages 188–191, Morristown, NJ, USA. Association for Computational Linguistics.

Andrei Mikheev, Marc Moens, and Claire Grover. 1999. Named Entity Recognition without gazetteers. In *Proceedings of the ninth conference on European chapter of the Association for Computational Linguistics*, pages 1–8, Morristown, NJ, USA. Association for Computational Linguistics.

Bh.Krishna Murthy and J.P.L.Gywnn. 1985. *A Grammar of Modern Telugu*. Oxford University Press, Delhi.

Georgios Petasis, Frantz Vichot, Francis Wolinski, Georgios Paliouras, Vangelis Karkaletsis, and Constantine D. Spyropoulos. 2001. Using machine learning to maintain rule-based named-entity recognition and classification systems. In *ACL '01: Proceedings of the 39th Annual Meeting on Association for Computational Linguistics*, pages 426–433, Morristown, NJ, USA. Association for Computational Linguistics.

Taku. http://crfpp.sourceforge.net/.

Tong Zhang and David Johnson. 2003. A Robust Risk Minimization based Named Entity Recognition system. In *Proceedings of the seventh conference on Natural language learning at HLT-NAACL 2003*, pages 204–207, Morristown, NJ, USA. Association for Computational Linguistics.

GuoDong Zhou and Jian Su. 2001. Named Entity Recognition using an HMM-based chunk tagger. In *ACL '02: Proceedings of the 40th Annual Meeting on Association for Computational Linguistics*, pages 473–480, Morristown, NJ, USA. Association for Computational Linguistics.

Bengali Named Entity Recognition using Support Vector Machine

Asif Ekbal
Department of Computer Science and Engineering, Jadavpur University
Kolkata-700032, India
asif.ekbal@gmail.com

Sivaji Bandyopadhyay
Department of Computer Science and Engineering, Jadavpur University
Kolkata-700032, India
svaji_cse_ju@yahoo.com

Abstract

Named Entity Recognition (NER) aims to classify each word of a document into predefined target named entity classes and is nowadays considered to be fundamental for many Natural Language Processing (NLP) tasks such as information retrieval, machine translation, information extraction, question answering systems and others. This paper reports about the development of a NER system for Bengali using Support Vector Machine (SVM). Though this state of the art machine learning method has been widely applied to NER in several well-studied languages, this is our first attempt to use this method to Indian languages (ILs) and particularly for Bengali. The system makes use of the different contextual information of the words along with the variety of features that are helpful in predicting the various named entity (NE) classes. A portion of a partially NE tagged Bengali news corpus, developed from the archive of a leading Bengali newspaper available in the web, has been used to develop the SVM-based NER system. The training set consists of approximately 150K words and has been manually annotated with the sixteen NE tags. Experimental results of the 10-fold cross validation test show the effectiveness of the proposed SVM based NER system with the overall average Recall, Precision and F-Score of 94.3%, 89.4% and 91.8%, respectively. It has been shown that this system outperforms other existing Bengali NER systems.

1 Introduction

Named Entity Recognition (NER) is an important tool in almost all NLP application areas such as information retrieval, machine translation, question-answering system, automatic summarization etc. Proper identification and classification of NEs are very crucial and pose a very big challenge to the NLP researchers. The level of ambiguity in NER makes it difficult to attain human performance

NER has drawn more and more attention from the NE tasks (Chinchor 95; Chinchor 98) in Message Understanding Conferences (MUCs) [MUC6; MUC7]. The problem of correct identification of NEs is specifically addressed and benchmarked by the developers of Information Extraction System, such as the GATE system (Cunningham, 2001). NER also finds application in question-answering systems (Maldovan et al., 2002) and machine translation (Babych and Hartley, 2003).

The current trend in NER is to use the machine-learning approach, which is more attractive in that it is trainable and adoptable and the maintenance of a machine-learning system is much cheaper than that of a rule-based one. The representative machine-learning approaches used in NER are Hidden Markov Model (HMM) (BBN's IdentiFinder in (Bikel, 1999)), Maximum Entropy (New York University's MEME in (Borthwick, 1999)), Decision Tree (New York University's system in (Sekine, 1998) and Conditional Random Fields (CRFs) (Lafferty et al., 2001). Support Vector Machines (SVMs) based NER system was proposed by Yamada et al. (2002) for Japanese. His system is an extension of Kudo's chunking system (Kudo and Matsumoto, 2001) that gave the best performance at CoNLL-2000 shared tasks. The other SVM-based NER systems can be found in (Takeuchi and Collier, 2002) and (Asahara and Matsumoto, 2003).

Named entity identification in Indian languages in general and particularly in Bengali is difficult and challenging. In English, the NE always appears with capitalized letter but there is no concept of capitalization in Bengali. There has been a very

Proceedings of the IJCNLP-08 Workshop on NER for South and South East Asian Languages, pages 51–58,
Hyderabad, India, January 2008.

little work in the area of NER in Indian languages. In Indian languages, particularly in Bengali, the works in NER can be found in (Ekbal and Bandyopadhyay, 2007a; Ekbal and Bandyopadhyay, 2007b) with the pattern directed shallow parsing approach and in (Ekbal et al., 2007c) with the HMM. Other than Bengali, a CRF-based Hindi NER system can be found in (Li and McCallum, 2004).

The rest of the paper is organized as follows. Support Vector Machine framework is described briefly in Section 2. Section 3 deals with the named entity recognition in Bengali that describes the named entity tagset and the detailed descriptions of the features for NER. Experimental results are presented in Section 4. Finally, Section 5 concludes the paper.

2 Support Vector Machines

Support Vector Machines (SVMs) are relatively new machine learning approaches for solving two-class pattern recognition problems. SVMs are well known for their good generalization performance, and have been applied to many pattern recognition problems. In the field of NLP, SVMs are applied to text categorization, and are reported to have achieved high accuracy without falling into over-fitting even though with a large number of words taken as the features.

Suppose we have a set of training data for a two-class problem: $\{(x_1, y_1), ...(x_N, y_N)\}$, where $x_i \in R^D$ is a feature vector of the i-th sample in the training data and $y_i \in \{+1, -1\}$ is the class to which x_i belongs. The goal is to find a decision function that accurately predicts class y for an input vector x. A non-linear SVM classifier gives a decision function $f(x) = sign(g(x))$ for an input vector where,

$$g(x) = \sum_{i=1}^{m} w_i K(x, z_i) + b$$

Here, $f(x) = +1$ means x is a member of a certain class and $f(x) = -1$ means x is not a member. z_i s are called support vectors and are representatives of training examples, m is the number of support vectors. Therefore, the computational complexity of $g(x)$ is proportional to m. Support vectors and other constants are determined by solving a certain quadratic programming problem. $K(x, z_i)$ is a *kernel* that implicitly maps vectors into a higher dimensional space. Typical kernels use dot products: $K(x, z_i) = k(x.z)$. A polynomial kernel of degree d is given by $K(x, z_i) = (1 + x)^d$. We can use various kernels, and the design of an appropriate kernel for a particular application is an important research issue.

We have developed our system using SVM (Jochims, 1999) and (Valdimir, 1995), which performs classification by constructing an N-dimensional hyperplane that optimally separates data into two categories. Our general NER system includes two main phases: training and classification. Both the training and classification processes were carried out by *YamCha*[1] toolkit, an SVM based tool for detecting classes in documents and formulating the NER task as a sequential labeling problem. Here, the pair wise multi-class decision method and *second degree polynomial kernel function* were used. We have used TinySVM-0.07[2] classifier that seems to be the best optimized among publicly available SVM toolkits.

3 Named Entity Recognition in Bengali

Bengali is one of the widely used languages all over the world. It is the seventh popular language in the world, second in India and the national language of Bangladesh. A partially NE tagged Bengali news corpus (Ekbal and Bandyopadhyay, 2007d), developed from the archive of a widely read Bengali newspaper. The corpus contains around 34 million word forms in ISCII (Indian Script Code for Information Interchange) and UTF-8 format. The *location, reporter, agency* and different *date* tags (*date, ed, bd, day*) in the partially NE tagged corpus help to identify some of the location, person, organization and miscellaneous names, respectively that appear in some fixed places of the newspaper. These tags cannot detect the NEs within the actual news body. The date information obtained from the news corpus provides example of miscellaneous names. A portion of this partially NE tagged corpus has been manually annotated with the sixteen NE tags as described in Table 1.

3.1 Named Entity Tagset

A SVM based NER system has been developed in this work to identify NEs in Bengali and classify

[1]http://chasen-org/~taku/software/yamcha/
[2]http://cl.aist-nara.ac.jp/~taku-ku/software/TinySVM

them into the predefined four major categories, namely, 'Person name', 'Location name', 'Organization name' and 'Miscellaneous name'. In order to properly denote the boundaries of the NEs and to apply SVM in NER task, sixteen NE and one non-NE tags have been defined as shown in Table 1. In the output, sixteen NE tags are replaced appropriately with the four major NE tags by some simple heuristics.

NE tag	Meaning	Example
PER	Single word person name	*sachin* / PER
LOC	Single word location name	*jdavpur*/LOC
ORG	Single word organization name	*infosys* / ORG
MISC	Single word miscellaneous name	100%/ MISC
B-PER I-PER E-PER	Beginning, Internal or the End of a multiword person name	*sachin*/B-PER *ramesh*/I-PER *tendulkar*/E-PER
B-LOC I-LOC E-LOC	Beginning, Internal or the End of a multiword location name	*mahatma*/B-LOC *gandhi*/I-LOC *road*/E-LOC
B-ORG I-ORG E-ORG	Beginning, Internal or the End of a multiword organization name	*bhaba*/B-ORG *atomic*/I-ORG *research*/I-ORG *center*/E-ORG
B-MISC I-MISC E-MISC	Beginning, Internal or the End of a multiword miscellaneous name	*10e*/B-MISC *magh*/I-MISC *1402*/E-MISC
NNE	Words that are not named entities	*neta*/NNE, *bidhansabha*/NNE

Table 1. Named Entity Tagset

3.2 Named Entity Feature Descriptions

Feature selection plays a crucial role in the Support Vector Machine (SVM) framework. Experiments have been carried out in order to find out the most suitable features for NER in Bengali. The main features for the NER task have been identified based on the different possible combination of available word and tag context. The features also include prefix and suffix for all words. The term prefix/suffix is a sequence of first/last few characters of a word, which may not be a linguistically meaningful prefix/suffix. The use of prefix/suffix information works well for highly inflected languages like the Indian languages. In addition, various gazetteer lists have been developed for use in the NER task. We have considered different combination from the following set for inspecting the best feature set for NER task:

F={ $w_{i-m},...,w_{i-1},w_i,w_{i+1},...,w_{i+n}$, |prefix|≤n, |suffix|≤n, previous NE tags, POS tags, First word, Digit information, Gazetteer lists}

Following are the details of the set of features that have been applied to the NER task:

- Context word feature: Previous and next words of a particular word might be used as a feature.
- Word suffix: Word suffix information is helpful to identify NEs. This feature can be used in two different ways. The first and the naïve one is, a fixed length word suffix of the current and/or the surrounding word(s) might be treated as feature. The second and the more helpful approach is to modify the feature as binary valued. Variable length suffixes of a word can be matched with predefined lists of useful suffixes for different classes of NEs. The different suffixes that may be particularly helpful in detecting person (e.g., *-babu*, *-da*, *-di* etc.) and location names (e.g., *-land*, *-pur*, *-lia* etc.) are also included in the lists of variable length suffixes. Here, both types of suffixes have been used.
- Word prefix: Prefix information of a word is also helpful. A fixed length prefix of the current and/or the surrounding word(s) might be treated as features.
- Part of Speech (POS) Information: The POS of the current and/or the surrounding word(s) can be used as features. Multiple POS information of the words can be a feature but it has not been used in the present work. The alternative and the better way is to use a coarse-grained POS tagger.

Here, we have used a CRF-based POS tagger, which was originally developed with the help of 26 different POS tags[3], defined for Indian languages. For NER, we have considered a coarse-grained POS tagger that has only the following POS tags:

NNC (Compound common noun), NN (Common noun), NNPC (Compound proper noun), NNP (Proper noun), PREP (Postpositions), QFNUM (Number quantifier) and Other (Other than the above).

[3]http://shiva.iiit.ac.in/SPSAL2007/iiit_tagset_guidelines.pdf

The POS tagger is further modified with two POS tags (Nominal and Other) for incorporating the nominal POS information. Now, a binary valued feature 'nominalPOS' is defined as: If the current/surrounding word is 'Nominal' then the 'nominalPOS' feature of the corresponding word is set to '+1'; otherwise, it is set to '-1'. This binary valued 'nominalPOS' feature has been used in addition to the 7-tag POS feature. Sometimes, postpositions play an important role in NER as postpositions occur very frequently after a NE. A binary valued feature 'nominalPREP' is defined as: If the current word is nominal and the next word is PREP then the feature 'nomianlPREP' of the current word is set to '+1', otherwise, it is set to '-1'.

•Named Entity Information: The NE tag(s) of the previous word(s) can also be considered as the feature. This is the only dynamic feature in the experiment.

•First word: If the current token is the first word of a sentence, then the feature 'FirstWord' is set to '+1'; Otherwise, it is set to '-1'.

•Digit features: Several digit features have been considered depending upon the presence and/or the number of digit(s) in a token (e.g., ContainsDigit [token contains digits], FourDigit [token consists of four digits], TwoDigit [token consists of two digits]), combination of digits and punctuation symbols (e.g., ContainsDigitAndComma [token consists of digits and comma], ConatainsDigitAndPeriod [token consists of digits and periods]), combination of digits and symbols (e.g., ContainsDigitAndSlash [token consists of digit and slash], ContainsDigitAndHyphen [token consists of digits and hyphen], ContainsDigitAndPercentage [token consists of digits and percentages]). These binary valued features are helpful in recognizing miscellaneous NEs such as time expressions, monetary expressions, date expressions, percentages, numerical numbers etc.

•Gazetteer Lists: Various gazetteer lists have been developed from the partially NE tagged Bengali news corpus (Ekbal and Bandyopadhyay, 2007d). These lists have been used as the binary valued features of the SVM framework. If the current token is in a particular list, then the corresponding feature is set to '+1' for the current and/or surrounding word(s); otherwise, it is set to '-1'. The following is the list of gazetteers:

(i). Organization suffix word (94 entries): This list contains the words that are helpful in identifying organization names (e.g., *kong*, *limited* etc.). The feature 'OrganizationSuffix' is set to '+1' for the current and the previous words.

(ii). Person prefix word (245 entries): This is useful for detecting person names (e.g., *sriman*, *sree*, *srimati* etc.). The feature 'PersonPrefix' is set to '+1' for the current and the next two words.

(iii). Middle name (1,491 entries): These words generally appear inside the person names (e.g., *chandra*, *nath* etc.). The feature 'MiddleName' is set to '+1' for the current, previous and the next words.

(iv). Surname (5,288 entries): These words usually appear at the end of person names as their parts. The feature 'SurName' is set to '+1' for the current word.

(v). Common location word (547 entries): This list contains the words that are part of location names and appear at the end (e.g., *sarani*, *road*, *lane* etc.). The feature 'CommonLocation' is set to '+1' for the current word.

(vi). Action verb (221 entries): A set of action verbs like *balen*, *ballen*, *ballo*, *shunllo*, *haslo* etc. often determines the presence of person names. The feature 'ActionVerb' is set to '+1' for the previous word.

(vii). Frequent word (31,000 entries): A list of most frequently occurring words in the Bengali news corpus has been prepared using a part of the corpus. The feature 'RareWord' is set to '+1' for those words that are not in this list.

(viii). Function words (743 entries): A list of function words has been prepared manually. The feature 'NonFunctionWord' is set to '+1' for those words that are not in this list.

(ix). Designation words (947 entries): A list of common designation words has been prepared. This helps to identify the position of the NEs, particularly person names (e.g., *neta*, *sangsad*, *kheloar* etc.). The feature 'DesignationWord' is set to '+1' for the next word.

(x). Person name (72, 206 entries): This list contains the first name of person names. The feature 'PersonName' is set to '+1' for the current word.

(xi). Location name (7,870 entries): This list contains the location names and the feature 'LocationName' is set to '+1' for the current word.

(xii). Organization name (2,225 entries): This list contains the organization names and the feature 'OrganizationName' is set to '+1' for the current word.

(xiii). Month name (24 entries): This contains the name of all the twelve different months of both

English and Bengali calendars. The feature 'MonthName' is set to '+1' for the current word.
(xiv). Weekdays (14 entries): It contains the name of seven weekdays in Bengali and English both. The feature 'WeekDay' is set to '+1' for the current word.

4 Experimental Results

A partially NE tagged Bengali news corpus (Ekbal and Bandyopadhyay, 2007d) has been used to create the training set for the NER experiment. Out of 34 million wordforms, a set of 150K wordforms has been manually annotated with the 17 tags as shown in Table 1 with the help of *Sanchay Editor*[4], a text editor for Indian languages. Around 20K NE tagged corpus is selected as the development set and the rest 130K wordforms are used as the training set of the SVM based NER system.

We define the *baseline* model as the one where the NE tag probabilities depend only on the current word:

$$P(t_1, t_2, t_3 ..., t_n \mid w_1, w_2, w_3 ..., w_n) = \prod_{i=1...n} P(t_i, w_i)$$

In this model, each word in the test data is assigned the NE tag that occurs most frequently for that word in the training data. The unknown word is assigned the NE tag with the help of various gazetteers and NE suffix lists.

Seventy four different experiments have been conducted taking the different combinations from the set 'F' to identify the best-suited set of features for NER in Bengali. From our empirical analysis, we found that the following combination gives the best result for the development set.

F={ $w_{i-3}w_{i-2}w_{i-1}w_iw_{i+1}w_{i+2}$, |prefix|<=3, |suffix|<=3, NE information of the window [-2, 0], POS information of the window [-1, +1], nominalPOS of the current word, nominalPREP, FirstWord, Digit features, Gazetteer lists}

The meanings of the notations, used in experimental results, are defined below:

pw, cw, nw: Previous, current and the next word; pwi, nwi: Previous and the next ith word from the current word; pt: NE tag of the previous word; pti: NE tag of the previous ith word; pre, suf: Prefix and suffix of the current word; ppre, psuf: Prefix and suffix of the previous word; npre, nsuf: Prefix and suffix of the next word; pp, cp, np: POS tag of the previous, current and the next word; ppi, npi: POS tag of the previous and the next ith word; cwnl: Current word is nominal.

Evaluation results of the development set are presented in Tables 2-4.

Feature (word, tag)	FS (%)
pw, cw, nw, FirstWord	71.23
pw2, pw, cw, nw, nw2, FirstWord	73.23
pw3, pw2, pw, cw, nw, nw2, FirstWord	**74.87**
pw3, pw2, pw, cw, nw, nw2, nw3, FirstWord	74.12
pw4, pw3, pw2, pw, cw, nw, nw2, FirstWord	74.01
pw3, pw2, pw, cw, nw, nw2, First Word, pt	75.30
pw3, pw2, pw, cw, nw, nw2, First Word, pt, pt2	**76.23**
pw3, pw2, pw, cw, nw, nw2, First Word, pt, pt2, pt3	75.48
pw3, pw2, pw, cw, nw, nw2, First Word, pt, pt2, | |suf|<=4, pre|<=4	78.72
pw3, pw2, pw, cw, nw, nw2, First Word, pt, pt2, |suf|<=3, |pre|<=3	**81.2**
pw3, pw2, pw, cw, nw, nw2, First Word, pt, pt2, |suf|<=3, |pre|<=3 |psuf|<=3	80.4
pw3, pw2, pw, cw, nw, nw2, First Word, pt, pt2, |suf|<=3, |pre|<=3, |psuf|<=3, |nsuf|<=3, |ppre|<=3, |npre|<=3	78.14
pw3, pw2, pw, cw, nw, nw2, First Word, pt, pt2, |suf|<=3, |pre|<=3, |nsuf|<=3, |npre|<=3	79.90
pw3, pw2, pw, cw, nw, nw2, First Word, pt, pt2, |suf|<=3, |pre|<=3, |psuf|<=3, |ppre|<=3,	80.10
pw3, pw2, pw, cw, nw, nw2, First Word, pt, pt2, |suf|<=3, |pre|<=3, Digit	**82.8**

Table 2. Results on the Development Set

It is observed from Table 2 that the word window [-3, +2] gives the best result (4th row) with the 'FirstWord' feature and further increase or decrease in the window size reduces the overall F-Score value. Results (7th-9th rows) show that the inclusion of NE information increases the F-Score value and the NE information of the previous two words gives the best results (F-Score=81.2%). It is indicative from the evaluation results (10th and 11th

[4]Sourceforge.net/project/nlp-sanchay

rows) that prefixes and suffixes of length up to three of the current word are very effective. It is also evident (12th-15th rows) that the surrounding word prefixes and/or suffixes do not increase the F-Score value. The F-Score value is improved by 1.6% with the inclusion of various digit features (15th and 16th rows).

Feature (word, tag)	FS (%)
pw3, pw2, pw, cw, nw, nw2, First Word, pt, pt2, |suf|<=3, |pre|<=3, Digit, pp, cp, np	**87.3**
pw3, pw2, pw, cw, nw, nw2, First Word, pt, pt2, |suf|<=3, |pre|<=3, Digit, pp2, pp, cp, np, np2	85.1
pw3, pw2, pw, cw, nw, nw2, First Word, pt, pt2, |suf|<=3, |pre|<=3, Digit, pp, cp	86.4
pw3, pw2, pw, cw, nw, nw2, First Word, pt, pt2, |suf|<=3, |pre|<=3, Digit, cp, np	85.8
pp2, pp, cp, np, np2, pt, pt2, |pre|<=3, |suf|<=3, FirstWord, Digit	41.9
pp, cp, np, pt, pt2, |pre|<=3, |suf|<=3, FirstWord, Digit	36.4
pw3, pw2, pw, cw, nw, nw2, First Word, pt, pt2, |suf|<=3, |pre|<=3, Digit, cp	86.1

Table 3. Results on the Development Set

Experimental results (2nd-5th rows) of Table 3 suggest that the POS tags of the previous, current and the next words, i.e., POS information of the window [-1, +1] is more effective than the window [-2, +2], [-1, 0], [0, +1] or the current word alone. In the above experiment, the POS tagger was developed with 7 POS tags. Results (6th and 7th rows) also show that POS information with the word is helpful but only the POS information without the word decreases the F-Score value significantly. Results (4th and 5th rows) also show that the POS information of the window [-1, 0] is more effective than the POS information of the window [0, +1]. So, it can be argued that the POS information of the previous word is more helpful than the POS information of the next word.

