Northern European Journal of Language Technology, 2009, Vol. 1, Article 2, pp 26–58
DOI 10.3384/nejlt.2000-1533.091226

Named Entity Recognition in Bengali: A Multi-Engine Approach

Asif Ekbal

University of Heidelberg, Germany
Department of Computational Linguistics
ekbal@cl.uni-heidelberg.de, asif.ekbal@gmail.com

Sivaji Bandyopadhyay

Jadavpur University, India
Department of Computer Science and Engineering
sivaji_cse_ju@yahoo.com, sbandyopadhyay@cse.jdvu.ac.in

Abstract

This paper reports about a multi-engine approach for the development of a Named Entity Recognition (NER) system in Bengali by combining the classifiers such as Maximum Entropy (ME), Conditional Random Field (CRF) and Support Vector Machine (SVM) with the help of weighted voting techniques. The training set consists of approximately 272K wordforms, out of which 150K wordforms have been manually annotated with the four major named entity (NE) tags, namely Person name, Location name, Organization name and Miscellaneous name. An appropriate tag conversion routine has been defined in order to convert the 122K wordforms of the IJCNLP-08 NER Shared Task on South and South East Asian Languages (NERSSEAL)[1] *data into the desired forms. The individual classifiers make use of the different contextual information of the words along with the variety of features that are helpful to predict the various NE classes. Lexical context patterns, generated from an unlabeled corpus of 3 million wordforms in a semi-automatic way, have been used as the features of the classifiers in order to improve their performance. In addition, we propose a number of techniques to post-process the output of each classifier in order to reduce the errors and to improve the performance further. Finally, we use three weighted voting techniques to combine the individual models. Experimental results show the effectiveness of the proposed multi-engine approach with the overall Recall, Precision and F-Score values of 93.98%, 90.63% and 92.28%, respectively, which shows an improvement of 14.92% in F-Score over the best performing baseline SVM based system and an improvement*

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

of 18.36% in F-Score over the least performing baseline ME based system. Comparative evaluation results also show that the proposed system outperforms the three other existing Bengali NER systems.

Keywords: Named Entity Recognition, Maximum Entropy, Conditional Random Field, Support Vector Machine, Weighted Voting, Bengali.

1 Introduction

Named Entity Recognition (NER) has important applications in almost all Natural Language Processing (NLP) application areas that include Information Retrieval, Information Extraction, Machine Translation, Question Answering and Automatic Summarization etc. The objective of NER is to identify and classify every word/term in a document into some predefined categories like person name, location name, organization name, miscellaneous name (date, time, number, percentage, monetary expressions etc.) and "none-of-the-above". The challenge in detection of NEs is that such expressions are hard to analyze using rule-based NLP because they belong to the open class of expressions, i.e., there is an infinite variety and new expressions are constantly being invented.

Nowadays, machine-learning (ML) approaches are popularly used in NER because these are easily trainable, adoptable to different domains and languages as well as their maintenance are also less expensive. Some of the representative machine-learning approaches used in NER are Hidden Markov Model (HMM) (BBN's IdentiFinder in Bikel et al. (1999)), Maximum Entropy (ME) (New York University's MENE in Borthwick (1999)) and Conditional Random Field (CRF) (Lafferty et al. 2001). Support Vector Machine (SVM) based NER system was proposed by Yamada et al. (2001) for Japanese. This system is an extension of Kudo's chunking system (Kudo and Matsumoto 2001) that gave the best performance at CoNLL-2000 shared task. The process of stacking and voting method for combining strong classifiers like boosting, SVM and TBL, on NER task can be found in Wu et al. (2003). Different methods are tested for combining the results of four systems in Florian et al. (2003) and they found that robust risk minimization worked best. Voting and bagging techniques are employed for combining classifiers in Munro et al. (2003). The work reported in this paper differs from the existing works in terms of the following points:

1. Useful features for NER in Bengali, a less computerized language, have been identified. We have used both the language independent features that are applicable to almost all the languages and the language dependent features extracted from the various language specific resources.

2. A method for semi-automatically generating lexical context patterns from an unlabeled corpus of 3 million wordforms has been reported. These lexical patterns are used as the features in each of the classifiers to improve their performance.

3. A number of post-processing techniques have been used in order to reduce the errors and to improve the performance of the classifiers.

4. The models are combined together into a final multi-engine NER system with the help of three weighted voting techniques. In the literature, a few works can be found where voting has been used in NER. However, these works are in non-Indian languages. The use of voting for NER in Indian languages in general and Bengali in particular is a very new concept. Moreover, the voting weights have been computed in a different way from that of the existing works.

Named Entity (NE) identification in Indian languages as well as in Bengali is more difficult and challenging as:

1. Unlike English and most of the European languages, Bengali lacks capitalization information, which plays a very important role in identifying NEs.

2. Indian person names are more diverse and a lot of these words can be found in the dictionary with specific meanings. For example, *kabitA* [Kabita] is a person name that can be found in the dictionary as a common noun with the meaning 'poem'.

3. Bengali is a highly inflectional language providing one of the richest and most challenging sets of linguistic and statistical features resulting in long and complex wordforms. For example, the person name *sachin* [root] can appear as *sachin**er*** [inflection:-*er*], *sachIn**ke*** [inflection:-*ke*], *sachIn**bAbu*** [inflection: -*bAbu*], *sachIn**dA*** [inflection:-*dA*] etc. The location name *kolkAtA* [root] can appear in different wordforms like *kolkAtA**r*** [inflection:-*r*], *kolkAtA**te*** [inflection:-*te*], *kolkAtA**i*** [inflection:-*i*] etc.

4. Bengali is a relatively free word order language. Thus, NEs can appear in any position of the sentence making the NER task more difficult.

5. Bengali, like other Indian languages, is a resource poor language. The annotated corpora, name dictionaries, good morphological analyzers, Part of Speech (POS) taggers etc. are not yet available in the required measure.

6. Although Indian languages have a very old and rich literary history, technological developments are of recent origin.

7. Web sources for name lists are available in English, but such lists are not available in Bengali forcing the use of transliteration.

