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. ©2008 Asian Federation of Natural Language Processing

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-	AP-1			AP-2		
	Size	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	Without Majority Tag			With Majority Tag		
Corpus	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	Without Majority Tag			With Majority Tag		
Corpus	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.