In another experiment, the POS tagger was developed with 26 POS tags and the use of this tagger has shown the F-Score value of 85.6% with the feature (word, tag)=[pw3, pw2, pw, cw, nw, nw2, FirstWord, pt, pt2, |suf|<=3, |pre|<=3, Digit, pp, cp, np]. So, it can be decided that the smaller POS tagset is more effective than the larger POS tagset in NER. We have observed from two different experiments that the overall F-Score values can further be improved by 0.5% and 0.3%, respectively, with the 'nominalPOS' and 'nominalPREP' features. It has been also observed that the 'nominalPOS' feature of the current word is only helpful and not of the surrounding words. The F-Score value of the NER system increases to 88.1% with the feature: feature (word, tag)=[pw3, pw2, pw, cw, nw, nw2, FirstWord, pt, pt2, |suf|<=3, |pre|<=3, Digit pp, cp, np, cwnl, nominalPREP].

Experimental results with the various gazetteer lists are presented in Table 4 for the development set. Results demonstrate that the performance of the NER system can be improved significantly with the inclusion of various gazetteer lists. The overall F-Score value increases to 90.7%, which is an improvement of 2.6%, with the use of gazetteer lists.

The best set of features is identified by training the system with 130K wordforms and tested with the help of development set of 20K wordforms. Now, the development set is included as part of the training set and resultant training set is thus consisting of 150K wordforms. The training set has 20,455 person names, 11,668 location names, 963 organization names and 11,554 miscellaneous names. We have performed 10-fold cross validation test on this resultant training set. The Recall, Precision and F-Score values of the 10 different experiments for the 10-fold cross validation test are presented in Table 5. The overall average Recall, Precision and F-Score values are 94.3%, 89.4% and 91.8%, respectively.

The other existing Bengali NER systems along with the *baseline* model have been also trained and tested with the same data set. Comparative evaluation results of the 10-fold cross validation tests are presented in Table 6 for the four different models. It presents the average F-Score values for the four major NE classes: 'Person name', 'Location name', 'Organization name' and 'Miscellaneous name'. Two different NER models, A and B, are defined in (Ekbal and Bandyopadhyay, 2007b). The model A denotes the NER system that does not use linguistic knowledge and B denotes the system that uses linguistic knowledge. Evaluation results of Table 6 show that the SVM based NER model has reasonably high F-Score value. The average F-Score value of this model is 91.8%, which is an improvement of 7.3% over the best-reported

HMM based Bengali NER system (Ekbal et al., 2007c). The reason behind the rise in F-Score value might be its better capability to capture the morphologically rich and overlapping features of Bengali language.

Feature (word, tag)	FS (%)
pw3, pw2, pw, cw, nw, nw2, First Word, pt, pt2, |suf|<=3, |pre|<=3, Digit pp, cp, np, cwnl, nominal-PREP, DesignationWord, Non-FunctionWord	89.2
pw3, pw2, pw, cw, nw, nw2, First Word, pt, pt2, |suf|<=3, |pre|<=3, Digit pp, cp, np, cwnl, nominal-PREP, DesignationWord, Non-FunctionWord	89.5
pw3, pw2, pw, cw, nw, nw2, First Word, pt, pt2, |suf|<=3, |pre|<=3, Digit pp, cp, np, cwnl, nominal-PREP, DesignationWord, Non-FunctionWord OrganizationSuffix, PersonPrefix	90.2
pw3, pw2, pw, cw, nw, nw2, First Word, pt, pt2, |suf|<=3, |pre|<=3, Digit pp, cp, np, cwnl, nominal-PREP, DesignationWord, Non-FunctionWord OrganizationSuffix, PersonPrefix, MiddleName, CommonLocation	90.5
pw3, pw2, pw, cw, nw, nw2, First Word, pt, pt2, |suf|<=3, |pre|<=3, Digit pp, cp, np, cwnl, nominal-PREP, DesignationWord, No-FunctionWord OrganizationSuffix, PersonPrefix, MiddleName, CommonLocation, Other gazetteers	**90.7**

Table 4. Results on the Development Set

The F-Score value of the system increases with the increment of training data. This fact is represented in Figure 1. Also, it is evident from Figure 1 that the value of 'Miscellaneous name' is nearly close to 100% followed by 'Person name', 'Location name' and 'Organization name' NE classes with the training data of 150K words.

Test set no.	Recall	Precision	FS (%)
1	92.5	87.5	89.93
2	92.3	87.6	89.89
3	94.3	88.7	91.41
4	95.4	87.8	91.40
5	92.8	87.4	90.02
6	92.4	88.3	90.30
7	94.8	91.9	93.33
8	93.8	90.6	92.17
9	96.9	91.8	94.28
10	97.8	92.4	95.02
Average	**94.3**	**89.4**	**91.8**

Table 5. Results of the 10-fold cross validation test

Model	F_P	F_L	F_O	F_M	F_T
Baseline	61.3	58.7	58.2	52.2	56.3
A	75.3	74.7	73.9	76.1	74.5
B	79.3	78.6	78.6	76.1	77.9
HMM	85.5	82.8	82.2	92.7	84.5
SVM	91.4	89.3	87.4	99.2	**91.8**

Table 6. Results of the 10-fold cross validation test (F_P: Avg. f-score of 'Person', F_L: Avg. f-score of 'Location', F_O: Avg. f-score of 'Organization', F_M: Avg. f-score of 'Miscellaneous' and F_T: Overall avg. f-score of all classes)

5 Conclusion

We have developed a NER system using the SVM framework with the help of a partially NE tagged Bengali news corpus, developed from the archive of a leading Bengali newspaper available in the web. It has been shown that the contextual window of size six, prefix and suffix of length up to three of the current word, POS information of the window of size three, first word, NE information of the previous two words, different digit features and the various gazetteer lists are the best-suited features for NER in Bengali. Experimental results with the 10-fold cross validation test have shown reasonably good Recall, Precision and F-Score values. The performance of this system has been compared with the existing three Bengali NER systems and it has been shown that the SVM-based system outperforms other systems. One possible reason behind the high Recall, Precision and F-Score values of the SVM based system might be its effectiveness to handle the diverse and overlapping features of the highly inflective Indian languages.

The proposed SVM based system is to be trained and tested with the other Indian languages, particularly Hindi, Telugu, Oriya and Urdu. Analyzing the performance of the system using other methods like MaxEnt and CRFs will be other interesting experiments.

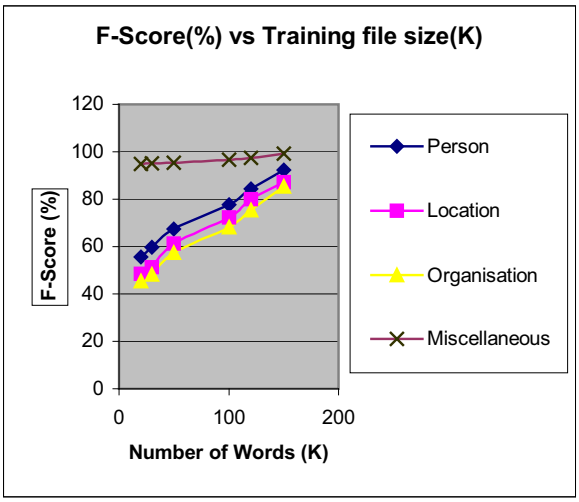

Fig. 1. F-Score VS Training file size

References

Anderson, T. W. and Scolve, S. 1978. Introduction to the Statistical Analysis of Data. *Houghton Mifflin.*

Asahara, Masayuki and Matsumoto, Yuji. 2003. Japanese Named Entity Extraction with Redundant Morphological Analysis. In *Proc. of HLT-NAACL.*

Babych, Bogdan, A. Hartley. 2003. Improving Machine Translation Quality with Automatic Named Entity Recognition. In *Proceedings of EAMT/EACL 2003 Workshop on MT and other language technology tools*, 1-8, Hungary.

Bikel, Daniel M., R. Schwartz, Ralph M. Weischedel. 1999. An Algorithm that Learns What's in Name. *Machine Learning (Special Issue on NLP),* 1-20.

Bothwick, Andrew. 1999. A Maximum Entropy Approach to Named Entity Recognition. *Ph.D. Thesis*, New York University.

Chinchor, Nancy. 1995. MUC-6 Named Entity Task Definition (Version 2.1). *MUC-6*, Maryland.

Chinchor, Nancy. 1998. MUC-7 Named Entity Task Definition (Version 3.5). *MUC-7*, Fairfax, Virginia.

Cunningham, H. 2001. GATE: A General Architecture for Text Engineering. *Comput. Humanit.* (36), 223-254.

Ekbal, Asif, and S. Bandyopadhyay. 2007a. Pattern Based Bootstrapping Method for Named Entity Recognition. In *Proceedings of ICAPR*, India, 349-355.

Ekbal, Asif, and S. Bandyopadhyay. 2007b. Lexical Pattern Learning from Corpus Data for Named Entity Recognition. In *Proc. of ICON*, India, 123-128.

Ekbal, Asif, Naskar, Sudip and S. Bandyopadhyay. 2007c. Named Entity Recognition and Transliteration in Bengali. *Named Entities: Recognition, Classification and Use, Special Issue of Lingvisticae Investigationes Journal, 30:1 (2007), 95-114.*

Ekbal, Asif, and S. Bandyopadhyay. 2007d. A Web-based Bengali News Corpus for Named Entity Recognition. *Language Resources and Evaluation Journal* (To appear December).

Joachims , T. 1999. Making Large Scale SVM Learning Practical. In *B. Scholkopf, C. Burges and A. Smola editions, Advances in Kernel Methods-Support Vector Learning, MIT Press*.

Kudo, Taku and Matsumoto, Yuji. 2001. Chunking with Support Vector Machines. In *Proceedings of NAACL,* 192-199.

Kudo, Taku and Matsumoto, Yuji. 2000. Use of Support Vector Learning for Chunk Identification. In *Proceedings of CoNLL-2000.*

Lafferty, J., McCallum, A., and Pereira, F. 2001. Conditional Random Fields: Probabilistic Models for Segmenting and Labeling Sequence Data. In *Proc. of 18th International Conference on Machine learning,* 282-289.

Li, Wei and Andrew McCallum. 2003. Rapid Development of Hindi Named Entity Recognition Using Conditional Random Fields and Feature Inductions. *ACM TALIP*, 2(3), (2003), 290-294.

Moldovan, Dan I., Sanda M. Harabagiu, Roxana Girju, P. Morarescu, V. F. Lacatusu, A. Novischi, A. Badulescu, O. Bolohan. 2002. LCC Tools for Question Answering. In *Proceedings of the TREC*, 1-10.

Sekine, Satoshi. 1998. Description of the Japanese NE System Used for MET-2. *MUC-7*, Fairfax, Virginia.

Takeuchi, Koichi and Collier, Nigel. 2002. Use of Support Vector Machines in Extended Named Entity Recognition. In *Proceedings of 6th CoNLL,* 119-125.

Vapnik, Valdimir N. 1995. The Nature of Statistical Learning Theory. *Springer*.

Yamada, Hiroyasu, Taku Kudo and Yuji Matsumoto. 2002. Japanese Named Entity Extraction using Support Vector Machine. In *Transactions of IPSJ,* Vol. 43, No. 1, 44-53.

Domain Focused Named Entity Recognizer for Tamil Using Conditional Random Fields

Vijayakrishna R
AU-KBC Research Centre
MIT Campus, Anna University
Chennai, India
vijayakrishna@au-kbc.org

Sobha L
AU-KBC Research Centre
MIT Campus, Anna University
Chennai, India
sobha@au-kbc.org

Abstract

In this paper, we present a domain focused Tamil Named Entity Recognizer for tourism domain. This method takes care of morphological inflections of named entities (NE). It handles nested tagging of named entities with a hierarchical tagset containing 106 tags. The tagset is designed with focus to tourism domain. We have experimented building Conditional Random Field (CRF) models by training the noun phrases of the training data and it gives encouraging results.

1 Introduction

Named Entity Recognition (NER) is the task of identifying and classifying the entities such as person names, place names, organization names etc, in a given document. Named entities play a major role in information extraction. NER has been a defined subtask in Message Understanding Conference (MUC) since MUC 6. A well performing NER is important for further level of NLP techniques.

In general NER is a hard problem.. Words can have multiple uses and there is an unbounded number of possible names. Many techniques have been applied in Indian and European languages for NER. Some of them are rule based system (Krupka and Hausman, 1998), which makes use of dictionary and patterns of named entities, Decision trees (Karkaletsis et al., 2000), Hidden Morkov Model (HMM) (Biker, 1997), Maximum Entropy Morkov Model (MEMM) (Borthwick et al., 1998), Conditional Random Fields (CRF) (Andrew McCallum and Wei Li, 2003) etc. In short, the approaches can be classified as rule-based approach, machine learning approach or hybrid approach.

For Indian languages, many techniques have been tried by different people. MEMM system for Hindi NER (Kumar and Pushpak, 2006) gave an average F1 measure of 71.9 for a tagset of four named entity tags.

NER has been done generically and also domain specific where a finer tagset is needed to describe the named entities in a domain. Domain specific NER is common and has been in existence for a long time in the Bio-domain (Settles 2004) for identification of protein names, gene names, DNA names etc.

We have developed a domain specific hierarchical tagset consisting of 106 tags for tourism domain. We have used Conditional Random Fields, a machine learning approach to sequence labeling task, which includes NER.

Section 2 gives a brief introduction to Conditional Random Fields (CRF). Section 3 discusses the nature of named entities in Tamil, followed by section 4 describing the tagset used in tourism domain. Section 5 describes how we have presented the training data to build CRF models and how we have handled nested tagging. Sections 6 and 7 explain the experiments and results. The paper is concluded in section 8.

Proceedings of the IJCNLP-08 Workshop on NER for South and South East Asian Languages, pages 59–66,
Hyderabad, India, January 2008.

2 Conditional Random Fields (CRF)

Conditional Random Fields (CRF) (Lafferty et al., 2001) is a machine learning technique. CRF overcomes the difficulties faced in other machine learning techniques like Hidden Markov Model (HMM) (Rabiner, 1989) and Maximum Entropy Markov Model (MEMM) (Berger et al., 1996). HMM does not allow the words in the input sentence to show dependency among each other. MEMM shows a label bias problem because of its stochastic state transition nature. CRF overcomes these problems and performs better than the other two. HMM, MEMM and CRF are suited for sequence labeling task. But only MEMM and CRF allows linguistic rules or conditions to be incorporated into machine learning algorithm.

Lafferty et al, define Conditional Random Fieds as follows**:** "Let **G** = (**V**,**E**) be a graph such that **Y** = $(\mathbf{Y}_v)_{v \in \mathbf{V}}$, so that **Y** is indexed by the vertices of **G**. Then (**X**,**Y**) is a conditional random field in case, when conditioned on **X**, the random variables $\mathbf{Y}_v$ obey the Markov property with respect to the graph: $p(\mathbf{Y}_v|\mathbf{X},\mathbf{Y}_w,w?v) = p(\mathbf{Y}_v|\mathbf{X},\mathbf{Y}_w,w\sim v)$, where w~v means that w and v are neighbors in G".

Here **X** denotes a sentence and **Y** denotes the label sequence. The label sequence **y** which maximizes the likelihood probability $p_?(\mathbf{y}|\mathbf{x})$ will be considered as the correct sequence, while testing for new sentence **x** with CRF model **?** . The likelihood probability $p_?(\mathbf{y}|\mathbf{x})$ is expressed as follows.

$$p_\theta(y \mid x) \propto$$

$$\exp\left(\sum_{e \in E,k} \lambda_k f_k(e, y|_e, x) + \sum_{v \in V,k} \mu_k g_k(v, y|_v, x)\right)$$

where $?_k$ and μ_k are parameters from CRF model ? and f_k and g_k are the binary feature functions that we need to give for training the CRF model. This is how we integrate linguistic features into machine learning models like CRF.

In NER task, the sequence of words which forms a sentence or a phrase can be considered as the sequence **x** and the sequence formed by named entity label for each word in the sequence **x** is the label sequence **y**. Now, the task of finding **y** that best describes **x** can be found by maximizing the likelihood probability $p_?(\mathbf{y}|\mathbf{x})$. Thus, NER task can be considered as a sequence labeling task. Hence CRF can be used for NER task.

3 Characteristics of Named Entities in Tamil

Unlike English, there is no concept of capital letters in Tamil and hence no capitalization information is available for named entities in Tamil. All named entities are nouns and hence are Noun Phrases. But not all Noun Phrases are Named Entities. Since named entities are noun phrases, they take all morphological inflections. This makes a single named entity to appear as different words in different places. By applying Morphological analysis on words, the root words of inflected Named Entities can be obtained. These roots will be uninflected Named Entities which is what is required in most applications. Some type of named entities like date, money etc, occur in specific patterns.

Example for inflected named entity:

ceVnYnYE**kku** ("to Chennai").

Example for pattern in named entity:

2006 aktopar 25Am wewi ("25th October, 2006")

Pattern: <4 digits> <month> <1-2 digit> [Am wewi]

4 Named Entity Tagset used

The tagset which we use here for NER contains 106 tags related to each other hierarchically. This type of tagset is motivated from "ACE English Annotation Guidelines for Entities" developed by Linguistic Data Consortium. The tagset which we use is built in-house with focus to tourism domain.

4.1 Sample Tags

Sample tags from the entire tagset is shown below with their hierarchy.

1. Enamex
    1.1. Person
        1.1.1. Individual
            1.1.1.1. Family Name
            1.1.1.2. Title

1.1.2.Group
1.2. Organization
. . . .
1.3. Location
. . . .
1.4. Facilities
. . . .
1.5. Locomotive
. . . .
1.6. Artifact
. . . .
1.7. Entertainment
. . . .
1.8. Materials
. . . .
1.9. Livthings
. . . .
1.10. Plants
. . . .
1.11. Disease
. . . .
2. Numex
2.1. Distance
2.2. Money
2.3. Quantity
2.4. Count
3. Timex
3.1. Time
3.2. Year
3.3. Month
3.4. Date
3.5. Day
3.6. Period
3.7. Sday

Certain tags in this tagset are designed with focus to Tourism and Health Tourism domain, such as place, address, water bodies (rivers, lakes etc.,), religious places, museums, parks, monuments, airport, railway station, bus station, events, treatments for diseases, distance and date.

The tags are assigned with numbers 1,2,3 for zeroth level, the tags with numbers 1.1, 1.11, 2.1 ,2.4 and 3.1 ,3.7 etc for level-1, the tags with numbers 1.1.1, 1.1.2, 1.2.1 etc as level-2 and the tags with numbers 1.1.1.1, 1.1.1.2, 1.2.4.1 etc for level-3 because they occur in the hierarchy in corresponding levels. We have 3 tags in zeroth level, 22 tags in level-1, 50 tags in level-2 and 31 tags in level-3.

4.2 Sample Annotation

Tamil :

<person> <city> mawurE </city> <individual> manYi <familyname> Eyar </familyname> </individual> </person> <city> ceVnYnYEkku </city> vanwAr.

English equivalent :

<person> <city> Madhurai </city> <individual> Mani <familyname> Iyer </familyname> </individual> </person> came to <city> Chennai </city>.

5 NER using CRF

We used CRF++ (Taku Kudo, 2005), an open source toolkit for linear chain CRF. This tool when presented with the attributes extracted from the training data builds a CRF model with the feature template specified by us. When presented with the model thus obtained and attributes extracted from the test data, CRF tool outputs the test data tagged with the labels that has been learnt.

5.1 Presenting training data

Training data will contain nested tagging of named entities as shown in section 4.2. To handle nested tagging and to avoid ambiguities, we isolate the tagset into three subsets, each of which will contain tags from one level in the hierarchy. Now, the training data itself will be presented to CRF as three sets of training data. From this, we will get three CRF models, one for each level of hierarchy.

Example:

The sample sentence given in section 4.2 will be presented to CRF training for each level of hierarchy as follows:

Level-1:

<location> mawurE </location> <person> manYi Eyar </person> <location> ceVnYnYEkku </location> vanwAr.

Level-2:

<place> mawurE </place> <individual> manYi Eyar </individual> <place> ceVnYnYEkku </place> vanwAr.

Level-3:

<city> mawurE </city> manYi <familyname> Eyar </familyname> <city> ceVnYnYEkku </city> vanwAr.

Notice that the tags ‘location’ and ‘place’ are not specified in the input sentence. In the

hierarchy, the 'location' tag is the parent tag of 'place' tag which is a parent tag of 'city' tag. Thus for the word "mawurE", level-1 tag is 'location', level-2 tag is 'place' and level-3 tag is 'city'.

5.2 Attributes and Feature Templates

Attributes are the dependencies from which the system can infer a phrase to be named entity or not. Features are the conditions imposed on these attributes. Feature templates help CRF engine to form features from the attributes of the training data. From the characteristics of named entities in Tamil, we see that it is only the noun phrases that are possible candidates for Named Entities. So we apply Noun Phrase Chunking and consider only noun phrases and train on them. The attributes that we arrived at are explained below:

1. **Roots of words**: This is to ignore inflections in named entities. Also to learn the context in which the named entity occurs, we consider two words prior and two words subsequent to the word under analysis and take unigram, bigram and trigram combinations of them as attributes.
2. **Their Parts of Speech (POS)**: This will give whether a noun is proper noun or common noun. POS of current word is considered.
3. **Words and POS combined**: The present word combined with the POS tag of the previous two words and the present word combined with POS of the next two words are taken as features.
4. **Dictionary of Named Entities**: A list of named entities is collected for each type of named entities. Root words are checked against the dictionary and if present in the dictionary, the dictionary feature for the corresponding type of named entity is considered positive.
5. **Patterns**: Certain types of named entities such as date, time, money etc., show patterns in their occurrences. These patterns are listed out. The current noun phrase is checked against each pattern. The feature is taken as true for those patterns which are satisfied by the current noun phrase.