A pattern directed shallow parsing approach for NER in Bengali is reported in Ekbal and Bandyopadhyay (2007a). A HMM based NER system for Bengali has been reported in Ekbal et al. (2007b), where additional contextual information has been considered during emission probabilities and NE suffixes are kept for unknown word handling. More recently, the works in the area of Bengali NER can be found in Ekbal et al. (2008), and Ekbal and Bandyopadhyay (2008a) with the CRF, and SVM approach, respectively. Other than Bengali, the works on Hindi can be found in Li and McCallum (2004) with CRF, Cucerzan and Yarowsky (1999) with a language independent method and in Kumar and Bhattacharyya (2006) using MEMM. As part of the IJCNLP-08 NER shared task, various works of NER

involving Indian languages using various approaches can be found in the proceedings of the IJCNLP-08 NER Shared Task on South and South East Asian Languages (NERSSEAL)[2].

2 Named Entity Recognition in Bengali

In terms of the native speakers, Bengali is the seventh most spoken language in the world, second in India and the national language of Bangladesh. We have used a Bengali news corpus (Ekbal and Bandyopadhyay 2008b) developed from the web-archive of a widely read Bengali newspaper for NER. The news corpus has been classified on geographic domain (International, National, State, District, Metro [Kolkata]) as well as on topic domain (Politics, Sports, Business). Out of 34 million wordforms of this corpus, 200K wordforms have been manually annotated with the four NE tags namely, *Person name*, *Location name*, *Organization name* and *Miscellaneous name*. The *Miscellaneous name* tag includes date, time, number, percentages, monetary expressions and measurement expressions. The annotation has been carried out by one of the authors and verified by a linguistic expert. The data has been collected from the International, National, State and Sports domains. In addition to this, we have used the IJCNLP-08 NERSSEAL shared task data that were tagged with the twelve NE tags. This data were collected mostly from the literature, agriculture and scientific domains. The fine-grained tagset consists of more tags than the four tags of CoNLL 2003 shared task on NER. The IJCNLP-08 NERSSEAL shared task tagset is shown in Table 1. The underlying reason to adopt this finer NE tagset was to use the NER system in various NLP applications, particularly in machine translation. One important aspect of the shared task was to identify and classify the maximal NEs as well as the nested NEs, i.e, the constituent parts of a larger NE. But, the training data were provided with the type of the maximal NE only. For example, *mahatmA gAndhi roDa* (Mahatma Gandhi Road) was annotated as location and assigned the tag 'NEL' even though *mahatmA* (Mahatma) and *gAndhi*(Gandhi) are NE title person (NETP) and person name (NEP), respectively. The task was to identify *mahatmA gAndhi roDa* as a NE and classify it as NEL. In addition, *mahatmA*, and *gAndhi* were to be recognized as NEs of the categories NETP (Title person), and NEP (Person name), respectively. Some NE tags are hard to distinguish in some contexts. For example, it is not always clear whether something should be marked as 'Number' or as 'Measure'. Similarly, 'Time' and 'Measure' is another confusing pair of NE tags. Another difficult class is 'Technical terms' and it is often confusing whether any expression is to be tagged as the 'NETE' (NE term expression) or not. For example, it is difficult to decide whether 'Agriculture' is 'NETE', and if no then whether 'Horticulture' is 'NETE' or not. The annotator may find the NETE tag as the most difficult class to tag correctly. Two other most ambiguous NE tags are 'NETE' (technical terms) and 'NETO' (title object).

An appropriate tag conversion routine, as shown in Table 2, has been defined in order to convert the shared task data into the forms tagged with the four NE tags. Here, the *Miscellaneous name* category includes only the number, time expressions and measurement expressions. The person name designations (NED), title-persons (NETP), title-objects (NETO), term expressions (NETE), abbreviations (NEA) and brand names (NEB) in the

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

Table 1: Named entity tagset for Indian languages (IJCNLP-08 NERSSEAL Shared Task Tagset)

NE Tag	Meaning	Example
NEP	Person name	*sachIna*/NEP, *sachIna ramesha tenDUlkara* / NEP
NEL	Location name	*kolkAtA*/NEL, *mahatmA gAndhi roDa* / NEL
NEO	Organization name	*yadabpUra bishVbidyAlYa*/NEO, *bhAbA eytOmika risArcha sentAra* / NEO
NED	Designation	*cheYArmAn*/NED, *sA.msada*/NED
NEA	Abbreviation	*bi e*/NEA, *ci em di a*/NEA, *bi je pi*/NEA, *Ai.bi.em*/ NEA
NEB	Brand	*fYAntA*/NEB
NETP	Title-person	*shrImAna*/NED, *shrI*/NED, *shrImati*/NED
NETO	Title-object	*AmericAn biUti*/NETO
NEN	Number	*10*/NEN, *dasha*/NEN
NEM	Measure	*tina dina*/NEM, *p.NAch keji*/NEM
NETE	Terms	*hidena markbha madela*/NETE, *kemikYAla riYYAkchYAna*/NETE
NETI	Time	*10 i mAgha 1402* / NETI, *10 ema*/NETI

shared task data are replaced by the NNE tags that denote other than NEs.

In order to properly denote the boundaries of the NEs, the four NE tags are further subdivided as shown in Table 3. It has been observed that some specific NE tags cannot be multiword, whereas some cannot appear as a single word entity. The NEN (NE number) tag generally appears as the single word entity but the NETI (NE time expressions) and NEM (NE measurements) tags can not be single word entities. In the output, these sixteen NE tags are directly mapped to the four major NE tags, namely *Person name*, *Location name*, *Organization name* and *Miscellaneous name*.