Example Patterns:

Date: <4 digits> <month> <1-2 digit> [Am wewi]

Money: rU. <digits> [Ayiram|latcam|koti]

(English Equivalent:

Rs. <digits> [thousands|lakhs|crores])

6. Bigram of Named Entity label

A feature considering the bigram occurrences of the named entity labels in the corpus is considered. This is the feature that binds the consecutive named entity labels of a sequence and thus forming linear chain CRFs. Sample noun phrase with level-1 tags:

arulYmiku JJ person
cupramaNiya NNPC person
cuvAmi NNPC person
wirukoyil NNC location
vayalUr NNP location

English Equivalent:

Gracious	JJ	person
Subramaniya	NNPC	person
Swami	**NNPC**	**person**
Temple	NNC	location
Vayalore	NNP	location

Attributes are extracted for each token in the noun phrase. For example, the attributes for third token in the sample noun phrase given are as follows.

1. Unigram: arulYmiku, cupramaNiya, cuvAmi, wirukoyil, vayalUr.
2. Bigram: cupramaNiya/cuvAmi, cuvAmi/ wirukoyil
3. Trigram: cupramaNiya/cuvAmi/wirukoyil
4. POS of current word: NNPC
5. Word and previous 2 POS: JJ/NNPC/ cuvAmi
6. Word and next 2 POS: cuvAmi/NNC/NNP
7. Bigram of NE labels: person/person

The CRF training process described above is illustrated in Figure-1.

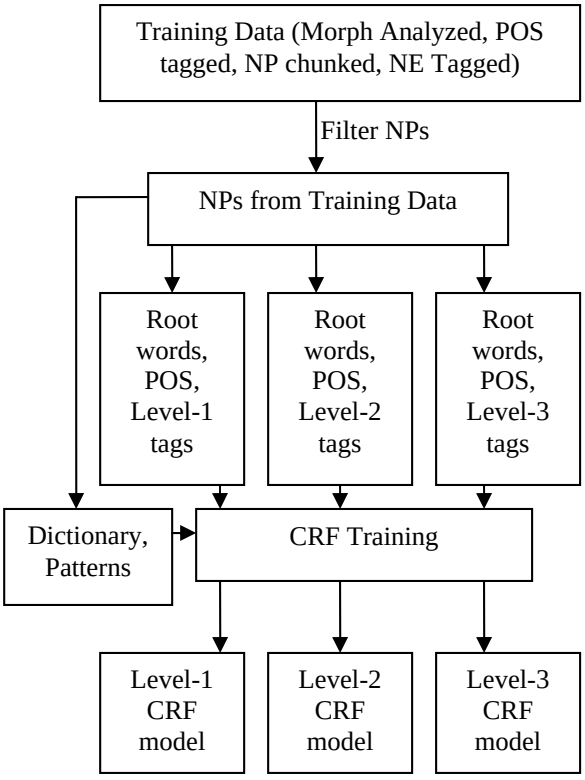

Figure 1.Training CRF for NER

5.3 Presenting testing data

Test data will also be presented in way similar to how we presented the training data. Test data is processed for Morph analysis, POS (Arulmozhi et al., 2004) and NP chunking (Sobha and Vijay Sundar Ram, 2006). Here also, the same set of attributes and feature templates are used. Now, the test data is tagged with each of the CRF models built for three levels of hierarchy. All the three outputs are merged to get a combined output. The CRF testing is illustrated in Figure 2.

6 Experiments

A 94k words corpus is collected in Tamil for tourism domain. Morph Analysis, POS tagging, NP chunking and named entity annotation are done manually on the corpus. This corpus contains about 20k named entities. This corpus is split into two sets. One forms the training data and the other forms the test data. They consist of 80% and 20% of the total data respectively. CRF is trained with training data and CRF models for each of the levels in the hierarchy are obtained. With these models the test data is tagged and the output is evaluated manually.

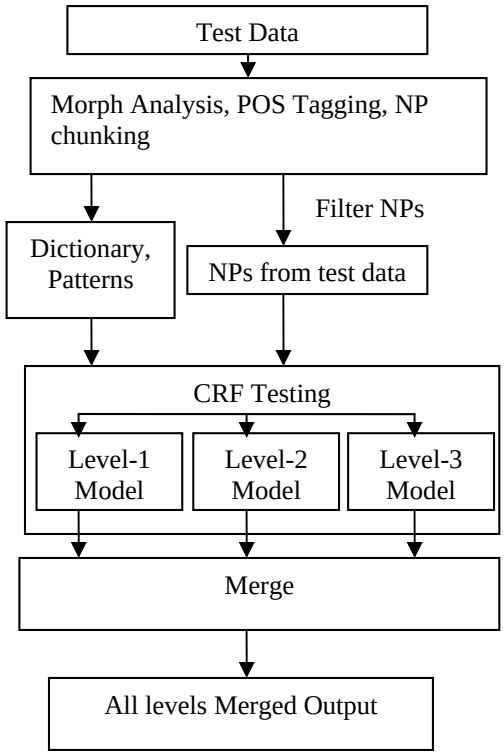

Figure 2. CRF Testing for NER

7 Results

The results of the above experiment are as follows. Here, NE means Named Entity, NP means noun phrase.

Number of NPs in test data = 7922

There are totally 4059 NEs in the test data. All of them bear level-1 tags. Out of 4059 NEs, 3237 NEs bear level-2 tags and 727 NEs bear level-3 tags. The result from the system is shown in Table 1 and Table 2.

The system performs well for domain focused corpus. It identifies inflected named entities efficiently by considering the root form of each word in noun phrases. The reason for good

precision is that tagging is done only when the root word that it is seeing is already learnt from the training corpus or the context of the current word is similar to the context of the named entities that it has learnt from the training corpus. However, in some words like 'arccunYAnawi' (Arjuna River), the Morph Analyzer gives two root words which are 'arccunYa' and 'nawi'. For our case, only the first word is considered and the system tags it as 'person' instead of 'waterbodies'.

Named Entity Level	Level-1	Level-2	Level-3
Number of NEs in data	4059	3237	727
Number of NEs identified by NER engine	3414	2667	606
Number of NEs identified correctly	3056	2473	505
Precision %	89.51	**92.73**	83.33
Recall %	75.29	**76.40**	69.46
F1 measure %	81.79	**83.77**	75.77

Table 1. Evaluation of output from NER engine for each level

Performance Measure	Value in %
Precision	88.52
Recall	73.71
F1 Measure	80.44

Table 2. Overall result from NER engine

When there are new named entities which are not in training corpus, CRF tries to capture the context and tags accordingly. In such cases irrelevant context that it may learn while training will cause problem resulting in wrong tagging. This affects the precision to some extent. When the named entities and their context are new to CRF, then they are most likely not tagged. This affects the recall.

From Table 1, we see that the system performs better for level-2 tags than for level-1 tags even though level-1 tags are less in number than level-2 tags and occur more frequently than level-2 tags. This is so because the named entities with level-2 tags have relatively more context and are lesser in length (number of words in the named entity) than the named entities in level-1 tags. Level-3 tags contain lesser number of tags than level-2 tags and also occur less frequently. Because of relatively more data sparseness, the system is unable to perform well for level-3 tags as it can for other levels.

8 Conclusion

We see that Conditional Random Fields is well suited for Named Entity recognition task in Indian languages also, where the inflection of named entities can be handled by considering their root forms. A good precision can be obtained by presenting only the noun phrases for both testing and training.

References

Arulmozhi P, Sobha L and Kumara Shanmugam B. 2004. *Parts of Speech Tagger for Tamil*, Symposium on Indian Morphology, Phonology & Language Engineering, March 19-21, IIT Kharagpur. :55-57.

Berger A, Della Pietra S and Della Pietra V. 1996. *A Maximum Entropy Approach to Natural Language Processing*. Computational Linguistics, 22(1).

Bikel D M. 1997. *Nymble: a high-performance learning name-finder*. In Proceedings of the Fifth Conference on Applied Natural Language Processing. :194-201.

Borthwick A, Sterling J, Agichtein E and Grishman R. 1998. *Description of the MENE named Entity System*, In Proceedings of the Seventh Machine Understanding Conference (MUC-7).

Karkaletsis V, Pailouras G and Spyropoulos C D. 2000. *Learning decision trees for named-entity recognition and classification*. In Proceedings of the ECAI Workshop on Machine Learning for Information Extraction.

Krupka G R and Hausman K. 1998. *Iso Quest Inc: Description of the NetOwl Text Extraction System as used for MUC-7*. In Proceedings of Seventh Machine Understanding Conference (MUC 7).

Kumar N, Pushpak Bhattacharyya. 2006. *Named Entity Recognition in Hindi using MEMM*.

John Lafferty, Andrew McCallum, Fernando Pereira. 2001. *Conditional Random Fields: Probabilistic Models for Segmenting and Labeling Sequence Data*. In Proceedings of the Eighteenth International

Conference on Machine Learning (ICML-2001). 282-289.

Andrew McCallum and Wei Li. 2003. *Early Results for Named Entity Recognition with Conditional Random Fields, Feature Induction and Web-Enhanced Lexicons*. Seventh Conference on Natural Language Learning (CoNLL).

Lawrence R. Rabiner. 1989. *A Tutorial on Hidden Markov Models and Selected Applications in Speech Recognition*. In Proceedings of the IEEE, 77(2):257–286.

Settles B. (2004). *Biomedical Named Entity Recognition Using Conditional Random Fields and Rich Feature Sets*. In Proceedings of the International Joint Workshop on Natural Language Processing in Biomedicine and its Applications (NLPBA), Geneva, Switzerland. pp:104-107.

Sobha L, Vijay Sundar Ram R. 2006. *Noun Phrase Chunking in Tamil*. In proceedings of the MSPIL-06, IIT Bombay. pp:194-198.

Taku Kudo. 2005. *CRF++, an open source toolkit for CRF*, http://crfpp.sourceforge.net .

A Character n-gram Based Approach for Improved Recall in Indian Language NER

Praneeth M Shishtla
praneethms
@students.iiit.ac.in

Prasad Pingali
pvvpr
@iiit.ac.in

Vasudeva Varma
vv@iiit.ac.in

Language Technologies Research Centre
International Institute of Information Technology
Hyderabad, India

Abstract

Named Entity Recognition (NER) is the task of identifying and classifying all proper nouns in a document as person names, organization names, location names, date & time expressions and miscellaneous. Previous work (Cucerzan and Yarowsky, 1999) was done using the complete words as features which suffers from a low recall problem. Character n-gram based approach (Klein et al., 2003) using generative models, was experimented on English language and it proved to be useful over the word based models. Applying the same technique on Indian Languages, we experimented with Conditional Random Fields (CRFs), a discriminative model, and evaluated our system on two Indian Languages Telugu and Hindi. The character n-gram based models showed considerable improvement over the word based models. This paper describes the features used and experiments to increase the recall of Named Entity Recognition Systems which is also language independent.

1 Introduction

The objective of NER is to classify all tokens in a text document into predefined classes such as person, organization, location, miscellaneous. NER is a precursor to many language processing tasks. The creation of a subtask for NER in Message Understanding Conference (MUC) (Chinchor, 1997) reflects the importance of NER in Information Extraction (IE). NER also finds aplication in question answering systems (Toral et al., 2005; Molla et al., 2006), and machine translation (Babych and Hartley, 2003). NER is an essential subtask in organizing and retrieving biomedical information (Tsai, 2006). NER can be treated as a two step process

- identification of proper nouns.
- classification of these identified proper nouns.

Challenges in named entity recognition.

Many named entities (NEs) occur rarely in corpus if at all.

Ambiguity of NEs. Ex *Washington* can be a person's name or location.

There are many ways of mentioning the same NE. *Ex: Mahatma Gandhi, M.K.Gandhi, Mohandas Karamchand Gandhi, Gandhi* all refer to the same person. *New Jersey, NJ* both refer to the same location.

In English, the problem of identifying NEs is solved to some extent by using the capitalization feature. Most of the named entities begin with a capital letter which is a discriminating feature for classifying a token as named entity. In addition to the above challenges, the complexity of Indian Languages pose few more problems. In case of Indian languages there is no concept of capitalization. Ex: The person name *Y.S.R (in english)* is represented as *ysr* in the Indian Languages.

Agglutinative property of the Indian Languages makes the identification more difficult. *For example: hyderabad, hyderabad ki, hyderabadki, hyderabadlo, hyderabad ni, hyderabad ko* etc .. all refer to the place Hyderabad. where *lo, ki, ni* are all postpostion markers in Telugu and *ko* is a postposition

Proceedings of the IJCNLP-08 Workshop on NER for South and South East Asian Languages, pages 67–74,
Hyderabad, India, January 2008.

marker in Hindi.

There are many ways of representing acronyms. The letters in acronyms could be the English alphabet or the native alphabet. Ex: *B.J.P* and *BaJaPa* both are acronyms of *Bharatiya Janata Party*. Indian Languages lack particular standard for forming acronyms.

Due to these wide variations and the agglutinative nature of Indian languages, probabilistic graphical models result in very less recall. If we are able to identify the presence of a named entity with a fairly good amount of accuracy, classification then can be done efficiently. But, when the machine fails to identify the presence of named entities, there is no chance of entity classification because we miss many of the named entities (less recall which results in less F-measure,$F_{\beta=1}$). So we focus mainly on the ways to improve the recall of the system. Also, Indian Languages have a relatively free word order, i.e. the words (named entities) can occupy any place in the sentence. This change in the word position is compensated using case markers.

2 Related Work & Our Contributions

The state-of-art techniques for Indic languages(Telugu and Hindi) use word based models which suffer from low recall, use gazetteers and are language dependent. As such there is no NER system for Telugu. Previously (Klein et al., 2003) experimented with character-level models for English using character based HMM which is a generative model. We experimented using the discriminative model for English, Hindi and Telugu.

- We propose an approach that increases the recall of Indic languages (even the agglutinative languages).
- The model is language independent as none of the language resources is needed.

3 Problem Statement

3.1 NER as sequence labelling task

Named entity recognition (NER) can be modelled as a sequence labelling task (Lafferty et al., 2001). Given an input sequence of words $W_1^n = w_1 w_2 w_3 ... w_n$, the NER task is to construct a label sequence $L_1^n = l_1 l_2 l_3 ... l_n$, where label l_i either belongs to the set of predefined classes for named entities or is none (representing words which are not proper nouns). The general label sequence l_1^n has the highest probability of occuring for the word sequence W_1^n among all possible label sequences, that is

$$\hat{L}_1^n = \text{argmax} \{\Pr(L_1^n \mid W_1^n)\}$$

3.2 Tagging Scheme

We followed the IOB tagging scheme (Ramshaw and Marcus, 1995) for all the three languages (English, Hindi and Telugu). In this scheme each line contains a word at the beginning followed by its tag. The tag encodes the type of named entity and whether the word is in the beginning or inside the NE. Empty lines represent sentence (document) boundaries. An example of the IOB tagging scheme is given in Table 1.

Words tagged with O are outside of named entities

Token	Named Entity Tag
Dr.	B-PER
Talcott	I-PER
led	O
a	O
team	O
of	O
researchers	O
from	O
the	O
National	B-ORG
Cancer	I-ORG
Institute	I-ORG

Table 1: IOB tagging scheme.

and the I-XXX tag is used for words inside a named entity of type XXX. Whenever two entities of type XXX are immediately next to each other, the first word of the second entity will be tagged B-XXX in order to show that it starts another entity. This tagging scheme is the IOB scheme originally put forward by Ramshaw and Marcus (Ramshaw and Marcus, 1995).

4 Conditional Random Fields

Conditional Random Fields (CRFs) (Wallach, 2004) are undirected graphical models used to calculate

the conditional probability of values on designated output nodes given the values assigned to other designated input nodes. In the special case in which the output nodes of the graphical model are linked by edges in a linear chain, CRFs make a first-order Markov independence assumption, and thus can be understood as conditionally-trained Finite State Machines (FSMs).

Let o = $\langle O_1, O_2, ... O_T \rangle$ be some observed input data sequence, such as a sequence of words in text in a document, (the values on n input nodes of the graphical model). Let **S** be a set of Finite State Machine (FSM) states, each of which is associated with a label, l $\in \mathscr{L}$.
Let **s** = $\langle s_1, s_2, ... s_T, \rangle$ be some sequence of states,(the values on T output nodes). By the Hammersley-Clifford theorem CRFs define the conditional probability of a state sequence given an input sequence to be

$$P(s|o) = \frac{1}{Z_o} * exp(\sum_{t=1}^{T} \sum_{k} \lambda_k f_k(s_{t-1}, s_t, o, t))$$

where Z_o is a normalization factor over all state sequences, is an arbitrary feature function over its arguments, and λ_k is a learned weight for each feature function. A feature function may, for example, be defined to have value 0 or 1. Higher λ weights make their corresponding FSM transitions more likely.

CRFs define the conditional probability of a label sequence based on total probability over the state sequences, $P(l|o) = \sum_{s:l(s)=l} P(s|o)$ where l(s) is the sequence of labels corresponding to the labels of the states in sequence s. Note that the normalization factor, Z_o, (also known in statistical physics as the partition function) is the sum of the scores of all possible state sequences,

$$Z_o = \sum_{s \in S^T} * exp(\sum_{t=1}^{T} \sum_{k} \lambda_k f_k(s_{t-1}, s_t, o, t))$$

and that the number of state sequences is exponential in the input sequence length, T. In arbitrarily-structured CRFs, calculating the partition function in closed form is intractable, and approximation methods such as Gibbs sampling, or loopy belief propagation must be used.

5 Features

There are many types of features used in NER systems.

Many systems use binary features i.e. the word-internal features, which indicate the presence or absence of particular property in the word. (Mikheev, 1997; Wacholder et al., 1997; Bikel et al., 1997). Following are examples of commonly used binary features: All-Caps (IBM), internal capitalization (eBay), initial capital (Abdul Kalam), uncapitalized word (can), 2-digit number (83, 73), 4-digit number (1983, 2007), all digits (8, 28, 1273) etc. The features that correspond to the capitalization are not applicable to Indian languages. Also, we have not used any of the binary features in any of our models.

Dictionaries: Dictionaries are used to check if a part of the named entity is present in the dictionary. These dictionaries are called as gazetteers. The problem with the Indian languages is that there are no proper gazetteers in Indian languages.

Lexical features like a sliding window [w_{-2},w_{-1},w_o,w_1,w_2] are used to create a lexical history view. Prefix and suffix tries were also used previously (Cucerzan and Yarowsky, 1999).

Linguistics features like Part Of Speech, Chunk, etc are also used. In our approach we don't use any of these language specific (linguistic) information.

5.1 Our Features

In our experiments, we considered and character n-grams (ASCII characters) as tokens.
For example for the word *Vivekananda*, the 4-gram model would result in 8 tokens namely *Vive, ivek, veka, ekan, kana, anan, nand* and *anda*. If our current token (w_0) is *kana*

Feature	Example
current token: w_0	*kana*
previous 3 tokens: w_{-3},w_{-2},w_{-1}	*ivek,veka,ekan*
next 3 tokens: w_1,w_2,w_3	*anan,nand,anda*
compound feature: w_0 w_1	*kanaanan*
compound feature: w_{-1} w_0	*ekankana*

In Indian Languages suffixes and other inflections get attached to the words increasing the length of the word and reducing the number of occurences of that word in the entire corpus. The character n-grams

can capture these variations. The compound features also help in capturing such variations. The sliding window feature helps in guessing the class of the entity using the context. In total 9 features were used in training and testing. All the features are languge independent and no binary features are used.

6 Experimental Setup

6.1 Corpus

We conducted the experiments on three languages namely Telugu, Hindi and English. We collected the Telugu corpus from Eenadu, a telugu daily newspaper. The topics included politics, health and medicine, sports, education, general issues etc. The annotated corpus had 45714 tokens, out of which 4709 were named entities. We collected the English corpus from the Wall Street Journal (WSJ) news articles. The corpus had 45870 tokens out of which 4287 were named entities. And we collected the hindi corpus from various sources. The topics in the corpus included social sciences, biological sciences, financial articles, religion, etc. The hindi corpus is not a news corpus. The corpus had 45380 tokens out of which 3140 were named entities. We evaluated the hand-annotated corpus once to check for any errors.

6.2 Experiments

We conducted various experiments on Telugu and Hindi. Also, to verify the correctness of our model for other languages, we have conducted some experiments on English data also. In this section we describe the various experiments conducted on the Telugu, Hindi and English data sets.

We show the average performance of the system in terms of precision, recall and F-measure for Telugu, Hindi and English in Table 6 and then for the impact of training data size on performance of the system in Table 7 (Telugu), Table 8 (English) and Table 9 (Hindi). Here, precision measures the number of correct Named Entities (NEs) in the machine tagged file over the total number of NEs in the machine tagged file and the recall measures the number of correct NEs in the machine tagged file over the total number of NEs in the golden standard file while F-measure is the weighted harmonic mean of precision and recall:

$$F = \frac{(\beta^2+1)\ RP}{\beta^2 R + P}$$

with

$$\beta^2 = 1$$

where P is Precision, R is Recall and F is F-measure.

	Precision	Recall	$F_{\beta=1}$
words	89.66%	29.21%	44.07
n=2	77.36%	46.07%	57.75
n=3	85.45%	**52.81%**	**65.28**
n=4	79.63%	48.31%	60.14
n=5	74.47%	39.33%	51.47
n=6	76.32%	32.58%	45.67

Table 2: Precision,Recall and $F_{\beta=1}$ measure for Date & Time expressions in Telugu.

	Precision	Recall	$F_{\beta=1}$
words	83.65%	28.71%	42.75
n=2	80.29%	**36.30%**	**50**
n=3	78.26%	35.64%	48.98
n=4	81.03%	31.02%	44.87
n=5	75.42%	29.37%	42.28
n=6	53.21%	27.39%	36.17

Table 3: Precision,Recall & $F_{\beta=1}$ measure values for location names in Telugu.