2.1 Our Approaches to NER

Natural Language Processing (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. Applying stochastic models to the NER problem requires large amount of annotated corpus in order to achieve reasonable performance. Simple HMMs do not work well when small amount of labeled data are used to estimate the model parameters. Incorporating diverse features in an HMM-based NE tagger is difficult and complicates the smoothing typically used in such taggers. In contrast, ME,

Table 2: Tagset mapping table

IJCNLP-08 tagset	Our tagset	Meaning
NEP	*Person name*	Single/multiword person name
NEL	*Location name*	Single/multiword location name
NEO	*Organization name*	Single/multiword organization name
NEN, NEM, NETI	*Miscellaneous name*	Single/multiword miscellaneous name
NEA, NED, NEB, NETP, NETE, NETO	*NNE*	Other than NEs

Table 3: Named entity tagset (B-I-E format)

Named Entity Tag	Meaning	Example
PER	Single word person name	*sachIna*/PER, *rabIndranAtha*/PER
LOC	Single word location name	*kolkAtA*/LOC, *mUmvAi*/LOC
ORG	Single word organization name	*infOsIsa*/ORG
MISC	Single word miscellaneous name	*10*/MISC, *dasha*/MISC
B-PER I-PER E-PER	Beginning, Internal or the End of a multiword person name	*sachIna*/B-PER *ramesha*/I-PER *tenDUlkara* /E-PER, *rabIndranAtha*/B-PER *ThAkUra*/E-PER
B-LOC I-LOC E-LOC	Beginning, Internal or the End of a multiword location name	*mahatmA*/B-LOC *gAndhi* /I-LOC *roDa* /E-LOC, *niU*/B-LOC *iYorka*/E-LOC
B-ORG I-ORG E-ORG	Beginning, Internal or the End of a multiword organization name	*yadabpUra* /B-ORG *bishVbidyAlYa*/E-ORG, *bhAbA* /B-ORG *eytOmika*/I-ORG *risArcha*/I-ORG *sentAra* /E-ORG
B-MISC I-MISC E-MISC	Beginning, Internal or the End of a multiword miscellaneous name	*10 i* /B-MISC *mAgha*/I-MISC *1402*/E-MISC, *10*/B-MISC *ema*/E-MISC
NNE	Other than NEs	*karA*/NNE, *jala*/NNE

CRF or SVM based method can deal with the diverse and morphologically complex features of the Indian languages more efficiently.

The present work deals with the development of a multi-engine NER system in Bengali by combining the ME, CRF and SVM frameworks. Two different models have been developed based on the SVM framework, one using **forward parsing** that parses from left to right and other using **backward parsing** that parses from right to left. Lexical context patterns, generated from an unlabeled corpus of 3 million wordforms, have been used as the features to improve the performance in each of the classifiers. The output of each classifier has been post-processed in order to improve the performance further. The individual system have been combined together into a single system using three weighted voting techniques.

We have used the C^{++} based Maximum Entropy package[3] and C^{++} based OpenNLP CRF^{++} package[4] for the ME and CRF based NER, respectively. The general purpose Limited Memory BFGS method as described in Malouf (2002) has been used for the estimation of ME parameters. For CRF based NER, the LBFGS method of Sha and Pereira (2003) has been used for parameter estimation.

Support Vector Machine (SVM) predicts the NEs based on feature information of words collected in a predefined window size while ME or CRF predicts them based on the information of the whole sentence. So, CRF can handle the NEs with outside tokens, which SVM always tags as NNE. A CRF has different characteristics from SVM, and is good at handling different kinds of data. SVMs have advantages over conventional statistical learning algorithms, such as Decision Tree, HMM and ME from the following two aspects:

1. SVMs have high generalization performance independent of dimension of feature vectors. Other algorithms require careful feature selection, which is usually optimized heuristically, to avoid overfitting.

2. SVMs can carry out their learning with all combinations of given features without increasing computational complexity by introducing the *Kernel function*. Conventional algorithms cannot handle these combinations efficiently.

In this work, the SVM system has been developed based on Joachims (1999) and Vapnik (1995), which perform classification by constructing an N-dimensional hyperplane that optimally separates data into two categories. Basically, SVMs are binary classifiers, thus we must extend SVMs to multi-class classifiers in order to classify more than two classes. There are two methods to extend a binary classification task to that of n classes. One is *one vs rest* and the other is the *pairwise* classification. In *one vs rest* strategy, the idea is to build n classifiers so as to separate one class from others. In *pairwise* classification, the idea is to build $\frac{n\times(n-1)}{2}$ classifiers considering all pairs of classes, and the final decision is given by their weighted voting. We have used the *pairwise* classifiers because of the following facts:

1. Generally, SVMs require $O(n^2) \sim O(n^3)$ training cost if the size of the training data is n. The training cost can be significantly reduced if the size of the training data for the individual binary classifier is small. Although *pairwise* classifiers tend to build a

[3]http://homepages.inf.ed.ac.uk/s0450736/software/maxent/maxent-20061005.tar.bz2
[4]http://crfpp.sourceforge.net

larger number of binary classifiers, the training cost required for *pairwise* method is much more tractable compared to the *one vs rest*.

2. Some experiments (Krebel 1999) report that a combination of *pairwise* classifiers performs better than the *one vs rest*.

We have used *YamCha* toolkit[5], an SVM based tool for detecting classes in documents and formulating the NER task as a sequential labeling problem. Here, in the present work, various degrees of the *polynomial kernel function* have been used. The TinySVM-0.07 [6] classifier has been used and it seems to be the best optimized among publicly available SVM toolkits. This means TinySVM-0.07 runs faster than the other SVM toolkits. Famous SVM-Light 3.50 (Joachims 1999) took 1.2 days to classify 569,994 vectors derived from 2 MB documents. That is, it runs at only 19 bytes/sec. But, TinySVM's classifier works at 92 bytes/sec.

During testing, it is possible that the classifier produces a sequence of inadmissible classes (e.g., B-PER followed by LOC). To eliminate such sequences, we define a transition probability between word classes $P(c_i|c_j)$ to be equal to 1 if the sequence is admissible, and 0 otherwise. The probability of the classes $c_1, c_2, \ldots, c_n$ assigned to the words in a sentence 'S' in a document 'D' is defined as follows:

$$P(c_1, c_2, \ldots, c_n|S, D) = \prod_{i=1}^{n} P(c_1|S, D) \times P(c_i|c_{i-1}) \quad (1)$$

where, $P(c_i|S, D)$ is determined by the ME/CRF/SVM classifier.