	Precision	Recall	$F_{\beta=1}$
words	51.11%	18.70%	27.38
n=2	53.41%	38.21%	44.55
n=3	**69.35%**	**34.96%**	**46.49**
n=4	**69.35%**	**34.96%**	**46.49**
n=5	55.00%	26.83%	36.07
n=6	50.98%	21.14%	29.89

Table 4: Precision,Recall and $F_{\beta=1}$ measure values for organisation names in Telugu.

Table:6 shows the average precison(P),recall(R) and F-measure(F) values for NEs in Telugu.

Tables 2 to 5 show the P,R,F values for the individual categories of NEs in Telugu. Interestingly,

	Precision	Recall	$F_{\beta=1}$
words	57.32%	18.65%	28.14
n=2	55.77%	34.52%	42.65
n=3	**61.04%**	**37.30%**	**46.31**
n=4	56.92%	29.37%	38.74
n=5	60.50%	28.57%	38.81
n=6	54.21%	23.02%	32.31

Table 5: Precision,Recall and $F_{\beta=1}$ measure values for Person names in Telugu.

though we have not used any of the features pertaining to years and numbers we have acheived an appreciable F-measure of 65.28 for date & time expressions.

In each table the model with the highest F-measure is higlighted in bold. And, the tri-gram model performed best in most of the cases except with locations where bi-gram model performed well. But, even the tri-gram model ($F_{\beta=1}$=48.98) performed close to the bi-gram model (($F_{\beta=1}$=50).

For Hindi, the recall of the n-gram models(Table 6) is more than the word based models but the amount of increase in recall and F-measure is less. On examining, we found that the average number of named entities in the Hindi data were quite less. This is because the articles for hindi were taken from general articles. Whereas in case of English and Telugu, the corpus was collected from news articles, which had more probability of having new and more named entities, which can occur in a similar repeating pattern.

The character n-gram approach showed considerable improvement in recall and F-measure (with a drop in precision) in Telugu and Hindi, which are agglutinative in nature. In Telugu, there is a difference of 14.19 and 14.02 in recall and F-measure respectively between the word based model and the best performing n-gram model (n=3) of size 3. In Hindi, there is a difference of 2.34 and 2.33 in recall and F-measure respectively between the word based model and the best performing n-gram model (n=5). Even in case of non-agglutinative language like English there is a considerable improvement of 1.48 and 1.91 in recall and F-measure respectively between the word based model and best performing n-gram model (n=2) of size 2.

In almost all the cases the character based models performed better in terms of recall and F-measure than the word based models.

We also experimented changing the training data size keeping the testing data size unchanged for Telugu(Table 7) and English(Table 8) and Hindi(Table 9). From Table 7:All the models (words,character n-gram models) are able to learn as we increase the training data size. And the recall of the character n-gram models is considerably more than recall of the word based model. Also the 3-gram model performed well in almost all the runs. The rate of learning is more in case of 30K.

From Table 8, in all the runs, the bi-gram character model constantly performed the best. Also interestingly the model is able to achieve a least F-measure of 44.75 with just 10K words of training data. But, in case of Telugu,(Table 7) an F-measure of 44+ was reached with training data of size 35K i.e the learning rate for english is more for less amount of data. This is due to the reason that Telugu (Entropy=15.625 bits per character) (Bharati et al., 1998) is comparitively a high entropy language than English (Brown and Pietra, 1992). However for Hindi, the relative jump in the performance (compared to Telugu and English)is less. Even the entropy of Hindi (Entorpy=11.088) (Bharati et al., 1998) is more than English. This is also observed from the table (Table 10). The numbers in the second, third and fourth columns are the number of features for English,Telugu and Hindi respectively.

	English	Telugu	Hindi
words	29145	320260	685032
n=2	27707	267340	647109
n=3	45580	680720	1403352
n=4	64284	1162320	1830438
n=5	65248	1359980	1735614
n=6	57297	1278790	1433322

Table 10: Number of features calculated in the word based model for English,Telugu and Hindi.

7 Conclusion & Future Work

The character based n-gram approach worked better than the word based approach even with agglutinative languages. A considerably good NER for

Language	English			Telugu			Hindi		
	Precision	Recall	$F_{\beta=1}$	Precision	Recall	$F_{\beta=1}$	Precision	Recall	$F_{\beta=1}$
Words	92.42%	47.29%	62.56	70.38%	23.83%	35.6	51.66%	36.45%	42.74
n=2	81.21%	68.77%	74.47	65.67%	37.11%	47.42	37.30%	36.06%	36.67
n=3	88.37%	62.45%	73.18	71.39%	38.02%	49.62	54.89%	37.23%	44.37
n=4	93.17%	59.19%	72.39	70.17%	33.07%	44.96	54.67%	37.62%	44.57
n=5	90.71%	58.30%	70.98	66.57%	29.82%	41.19	53.78%	38.79%	45.07
n=6	91.03%	56.14%	69.45	55.68%	25.52%	35	51.79%	36.65%	42.92

Table 6: Average Precision, Recall and $F_{\beta=1}$ measure for English, Telugu and Hindi 'n' indicates the number of n-gram characters

Size	10K			20K			30K			35K		
Model	$P_{(\%)}$	$R_{(\%)}$	$F_{\beta=1}$	$P_{(\%)}$	$R_{(\%)}$	$F_{\beta=1}$	$P_{(\%)}$	$R_{(\%)}$	$F_{\beta=1}$	$P_{(\%)}$	$R_{(\%)}$	$F_{\beta=1}$
words	58.04	8.46	14.77	56.54	14.06	22.52	67.90	21.48	32.64	71.03	23.31	35.1
n=2	53.81	13.80	21.97	60.31	**25.52**	**35.86**	63.68	31.51	42.16	65.16	35.55	46
n=3	68.07	**14.71**	**24.2**	64.71	24.35	35.38	70.22	**32.55**	**44.48**	71.79	**37.11**	**48.93**
n=4	71.23	13.54	22.76	63.42	21.22	31.8	68.14	28.12	39.82	68.16	31.77	43.34
n=4	69.92	11.20	19.3	61.20	19.92	30.06	63.90	26.04	37	66.96	29.30	40.76
n=6	52.38	8.59	14.77	52.70	16.54	25.17	56.13	22.66	32.28	55.16	24.35	33.79

Table 7: Effect of training data size on Average Precision,Recall and $F_{\beta=1}$ measure for Telugu.

Size	10K			20K			30K			35K		
Model	$P_{(\%)}$	$R_{(\%)}$	$F_{\beta=1}$	$P_{(\%)}$	$R_{(\%)}$	$F_{\beta=1}$	$P_{(\%)}$	$R_{(\%)}$	$F_{\beta=1}$	$P_{(\%)}$	$R_{(\%)}$	$F_{\beta=1}$
words	81.84	30.79	44.75	86.54	40.93	55.57	89.04	45.95	60.62	89.80	46.35	61.14
n=2	71.49	**42.00**	**52.92**	74.80	**58.40**	**65.59**	75.46	**61.03**	**67.49**	76.63	**61.87**	**68.46**
n=3	76.09	28.85	41.84	81.15	50.03	61.9	81.31	54.28	65.11	82.18	56.84	67.2
n=4	83.42	25.75	39.36	83.35	42.93	56.67	88.01	48.70	62.7	87.40	50.25	63.81
n=5	81.95	25.64	39.06	84.48	41.00	55.21	86.81	44.47	58.81	88.07	47.43	61.66
n=6	79.24	26.89	40.16	83.31	38.18	52.36	89.34	42.88	57.95	87.71	44.32	58.88

Table 8: Effect of training data size on Average Precision,Recall and $F_{\beta=1}$ measure for English.

Size	10K			20K			30K			35K		
Model	$P_{(\%)}$	$R_{(\%)}$	$F_{\beta=1}$	$P_{(\%)}$	$R_{(\%)}$	$F_{\beta=1}$	$P_{(\%)}$	$R_{(\%)}$	$F_{\beta=1}$	$P_{(\%)}$	$R_{(\%)}$	$F_{\beta=1}$
words	43.13	30.60	35.80	47.97	34.50	40.14	48.67	35.67	41.17	51.92	36.84	43.10
n=2	39.29	30.41	34.29	40.73	34.70	37.47	37.58	36.26	36.90	37.91	36.06	36.96
n=3	48.17	33.33	39.40	50.56	35.28	41.56	47.72	36.65	41.46	50.68	36.06	42.14
n=4	49.18	**35.09**	**40.96**	49.21	**36.26**	**41.75**	52.14	35.67	**42.36**	54.87	38.40	45.18
n=5	41.08	34.11	37.27	41.93	33.92	37.50	48.72	37.23	42.21	53.12	**39.77**	**45.48**
n=6	41.43	31.58	35.84	44.59	33.72	38.40	46.35	35.87	40.44	50.67	36.84	42.66

Table 9: Effect of training data size on Average Precision,Recall and $F_{\beta=1}$ measure for Hindi.

English can be built with less amount of data when we use character based models and for high entropy languages large amount of training data is necessary to build a considerably good NER. We are able to achieve an F-measure (49.62 for Telugu and 45.07 for Hindi) even without any extra features like regu-

lar expressions and gazetteer information. The character based n-gram models have worked well even with the discriminative models. A total of 9 features were used in training and testing. We have not used any of the language dependent resources and any binary features. To improve the efficiency of the system we plan to experiment with language specific resources like Part Of Speech (POS) Taggers, Chunkers, Morphological analyzers.. etc and also include some regular expressions and gazetteer information.

References

Bogdan Babych and Anthony Hartley. 2003. Improving machine translation quality with automatic named entity recognition.

Akshar Bharati, Prakash Rao K, Rajeev Sangal, and S.M.Bendre. 1998. Basic statistical analysis of corpus and cross comparison among corpora. Technical report, International Institute of Information Technology-Hyderabad(IIIT-H).

D. Bikel, S. Miller, R. Schwartz, and R. Weischedel. 1997. Nymble: a high-performance learning namefinder.

Peter E Brown and Vincent J. Della Pietra. 1992. An estimate of an upper bound for the entropy of english.

Nancy Chinchor. 1997. Muc-7 named entity task definition. Technical Report Version 3.5, Science Applications International Corporation, Fairfax, Virginia.

S. Cucerzan and D. Yarowsky. 1999. Language independent named entity recognition combining morphological and contextual evidence.

D. Klein, J. Smarr, H. Nguyen, and C. Manning. 2003. Named entity recognition with character-level models.

John Lafferty, Andrew McCallum, and Fernando Pereira. 2001. Conditional random fields: Probabilistic models for segmenting and labeling sequence data. In *Proc. 18th International Conf. on Machine Learning*, pages 282–289. Morgan Kaufmann, San Francisco, CA.

Andrei Mikheev. 1997. Automatic rule induction for unknown-word guessing. *Comput. Linguist.*, 23(3):405–423.

Diego Molla, Menno van Zaanen, and Daniel Smith. 2006. Named entity recognition for question answering. In *Proceedings of Australasian Language Technology Workshop 2006*, Sydney, Australia.

Lance Ramshaw and Mitch Marcus. 1995. Text chunking using transformation-based learning. In David Yarovsky and Kenneth Church, editors, *Proceedings of the Third Workshop on Very Large Corpora*, pages 82–94, Somerset, New Jersey. Association for Computational Linguistics.

Antonio Toral, Elisa Noguera, Fernando Llopis, and Rafael Muñoz. 2005. Improving question answering using named entity recognition. In *Proceedings of the 10th NLDB congress*, Lecture notes in Computer Science, Alicante, Spain. Springer-Verlag.

Richard Tzong-Han Tsai. 2006. A hybrid approach to biomedical named entity recognition and semantic role labeling. In *Proceedings of the 2006 Conference of the North American Chapter of the Association for Computational Linguistics on Human Language Technology*, pages 243–246, Morristown, NJ, USA. Association for Computational Linguistics.

N. Wacholder, Y. Ravin, and M. Choi. 1997. Disambiguation of proper names in text.

Hanna M. Wallach. 2004. Conditional random fields: An introduction. Technical Report MS-CIS-04-21, University of Pennsylvania, Department of Computer and Information Science, University of Pennsylvania.

An Experiment on Automatic Detection of Named Entities in Bangla

Bidyut Baran Chaudhuri
Head- CVPR Unit
Indian Statistical Institute
203,B.T. Road, Kolkata-108
bbc@isical.ac.in

Suvankar Bhattacharya
Systems Executive
ABP Pvt. Ltd.
6, P.S. Street, Kolkata-1
suvankar.bhattacharya@abp.in

Abstract

Several preprocessing steps are necessary in various problems of automatic Natural Language Processing. One major step is named-entity detection, which is relatively simple in English, because such entities start with an uppercase character. For Indian scripts like Bangla, no such indicator exists and the problem of identification is more complex, especially for human names, which may be common nouns and adjectives as well. In this paper we have proposed a three-stage approach of named-entity detection. The stages are based on the use of Named-Entity (NE) dictionary, rules for named-entity and left-right co-occurrence statistics. Experimental results obtained on Anandabazar Patrika (Most popular Bangla newspaper) corpus are quite encouraging.

1 Introduction

The discipline of Natural Language Processing (NLP) is concerned with the design and implementation of computational approaches that communicate with human using natural language. Name searching, matching and recognition have been active areas of research in the field of NLP and Information retrieval for a long period. This is an important problem since search queries are often proper nouns while all proper nouns cannot be exhaustively maintained in the dictionary for automatic identification. Moreover, human names may be picked from common nouns and adjective words (e.g. Surya, Anindya) and hence dictionary-based syntactic information can confuse the Natural Language Processor in such a situation. Pet and other animal names, organization and place names, can also come from common nouns and adjectives e.g. Shyamali (cow name), Bardhaman (Place name), Bhalobasha (Building name), Nandan (Auditorium name) etc. So, it becomes a non-trivial problem to automatically detect the named entity from a sentence.

This paper aims at attacking this problem for Bangla language, especially on the NE detection from newspaper text. Name recognition in English is somewhat easier since quite often the proper noun starts with an uppercase character. Bangla names cannot be identified by such case information because Bangla has single-case alphabet.

Some studies on Named Entity (NE) identification are reported in the literatures (from Zhou and 2007 to Narayanswamy et al., Narayanswamy as listed in the reference section of this paper). The approaches mainly employ dictionary based,rule based and statistical tools such as HMM, Maximum entropy, Support vector machine and conditional random field for this purpose. Name searching in the context of information retrieval and query answering are also reported in the literature (Thompson and Dozier, 2007). However, these studies are done on non-Indian languages. Among Indian languages typical efforts based on HMM and CRF are presented by EKbal et al. (2007) and Li and McCallum (2003) respectively.

The NE identification approach presented here employs a three tier combination of dictionary-based, rule-based and statistical information. The approach employed here is explained in Section 2 where use of the hybrid approach is also justified. In Section 3, the data collection and experimental

Proceedings of the IJCNLP-08 Workshop on NER for South and South East Asian Languages, pages 75–82,
Hyderabad, India, January 2008.

setup is described. Tests have been made on a moderate size Anandabazar (most popular Bangla newspaper) news corpus. The results are presented in Section 4.

2 Proposed Named Entity (NE) detection approach

As mentioned before, our method of NE detection is a combination of dictionary-based, rule-based and statistical (n-gram based) approaches. In the dictionary based approach, we need a word-level morphological parser as well. The approaches are sequentially described here and demonstrated in Fig.1. However, at first, we describe some properties of named entity.

2.1 Properties of named entity

If we look at a corpus of reasonable size from the perspective of NEs, we note that the words may belong to three categories: (a) words that almost never act as NE, (b) the words that almost always act as NE, (c) the words that sometimes act as names and sometimes as common nouns or adjectives. Words like *I, the, to, from, go* belong to category (a) while words like *India, Ganges, Paris, Himalayas* belong to category (b). Words like *Nirmal, Swapan, Rabi* belong to category (c). The English meanings of these third category words are clean, dream and sun, respectively, but they are used as names of persons in Bangla and thus can create problems for the NLP of Bangla language. In English, the names begin with uppercase, and are less problematic in nature.

Another point to note is that the named entity may be a single word or a multi word expression. The multi-word names pose additional difficulty for automatic identification of NE. A multi-word may have a component that alone is also a name, like *England* in *New England* or it may consist of adjective and common noun, like *White House*. Such multi-words generate additional problems for NE detection.

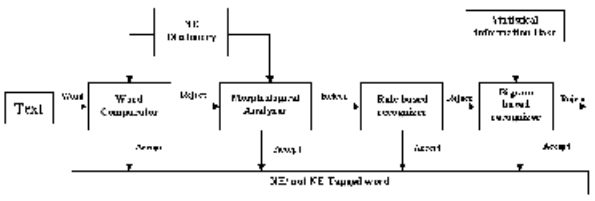

Fig 1. Flow chart for NE detection

2.2 Justification of hybrid approach

In the NE detection tasks, the entries that are considered are person, organization, location, date, time, money, percentage. In case of English, there are indicators like uppercase character, dot mark, Dollar and Pound symbol etc. to identify them. In addition, rule-base or machine learning approaches are employed and hence an impressive result is obtained.

In Bangla, date, time, money, percentage also use special symbols in some occasions, but for person, organization or location name this is not true. Moreover, nouns and adjectives are very frequently used for single-word or multi-word names of above types. Now, a dictionary or some special kind of word data-base is used in most NLP problems. If we equip the same dictionary or data-base which have information about NE, then every word of a text need not pass through more sophisticated NE detection software. We have noted that even for NE-rich text like news, the percentage of such words does not exceed 7% (See Table-1). The dictionary helps us to detect 65% of Nes and discard more than 90% of the non-NE words. For that purpose, we have to tag the dictionary in the manner described in Section 2.3. We can be left with about 1.4% words, which may be ambiguous (can be or cannot be NE) and about 1.15% words, which are not there in the dictionary (hence nothing can be said using dictionary).

Newspaper	Total Word	Total NE	Person Name	Place Name	Other Names
Anandabazar	42104	6.53% [2753]	2.50% [1023]	2.30% [972]	1.80% [758]
Ajkaal	39452	6.93% [2734]	2.89% [1143]	1.92% [755]	2.11% [836]
Bartamaan	40323	6.50% [2621]	3.43% [1383]	1.60% [645]	1.47% [593]

Table 1. NEs from different Bangla Newspapers

So, we can use the rule base at the second stage. Compared to statistical learning methods, rule-based system has both power and limitation. Consider a robust simulation where each person name and place name of West Bengal, Tripura and Bangladesh (the Bangla-language-using places) can appear. Note that there are about 240 million Bengali names and a few tens of thousands of place names. Of course, not all are distinct names, but the distinct names are also huge in number. To explain it better, let there be 1000 distinct first names, 50

distinct middle names and 500 distinct last names (title) of persons. Then the total number of distinct human names that can be created is 1000 X 50 X 500 = 25 million. If the full names appear in the test, then they could be very easily tackled by a rule and a database of middle names and titles. On the other hand, any statistical technique is based on probability, and estimation of probability needs a reasonable corpus size that is costly and may not be available for design. Even if the corpus is available, the statistical approach will perhaps discover the same rule along with the same database in a different manner. Moreover, extension for a few more names can be quickly accommodated in the database or another rule, but the statistical approach will need re-training, resulting in a new set of absolute and conditional probabilities.

On the other hand, rule-based system cannot tackle ambiguous situations very well. So, when it is the question of a noun or adjective word being used as NE or not NE, good rules cannot be formulated for every situation. Rule-based system is also useless for a word not falling under any of the rules generated so far. In such a situation the statistical learning technique may be very useful.

In this way, we believe that the combination of three approaches will help us in detecting NE in a robust way. Moreover, we believe that it will be easily adapted to changed environment of the test set.

2.3 Dictionary based NE detection

If a dictionary is maintained where one of the above three category tags are attached to each word and if a word morphological analyzer is developed, then the combination of these two can act as a NE detector for a text file. The dictionary should be generated from a corpus of reasonable size, say 5-10 million words, as well as from conventional dictionary book of say 50,000 root words. Normally, 10 million word corpus of Bangla contains between 100,000 and 200,000 surface words. A small fraction of these words belong to the set of NEs not found in the conventional dictionary. These surface words should be properly NE tagged as per three types described above and entered in the NE dictionary. The corpus provides important information about the inflectional nature of root words, which, in turn, helps in building the morphological analyzer. On the other hand, if we want to avoid building sophisticated morph analyzer, the most common inflected surface words of the corpus may also be included in the dictionary with the three tags described above. We have followed this procedure for our NE detection approach.

The detection algorithm will proceed as follows. Given a test word W, at first, a match is searched in the NE tagged dictionary. If no match is found, W is rejected and the next word is considered for examination. But if a match occurs, we look at the tag of the matched word. If the tag is '*almost always NE*' then we declare this W as NE with weight 1. If the tag is '*almost never NE*' then W is declared as not NE (ie with weight 0). But if the tag is '*may or may not be NE*' then again W has to be rejected (say with weight 0.5), which makes this approach uncertain for such word. To remedy this drawback, we next employ some rule-based approach described in the next Section.

However, before sending to the rule-based module, each of the words with weight 0.5 is subject to morphological analysis. Here for each word, the suffix is stripped using a previously stored suffix database. If no database suffix matches, then the whole word is sent to rule based method. Else, the suffix-stripped word is again matched in the NE dictionary. If a match is found, then it is checked if the suffix can be morphologically accepted by the dictionary root word category. Then W is properly tagged with weight 1 or 0. Else, it is sent to the module for rule-based approach described below with the hope for better decision.

2.4 Rule-based NE detection

Rule-based approaches rely on some rules, one or more of which is to be satisfied by the test word W. There may be *positive* and *negative* rules. The positive rules make the inference system biased towards NE while the negative rules tend to be biased against NE. Some small databases may be needed to execute the rules. For Bangla text NEs, some typical rules are given below. Here, 1-8 are positive and 9-12 are negative rules.

Rule1. If there are two or more words in a sequence that represent the characters or spell like the characters of Bangla or English, then they belong to the named entity (with high weight). For example, বি এ (B A), সি এম ডি এ (C M D A), অ গ প are all NEs. Note that the rule will not distinguish between a proper name and common name.

Rule 2. If the previous word of W is a pre-name word like শ্রী, শ্রীযুক্ত, ডঃ, মিঃ, মিস, মিসেস, বেগম, বিবি, মোঃ, মুঃ, ডক্টর, স্বামী, সৈয়দ, রেভারেন্ড, then W belongs to the named entity (with high weight). To detect them, all words of this type can be maintained in a database.

Rule 3. If after W there are title words and mid-name words to human names like ঘোষ, বোস, বসু, মিত্র, রায়, সরকার, খান, আহমেদ, রহমান, হক etc. and কুমার, চন্দ্র, রঞ্জন, শেখর, প্রসাদ, আলী, আলম etc., respectively, then W along with such words are likely to constitute a multi-word NE (with high weight).. For example, রবি বসাক, পল্লব কুমার মল্লিক are all NEs. A set of title and mid-name words should be collected and maintained in a database.

Rule 4. If a substring like –বাবু, -দাদা, -দা, -সাহেব, -কাকু, -গঞ্জ, -গ্রাম, -পুর, -গড়, -নগর occurs at the end of the word W, then W is likely to be a NE (with high weight). These strings can be collected in a database for future use.

Rule 5. If at the end of a word W there are strings like –এ-কার, -এর, -র, -রা, -এরা, -কে, -দের, -তে, -য় then W is likely to be a name (with high weight).

Rule 6. If a word like সরণী, রোড, স্ট্রিট, লেন, থানা, স্কুল, বিদ্যালয়, কলেজ, নদী, লেক, হ্রদ, সাগর, মহাসাগর, পাহাড়, পর্বত is found after W of type unknown in dictionary then W along with such word may belong to NE (with high weight). For example, বিধান সরণী, রাসেল স্ট্রীট, ডালহৌসি পাহাড় are all NEs.

Rule 7. We note that only a few names or words in Bangla consist of characters ঁ (Chandrabindu) or ৎ (Khanda Ta)**.** So, if W does not belong to those words and has the occurrence of any of these two characters, then W may be a named entity (with high weight). For example, ‘‘অঁরি’’ is a French name.

Rule 8. If in the sentence containing unknown word W or a word W with *may or may not be NE* tag, the following words are বলেন, বললেন, বলল, শুনল, হাসল, লিখল, লিখলেন, খেলেন, দেখল which imply action that can be done by human being, then W is likely to be a name (with high weight). A database of action verbs of various types is needed to check this rule.

Rule 9. If W of the type given in rule 8 is followed by verb not in the set of verbs described above, then W is not likely to be a NE. So, the weight should be reduced from 0.5 to a smaller value.

Rule 10. If there is re-duplication of W in a sentence then W is not likely to be a named entity. This is so because rarely name words are reduplicated. In fact, reduplicated name word may signify something else. For example রাম রাম is used to greet a person. So, the NE weight should be reduced in a such case to near zero.

Rule 11. If at the end of W there are suffixes like –টা, -খানা, -খানি, -টাতে, -টায়, -টিকে, -টাকে, -টকুন, -গুলা, -গুলো, -গুলি etc., then W is usually not a named entity.

Rule 12. If there is an echo-word after W e.g. গাছ টাছ, then none of these two words is a named entity. The exact value of the weight for a rule is decided from training dataset. We increase or decrease the weight of the test word if a rule fires. To be consistent, we have included the dictionary-based approach under the same framework.

Thus, in our scheme, if the weight is more than certain value (say 0.75) then the word is finally accepted to be NE. On the other hand, if the weight is less than certain value (say 0.25) then the word is rejected to be NE. For intermediate cases, the word may be subject to the n-gram based technique described below.

2.5 n-gram based NE detection

The n-gram based approach relies on the co-occurrence of other words before and after a NE. To generate the n-gram we need a corpus where the NE words are tagged manually. From these tagged words the left neighbor and right neighbor words are checked (for a 2-gram model). The frequencies of each pair of left-right neighbor are counted from the corpus. The probability of each left-right pair with respect to W may be estimated as

P_{lr}(W) = No of this left-right word pair around W/ total no of all left-right words around W in the training corpus.

If a particular left-right neighbors occur about a word W, then W has a *positive likelihood* of being NE, or a *negative likelihood* that W is not a NE. For example, in the sentence ‘Mark the answer script properly’ the word ‘Mark’ is a negative instance for NE. But in the sentence ‘Mark is a good boy’, ‘Mark’ is a positive instance. Here the left-right pair is ‘blank’ and ‘is’. We have to count from the test corpus how many times the particular left-right neighbor give positive instances of W being a NE, while how many are the instances of

non-NE. From these positive and negative instance counts, a NE weight value is found for a particular pair of left-right word pair around W as

$$w_{lr}(W) = P_{lr}(W)\, R_{lr}(W)$$

where $R_{lr}(W)$ = No of positive instances /(No of positive instances + No of negative instances).

However, a large number of words will be negative instances at all times, so their $w_{lr}(W)$ value will come out as zero. Examples are the so-called *stop* words. They can be dealt in the dictionary itself, as discussed in Sec 2.2, reducing a lot of computational effort for this n-gram based approach. Some words which will also be positive instance, irrespective of the left right words. The NE dictionary described in Section 2 can deal them as well. This fact partly justifies the scheme of having three approaches combined in our NE detection algorithm.

Thus, the generation of training phase is completed. Now, in the test phase, if a word W has left-right neighbors whose weight is $w_{lr}(W)$ based on the training phase, then W may be assigned this weight of being named entity. This is the modified weight over and above what was given in the previous phases. For the test phase, a threshold t is set on the weight. If the weight for the test word W is $w > t$ then we declare W as a NE. Otherwise, we call it not-NE.

There may be left-right pair for a test word that is absent in our probability list. If none of the pair exist then the word is rejected since no decision can be made. If only left or right word is present then we take a pessimistic estimate based on it. In other words, we take the minimum of probabilities individually this W and the said left word.

3 Data collection

To obtain the corpus for our experiment, we browsed the net and found the site of Anandabazar Patrika, the largest Bangla daily newspaper. We downloaded the e-newspapers for the years 2001-2004. Of this huge data, a portion for the years 2001-2003 were used for training the system (about 20 million words) and a portion from 2004 (about 100, 000 words) was used for testing. The data could not be utilized in a straightforward way, since the newspaper authority used a proprietary glyph code. So, we had to discover which glyph code denotes which character of Bangla script and then convert the text into ISCII coding format. After that, all the developed softwares were run on these ISCII files. At first a program was written was used to collect all distinct surface words from this corpus of 20 million words. These distinct words were ranked in descending order of frequency and the top 20,000 ranked words were chosen for manual tagging of named entity by giving weight 0, 0.5 or 1.0.

The manual tagging was done by the linguists based on their global knowledge. However, if the person is in doubt, (s)he would consult a few examples in the original corpus involving the word in question. Using the contextual information, most problematic cases could be disambiguated. Those which still appeared unclear were given 'may or may not be' status. A morphological analyzer was previously developed in connection with the design of a spell checker in Bangla (Chaudhuri, 2001). That analyzer has been employed for stemming of the type-words in the current NE detection problem also. Moreover, a rule-based system as described in Section 2.3 is also developed. The database needed for each rule is being continuously updated to give better experimental results.

Experimental results:

The software was trained with the Anandabazar Patrika web corpus of the year 2001-2003. Some geographical names were further added to enrich the database. Then several files of the corpus of the same newspaper of the year 2004 were used for testing. The results are presented in the form of recall(R), precision (P) and F-measure percentage. Here the recall is the ratio of number of NE words retrieved and the number of NE words actually present in the file, expressed in percent. In other words,

$$R\% = \frac{Number\ of\ NE\ words\ retrived}{Total\ Number\ of\ NE\ words\ in\ the\ text} X\,100\%$$

Precision is the number of correctly retrieved NE words to the total number of words retrieved, expressed in percent. So, we can write

$$P\% = \frac{Number\ of\ correct\ NE\ words\ retrieved}{Total\ Number\ of\ NE\ words\ retrieved} X\,100\%$$

The F-measure is often used in the Information Retrieval and Natural Language Processing problems. This class of measures was introduced by C.

J. van Rijsbergen. F_1- measure is the ratio of the twice of the multiplication of precision (P) and recall (R) and the sum of these two. In other words,

$$F1\% = \frac{2P\%R\%}{P\% + R\%} X 100\%$$

F_1 measure combines recall (R) and precision (P) with an equal weight and hence is the harmonic mean of the two quantities. Note that F_1 cannot exceed 100%. Experimental results on 10 sets of test documents are shown in Table 2.

NO. OF WORDS	NO. OF NE	CORRECTLY DETECTED	NO. OF ERROR	RECALL %	PRECISION %	F_1-MEASURE %
2592	165	138	7	79.39	95.00	86.00
2938	186	157	6	81.10	96.20	88.00
2477	247	176	6	76.25	97.60	85.00
3816	336	268	7	79.76	97.40	87.00
2944	192	144	5	75.00	96.52	84.41
4843	255	210	13	82.35	93.50	87.85
2899	202	192	7	95.04	96.35	95.44
3420	232	201	9	86.63	95.52	90.85
4428	243	209	11	86.00	94.73	90.15
4228	210	177	16	84.28	90.96	87.42
4528	292	261	11	89.38	95.78	92.46
2991	193	168	5	87.04	97.02	91.75
AVERAGE				**85.50**	**94.24**	**89.51**

Table 2. Results of the experiment

It is noted from Table 1 that the precision is reasonably high but the recall is somewhat moderate. The reason of moderate occurrence of recall is that the training has been done with only 20,000 corpus words, while actual number of corpus words was about 200,000. Also, we have to improve the database for rules, as well as search for other potential rules that we have not included here. The front back 2-grams are also at present aggregated over all NE words tagged manually. Such global occurrence statistics can mask the local phenomenon. We are working towards improving our NE detection approach.

Every detection system is to be judged by some automatic evaluation techniques, e.g. BLEU (Bilingual Evaluation Understudy) (Papineni, 2002) and several others. So, in case of ours we introduced an Automatic Evaluation approach for the main detection algorithm. The evaluation system is actually based upon a manually annotated dataset of almost 70,000 words. These datasets are tagged in a "non-NE <NE *Name* NE> non-NE" format and are available at Chaudhuri (2007). After the system detects and tags the names, the detection system treats the NE-detected file location as the *"Target Location"*. In our annotated dataset the annotated corpus is available for the same documents. That location is treated as the "*Annotated Location"*. As the evaluation system starts evaluating, a word by word comparison is done between the target and annotated locations. At the end of evaluation number of correctly detected words, the number of wrong detection and the number of real NE is found and so the Precision, Recall and F1-Measure is calculated easily. We have also observed that our evaluation system gives almost the same result as found by manual evaluation.

References

G. Zhou and J. Su 2002. *Named Entity recognitionusing HMM Based chunk tagger*, Proc. 40-th Annual Meeting of ACL,Philadelphia,pp. 473-480.

A. Borthwick,1999. *A maximum Entropy approach to Named Entity recognition*, Computer Sc. Dept. New York University.

J. R. Finkel, T. Grnagar & C. Maning, 2005. *Incorporating non-local information into information extraction systems by Gibbs sampling*, Proc. 43-rd Annual meeting of ACL, pp. 363-370.

U. Pfeiffer; T. Poersch, and N. Fuhr. 1996. *Retrieval effectiveness of proper name search methods*, Information Processing & Management, Vol. 32, pp. 667-679.

K. Takeuchi and N. Collier, 2002, *Use of support vector machines in extended Named Entity recognition*, Proc. 6th Conference Natural Language Learning, Taipei, pp. 119-125.

D. Marynard, V.Tablan, K. Cunningham and Y. Wilks. 2003. *Muse : a multisource entity recognition system.* Computers and the Humanities. Website Reference: http://gate.ac.uk/sale/muse/muse.pdf

D. Maynard, K. Bonteheva, H. Cunningham, 2003. *Towards a semantic extraction of named entity,*. Website reference: http://eprints.akfors.org /267/01/maynard.pdf.

P. Thompson and C. C. Dozier, on *Name Searching and Information Retrieval.* Website reference: http://arxiv.org/html/cmplg/9706017.

H. Cunningham. 2002. *Gate, a general architecture for text engineering.* Computers and the Humanities, Vol. 36, pp. 223- 254.

M. Narayanswamy, K.E. Ravikumar, and V.K. Shanker. 2003. *A biological named entity recognizer.* Proceed-

ings of the Pacific Symposium on Biocomputing, Hawaii.

A. EKbal ,S. Naskar and S. Bandopadhyay, 2007, *Named Recognition and Transliteration in Bengali*, Special Issue of Lingvisticae Investigationes Journal,30:1 (2007), pp. 95-114, John Benjamins Publishing Company.

S. Cucerzon and D. Yarowsky. 1999. *Language independent named entity recognition combining morphological and contextual evidence.* Proceedings of the 1999 Joint SIGDAT conference on EMNLP and VLC.

B.B. Chaudhuri. 2001. *A Novel Spell-checker for Bangla Text Based on Reversed-Word Dictionary.* Vivek Vol. 14 No. 4,pp. 3-12.

W. Li, A. McCallum, 2003. *Rapid development of Hindi Named Entity recognition using Conditional Random Fields and Feature Extraction.* ACM Trans on Asian Language Information Processing, Vol 2, No. 3, pp. 290-293.

D. Okanohara, Y. Miyao, Y. Tsuruoka and J. Tsujii. 2006. *Improving the Scalability of Semi-Markov Conditional Random Fields for Named Entity Recognition.* Proceedings of the COLING-ACL, Sydney, Australia, 17-21 July, pp. 465-472.

Papineni, K., Roukos**,** S., Ward, T., and Zhu, W. J. 2002. *BLEU: a method for automatic evaluation of machine translation* in ACL-2002: 40th Annual meeting of the Association for Computational Linguistics pp. 311--318

Annotated Bengali Corpus by Prof. B. B. Chaudhuri, ISI, Calcutta and Suvankar Bhattacharya. Website Reference: http://cybersuv.googlepages.com /Annotated.zip

Rapid Development of Hindi Named Entity Recognition using Conditional Random Fields and Feature Extraction by Wei Li, University of Massachusetts Amherst and Andrew McCallum, University of Massachusetts Amherst.

A Hybrid Named Entity Recognition System for South Asian Languages

Praveen Kumar P
Language Technologies Research Centre
International Institute of Information Technology - Hyderabad
praveen_p@students.iiit.ac.in

Ravi Kiran V
Language Technologies Research Centre
International Institute of Information Technology - Hyderabad
ravikiranv@students.iiit.ac.in

Abstract

This paper is submitted for the contest NERSSEAL-2008. Building a statistical based Named entity Recognition (NER) system requires huge data set. A rule based system needs linguistic analysis to formulate rules. Enriching the language specific rules can give better results than the statistical methods of named entity recognition. A Hybrid model proved to be better in identifying Named Entities (NE) in Indian Language where the task of identifying named entities is far more complicated compared to English because of variation in the lexical and grammatical features of Indian languages.

1 Introduction

Named Entities (NE) are phrases that contain person, organization, location, number, time, measure etc. Named Entity Recognition is the task of identifying and classifying the Named Entities into predefine categories such as person, organization, location, etc in the text.

NER has several applications. Some of them are Machine Translation (MT), Question-Answering System, Information Retrieval (IR), and Cross-lingual Information Retrieval.

The tag set used in the NER-SSEA contest has12 categories. This is 4 more than the CONLL-2003 shared task on NER tag-set. The use of finer tag-set aims at improving Machine Translation (MT). Annotated data for Hindi, Bengali, Oriya, Telugu and Urdu languages was provided to the contestants.

Significant work in the field of NER was done in English, European languages but not in Indian languages. There are many rule-based, HMM based; Conditional Random Fields (CRF) based NER systems. MEMM were used to identify the NE in Hindi (Kumar and Bhattacharyya, 2006). Many techniques were used in CoNLL-2002 shared task on NER which aimed at developing a language independent NER system.

2 Issues: Indian Languages

The task of NER in Indian Languages is a difficult task when compared to English. Some features that make the task difficult are

2.1 No Capitalization

Capitalization is an important feature used by the English NER systems to identify the NE. The absence of the lexical features such as capitalization in Indian languages scripts makes it difficult to identify the NE.

2.2 Agglutinative nature

Some of the Indian language such as Telugu is agglutinative in nature. Telugu allows polyagglutination, the unique feature to being able to add multiple suffixes to words to denote more complex words.
Ex: “hyderabadlonunci” = hyderabad+ lo + nunchi

2.3 Ambiguities

There can be ambiguity among the names of persons, locations and organizations such as *Washington* can be either a person name as well as location name.

2.4 Proper-noun & common noun Ambiguity

In India the common-nouns often occur as the person names. For instance *Akash* which can mean ‘*sky*’ is also name of a person.

Proceedings of the IJCNLP-08 Workshop on NER for South and South East Asian Languages, pages 83–88,
Hyderabad, India, January 2008.

2.5 Free-word order

Some of the Indian languages such as Telugu are free word order languages. The heuristics such as position of the word in the sentence can not be used as a feature to identify NE in these languages.

3 Approaches

A NER system can be either a Rule based or statistical or hybrid. A Rule-based system needs linguistic analysis to formulate the rules. A statistical NER system needs annotated corpus. A hybrid system is generally a rule based system on top of statistical system.

For the NER-SSEAL contest we developed CRF based and HMM based hybrid system.

3.1 Hidden Markov Model

We used a second order Markov model for Named entity tagging. The tags are represented by the states, words by the output. Transition probabilities depend on the states. Output probabilities depend on the most recent category. For a given sentence $w_1 \ldots w_T$ of length T. $t_1, t_2 .. t_T$ are elements of the tag-set. We calculate

$$\text{Argmax}_{t1 \ldots tT} [\prod_1^T P(t_i|t_{i-1},t_{i-2})P(w_i|t_i)](P(t_{T+1}|t_T)$$

This gives the tags for the words. We use linear interpolation of unigrams, bigrams and trigrams for transition probability smoothing and suffix trees for emission probability smoothing.

3.1.1 HMM based hybrid model

In the first phase HMM models are trained on the training corpus and are used to tag the test data.
The first layer is purely statistical method of solving and the second layer is pure rule based method of solving. In order to extend the tool for any other Indian language we need to formulate rules in the second layer. In the first layers HMM models are training from the annotated training corpus. The annotation follows as: Every word in the corpus if belongs to any Named entity class is marked with the corresponding class name. And the one's which don't fall into any of the named entity class fall into the class of words that are not named entities. The models obtained by training the annotated training corpus are used to tag the test data. In the first layer the class boundaries may not be identified correctly. This problem of correctly identifying the class boundaries and nesting is solved in the second layer.

In the second layer, the chunk information of the test corpus is used to identify the correct boundaries of the named entities identified from the first layer. It's a type of validation of result from the first layer. Simultaneously, few rules for every class of named entities are used in order to identify nesting of named entities in the chunks and to identify the unidentified named entities from the first layer output. For Telugu these rules include suffixes with which Named Entities can be identified

3.2 Conditional Random Fields

Conditional Random Fields (CRFs) are undirected graphical models, a special case of which corresponds to conditionally-trained finite state machines. CRFs are used for labeling sequential data.

In the special case in which the output nodes of the graphical model are linked by edges in a linear chain, CRFs make a first-order Markov independence assumption, and thus can be understood as conditionally-trained finite state machines (FSMs).

Let $o = (o, o_2, o_3, o_4, \ldots o_T)$ be some observed input data sequence, such as a sequence of words in text in a document,(the values on n input nodes of the graphical model). Let S be a set of FSM states, each of which is associated with a label, l ? £.Let $s = (s_1, s_2, s_3, s_4, \ldots s_T)$ be some sequence of states, (the values on T output nodes). By the Hammersley- Clifford theorem, CRFs define the conditional probability of a state sequence given an input sequence to be:

$$P(s|o) = \frac{1}{Z_o} * exp(\sum_{t=1}^{T} \sum_{k} \lambda_k f_k(s_{t-1}, s_t, o, t))$$

where Z_o is a normalization factor over all state sequences is an arbitrary feature function over its arguments, and $?_k$ is a learned weight for each feature function. A feature function may, for example, be defined to have value 0 or 1. Higher ? weights make their corresponding FSM transitions more likely. CRFs define the conditional probability of a label sequence based on the total probability over the state sequences,

$$P(l|o) = \sum_{s:l(s)=l} P(s|o)$$

where l***(s)*** is the sequence of labels corresponding to the labels of the states in sequence ***s***.

$$Z_o = \sum_{s \in S^T} \exp\left(\sum_{t=1}^{T} \sum_{k} \lambda_k f_k(s_{t-1}, s_t, \mathbf{o}, t)\right),$$

Note that the normalization factor, Z_o, (also known in statistical physics as the partition function) is the sum of the scores of all possible states.

And that the number of state sequences is exponential in the input sequence length T. In arbitrarily structured CRF's calculating the normalization factor in closed form is intractable, but in liner-chain-structure CRFs, the probability that a particular transition was taken between two CRF states at a particular position in the input can be calculated by dynamic programming.

3.2.1 CRF based model

CRF models were used to perform the initial tagging. The features for the Hindi and Telugu models include the Root, number and gender of the word from the morphological analyzer. From our previous experiments it is observed that the system performs better with the suffix and the prefix as features. So the first 4, first 3, first 2 and the 1st letter of the word (prefix) and the last 4, 3, 2, 1 letters of the word (suffix) are used as features.

The word is a Named Entity depends on the POS tag. So the POS tag is used as a feature. The chunk information is important to identify the Named entities with more than one word. So the chunk information is also included in the feature list.

The resources for the rest of the three languages (Oriya, Urdu and Bengali) are limited. Since we couldn't find the morphological analyzer for these languages, the first 4,3,2,1 letters and the last 4,3,2,1 letters are used as features.

The word being classified as a named entity also depends on the previous and next words. So these are used as features for all the languages

4 Evaluation

Precision, Recall and F-measure are used as metric to evaluate the system. These are calculated for Nested (both nested and largest possible NE match), Maximal (largest possible NE match) and Lexicon matches

Nested matches (n): The largest possible as well as the nested NE

Maximal matches (m): The largest possible NE matched with reference data.