2.2 Named Entity Features

Feature plays a crucial role in any statistical model. Appropriate feature selection is very essential in the ME model in order to achieve high accuracy. It does not provide a method for automatic selection of given feature sets. Usually, heuristics are used for selecting effective features and their combinations. It is not possible to add arbitrary features in a ME framework as that will result in overfitting. On the other hand, CRF has the freedom to include arbitrary features, and the ability of feature induction to automatically construct the most useful feature combinations. Since, CRFs are log-linear models, and high accuracy may require complex decision boundaries that are non-linear in the space of original features, the expressive power of the models is often increased by adding new features that are conjunctions of the original features. For example, a conjunction feature might ask if the current word is in the person name list and the next word is an action verb '*bollen*'(told). One could create arbitrary complicated features with these conjunctions. However, it is infeasible to incorporate all possible conjunctions as these might result in overflow of memory as well as overfitting. SVM technique takes a strategy that maximizes the margin between the critical samples and the separating hyperplane. In particular, SVMs achieve high generalization even with training data of a very high dimension. Moreover, with the use of *Kernel*

[5]http://chasen-org/~taku/software/yamcha/

[6]http://cl.aist-nara.ac.jp/~taku-ku/software/TinySVM

function, SVMs can handle non-linear feature spaces, and carry out the training considering combinations of more than one feature. Detailed experiments have been carried out in order to find out the most suitable features for NER in Bengali.

We have considered different combinations from the following set for inspecting the best feature set for NER in each of the classifiers for Bengali:

$F = \{w_{i-m}, \ldots, w_{i-1}, w_i, w_{i+1}, \ldots, w_{i+m}, |prefix| \leq n, |suffix| \leq n$, previous NE tag(s), POS tags, First word, Length of the word, Infrequent word, Digit information, Position of the word, Gazetteer lists$\}$, where w_i is the current word, w_{i-m} is the mth previous word from the current position and w_{i+n} is the nth next word from the current position.

The set 'F' contains both language independent as well as language dependent features. The set of language independent features includes the context words, prefixes and suffixes of all the words, NE information of the previous word(s), first word, length of the word, digit information, infrequent word and POS information. Language dependent features include the set of known suffixes that may appear with the various NEs, clue words that help in predicting the location and organization names, words that help to recognize measurement expressions, designation words that help to identify person names, various gazetteer lists that include the first names, middle names, last names, location names, organization names, function words, weekdays and month names. Language independent NE features can be applied for NER in any language without any prior knowledge of that language. The lists or gazetteers are basically language dependent at the lexical level and not at the morphology or syntax level. Evaluation results suggest that the use of language dependent (or, specific) features is helpful to improve the performance of the NER system. In the resource-constrained Indian language environment, the non-availability of language specific resources and tools such as POS taggers, gazetteers, morphological analyzers etc. acts as a stimulant for the development of such resources to use in the NER systems. This leads to the necessity of *a priori* knowledge of the respective language.

2.2.1 Language Independent Named Entity Features

Following are the descriptions of the set of language independent features that have been applied to the NER task:

- Context words: Preceding and following words of a particular word can be used as the features. This is based on the observation that the surrounding words are very effective in the identification of NEs.

- Word suffix: Word suffix information is helpful to identify NEs. This is based on the observation that the NEs share some common suffix strings. This feature can be used in two different ways. The first and the naïve one is to use a fixed length (say, n) word suffix of the current and/or the surrounding word(s) as the features. These are actually the various fixed length character strings stripped from the word endings. For example, the fixed length word suffixes of the word '*sachIn*[Sachin]' are '*n*[n]' of length 1, '*In*[in]' of length 2 and '*chIn*[chin]' of length 3 etc. These fixed length suffixes are not linguistically meaningful suffixes. If the length of the corresponding word is less than or equal to $n - 1$ then the feature values are not defined and denoted by ND.

The feature value is also not defined (ND) if the token itself is a punctuation symbol or contains any special symbol or digit. The value of ND is set to 0. The second and the more helpful approach is to modify the feature as binary valued. Variable length suffixes of a word are matched with the predefined lists of useful suffixes for different classes of NEs. The variable length word suffixes include the list of NE suffixes that have been manually compiled by analyzing the various wordforms of the Bengali news corpus (Ekbal and Bandyopadhyay 2008b). It is to be noted that a morphological analyzer and/or a stemmer would be more effective to identify the linguistic suffixes of the various wordforms. The unavailability of these resources forces us to define the suffix features in this way.

- Word prefix: Word prefixes are also helpful and based on the observation that NEs share some common prefix strings. Fixed length word prefixes are basically the character strings of fixed lengths that are stripped from the beginning positions of the various wordforms. For example, the fixed length word prefixes of the word '*sachIn*[Sachin]' are '*sa*[s]' of length 1, '*sach*[sach]' of length 2 and '*sachI*[sachi]' of length 3 etc. This feature has been defined in a similar way as that of the fixed length suffixes. The fixed length prefixes are not linguistically meaningful prefixes. A morphological analyzer and/or a stemmer can be more effective to identify the meaningful prefixes.

- Named Entity Information: The NE tag(s) of the previous word(s) carry very effective information in determining the NE tag of the current word. This is the only dynamic feature in the experiment.

- First word: This feature is used to check whether the current token is the first word of the sentence or not. The first word of the sentence is most likely a NE.

- Position of the word: Position of the word in a sentence is a good indicator of NEs. Generally, verbs occur at the last position of the sentence.

- Length of the word: The training corpus has been analyzed to prepare a list containing each wordform along with its 'type' (whether NE or not) and length. It has been found that only 231 out of total 22,488 NEs in the training set have the lengths less than three. This observation leads us to decide that the very short words are rarely NEs.

- Infrequent word: The frequencies of the words in the training corpus have been calculated. A cut off frequency has been chosen in order to consider the words that occur more than the cut off frequency in the training corpus. Here, the cut off frequency is set to 10. A binary valued feature is defined to check whether the current word appears in this list of frequently occurring words. It is based on the observation that frequently occurring words are rarely NEs.

- 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, FourDigit, TwoDigit), combination of digits and punctuation symbols (e.g., ContainsDigitAndComma, ConatainsDigitAndPeriod), combination of digits and symbols (e.g., ContainsDigitAndSlash,

ContainsDigitAndHyphen, ContainsDigitAndPercentage). These binary valued features are helpful in recognizing miscellaneous NEs such as time expressions, monetary expressions, date expressions, percentages, numerical numbers etc.

- Part of Speech (POS) Information: We have used a CRF-based POS tagger (Ekbal et al. 2007a) that was originally developed with a POS tagset of 26 POS tags, defined for the Indian languages. The SVM based NER systems make use of the POS information extracted from this fine-grained POS tagger. However, for CRF and ME models, a coarse-grained POS tagger has been considered that has only three POS tags, namely Nominal, PREP (Postpositions) and Other. The postpositions have been considered as these often appear after the NEs.