Lexical item (l): The lexical item inside the NE are matched

5 Results

P_m, P_n ,P_l are the precision of maximal, nested, lexical matches respectively. R_m, R_n, R_l are the recall of maximal, nested, lexical matches respectively. Similarly F_m, F_n, F_l are the F-measure of maximal, nested, lexical matches.

The precision, recall, F-measure of five languages for CRF system is given in Table1. Table 2 has the lexical F-measure for each category. Similarly Table3 and Table4 give the precision, recall and F-measure for the five languages and the lexical F-measure for each category of HMM based system.

The performance of the NER system for five languages using a CRF based system is shown in Table-1.

	Precision			Recall			F-Measure		
Language	Pm	Pn	Pl	Rm	Rn	Rl	Fm	Fn	Fl
Bengali	61.28	61.45	66.36	21.18	20.54	24.43	31.48	30.79	***35.71***
Hindi	69.45	72.53	73.30	30.38	29.12	27.97	***42.27***	***41.56***	***40.49***
Oriya	37.27	38.65	64.20	19.56	16.19	25.75	25.66	22.82	***36.76***
Telugu	33.50	36.18	61.98	15.90	11.13	36.10	21.56	17.02	***45.62***
Urdu	45.55	46.11	52.35	26.08	24.24	30.13	33.17	31.78	***38.25***
m: Maximal n: Nested l: lexical									

Table 1: Performance of NER system for five languages (CRF)

	Bengali	Hindi	Oriya	Telugu	Urdu
NEP	33.06	***42.31***	***51.50***	15.70	11.72
NED	00.00	***42.85***	01.32	00.00	04.76
NEO	11.94	34.83	12.52	02.94	20.92
NEA	00.00	36.36	00.00	00.00	00.00
NEB	NP	NP	00.00	00.00	00.00
NETP	29.62	00.00	18.03	00.00	00.00
NETO	28.96	08.13	03.33	00.00	00.00
NEL	34.41	***61.08***	***46.73***	12.26	***54.59***
NETI	***63.86***	***70.37***	35.22	***90.49***	***62.22***
NEN	***75.34***	***74.07***	21.03	26.32	13.44
NEM	***46.96***	***58.33***	14.19	***42.01***	***77.72***
NETE	12.54	13.85	NP	08.63	00.00
NP: Not present in reference data					

Table 2: Class specific F-Measure for nested lexical match (CRF)

Measure	Precision			Recall			F-Measure		
Language	Pm	Pn	Pl	Rm	Rn	Rl	Fm	Fn	Fl
Bengali	50.66	50.78	58.00	25.03	24.26	30.26	33.50	32.83	***39.77***
Hindi	69.89	73.37	73.59	36.90	35.75	34.34	***48.30***	***47.16***	***46.84***
Oriya	33.10	34.70	60.98	24.63	20.61	36.72	28.24	25.86	***45.84***
Telugu	15.61	49.67	62.00	11.64	24.00	37.30	13.33	32.37	***46.58***
Urdu	42.81	47.14	56.21	29.37	29.69	37.15	34.48	36.83	***44.73***
m: Maximal n: Nested l: lexical									

Table 3: Performance of NER system for five languages (HMM)

	Bengali	Hindi	Oriya	Telugu	Urdu
NEP	***38.10***	***53.19***	***63.04***	23.14	34.96
NED	00.00	***52.94***	08.75	06.18	***49.18***
NEO	05.05	***40.42***	28.52	04.28	31.53
NEA	00.00	25.00	10.00	00.00	04.00
NEB	NP	NP	00.00	00.00	00.00
NETP	36.25	00.00	19.92	00.00	09.09
NETO	07.44	16.39	09.09	05.85	00.00
NEL	***49.35***	***72.03***	***50.09***	29.26	***58.59***
NETI	***50.81***	***62.56***	***46.30***	***70.75***	***53.98***
NEN	***66.66***	***81.96***	30.43	***86.29***	23.63
NEM	***62.98***	***54.44***	20.68	35.44	***82.64***
NETE	12.56	17.43	NP	11.67	00.00
NP: Not present in reference data					

Table 4: Class specific F-measure for nested lexical match (HMM)

Table-2 shows the performance for specific classes of named entities. Table-3 presents the results for the HMM based system and Table-4 gives the class specific performance of the HMM based system.

6 Error Analysis

In both HMM, CRF based system the pos-tag and the chunk information are being used. NEs are generally the noun chunks. The pos-tagger and the chunker that we used had low accuracy. These errors in the POS-Tag contributed significantly to errors in NER.

In Telugu the F-measure for the maximal named entities is low for both the CRF, HMM models. This is because the test data had a large number of TIME named entities which are 5-6 words long. These entities further had nested named entities. Both the models are able to identify the nested named entities. We chose not to consider the Time entities as a maximal entity since it was not tagged as a maximal NE as in some places. Considering it as a maximal NE the F-measure of the system increased significantly to over 30 for both HMM and CRF based systems.

It is also observed that many NE's were retrieved correctly but were wrongly classified. Working with fewer tag-set will help to increase the performance of the system but this is not suggested.

7 Conclusion

The overall performance of the HMM model based hybrid system is better than the CRF model for all the languages. The performance of HMM based system is less that that of CRF. We obtained a decent Lexical F-measure of 39.77, 46.84, 45.84, 46.58, 44.73for Bengali, Hindi, Oriya, Telugu and Urdu using rules over HMM model. HMM based model has a better F-measure for NEP, NEL, NEO classes when compared to CRF model

References

CRF++: http://crfpp.sourceforge.net

P. Avinesh, G. Karthik. 2007. *Parts-of-Speech Tagging and Chunking using Conditional Random Fields and Transformation Based learning*. Proceedings of SPSAL workshop IJCNLP 07

Thorsen Brants. 2000. *TnT: a statistical Part-of-Speech Tagger*. Proceeding of sixth conference on Applied Natural Language Processing.

N. Kumar and Pushpak Bhattacharyya. 2006. *NER in Hindi using MEMM*.

J. Lafferty, A. McCullam, F. Pereira. 2001. *Conditional Random Fields: Probabilistic models for segmenting and labeling sequence data.* 18th International Conference on Machine Learning

Wei Li and A. McCallum. 2003. *Rapid Development of Hindi Named Entity Recognition using Conditional Random Fields and Feature Induction*. Transactions on Asian Language Information Processing.

Named Entity Recognition for South Asian Languages

Amit Goyal
University of Utah, School of Computing
Salt Lake City, Utah
amitg@cs.utah.edu

Abstract

Much work has already been done on building named entity recognition systems. However most of this work has been concentrated on English and other European languages. Hence, building a named entity recognition (NER) system for South Asian Languages (SAL) is still an open problem because they exhibit characteristics different from English. This paper builds a named entity recognizer which also identifies nested name entities for the Hindi language using machine learning algorithm, trained on an annotated corpus. However, the algorithm is designed in such a manner that it can easily be ported to other South Asian Languages provided the necessary NLP tools like POS tagger and chunker are available for that language. I compare results of Hindi data with English data of CONLL shared task of 2003.

1 Introduction

Identifying and classifying named-entities into person, location, organization or other names in a text is an important task for numerous applications. I focus here on building a named entity recognition system that will automatically mark the boundaries and labels of the named entities (NEs) in running text. The system also identifies nested named entities which are a superset of the maximal entities. E.g. "Lal Bahadur Shastri National Academy of Administration" is an organization name and is referred as maximal entity. However it also contains "Lal Bahadur Shastri" as a person name present inside an organization name and which is referred as a part of nested entity along with "Lal Bahadur Shastri National Academy of Administration" as an organization name.

To make the problem simpler, I split the problem into three sub tasks. The first (NER module) of which identifies whether an entity is a NE or not; the second (NEC module) identifies the type of label associated with each entity; the third (NNE module) identifies the nested name entities (NNE). Labels considered for this task are: person, organization and location names, measure, time, number, domain specific terms, abbreviation, title and designation.

Conditional random fields (CRFs) (Lafferty et al. 2001) with a variety of novel and traditional features have been used as a classifier for above three modules. CRFs are undirected graphical models, a special case of which is linear chains which are well suited to sequence labeling tasks. They have shown to be useful in part of speech tagging (Lafferty et al. 2001), shallow parsing (Sha and Pereira 2003), and named entity recognition for Hindi newswire data (Li and McCallum 2003).

2 Related Work

Named Entity Recognition (NER) has been considered as subtask of Information Extraction. Different NER systems were evaluated as a part of the Sixth Message Understanding Conference in 1995 (MUC6). The target language was English. Palmer and Day (1997) have worked on Chinese, English, French, Japanese, Portuguese and Spanish and found that the difficulty of the NER task was different for the six languages but that a large part of the task could be performed with simple methods.

Proceedings of the IJCNLP-08 Workshop on NER for South and South East Asian Languages, pages 89–96,
Hyderabad, India, January 2008.

Cucerzan et al. (1999) used both morphological and contextual clues for identifying named entities in English, Greek, Hindi, Rumanian and Turkish. With minimal supervision, they obtained overall F measures between 40 and 70, depending on the languages used. Collins (1999) showed that use of unlabelled data for NER can reduce the requirements for supervision to just 7 simple seed rules. The CoNLL shared task of 2002 and 2003 focused on language independent NER and has performed evaluations on English, Spanish, Dutch and German and participating systems have performed well. Li and McCallum (2003) used CRFs and feature induction (McCallum 2003) to get an F-score of 71.50 for Hindi language on test-set. May et al. (2003) used HMM to create NER for Hindi and Cebuano. Ekbal et al. (2007) used lexical pattern learning from corpus data for NER for Bangla language.

3 My Contributions

I focus here on building a NER system for the Hindi language using conditional random fields (CRFs) using NLPAI Machine Learning Contest 2007 data. The system is built in such a manner that it could be easily ported to other languages. This method was evaluated on test set 1 and test set 2 and attains a maximal F1 measure around 49.2 and nested F1 measure around 50.1 for test-set 1; maximal F1 measure around 44.97 and nested F1 measure 43.70 around for test-set 2. However the system achieves an F-measure of 58.85 on development set. The great difference in the numbers could be due to some difference in test and development set. I have also compared my results on Hindi data with English data of CONLL shared task of 2003 by introducing interesting phenomena which are not present in English. I perform experiments on English after removing capitalization since Hindi lacks such overt marking. Also there is another interesting phenomenon in Hindi or any other SAL i.e. a word can be a common noun as well as a proper noun. For example "sambhab sinha" is a name of a person but when I use 'sambhab' in a sentence "yaha kaam mujse sambhab nahi" It acts as a common noun meaning 'possible' in English. Hindi is full of such cases making the task more difficult. Hence it becomes very difficult for NER system to classify it as person or not.

4 Features

The success of any machine learning algorithm depends on finding an appropriate combination of features. This section outlines three types of features.

4.1 Contextual features

- Word Window: A word window of size n centered in position i_w is the sequence of words in the sentence placed at $i_w + j_w$ positions, with $j_w \in [-n , +n]$. For each word in the window, word and it's POS + its relative position j_w forms a feature
- Chunk window: A chunk window of context size n centered in position i_c is the sequence of chunks in the sentence placed $i_c + j_c$ positions, with $j_c \in [-n , +n]$. The tags (labels) of the chunks in the window + its relative position j_c form a feature.

4.2 Statistical features

- Binary features: As name suggests these features have value 0 or 1. These features are not mutually exclusive features that test whether the following predicates hold in the word: all digits, 4 digit number, contains hyphen, punctuation mark, acronym, alphanumeric etc. I also modeled whether a particular word is a noun or not using the POS information.
- Trigger words: Using the annotated training data I find all those words which have a high probability of being a number, measure, abbreviation and time. I model 4 binary features giving value 1 to high probable words and 0 to the rest. For example, high probable words for number would be "eka", "xo", "wIna", "cAra" etc. (words here are in wx-notation) and will get a value as 1.

4.3 Word Internal Feature

- Affixes: Some prefixes and suffixes are good indicators for identifying certain classes of entities. Suffixes are typically even more informative. For example, suffixes like -bad , -pur, -pally are good indicators of a name of a location.

- Words are also assigned a generalized 'word class (WC)' similar to Collins (2002), which replaces all letters with 'a', digits with '0', punctuation marks with 'p', and other characters with '-'. There is a similar 'brief class (BWC) (Settles 2004)' which collapses consecutive characters into one. Thus the words "D.D.T." and "AB-1946" would both be given the features WC=apapap, BWC=apapap and WC=aap0000, BWC=ap0 respectively, in above example hyphen forms the part of punctuation marks. This feature has been modeled since this feature can be useful for both unseen words as well as solving the data sparsity problem.
- Stem of the Word was also obtained using a morph analyzer.

We have tried to use the different combination of all these features for all three modules which I am going to discuss in the next section. But before ending there are few features which I haven't used and would like to use in future. Bag of words i.e. form of the words in the window without considering their position. Gazetteer Features can also be useful. These features couldn't be used due to computational reasons, lack of resources and time.

5 Modules

5.1 NER module

This module identifies whether an entity is a NE or not. I use well-known BIO model. B denotes begin of an entity, I denotes inside an entity; O denotes outside and is not part of any entity. Here I have only one label i.e. NE. Hence it becomes a three class problem with B-NE, I-NE and O as output labels. Here I am identifying NEs as it's an easier task as compare to classifying them among named-entity tag-set. It is also done with a hope that this information can be useful for NEC module. For example in entity like "Raja Ram Mohun Roy" tags would be "Raja/B-NE Ram/I-NE Mohun/I-NE Roy/I-NE." Similarly for "Microsoft Corp." tags would be "Microsoft/B-NE Corp./I-NE." Words like "tiger", "eat", "happy" etc which are not NEs are tagged as O.

5.2 NEC module

Here I try to classify the NEs among various classes/labels like person (like Mahatma Gandhi), location(like Delhi) and organization(like Microsoft Corp.) names, number (like one, two etc), time (like one day), measure (like 5 kg), domain specific terms (Botany, zoology etc), title (Mr., The Seven Year Itch), abbreviation (D.D.T.) and designation (Emperor). Hence it becomes a 10 (labels/classes) * 2(B+I) = 20 + 1 (O which denotes remaining words) =21 class problem. This module is independent from the previous module. For example in entity like "Raja Ram Mohun Roy" tags would be "Raja/B-NEP Ram/I-NEP Mohun/I-NEP Roy/I-NEP." Similarly for "Microsoft Corp." tags would be "Microsoft/B-NEO Corp./I-NEO."

I could have tried labeling the identified named-entities from NER However; I found that this results in a drop in accuracy. Hence I use the output of the NER module as one of the features for NEC.

5.3 NNE module

The length of nested named entities is unbounded but the majority contains at most 3 words. Therefore, I try to train three classifiers to learn entities of length 1, 2 and 3 independently. This allows us to learn nested entities since the bigger entities can have different tags when compared to smaller entities. For example, Srinivas Bangalore will be tagged as a name of a person by a classifier who is trained to classify NEs of length 2. However, Srinivas and Bangalore will be tagged as a name of a person and location respectively by a classifier which is trained to classify entities of length 1.

In this module also I use the same BIO model and there will be 21 classes for each of the three classifiers.

6 Experiments and Discussion

In this section I describe the experiments I performed to evaluate presented algorithm with its variations.

NLPAI 2007 NER contest Corpus, I was provided annotated training and development data comprising of 19825 and 4812 sentences respectively for Hindi. The data is labeled with 10 labels described above in NEC module. The average sentence length of the corpus is 24.5. The first step was to enrich the data with POS, chunk information and root of the word using POS tagger, Chun-

ker (Avinesh et al. 2007) and IIIT-Hyderabad morph analyzer. Hence porting this algorithm to any other SAL would require these tools for that language.

In the training data, in about 50% sentences (i.e.10524 sentences) there was not even a single NE. Experimentally I found that the inclusion or exclusion of these sentences did not have a significant effect on system performance. Hence I carried all the remaining experiments with sentences containing NEs. The reason for choosing it is it takes less time to train and more experiments could be performed given the time constraints.

Then I tried to find an appropriate set of features for NER and NEC module. For NNE I used the same features as used in NEC module since I don't have explicitly labeled data for nested entities. Tweaking and tuning of feature doesn't affect the accuracy significantly.

For NER module, where I am trying to identify name entities; context information seems to be more informative than statistical features. I use a window of -1 to +1 for words, -2 to +2 POS and also use features which are combinations of consecutive POS tags and words. For example Ram/NNP eat/VB mangoes/NNS. Combination features for word 'eat' would be NNP/VB, VB/NNS, Ram/eat, eat/mangoes, NNP/VB/NNS, Ram/eat/mangoes. The stem of the word and chunk information also doesn't affect the accuracy. The prefixes and suffixes of length 3 and 4 are found to improve the accuracy of the classifier. For example Hyderabad will have Hyd, Hyde, bad, abad as prefixes and suffixes of length 3 and 4 respectively. The word class (WC) and Brief word class (BWC) features are also very useful features for recognizing named-entities. I have achieved an F-measure of 64.28 by combination of all these features for identifying name-entities on development set. Table 1 shows the detailed results of named entity recognition (NER) module.

For NEC module, the contextual features as well as statistical features are helpful in deciding to which class a name-entity belongs. I use word and POS window of -1 to +1 as context. No combination features are being used as introduction of such features degrades the accuracy rather than improving it. However the statistical features are found to be more useful in this case as compared to NER. Here also prefixes and suffixes of length 3 and 4 are found to be useful. BWC feature alone is suffi-

Features	Precision	Recall	F-measure
Contextual	64.19	60.53	62.31
Contextual+ Word Internal	64.84	63.73	64.28

Table1: Detailed performance of NER module using only contextual features and combining word internal features.

Entity	Precision	Recall	F-measure
Abbreviation	43.21	36.46	39.55
Designation	69.61	46.84	56.00
Location	67.51	63.08	65.22
Measure	73.98	72.84	73.41
Number	70.41	87.74	78.13
organization	49.71	39.73	44.16
Person	61.18	47.37	53.40
Title	31.82	14.00	**19.44**
Terms	30.81	16.72	**21.67**
Time	67.30	58.53	62.61
Overall	62.60	55.52	**58.85**

Table2: Detailed performance of the best feature set on development set for maximal/nested named entities.

-cient for classification, we don't need to use WC feature for improving the accuracy. Chunk information and stem of the word doesn't improve the accuracy.

I have modeled NER module so that the output of that module can be used as feature for NEC. But using it as a feature doesn't improve the classification accuracy. Also, I tried using the boundary information from the NER module and combining it with labels learned from NEC module. It also seems to be a futile attempt.

I have used unlabelled data i.e. 24630 sentences provided during the contest and used bootstrapping to make use of it. I have doubled the data i.e. 50% manually annotated data and rest is system output on unlabelled data i.e. 12323 sentences; we have used only those sentences which contains at least one NE. With this data I almost get the same accuracy as I got with only manually annotated data. Table 2 shows the detailed performance of the best feature set on development set for maximal/nested named entities using evaluation script of CONLL shared task of 2003. I have used the evaluation script of NLPAI contest to report results on Test set-1 and Test set-2 (which contains 1091 and 744 sentences) for two systems in Table 3 and 4. One

trained using only annotated data and the other trained on annotated and bootstrapped data for the same feature set which performed best on development set. For test-set 2, system trained using annotated and bootstrapped data performs better than the system trained using only annotated data. However, for test set1 both the systems perform almost same. One of the reasons for less results as compared to development set is I haven't further classified title tag into title object and title person tag and Test sets contain many such instances.

I have trained a single classifier for all the entities but we can use more classifiers and divide the tags in such a fashion that those which are closer to one another fall in one group. For example we can club number, time and measure in one group and call them as number group since these are closer to each other and train a classifier to automatically annotate these entities in running text. Similarly, we can group person, number, and location and call them as name group. I have attempted a similar experiment using the same features of NEC module for number and name group but still there is no improvement.

For NNE module, I have used the same set of features which I have used in NEC module and I am handling nested entities up to length of 3. Since the development set is not enriched with nested entities, it is difficult to optimize the features for this module and the results would be same as NER module since nested entities are superset of maximal entities. For Test set-1 and Test set-2 Table 3 and 4 are used to report results.

For NEs like title there are fewer instances in training data which is a reason for its low F-measure i.e. 19.44 on development set which is even less than terms (i.e. 21.67) which are most difficult to learn. Also here I have focused on a large tag set but it would be interesting to concentrate only on person, location and organization names, since most of the systems report accuracy for these entities. Hence I did some experiments with Hindi data concentrating only on person, location and Organization but there is not so much increase in the performance.

When I trained my system on English data (which I have made mono case) of Conll-2003 shared task, with only contextual features, system gets an overall F-measure of 84.09 on development set and 75.81 on test set which is far better than Hindi. I have just used contextual features with window size of -1 to +1 for words, POS and chunk to achieve the results reported in Table 5 for test set. The reason for using only contextual information is that these features give the maximum accuracy and the rest of the features don't increase the accuracy by such a great amount. Also the aim over here is to compare results with Hindi language and not to make the best NER system for English language.

Entity	Test set1	Test set 2
Maximal Precision	70.78	55.24
Maximal Recall	37.69	35.75
Maximal F-Measure	**49.19**	43.41
Nested Precision	74.28	58.62
Nested Recall	37.73	33.07
Nested F-Measure	50.04	42.29

Table3: System trained using only annotated data

Entity	Test set1	Test set 2
Maximal Precision	70.28	57.60
Maximal Recall	37.62	36.88
Maximal F-Measure	49.00	**44.97**
Nested Precision	73.90	60.98
Nested Recall	37.93	34.05
Nested F-Measure	**50.13**	**43.70**

Table 4: System trained using annotated and bootstrapped data

Entity	Precision	Recall	F-measure
Person	82.05	79.16	80.58
Location	84.16	79.32	81.67
Organization	70.76	67.01	68.83
Misc.	73.71	61.11	66.82
Overall	78.40	73.39	75.81

Table 5: System trained on English mono case data using contextual features

Also to include common noun phenomena in English I have taken 10 random person names from the data and replaced them with common nouns and the results are really surprising. By introducing this, system achieves an F-measure of 84.32 on development set and 76.19 on test set which is better than the results on normal system. The number of tokens corresponding to these names in training data is 500. Table 6 contains the detailed results.

Entity	Precision	Recall	F-measure
Person	81.92	79.84	80.86
Location	84.18	80.10	82.09
Organization	71.98	67.13	69.47
Misc.	73.04	60.97	66.46
Overall	78.71	73.83	76.19

Table 6: System trained on English mono case data with common noun phenomena using contextual features

The results for English are far better than Hindi language. The reason is English already has tools like POS tagger and chunker which achieves an F measure around 95 whereas for Hindi we only have an F-measure of 85 for tagger and 80 for chunker. This is the reason why the accuracy of English system didn't fall when I removed capitalization and introduced common noun phenomena since POS context and chunk context helps a lot. Since CONLL 2003 data is already POS tagged and chunked, hence POS and chunks correspond to capitalized data. To make it more even, I ran Stanford POS tagger (Toutanova et al. 2003) on the same mono case CONLL 2003 data and then train the model using only word and POS context. The numbers drop on test set by more than 15% as shown in Table 7. For development set the overall F-measure is around 74%.

Entity	Precision	Recall	F-measure
Person	66.97	53.93	59.75
Location	68.57	56.54	61.98
Organization	71.64	53.55	61.29
Misc.	74.71	55.98	64.01
Overall	69.69	54.84	61.38

Table7: System trained on POS tagger ran on mono-case data

These numbers are comparable to Hindi data. The reason is POS tagger performs badly after removing capitalization. Now the POS tagged data marks proper noun i.e. NNP as common noun i.e. NN or foreign word as FW. The reason is it uses capitalization to mark NNP tag. We still haven't included common noun phenomena. So to do that, I take the common noun phenomenon English data and train the model using the same features as used above. Here also the system performs in the same way. There is just a decrease of 1% in F-measure of person class. Table 8 contains the detailed results. The introduction of common noun phenomena doesn't seem to affect the performance too much. The reason can be context helps in disambiguating between the real 'cheese' and the 'cheese' which has been made up by replacing it with 'John'.

Entity	Precision	Recall	F-measure
Person	65.48	53.37	58.81
Location	68.23	56.18	61.62
Organization	73.95	53.01	61.75
Misc.	74.81	56.27	64.23
Overall	69.74	54.45	61.16

Table8: System trained on POS tagger ran on mono case data which contains common noun phenomenon

After looking at these results, we can easily say that if we can improve the performance of POS tagger, we can do very well on the NER task. Without that it's even difficult for English to give good numbers. It is correct that Hindi and SAL don't have capitalization but we could make use of morphological features since most of SAL are morphologically rich. A hybrid approach involving rules along with machine learning approach could help us to improve POS tagger and NER systems.

After seeing results on English we ask what are the actual reasons for lower numbers on Hindi data? Inconsistency of annotated data is one of the big problems but it's very difficult to create 100% correct manual data since we have chosen a finely grained tagset. Also the data used for Hindi is from different domains. Hence due to which the lot of terms doesn't occur in corpus more than once. One of the plausible reasons for bad results on test set for Hindi compared to development set could be

difference in domain of test set. Also due to lack of resources like gazetteer for SAL the task becomes more challenging to create everything from scratch. Also the accuracy of tagger, chunker and morph analyzer are not as good as when we compare results with English.

7 Conclusion

In conclusion, I have confirmed that use of machine learning algorithm on annotated data for Hindi language can be useful and the same algorithm can be useful for other languages. I only need to tune and tweak the features for a particular language. I have described some traditional and novel features for Hindi language. I have also shown that it's better to directly classify name-entities into various labels or classes rather than first recognizing them. Also the attempt to make use of unlabelled data didn't help much.

Also I have showed that capitalization is one of the important clues for high performance of English on various NLP applications. But we could also recognize some other important clues in SAL and can hope to do better than English without having capitalization.

Directions for future work include concentrating on a smaller tag set and trying to improve accuracy for each of the label. Since still we don't have enough labeled data for other SAL, it would be interesting to try out some unsupervised or semi-supervised approaches. Also I haven't tried rule based approach which could be very handy when combined with some machine learning approach. Hence adopting a hybrid approach should help in improving the accuracy of the system but still it's an open question.

8 Acknowledgements

I would like to thank Prof. Rajeev Sangal, Dr. Hal Daume III, Dr. Dipti Misra Sharma and Anil Kumar Singh for their helpful comments and suggestions.

References

Andrew McCallum. 2003. *Efficiently Inducing Features of Conditional Random Fields.* In Proceedings of the 19th Conference in UAI.

Andrew McCallum and Wei Li. 2003. *Early results for named entity recognition with conditional random fields, feature induction and web-enhanced lexicons*. In Proceedings CoNLL 2003.

Avinesh.PVS. and Karthik G. 2007. *Part-Of-Speech Tagging and Chunking using Conditional Random Fields and Transformation Based Learning.* In Proceedings of SPSAL2007

Asif Ekbal and Sivaji Bandyopadhyay. 2007. *Lexical Pattern Learning from Corpus Data for Named entity recognition*. In Proceedings of ICON 2007.

Burr Settles. 2004. *Biomedical Named Entity Recognition Using Conditional Random Fields and Rich Feature Sets*. In Proceedings of the International Joint Workshop on NLPBA.

David D. Palmer and David S. Day. 1997. *A Statistical Profile of the Named Entity Task*. In Proceedings of Fifth ACL Conference for ANLP.

Erik F. Tjong Kim Sang. 2002. Introduction to the CoNLL-2002 Shared Task: *Language-Independent Named Entity Recognition*. In Proceedings of the CoNLL 2002.

Erik F. Tjong Kim Sang and Fien De Meulder. 2003. *Introduction to the CoNLL-2003 Shared Task: Language-Independent Named Entity Recognition*. In Proceedings of the CoNLL 2003.

Fei Sha and Fernando Pereira. 2003. *Shallow parsing-with conditional random fields*. In Proceedings of the HLT and NAACL 2003.

John Lafferty, Andrew McCallum, and Fernando Pereira. 2001. *Conditional random fields: Probabilistic models for segmenting and labeling sequence data*. In Proceedings of ICML 2001.

Kristina Toutanova and Christopher D. Manning. 2000. *Enriching the Knowledge Sources Used in a Maximum Entropy Part-of-Speech Tagger.* Proceedings of the Joint SIGDAT Conference on (EMNLP/VLC-2000), Hong Kong.

Kristina Toutanova, Dan Klein, Christopher Manning, and Yoram Singer. 2003. *Feature-Rich Part-of-Speech Tagging with a Cyclic Dependency Network.* In Proceedings of HLT-NAACL 2003.

Michael Collins. 2002. *Ranking algorithms for named-entity extraction: Boosting and the voted perceptron*. In Proceedings of ACL 2002.

Michael Collins and Yoram Singer. 1999. *Unsupervised models for named entity classification.* In Proceedings of the Joint SIGDAT Conference on EMNLP *and Very Large Corpora*.

Silviu Cucerzan and David Yarowsky. 1999. *Language independent named entity recognition combining*

morphological and contextual evidence. In Proceedings of 1999 Joint SIGDAT Conference on EMNLP and VLC.

Wei Li and Andrew McCallum. 2003. *Rapid Development of Hindi Named Entity Recognition Using Conditional Random Fields and Feature Induction*. In Proceedings of ACM TALIP.

Named Entity Recognition for Indian Languages

Animesh Nayan, B. Ravi Kiran Rao, Pawandeep Singh,
Sudip Sanyal and Ratna Sanyal
Indian Institute of Information Technology
Allahabad, India
e-mail@domain

Abstract

Abstract Stub This paper talks about a new approach to recognize named entities for Indian languages. Phonetic matching technique is used to match the strings of different languages on the basis of their similar sounding property. We have tested our system with a comparable corpus of English and Hindi language data. This approach is language independent and requires only a set of rules appropriate for a language.

1 Introduction

Named Entity Recognition (NER) is a subtask of machine translation and information retrieval. Named entities are words which belong to certain categories like persons, places, organizations, numerical quantities, expressions of times etc. A large number of techniques have been developed to recognize named entities for different languages. Some of them are Rule based and others are Statistical techniques. The rule based approach uses the morphological and contextual evidence (Kim and Woodland, 2000) of a natural language and consequently determines the named entities. This eventually leads to formation of some language specific rules for identifying named entities. The statistical techniques use large annotated data to train a model (Malouf, 2002) (like Hidden Markov Model) and subsequently examine it with the test data. Both the methods mentioned above require the efforts of a language expert. An appropriately large set of annotated data is yet to be made available for the Indian Languages. Consequently, the application of the statistical technique for Indian Languages is not very feasible.

This paper deals with a new technique to recognize named entities of different languages. Our approach does not use the previously mentioned techniques. Instead, we use an approach that not only reduces the burden of collecting and annotating data, but is language independent as well. We use this method to build a multilingual named entity list that can be used by the named entity recognizer. Our method recognizes and finds the actual representation of the named entities in the target language from an untagged corpus. Our idea was to match the two representations of the same named entity in two different languages using a phonetic matching algorithm. This comes from the property of named entities that they sound similar when written in native script or any other script. However this cross-lingual matching is not a trivial task. First of all, the two strings to be matched have to be represented in a common script. So we face two choices here. Either we should convert the two strings into some common intermediate representation (ex. Phonemic representation) or transliterate the name written in Indian language to English and then look for phonetic equivalence. Our engine has been tested for Hindi. After making transliteration rules for Hindi, we used a variation of the Editex algorithm to match the transliterated string with entries in English named entity database to find a match. Here it is worthwhile to mention that certain class of name entities which are not similar sounding (mostly phrases) cannot be extracted through this cross-lingual matching. E.g. “United Nations”, “Government of India” etc. Abbreviations which are spelled character by charac-

Proceedings of the IJCNLP-08 Workshop on NER for South and South East Asian Languages, pages 97–104,
Hyderabad, India, January 2008.

ter in both the languages can however be extracted. E.g. BBC (बीबीसी), LTTE (एलटीटीई) etc.

In the next section we have given the system architecture. The logical flow and overall description of the system are discussed here. Our own set of transliteration rules in Hindi are given in the third section. In the fourth section we define our baseline task. Our system has been tested with a parallel corpus which consisted of both English and Hindi language data. The results obtained using our system is described in the fifth section together with an analysis. Conclusions are presented in the last section together with directions for future improvements.

2 System Architecture: Logical Flow and overall description of the System

The system architecture is shown in Figure 1. It consists of the following modules:

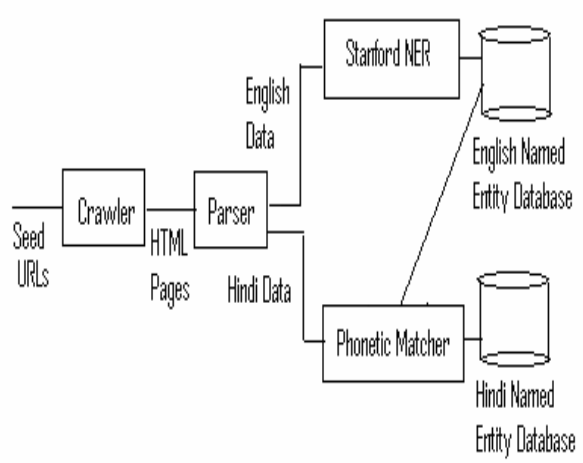

Figure 1: System Architecture

2.1 Crawler

The crawler is a web-bot or spider which browses the web in an automated manner. It starts with a list of Uniform Resource Locators (URL) that it is to visit, called the seeds. As the crawler visits these URL's it collects all the hyperlinks and adds them to a queue. URL's from the queue are crawled further. Since the crawler collects the data from web, the data collection is fully automated. The crawler gathers data for both English and other Indian languages. The data collected for English is used to populate the English named entity database which is significantly accurate. We have used the freely available Stanford Named Entity Recognizer (Finkel, Grenager, and Manning, 2005) in our engine. The data collected for Indian languages will be used to build a database of named entities for the given language.

2.2 Parser

The crawler saves the content in an html form onto the system. The parser parses these html files. Additionally the parser can also parse the PDF as well as RTF files. The output of the parser is passed to the corresponding modules for the two different languages.

2.3 Phonetic Matcher

Phonetic matching is the task of matching two representations of the same name. A name may have more than one representation in its native script itself. If the name is represented in a script other than its native script, there may be large number of potential variants for its representation. Phonetic matching is a fuzzy string matching technique in which we match strings on the basis of their similar sounding property and not identity. Most common phonetic matching techniques are Soundex and Editex. These techniques are used to match two representations of the same name in English. We survey the techniques in the following subsections.

2.3.1 Soundex

Soundex algorithm was designed by Odell and Russell in 1918 to find spelling variation of names. It represents classes of sounds which can be lumped together. The classes for the algorithm are shown in Appendix A. These classes are placed for phonetic matching according to the following algorithm:

1. Replace all but the first letter of the string by its phonetic code.
2. Eliminate any adjacent representation of codes.
3. Eliminate all occurrences of code 0 i.e. eliminate all vowels.
4. Return the first four characters of the resulting string.
5. Examples: Dickson = d25, Dikson = d25.

Two names match if they have the same soundex representation. This method does not account

for vowels and hence is not accurate for cross-lingual matching.

2.3.2 Editex

The Editex algorithm was designed by Zobel and Dart (Zobel and Dart,1996). It is an enhancement of the Levenshtein (Levenshtein, 1966) edit distance algorithm. The Levenshtein algorithm measures the edit distance between two strings where edit distance is defined as the minimum number of basic operations required to match one string to the other where the basic operations are insertion, deletion and substitution. Insertion and deletion costs are 1 and substitution cost is given by a function subst_cost (Xi, Yj) which returns 0 if the two characters Xi and Yj are same and 1, if they are different. The score dist [m, n] is returned as the edit distance between two strings. A score of zero implies a perfect match.

The algorithm has O (mn) time and space complexity where m and n are the lengths of the two strings respectively. The pseudo code for the Levenshtein edit distance algorithm is described in Appendix B. Editex groups similar sounding phonemes into equivalence classes. The substitution cost is determined by a function S (Xi, Yj) that returns 0 if the two characters Xi and Yj are same, 1 if they lie in the same equivalence class and 2 otherwise. The insertion and substitution costs are determined by a function D (Xi-1, Xi) which is almost same as S (Xi, Yj) except for the difference that it compares letters of the same string and it returns 1 if Xi-1 is 'h' or 'w' and Xi-1 is not equal to Xi. The editex equivalence classes and the editex pseudo-code are given in Appendix C.

Editex performs fairly better than Soundex and Leveinshtein edit distance algorithms. However further enhancements in Editex are also possible. "Tapering" is one enhancement in which we weigh mismatches at the beginning of the string with higher score than mismatches towards the end (Zobel and Dart, 1996). Other enhancements are those in which input strings are mapped to their phonemic representation, called phonometric methods (Zobel and Dart, 1996).

3 Transliteration rules

To perform phonetic matching of two different representations of a named entity, we need both of them in a common script. We choose to transliterate the named entity in Indian language to English. The transliteration rules for a language must be written for the same. We have written our own set of transliteration rules for Hindi. These can be described briefly as under

The entity to be transliterated is scanned character by character from left to right. Each character of Hindi is mapped to an equivalent character/set of character in English according to a mapping function. The character set generated by the function is appended into a string as per the rules. E.g. का = क् + अ is a single character representation in Unicode ('क') and maps to 'Ka'.

1. Start with an empty string. When a consonant or singleton vowel (not as 'matra') is encountered append the set of characters returned by mapping function.
2. When a consonant is followed by a vowel the preceding 'a' should be removed and the character set for the vowel should be appended. E.g. के consists of two characters क + े. Once we encounter क we append 'ka' and when े is encountered next we remove the 'a' and append the mapping for े i.e. 'e'. This rule applies in general to all the vowels.
3. If the transliterated string has 'a' as its last character while it doesn't have the vowel ा as last character of Hindi string, remove this occurrence of 'a'. The last vowel in Hindi is very important as two altogether different words may have the only difference in the last vowel. E.g. "कमल" and "कमला" are proper nouns having different genders. Their English representations are "Kamal" and "Kamla" respectively.

The transliteration always performs a one to one mapping of a character in Hindi to a set of characters in English. However the English representation may have different character sets for the same Hindi character in different names. E.g. "कमल" is "Kamal" while "क्रिकेट" is "**Cricket**". 'क' is often represented by 'K' for Hindi names, by 'C' for

English names and by ‘Q’ for Urdu names. The Editex algorithm groups these letters in the same equivalence class.

4 Baseline Task

At the core of our method lies the phonetic matching algorithm. We have modified the Editex algorithm as mentioned in Appendix C. Editex can be modified to take into account that there can be more than three (0, 1, 2) levels of acceptability for substitutions due to the inherent properties of particular languages. For example, say “ckq” is one equivalence class in Editex. ‘c’ and ‘k’ have a substitution cost of 1. We may reduce this substitution cost to 0.5 for a language in which it is highly probable that the same character maps to ‘c’ and ‘k’ in the English representation of its names. Thus the equivalence classes and the substitution costs in Editex can be modified for cross-lingual phonetic matching. There can also be further language specific enhancements. The following algorithm along with some language specific enhancements was implemented for Hindi.

4.1 Abbreviation Check

Abbreviations form an important class of named entities. So, we first check whether the Hindi string is an abbreviation in which the English characters are spelled individually. For each English alphabet we have some unique Hindi representation. The function performs accurately most of the time and extracts such named entities. If we are able to find out that the string is an abbreviation, the corresponding English representation can be returned by the function itself, hence there is no need of further matching. If the string is not an abbreviation, we proceed to the actual matching algorithm.

4.2 4.2. First letter matching

The first letters of the two strings must either be the same or should belong to the same equivalence class. The equivalence classes for first character matching are:

"ckq", "wbv", "iy","jz", "aeiou"

The English named entity database must be indexed according to the first letter of the named entity so that we only search for matches in those indexes which fall into the same equivalence class. This is very important for the computational efficiency of the engine as it reduces the search space.

4.3 Preprocessing

Often the phonetic inconsistencies in English lead to low matching score for two representation of the same name. To take this into account, before matching the two strings the named entity retrieved from English Named entity database is preprocessed to form a new string. We have used the famous “Mark Twain’s plan for the improvement of English spelling” (http://grammar.ccc.commnet.edu/grammar/twain.htm) added with some more rules. This way we tackle the problem of more than one possible character sets for some vowels since only one of them can be chosen during transliteration. We also tackle some other problems like silent-alphabets and repeated alphabets so that the probability of generating high matching score increases. The following set of rules for preprocessing was used.

1. Change all occurrences of “oo” to “u”. (both character sets are for the vowel ू)
2. Change all occurrences of “ee” to “i” (both character sets are for the vowel ी)
3. Change all occurrences of “f” to ph”
4. Change all occurrences of “au” to “o”
5. If a word starts with "x", replace the "x" with a "z". Change all the remaining "x"s to "ks"s.
6. If a "c" is directly followed by an "e" or "i", change the "c" to an "s"
7. If a "c" is directly followed by a "k", remove the "c". Keep applying this rule as necessary (Example: "cck" becomes "k".)
8. If a word starts with "sch", change the "sch" to a "sk".
9. If a "ch" is directly followed by an "r", change the "ch" to a "k".
10. After applying the above rules, change all "c"s that are not directly followed by an "h", to a "k". (This includes all "c"s that are last letter of a word)
11. If a word starts with "kn" change "kn" to“n”
12. Change all double consonants of the same letter to a single consonant. A consonant is any letter that is not one of "a, e, i, o, u." (Example: "apple" becomes "aple"). Keep

applying this rule as necessary (Example: "zzz" becomes "z".)

4.4 Editex Score

Now the transliterated string and the preprocessed string are compared to generate an editex score. The equivalence classes we used were similar to as proposed in the original editex algorithm except for some language specific changes for Hindi. Length of the two strings has to be considered while deciding the threshold score for a match otherwise there can be greater number of mismatches for small strings. So we normalize editex score as
d = [1- {editex(X, Y) / (length(X) + length(Y)}]

The decided threshold for match was 0.86. A score above threshold guarantees equivalence of the two representations. The results are shown in Table-1.

Hindi NE	English NE	Transliteration Output	Editex Score
हिन्दी	Hindi	Hindi	1.0
फ़लस्तीनी	Philistini	Phalastini	0.9
बांगलादेश	Bangladesh	Bangladesh	1.0
झारखण्ड	Jharkhand	Jharakhand	0.894
पश्चिम	Pashchim	Pashchim	1.0
बंगाल	Bengal	Bangal	0.916
भारत	Bharat	Bharat	1.0
क्रिकेट	Cricket	Kriket	0.923
ग्रेग	Greg	Greg	1.0
चैपल	Chappel	Chaipal	0.857
महेंद्र	Mahendra	Mahendr	0.933
राहुल	Rahul	Rahul	1.0
द्रविड	Dravid	Dravid	1.0
छत्तीसगढ	Chattisgarh	Chattisagadh	0.866

Table-1: Hindi named entities with transliteration output and normalized Editex scores

5 Results and Analysis

We have tested our system with a parallel corpus which consisted of both English and Hindi language data. Further we used the web crawler to populate our NE list of both the languages thus embedding the concept of comparable corpus. The results **for English** obtained using parallel corpus are:

Precision: 81.40% and Recall: 81.39%

This corpus carried named entities from the domain of travel, tourism and culture. Further for classifying the results for Hindi we used the definition of named entities as given by Chinchor (Chinchor, 1997) as for entity names organizations (OE), person names (PE) and location names (LE). The results for numeric expressions (monetary values and percentages) and temporal expressions (dates and times) were not considered for results because it is a trivial task to build grammar rules for such entities which appear quite regularly.

We have focused on OE, PE and LE named entities for Hindi so that we can analyze the performance on new and hitherto undiscovered entities which come into existence with the passage of time. This premise provides the real basis for challenging the performance of any NER technique for Indian Languages.