The above set of features along with their descriptions are shown in Table 4.

2.2.2 Language Dependent Named Entity Features

Language dependent features have been identified based on the earlier experiments (Ekbal and Bandyopadhyay 2007a), (Ekbal and Bandyopadhyay 2007b) on NER. Additional NE features have been identified from the Bengali news corpus (Ekbal and Bandyopadhyay 2008b). Various gazetteers used in the experiment are presented in Table 5. These gazetteer lists have been used as the features in each of the classifiers. A number of these gazetteers have been also used to post-process the outputs of the classifiers to improve their performance further. Some of the gazetteers are briefly described as below:

- NE Suffix list (variable length suffixes): Variable length suffixes of a word are matched with the predefined lists of useful suffixes that are helpful to detect person (e.g., -*bAbU*[-babu], -*dA*[-da], -*di*[-di] etc.) and location (e.g., -*lYAnDa*[land], -*pUra*[pur], -*liYA*[-liya] etc.) names.

- Organization suffix word list: This list contains the words that are helpful to identify organization names (e.g., *kO.m*[Co.], *limiteDa*[Limited] etc.). These are also the part of organization names.

- Person prefix word list: This is useful for detecting person names (e.g., *shrImAna*[Mr.], *shrI*[Mr.], *shrImati*[Mrs.] etc.). Person name generally appears after these clue words.

- Common location word list: This list contains the words (e.g., *saranI*[Sarani], *rOda*[Road], *lena*[Lane] etc.) that are part of the multiword location names and usually appear at their end.

- Action verb list: A set of action verbs like *balena*[says], *balalena*[told], *ballO*[says], *sUnllO*[heared], *h.AsalO*[smiled] etc. often determines the presence of person names. Person names generally appear before the action verbs.

- Designation words: A list of common designation words (e.g., *netA*[leader], *sA.msada*[MP], *khelOYAra*[player] etc.) has been prepared manually. This helps to identify the position of person names.

Feature	Description
ContexT	$ContexT_i = w_{i-m}, \ldots, w_{i-1}, w_i, w_{i+1}, w_{i+n}$, where w_{i-m}, and w_{i+n} are the previous mth, and the next nth word
Suf	$\text{Suf}_i(n) = \begin{cases} \text{Suffix string of length } n \text{ of } w_i & \text{if } \lvert w_i \rvert \geq n \\ ND(=0) & \text{if } \lvert w_i \rvert \leq (n-1) \\ & \text{or } w_i \text{ is a punctuation symbol} \\ & \text{or } w_i \text{ contains any special symbol or digit} \end{cases}$
Pre	$\text{Pre}_i(n) = \begin{cases} \text{Prefix string of length } n \text{ of } w_i & \text{if } \lvert w_i \rvert \geq n \\ ND(=0) & \text{if } \lvert w_i \rvert \leq (n-1) \\ & \text{or } w_i \text{ is a punctuation symbol} \\ & \text{or } w_i \text{ contains any special symbol or digit} \end{cases}$
NE	NE_i = NE tag of w_i
FirstWord	$\text{FirstWord}_i = \begin{cases} 1, \text{if } w_i \text{ is the first word of a sentence} \\ 0, \text{Otherwise} \end{cases}$
Position	$\text{Position}_i = \begin{cases} 1, \text{if } w_i \text{ is the last word of a sentence} \\ 0, \text{Otherwise} \end{cases}$
CntDgt	$\text{CntDgt}_i = \begin{cases} 1, \text{if } w_i \text{ contains digit} \\ 0, \text{otherwise} \end{cases}$
FourDgt	$\text{FourDgt}_i = \begin{cases} 1, \text{if } w_i \text{ consists of four digits} \\ 0, \text{otherwise} \end{cases}$
TwoDgt	$\text{TwoDgt}_i = \begin{cases} 1, \text{if } w_i \text{ consists of two digits} \\ 0, \text{otherwise} \end{cases}$
CntDgtCma	$\text{CntDgtCma}_i = \begin{cases} 1, \text{if } w_i \text{ contains digit and comma} \\ 0, \text{otherwise} \end{cases}$
CntDgtPrd	$\text{CntDgtPrd}_i = \begin{cases} 1, \text{if } w_i \text{ contains digit and period} \\ 0, \text{otherwise} \end{cases}$
CntDgtSlsh	$\text{CntDgtSlsh}_i = \begin{cases} 1, \text{if } w_i \text{ contains digit and slash} \\ 0, \text{otherwise} \end{cases}$
CntDgtHph	$\text{CntDgtHph}_i = \begin{cases} 1, \text{if } w_i \text{ contains digit and hyphen} \\ 0, \text{otherwise} \end{cases}$
CntDgtPrctg	$\text{CntDgtPrctg}_i = \begin{cases} 1, \text{if } w_i \text{ contains digit} \\ \quad \text{and percentage} \\ 0, \text{otherwise} \end{cases}$
Infrequent	$\text{Infrequent}_i = I_{\{\text{Infrequent word list}\}}(w_i)$
Length	$\text{Length}_i = \begin{cases} 1, \text{if } w_i \geq 3 \\ 0, \text{otherwise} \end{cases}$
Position	$\text{Position}_i = \begin{cases} 1, \text{if } w_i \text{ is the last word of the sentence} \\ 0, \text{otherwise} \end{cases}$
POS	POS_i=POS tag of the current word

Table 4: Descriptions of the language independent features (Here, i represents the position of the current word and w_i represents the current word)

Gazetteer	Number of entries	Source
NE suffix	115	Manually prepared from the news corpus
Organization suffix	94	Manually prepared from the news corpus
Person prefix	245	Manually prepared from the news corpus
Middle name	1491	Semi-automatically prepared from the news corpus
Surname	5,288	Semi-automatically prepared from the news corpus
Common Location	147	Manually prepared from the news corpus
Action verb	221	Manually prepared from the news corpus
Designation words	947	Semi-automatically prepared from news corpus
First names	72,206	Semi-automatically prepared from the news corpus
Location name	5,285	Semi-automatically prepared from the news corpus
Organization name	2,225	Manually prepared from the news corpus
Common word	334	Manually prepared from the news corpus
Lexicon	128,000	Prepared in an unsupervised way from the news corpus
Month name	24	Manually prepared
Weekdays	14	Manually prepared
Measurement expressions	52	Manually prepared

Table 5: Different gazetteers used in the experiment

- Common word: Most of the Indian languages NEs appear in the dictionary with some other meanings. For example, the word *kamal* may be the name of a person but also appears in the dictionary with another meaning *padmma*[lotus], the name of a flower; the word *dhar* may be a verb or also can be the part of a person name. We have manually created a list, containing the words that can be NEs as well as valid dictionary words.

- Lexicon (128,000 entries): A lexicon (Ekbal and Bandyopadhyay 2008b) has been developed from the Bengali news corpus in an unsupervised way. This lexicon has been used to handle the unknown word problems. Our assumption is that the words

Feature	Description
NESuf	$NESuf_i = I_{\{\text{NE suffix list}\}}(w_i)$
OrgSuf	$OrgSuf_i = I_{\{\text{Organization suffix word list}\}}(w_i)$ $\bigvee I_{\{\text{Organization suffix word list}\}}(w_{i+1})$
ComLoc	$ComLoc_i = I_{\{\text{Common location list}\}}(w_i)$
ActVerb	$ActVerb_i = I_{\{\text{Action verb list}\}}(w_i)$ $\bigvee I_{\{\text{Action verb ist}\}}(w_{i+1})$
DesG	$DesG_i = I_{\{\text{Designation word list}\}}(w_{i-1})$
PerPre	$PerPre_i = I_{\{\text{Person prefix word list}\}}(w_{i-1})$
COM	$COM_i = I_{\{\text{Common word list}\}}(w_i)$
LocName	$LocName_i = I_{\{\text{Location name list}\}}(w_i)$
OrgName	$OrgName_i = I_{\{\text{Organization name list}\}}(w_i)$
FirstName	$FirstName_i = I_{\{\text{First name list}\}}(w_i)$
MidName	$MidName_i = I_{\{\text{Middle name list}\}}(w_i)$
SurName	$SurName_i = I_{\{\text{Sur name list}\}}(w_i) \bigvee I_{\{\text{Sur name list}\}}(w_{i+1})$
Funct	$Funct_i = I_{\{\text{Function word list}\}}(w_i)$
MonthName	$MonthName_i = I_{\{\text{Month name list}\}}(w_i)$
WeekDay	$WeekDay_i = I_{\{\text{Week day list}\}}(w_i)$
MeasureMent	$Measurement_i = I_{\{\text{Measurement word list}\}}(w_{i+1})$ $\bigvee I_{\{\text{Measurement list}\}}(w_{i+1})$

Table 6: Descriptions of the language dependent features (Here, i represents the position of the current word and w_i represents the current word)

appearing in the lexicon are rarely NEs.

The set of language dependent features are summarized in Table 6.

3 Unsupervised Lexical Pattern Learning from the Unlabeled Corpus

We propose a technique to generate the lexical context patterns from a portion of the unlabeled Bengali news corpus (Ekbal and Bandyopadhyay 2008b) containing 3 million wordforms. We also generate the context patterns from the annotated corpus of 272K wordforms in a semi-automatic way. Given a small seed examples and an unlabeled corpus, the algorithm can generate the lexical context patterns in a bootstrapping manner. The seed name serves as a *positive example* for its own NE class, *negative example* for other NE classes and *error example* for non-NEs. In the literature, unsupervised algorithms (bootstrapping

from seed examples and unlabeled data) have been discussed in Collins and Singer (1999), Cucerzan and Yarowsky (1999), and Cucerzan and Yarowsky (2002). Using a parsed corpus, the proper names that appear in certain syntactic contexts were identified and classified in Collins and Singer (1999). The procedures to identify and classify proper names in seven languages, learning character-based contextual, internal, and morphological patterns are reported in Cucerzan and Yarowsky (2002). This algorithm does not strictly require capitalization but recall was much lower for the languages that do not have case distinctions. Others such as Phillips and Riloff (2002) relied on structures such as appositive and compound nouns. Contextual patterns that predict the semantic class of the subject, direct object, or prepositional phrase object are reported in Riloff and Jones (1999) and Thelen and Riloff (2002). The technique to use the windows of tokens to learn contextual and internal patterns without parsing is described in Strzalkowski and Wang (1996) and Yangarber et al. (2002). An algorithm for unsupervised learning and semantic classification of names and terms is reported in Yangarber et al. (2002). They considered the *positive example* and *negative example* for a particular name class. We have developed an unsupervised algorithm that can generate the lexical context patterns form the unlabeled corpus. This work differs from the previous works in the sense that here we have also considered the patterns that yield *negative examples*. These *negative examples* can be effective to generate new patterns. Apart from *accuracy*, we have also considered the *relative frequency* of a pattern in order to decide its inclusion into the final set of patterns. The final lexical context patterns have been used as features of the classifiers.

1. **Seed list preparation**: The frequently occurring words have been collected from a part of this Bengali news corpus (Ekbal and Bandyopadhyay 2008b) and the annotated training set of 272K wordforms to use as the seeds. There are 123, 87 and 32 entries in the person, location and organization seed lists, respectively.

2. **Lexical pattern generation**: The unlabeled corpus is tagged with the elements from the seed lists. For example,
   <Person name> soniYA gAndhi[Sonia Gandhi] *< /Person name>*,
   <Location name> kolkAtA[Kolkata] *< /Location name>* and
   <Organization name> yadabpUra bishVbidyAlYa[Jadavpur University] *< /Organization name>*

   For each tag T inserted in the training corpus, the algorithm generates a lexical pattern p using a context window of maximum width 6 (excluding the tagged NE) around the left and the right tags, e.g.,
   $p = [l_{-3}l_{-2}l_{-1} < T > \ldots < /T > l_{+1}l_{+2}l_{+3}]$,
   where, $l_{\pm i}$ are the context of p.

   Any of $l_{\pm i}$ may be a punctuation symbol. In such cases, the width of the lexical patterns will vary. All these patterns, derived from the different tags of the training corpus, are stored in a Pattern Table (or, set P), which has four different fields namely, pattern *id* (identifies any particular pattern), pattern *example* (the pattern itself), pattern *type* (*Person name*/*Location name*/*Organization name*)and relative frequency (indicates the

number of times any pattern of a particular *type* appears in the entire training corpus relative to the total number patterns generated of that *type*). This table has 17,986 entries, out of which 13,031 patterns are distinct. We have also generated the context patterns by extracting the examples from the labeled training data of 272K wordforms and this yields 5,488 number of new patterns. Finally, the set P has 17,233 distinct patterns.