The testing on the corpus of around 1000 sentences revealed the following results **for Hindi**:

- Precision for all named entities (PE+OE+LE): 80.2%
- Recall for PE (person entity names): 47.4%
- Recall for OE (organization entity names): 42.9%
- Recall for LE (location entity names): 74.6%

It is important to observe here that the engine shows good recall for location entity names (LE) which were more abundant in the corpus. Besides this, the corpus had a heterogeneous mix of named entities with tourism-related information not only from India but also from the continents of South America and Antarctica. A good recall percentage for Hindi location entity names is encouraging as the named entities related to South America and Antarctica did not have phonetic similarity with

the native entities available from tourism information from India. This gives good credence to the phonetic matching approach used above. Causes for the comparatively lower recall percentage among person entity names and organization entity names are under further investigation.

6 Conclusions

We have used the phonetic matching technique to match the strings of different languages on the basis of their similar sounding property. As the Phonetic Matcher module is tested for more data, more generic rules can be made to improve its accuracy. The Engine should be improved so that it may recognize phrasal named entities and abbreviations. The engine will work for any language if the phonetic matching rules are written for that language. We can also develop a crawler which will be focused upon a certain domain of interest. Focused crawlers are very important for generating resources for natural language processing. A focused crawler application is an intelligent agent that crawls the web for content related to a specific domain. This kind of crawler could be used in the future for purposes of data collection for a particular domain.

7 Acknowledgements

The authors gratefully acknowledge financial assistance from TDIL, MCIT (Govt. of India).

References

Chinchor, N. 1997. *MUC-7 Named entity task definition*. In Proceedings of the 7th Message Understanding Conference (MUC-7)

Finkel, Jenny Rose, Grenager, Trond and Manning, Christopher. 2005. *Incorporating Non-local Information into Information Extraction Systems by Gibbs Sampling*. Proceedings of the 43rd Annual Meeting of the Association for Computational Linguistics (ACL 2005), pp. 363-370.

Kim, J. and Woodland, P.C. 2000a. *Rule Based Named Entity Recognition*. Technical Report CUED/ FINFENG/TR.385, Cambridge University Engineering Department, 2000.

Malouf, Robert. 2002 *Markov models for language-independent named entity recognition*. In Proceedings of CoNLL-2002 Taipei, Taiwan, pages 591-599.

Levenshtein, V.I. 1966. *Binary codes capable of correcting deletions, insertions, and reversals*. Soviet Physics Doklady 10: 707–710.

Zobel, Justin and Dart, Philip. 1996. *Phonetic string matching: Lessons from information retrieval*. In Proceedings of the Eighteenth ACM SIGIR International Conference on Research and Development in Information Retrieval, Zurich, Switzerland, August 1996, pp. 166-173.

Appendix A: Soundex classes

Code	Letters	Code	Letters
0	aeiouyhw	4	l
1	bpfv	5	mn
2	cgjkqsxz	6	R
3	dt		

Appendix B: Pseudo code for Leveinshtein edit distance:

```
Input: Two strings, X and Y
Output: The minimum edit dis-
tance between X and Y

m ← length(X)
n ← length(Y)

for i =0 to m do
dist[i, 0] ← i

for j = 0 to n do
dist[0, j] ← j

for i = 1 to m do
for j = 1 to n do

dist[i, j] =
min{
    dist[i-1, j]+inser_cost,
    dist[i-1, j-1]
    + subst_cost[Xi, Yj],
    dist[i, j-1] + delet_cost
}
end
```

Appendix C: Editex Equivalence Classes:

aeiouy	bp	ckq	dt	lr	mn
gj	fpy	sxz	csz		

Pseudo code for Editex Algorithm

```
Input: Two strings, X and Y
Output: The editex distance
between X and Y
 m = length(X)
 n = length(Y)

editex_dist[0, 0] = 0

for i = 1 to m do

editex_dist[i, 0]
  = editex_dist[i-1, 0]
  + D(Xi-1, Xi)

for j = 0 to n do

editex_dist[0, j]
  = editex_dist[0, j-1]
  + D(Yj-1, Yj)

for i = 1 to m do
for j = 1 to n do

editex_dist[i, j] =
  min { editex_dist[i-1, j]
  + D(Xi-1, Xi),
  editex_dist[i-1, j-1]
  + S(X, Yj),
  editex_dist[i, j-1]
  + D(Yj-1, Yj)

end
```

Experiments in Telugu NER: A Conditional Random Field Approach

Praneeth M Shishtla, Karthik Gali, Prasad Pingali and Vasudeva Varma
{praneethms,karthikg}@students.iiit.ac.in,{pvvpr,vv}@iiit.ac.in
Language Technologies Research Centre
International Institute of Information Technology
Hyderabad, India

Abstract

Named Entity Recognition(NER) is the task of identifying and classifying tokens in a text document into predefined set of classes. In this paper we show our experiments with various feature combinations for Telugu NER. We also observed that the prefix and suffix information helps a lot in finding the class of the token. We also show the effect of the training data on the performance of the system. The best performing model gave an $F_{\beta=1}$ measure of 44.91. The language independent features gave an $F_{\beta=1}$ measure of 44.89 which is close to $F_{\beta=1}$ measure obtained even by including the language dependent features.

1 Introduction

The objective of NER is to identify and classify all tokens in a text document into predefined classes such as person, organization, location, miscellaneous. The Named Entity information in a document is used in many of the language processing tasks. NER was created as a subtask in Message Understanding Conference (MUC) (Chinchor, 1997). This reflects the importance of NER in the area of Information Extraction (IE). NER has many applications in the areas of Natural Language Processing, Information Extraction, Information Retrieval and speech processing. NER is also used in question answering systems (Toral et al., 2005; Molla et al., 2006), and machine translation systems (Babych and Hartley, 2003). It is also a subtask in organizing and retrieving biomedical information (Tsai, 2006).
The process of NER consists of two steps

- identification of boundaries of proper nouns.
- classification of these identified proper nouns.

The Named Entities(NEs) should be correctly identified for their boundaries and later correctly classified into their class. Recognizing NEs in an English document can be done easily with a good amount of accuracy(using the capitalization feature). Indian Languages are very much different from the English like languages.

Some challenges in named entity recognition that are found across various languages are: Many named entities(NEs) occur rarely in the corpus i.e they belong to the open class of nouns. Ambiguity of NEs. Ex *Washington* can be a person's name or a place name. There are many ways of mentioning the same Named Entity(NE). In case of person names, *Ex: Abdul Kalam, A.P.J.Kalam, Kalam* refer to the same person. And, in case of place names *Warangal, WGL* both refer to the same location. Named Entities mostly have initial capital letters. This discriminating feature of NEs can be used to solve the problem to some extent in English.

Indian Languages have some additional challenges: We discuss the challenges that are specific to Telugu. Absence of capitalization. Ex: The condensed form of the person name *S.R.Shastry* is written as *S.R.S* in English and is represented as *srs* in Telugu. Agglutinative property of the Indian Languages makes the identification more difficult. Agglutinative languages such as Turkish or Finnish, Telugu etc. differ from languages like English in

Proceedings of the IJCNLP-08 Workshop on NER for South and South East Asian Languages, pages 105–110,
Hyderabad, India, January 2008.

the way lexical forms are generated. Words are formed by productive affixations of derivational and inflectional suffixes to roots or stems. *For example: warangal, warangal ki, warangalki, warangallo, warangal ni* etc .. all refer to the place Warangal. where *lo, ki, ni* are all postpostion markers in Telugu. All the postpositions get added to the stem hyderabad. There are many ways of representing acronyms. The letters in acronyms could be the English alphabet or the native alphabet. Ex: *B.J.P* and *BaJaPa* both are acronyms of *Bharatiya Janata Party*. Telugu has a relatively free word order when compared with English. The morpohology of Telugu is very complex. The Named Entity Recognition algorithm must be able handle most of these above variations which otherwise are not found in languages like English. There are not rich and robust tools for the Indian Languages. For Telugu, though a Part Of Speech(POS) Tagger for Telugu, is available, the accuracy is less when compared to English and Hindi.

2 Problem Statement

NER as sequence labelling task
Named entity recognition (NER) can be modelled as a sequence labelling task (Lafferty et al., 2001). Given an input sequence of words $W_1^n = w_1 w_2 w_3 ...w_n$, the NER task is to construct a label sequence $L_1^n = l_1 l_2 l_3 ...l_n$, where label l_i either belongs to the set of predefined classes for named entities or is none(representing words which are not proper nouns). The general label sequence l_1^n has the highest probability of occuring for the word sequence W_1^n among all possible label sequences, that is

$$\hat{L}_1^n = \text{argmax} \{\Pr(L_1^n \mid W_1^n)\}$$

3 Conditional Random Fields

Conditional Random Fields (CRFs) (Wallach, 2004) are undirected graphical models used to calculate the conditional probability of values on designated output nodes given values assigned to other designated input nodes. In the special case in which the output nodes of the graphical model are linked by edges in a linear chain, CRFs make a first-order Markov independence assumption, and thus can be understood as conditionally-trained finite state machines(FSMs).

Let o = $\langle O_1, O_2, ...O_T \rangle$ be some observed input data sequence, such as a sequence of words in text in a document,(the values on n input nodes of the graphical model). Let **S** be a set of FSM states, each of which is associated with a label, $l \in \mathscr{L}$.
Let **s** = $\langle s_1, s_2, ... s_T, \rangle$ be some sequence of states,(the values on T output nodes). By the Hammersley-Clifford theorem CRFs define the conditional probability of a state sequence given an input sequence to be

$$P(s|o) = \frac{1}{Z_o} * exp(\sum_{t=1}^{T} \sum_{k} \lambda_k f_k(s_{t-1}, s_t, o, t))$$

where Z_o is a normalization factor over all state sequences, is an arbitrary feature function over its arguments, and λ_k is a learned weight for each feature function. A feature function may, for example, be defined to have value 0 or 1. Higher λ weights make their corresponding FSM transitions more likely.

CRFs define the conditional probability of a label sequence based on total probability over the state sequences, $P(l|o) = \sum_{s:l(s)=l} P(s|o)$ where l(s) is the sequence of labels corresponding to the labels of the states in sequence s. Note that the normalization factor, Z_o, (also known in statistical physics as the partition function) is the sum of the scores of all possible state sequences,

$$Z_o = \sum_{s \in S^T} * exp(\sum_{t=1}^{T} \sum_{k} \lambda_k f_k(s_{t-1}, s_t, o, t))$$

and that the number of state sequences is exponential in the input sequence length,T. In arbitrarily-structure CRFs, calculating the partition function in closed form is intractable, and approximation methods such as Gibbs sampling, or loopy belief propagation must be used.

4 Features

There are many types of features used in general NER systems. Many systems use binary features i.e. the word-internal features, which indicate the presence or absence of particular property in the word. (Mikheev, 1997; Wacholder et al., 1997; Bikel et al., 1997). Following are examples of binary features commonly used. All-Caps (IBM), Internal capitalization (eBay), initial capital (Abdul Kalam), uncapitalized word (can), 2-digit number

(83, 28), 4-digit number (1273, 1984), all digits (8, 31, 1228) etc. The features that correspond to the capitalization are not applicable to Telugu. We have not used any binary features in our experiments.

Gazetteers are used to check if a part of the named entity is present in the gazetteers. We don't have proper gazetteers for Telugu.

Lexical features like a sliding window [w_{-2},w_{-1},w_o,w_1,w_2] are used to create a lexical history view. Prefix and suffix tries were also used previously(Cucerzan and Yarowsky,1999).

Linguistics features like Part Of Speech, Chunk, etc are also used.

4.1 Our Features

We donot have a highly accurate Part Of Speech(POS) tagger. In order to obtain some POS and chunk information, we ran a POS Tagger and chunker for telugu (PVS and G, 2007) on the data. And from that, we used the following features in our experiments.

Language Independent Features
current token: w_0
previous 3 tokens: w_{-3},w_{-2},w_{-1}
next 3 tokens: w_1,w_2,w_3
compound feature:w_0 w_1
compound feature:w_{-1} w_0
prefixes (len=1,2,3,4) of w_0: pre_0
suffixes (len=1,2,3,4) of w_0: suf_0

Language Dependent Features
POS of current word: POS_0
Chunk of current word: $Chunk_0$

Each feature is capable of providing some information about the NE.

The word window helps in using the context information while guessing the tag of the token. The prefix and suffix feature to some extent help in capturing the variations that may occur due to agglutination.

The POS tag feature gives a hint whether the word is a proper noun. When this is a proper noun it has a chance of being a NE. The chunk feature helps in finding the boundary of the NE.

In Indian Languages suffixes and other inflections get attached to the words increasing the length of the word and reducing the number of occurences of that word in the entire corpus. The character n-grams can capture these variations.

5 Experimental Setup

5.1 Corpus

We conducted the experiments on the developement data released as a part of NER for South and South-East Asian Languages (NERSSEAL) Competetion. The corpus in total consisted of 64026 tokens out of which 10894 were Named Entities(NEs). We divided the corpus into training and testing sets. The training set consisted of 46068 tokens out of which 8485 were NEs. The testing set consisted of 17951 tokens out of which 2407 were NEs. The tagset as mentioned in the release, was based on AUKBC's ENAMEX,TIMEX and NAMEX, has the following tags: NEP (Person), NED (Designation), NEO (Organization), NEA (Abbreviation), NEB (Brand), NETP (Title-Person), NETO (Title-Object), NEL (Location), NETI (Time), NEN (Number), NEM (Measure) & NETE (Terms).

5.2 Tagging Scheme

The corpus is tagged using the IOB tagging scheme (Ramshaw and Marcus, 1995). In this scheme each line contains a word at the beginning followed by its tag. The tag encodes the type of named entity and whether the word is in the beginning or inside the NE. Empty lines represent sentence(document) boundaries. An example is given in table 1.

Words tagged with O are outside of named entities and the I-XXX tag is used for words inside a named entity of type XXX. Whenever two entities of type XXX are immediately next to each other, the first word of the second entity will be tagged B-XXX in order to show that it starts another entity. This tagging scheme is the IOB scheme originally put forward by Ramshaw and Marcus (1995).

5.3 Experiments

To evaluate the performance of our Named Entity Recognizer, we used three standard metrics namely precision, recall and f-measure. Precision measures the number of correct Named Entities(NEs) in the

Token	Named Entity Tag
Swami	B-NEP
Vivekananda	I-NEP
was	O
born	O
on	O
January	B-NETI
,	I-NETI
12	I-NETI
in	O
Calcutta	B-NEL
.	O

Table 1: IOB tagging scheme.

machine tagged file over the total number of NEs in the machine tagged file and the recall measures the number of correct NEs in the machine tagged file over the total number of NEs in the golden standard file while F-measure is the weighted harmonic mean of precision and recall:

$$F = \frac{\left(\beta^2+1\right)\ RP}{\beta^2 R + P}$$

with

$$\beta = 1$$

where P is Precision, R is Recall and F is F-measure.

W_{-n+n}: A word window :w_{-n}, w_{-n+1}, .., w_{-1}, w_0, w_1, .., w_{n-1}, w_n.
POS_n: POS n^{th} token.
Ch_n: Chunk of n^{th} token.
pre_n: Prefix information of n^{th} token. (prefix length=1,2,3,4)
suf_n: Suffix information of n^{th} token. (suffix length=1,2,3,4)

The more the features, the better is the performance. The inclusion of the word window, prefix and suffix features have increased the $F_{\beta=1}$ measure significantly. Whenever the suffix feature is included, the performance of the system increased. This shows that the system is able to caputure those agglutinative language variations. We also have experimented changing the training data size. While varying the training data size, we have tested the performance on the same amount of testing data of 17951 tokens.

6 Conclusion & Future Work

The inclusion of prefix and suffix feature helps in improving the $F_{\beta=1}$ measure (also recall) of the system. As the size of the training data is increased, the $F_{\beta=1}$ measure is increased. Even without the language specific information the system is able to perform well. The suffix feature helped improve the recall. This is due to the fact that the POS tagger also uses the same features in predicting the POS tags. Prefix, suffix and word are three non-linguistic features that resulted in good performance. We plan to experiment with the character n-gram approach (Klein et al., 2003) and include gazetteer information.

References

Bogdan Babych and Anthony Hartley. 2003. Improving machine translation quality with automatic named entity recognition. In *Proceedings of Seventh International EAMT Workshop on MT and other language technology tools*, Budapest, Hungary.

Daniel M. Bikel, Scott Miller, Richard Schwartz, and Ralph Weischedel. 1997. Nymble: a high-performance learning name-finder. In *Proceedings of the fifth conference on Applied natural language processing*, pages 194–201, San Francisco, CA, USA. Morgan Kaufmann Publishers Inc.

Nancy Chinchor. 1997. Muc-7 named entity task definition. Technical Report Version 3.5, Science Applications International Corporation, Fairfax, Virginia.

Dan Klein, Joseph Smarr, Huy Nguyen, and Christopher D. Manning. 2003. Named entity recognition with character-level models. In *Proceedings of the seventh conference on Natural language learning at HLT-NAACL 2003*, pages 180–183, Morristown, NJ, USA. Association for Computational Linguistics.

John Lafferty, Andrew McCallum, and Fernando Pereira. 2001. Conditional random fields: Probabilistic models for segmenting and labeling sequence data. In *Proc. 18th International Conf. on Machine Learning*, pages 282–289. Morgan Kaufmann, San Francisco, CA.

Andrei Mikheev. 1997. Automatic rule induction for unknown-word guessing. *Comput. Linguist.*, 23(3):405–423.

Features	Precision	Recall	$F_{\beta=1}$
Ch_0	51.41%	9.19%	15.59
POS_0	46.32%	9.52%	15.80
$POS_0.Ch_0$	46.63%	9.69%	16.05
$W_{-3+3}.Ch_0$	59.08%	19.50%	29.32
$W_{-3+3}.POS_0$	58.43%	19.61%	29.36
$Ch_0.pre_n$	53.97%	24.76%	33.95
$POS_0.pre_n$	53.94%	24.93%	34.10
$POS_0.Ch_0.pre_n$	53.94%	25.32%	34.46
$POS_0.suf_n$	47.51%	29.36%	36.29
$POS_0.Ch_0.suf_n$	48.02%	29.24%	36.35
$Ch_0.suf_n$	48.55%	29.13%	36.41
$W_{-3+3}.POS_0.pre_n$	62.98%	27.45%	38.24
$W_{-3+3}.POS_0.Ch_0.pre_n$	62.95%	27.51%	38.28
$W_{-3+3}.Ch_0.pre_n$	62.88%	27.62%	38.38
$W_{-3+3}.POS_0.suf_n$	60.09%	30.53%	40.49
$W_{-3+3}.POS_0.Ch_0.suf_n$	59.93%	30.59%	40.50
$W_{-3+3}.Ch_0.suf_n$	61.18%	30.81%	40.98
$POS_0.Ch_0.pre_n.suf_n$	57.83%	34.57%	43.27
$POS_0.pre_n.suf_n$	57.41%	34.73%	43.28
$Ch_0.pre_n.suf_n$	57.80%	34.68%	43.35
$W_{-3+3}.Ch_0.pre_n.suf_n$	64.12%	34.34%	44.73
$W_{-3+3}.POS_0.pre_n.suf_n$	64.56%	34.29%	44.79
$W_{-3+3}.POS_0.Ch_0.pre_n.suf_n$	64.07%	34.57%	44.91

Table 2: Average Precision,Recall and $F_{\beta=1}$ measure for different language dependent feature combinations.

Features	Precision	Recall	$F_{\beta=1}$
w	57.05%	20.62%	30.29
pre	53.65%	23.87%	33.04
suf	47.75%	29.19%	36.23
w.pre	63.08%	27.56%	38.36
w.suf	60.93%	30.76%	40.88
pre.suf	57.94%	34.96%	43.61
w.pre.suf	64.80%	34.34%	44.89

Table 3: Average Precision,Recall and $F_{\beta=1}$ measure for different language independent feature combinations.

Diego Molla, Menno van Zaanen, and Daniel Smith. 2006. Named entity recognition for question answering. In *Proceedings of Australasian Language Technology Workshop 2006*, Sydney, Australia.

Avinesh PVS and Karthik G. 2007. Part-of-speech tagging and chunking using conditional random fields and transformation based learning. In *In Proceedings of SPSAL-2007 Workshop*.

Lance Ramshaw and Mitch Marcus. 1995. Text chunking using transformation-based learning. In David Yarovsky and Kenneth Church, editors, *Proceedings of the Third Workshop on Very Large Corpora*, pages 82–94, Somerset, New Jersey. Association for Computational Linguistics.

Antonio Toral, Elisa Noguera, Fernando Llopis, and Rafael Muñoz. 2005. Improving question answering using named entity recognition. In *Proceedings of the 10th NLDB congress*, Lecture notes in Computer Science, Alicante, Spain. Springer-Verlag.

Number of Words	Precision	Recall	$F_{\beta=1}$
2500	51.37%	9.47%	15.99
5000	64.74%	11.93%	20.15
7500	61.32%	13.50%	22.13
10000	66.88%	23.31%	34.57
12500	63.42%	27.39%	38.26
15000	63.55%	31.26%	41.91
17500	60.58%	30.64%	40.70
20000	58.32%	30.03%	39.64
22500	57.72%	29.75%	39.26
25000	59.33%	29.92%	39.78
27500	60.91%	30.03%	40.23
30000	62.77%	30.42%	40.98
32500	62.66%	30.64%	41.16
35000	62.08%	30.81%	41.18
37500	61.02%	30.87%	41.00
40000	61.60%	31.09%	41.33
42500	62.12%	32.44%	42.62
45000	62.70%	32.77%	43.05
47500	63.20%	32.72%	43.12
50000	64.29%	34.29%	44.72

Table 4: The effect of training data size on the performance of the NER.

Richard Tzong-Han Tsai. 2006. A hybrid approach to biomedical named entity recognition and semantic role labeling. In *Proceedings of the 2006 Conference of the North American Chapter of the Association for Computational Linguistics on Human Language Technology*, pages 243–246, Morristown, NJ, USA. Association for Computational Linguistics.

Nina Wacholder, Yael Ravin, and Misook Choi. 1997. Disambiguation of proper names in text. In *Proceedings of the fifth conference on Applied natural language processing*, pages 202–208, San Francisco, CA, USA. Morgan Kaufmann Publishers Inc.

Hanna M. Wallach. 2004. Conditional random fields: An introduction. Technical Report MS-CIS-04-21, University of Pennsylvania, Department of Computer and Information Science, University of Pennsylvania.

Author Index