3. **Evaluation of patterns**: Every pattern p in the set P is matched against the same unannotated corpus. In a place, where the context of p matches, p predicts the occurrence of the left or right boundary of name. The POS information of the words as well as some linguistic rules and/or length of the entity have been used in detecting the other boundary of the entity. The extracted entity may fall in one of the following categories:

   (a) *positive example*: The extracted entity is of the same NE type as that of the pattern.

   (b) *negative example*: The extracted entity is of the different NE type as that of the pattern.

   (c) *error example*: The extracted entity is not at all a NE.

4. **Candidate pattern acquisition**: For each pattern p, we have maintained three different lists for the *positive*, *negative* and *error* examples. The *type* of the extracted entity is determined by checking whether it appears in any of the seed lists (person/location/organization); otherwise, its *type* is determined manually. The *positive* and *negative* examples are then added to the appropriate seed lists. Then, we compute the pattern's *accuracy* as follows:
   $accuracy(p) = \frac{|positive(p)|}{[|positive(p)|+|negative(p)|+|error(p)|]}$

   A threshold value of *accuracy* has been chosen and the patterns below this threshold value are discarded. A pattern is also discarded if its total *positive count* is less than a predetermined threshold value. The remaining patterns are ranked by their *relative frequency* values. The n top high frequent patterns are retained in the pattern set P and this set is denoted as *Accept Pattern*.

5. **Generation of new patterns**: All the *positive* and *negative* examples extracted by a pattern p in Step 4 can be used to generate further patterns from the same training corpus. Each new *positive* or *negative* instance (not appearing in the seed lists) is used to further tag the training corpus. We repeat steps 2-5 for each new NE until no new patterns can be generated. The threshold values of *accuracy*, *positive count* and *relative frequency* are chosen in such a way that in each iteration of the algorithm at least 5% new patterns are added to the set P. A newly generated pattern may be identical to a pattern that is already in the set P. In such case, the *type* and *relative frequency* fields in the Pattern Table (set, P) are updated accordingly. Otherwise, the newly generated pattern is added to the table with the *type* and *relative frequency* fields set properly.

Set	#of Sentences	#of Wordforms (approx.)	# of NEs	Avg. length of NE
Training	21,340	272K	22,488	1.5138
Development	3,367	50K	3,665	1.6341
Test	2,501	35K	3,178	1.6202

Table 7: Training, development and test set statistics

The algorithm terminates while no new NE is generated in two consecutive iterations. At the end of 17 iterations, there are 20,098 distinct entries in the set P.

4 Evaluation Results and Discussions

A portion of the Bengali news corpus (Ekbal and Bandyopadhyay 2008b) containing approximately 200K wordforms has been manually annotated with *Person name*, *Location name*, *Organization name* and *Miscellaneous name* NE tags with the help of *Sanchay Editor*[7], a text editor for the Indian languages. This manual annotation has been carried out by one of the authors and verified by an expert. We have also used the IJCNLP-08 NERSSEAL shared task data that was originally annotated with the fine-grained NE tagset of twelve tags. An appropriate tag conversion routine has been defined in order to convert the shared task data into the form tagged with the four NE tags. Out of 200K wordforms, 150K wordforms along with the IJCNLP-08 NERSSEAL shared task data have been used during training and the remaining 50K wordforms have been used as the development data. The system has been tested with a gold standard test set of 35K wordforms. Statistics of the training, development and test sets are presented in Table 7.

A number of experiments have been carried out taking the different combinations of the language independent features that include available words, context and orthographic word level features to identify the best-suited set of features in the ME, CRF and SVM frameworks for NER in Bengali. The SVM models that use **forward parsing** and **backward parsing** are denoted by SVM-F and SVM-B, respectively. The *baseline* models are developed with the language independent features that include context words, word prefixes and suffixes, dynamic NE information, first word, position of the word, length, infrequent words, digit features and the POS information. These *baseline* systems are referred to as the language independent versions. Detailed evaluation results of the *baseline* systems on the development set in terms of *F-Score* values are presented in Table 8 after removing the inadmissible tag sequences. Evaluation results have demonstrated that the ME based *baseline* system performs best (*F-Score*=73.32%) for the context window of size three (i.e., previous, current and next words), NE information of the previous word, POS information of the current word, prefixes and suffixes of length upto three characters of the current word along with the other

[7]http://Sourceforge.net/project/ nlp-sanchay

language independent features. The *baseline* CRF model has shown best performance (*F-Score*=75.71%) for the context window of size five (i.e., preceding two, current and the next two words), POS information of the current and previous words along with the other set of features similar to ME. The SVM-F based *baseline* system has performed best among the four models and has demonstrated the *F-Score* value of 76.3% for the context window of size five, NE information of the previous two words, POS information of the current, previous and the next words along with the other similar set of features as ME and CRF frameworks. The SVM-B model yields the *F-Score* value of 76.1% with the same set of features as that of the SVM-F model. For all the models, we have conducted several experiments by considering the various length word suffixes and/or prefixes and observed the best performance for the length upto three characters. A number of experiments have been also conducted with the suffixes and/or prefixes of the surrounding words. Results demonstrated that all the models perform best for the suffixes and/or prefixes of length upto three character of only the current word and inclusion of surrounding word suffixes and/or prefixes may degrade the overall performance of the system. In SVM models, a number of experiments have been conducted with the different degrees of the *polynomial kernel* functions and the models attain the highest performance with degree two. Also, the *pairwise* multi-class decision method yielded better performance compared to the *one vs rest* strategy. During all the experiments, we have observed that the word context, prefixes, suffixes, POS information, dynamic NE tag(s) and digit features are the most effective features for NER in each of the models. Overall evaluation results in terms of *Recall*, *Precision* and *F-Score* parameters are presented in Table 9.

Now, the various language dependent features extracted from the gazetteers are included in the feature set and the models are retrained. Detailed evaluation results on the development set are presented in Table 10. It is evident that all the gazetteers are not equally helpful to improve the performance of the models. Results show that the various suffixes that can occur with the different NEs are very effective to improve the overall performance of the system. We observe the improvement in the *F-Score* values by 0.85% in the ME model (2^{nd} experiment), 0.82% in the CRF model (1^{st} experiment), 0.93% in the SVM-F model (3^{rd} experiment) and 1.05% in the SVM-B model (3^{rd} experiment), respectively. We also observe the effectiveness of the use of organization suffix words, person prefix words, designations and common location words with the improvement in *F-Score* values by 0.94% in the ME model (5^{th} experiment), 0.59% in the CRF model (4^{th} experiment), 1.16% in the SVM-F model (6^{th} experiment) and 1.11% in the SVM-B model (6^{th} experiment). Other gazetteers improve the performance of the system though their effect are not very impressive. Finally, the models yield the *F-Score* values of 75.81%, 79.82%, 80.75% and 80.21% in the ME, CRF, SVM-F and SVM-B models, respectively. Thus, these are the improvement in the *F-Score* values by **2.49**%, **5.65**%, **4.45**% and **4.11**% in the ME, CRF, SVM-F and SVM-B, respectively, with the use of several language dependent (or, specific) features. The overall evaluation results in terms of *Recall*, *Precision* and *F-Score* parameters are reported in Table 11. Results suggest that adding all the available features may not be always helpful to achieve a reasonably high performance in the ME framework. This leads to conclude that careful feature selection has an important role to avoid overfitting in the ME model. On the other hand, CRF and SVM can include arbitrary set of features and can still avoid the

Experiment Number	Feature	SVM-F	SVM-B	CRF	ME
1	w_{i-1}, w_i, w_{i+1}, FirstWord, LEN	61.63	61.61	61.17	59.11
2	w_{i-2}+ Feature (1)+w_{i+2}	63.31	63.19	**63.31**	**61.01**
3	w_{i-2}+ Feature (1)	62.41	62.38	62.12	60.12
4	Feature (1)+w_{i+2}	62.35	62.29	62.03	60.07
5	w_{i-3}+Feature (2)+w_{i+3}	63.42	63.31	61.02	59.75
6	w_{i-3}+ Feature (2)	**63.53**	**63.48**	62.23	60.93
7	Feature (2)+w_{i+3}	63.43	63.39	62.01	60.81
8	Feature (2)+$Pre_4(w_i) + Suf_4(w_i)$	66.21	66.02	**65.92**	63.23
9	Feature (6) + $Pre_4(w_i) + Suf_4(w_i)$	66.73	66.64	64.89	62.84
10	Feature (1)+ $Pre_4(w_i) + Suf_4(w_i)$	64.02	64.01	63.29	64.02
11	Feature (6) + $Pre_3(w_i) + Suf_3(w_i)$	**67.81**	**67.73**	65.25	64.11
12	Feature (2)+$Pre_3(w_i) + Suf_3(w_i)$	66.21	66.02	**66.19**	63.55
13	Feature (1)+ $Pre_3(w_i) + Suf_3(w_i)$	64.73	64.34	64.02	**64.64**
14	Feature (11) + INFRQ + POSTION	**68.17**	**68.11**	67.37	65.18
15	Feature (12) + INFRQ+ POSTION	68.01	67.98	**67.87**	64.32
16	Feature (13) + INFRQ + POSTION	66.13	65.94	65.78	**65.39**
17	Feature (14) + Digit features + LEN	**70.91**	**70.34**	68.32	66.92
18	Feature (15) + Digit features + LEN	70.02	69.97	**69.82**	66.98
19	Feature (16) + Digit features + LEN	69.67	69.51	69.48	**67.56**
20	Feature (17) +$NE_{i-1} + FirstWord$	72.89	72.82	71.81	69.03
21	Feature (18) +$NE_{i-1} + FirstWord$	72.81	72.62	**72.98**	69.54
22	Feature (19) +$NE_{i-1} + FirstWord$	72.13	71.76	71.34	**70.45**
23	Feature (17)+$NE_{i-1} + NE_{i-2}$+ $FirstWord$	**73.30**	**73.10**	70.01	68.09
24	Feature (18)+$NE_{i-1} + NE_{i-2}$+ $FirstWord$	73.11	72.71	70.23	68.11
25	Feature (19)+ $NE_{i-1} + NE_{i-2}$+ $FirstWord$	72.89	72.11	69.98	68.45
26	Feature (21) +POS_i+POS_{i-1}	75.79	75.76	**75.71**	72.01
27	Feature (22) +POS_i	74.79	74.72	74.09	**73.32**
28	Feature (23) +POS_i+POS_{i-1}+POS_{i+1}	**76.30**	**76.10**	75.04	72.19

Table 8: Evaluation results on the development set in terms of the *F-Score* values for the *baseline* models

Model	Recall (in %)	Precision (in %)	F-Score (in %)
ME	73.57	73.07	73.32
CRF	75.97	75.45	75.71
SVM-F	77.14	75.48	76.30
SVM-B	77.09	75.14	76.10

Table 9: Overall evaluation results on the development set for the *baseline* models

overfitting problem.

4.1 Use of context patterns as features

High ranked patterns of the *Accept Pattern set* (discussed in Section 3) can be used as the features in each of the classifiers. Words in the left and/or the right contexts of person, location and organization names carry effective information that could be helpful for their identification. These words are used as the *trigger words*. A particular *trigger word* may appear in more than one pattern *type*. A feature 'ContextInformation' is defined as below by observing the three preceding and following words of the current word:

- If the window $W[-3, +3]$ (three words spanning to left and right) of the current word contains any *trigger word* of *Person name* then the feature value is set to 1.
- If the window $W[-3, +3]$ contains any *trigger word* of *Location name* then the feature value is set to 2.
- If the window $W[-3, +3]$ contains any *trigger word* of *Organization name* then the feature value is set to 3.
- If the window $W[-3, +3]$ contains any *trigger word* that appears in more than one NE type pattern then feature value is set to 4.
- Otherwise, the value of the feature is set to 0.

Experimental results of the system for the development set are presented in Table 12 by including the context features. Comparison between Table 11 and Table 12 show the effectiveness of the context features with the improvement in the *F-Score* values by **2.27**%, **3.08**%, **2.82**% and **3.28**% in the ME, CRF, SVM-F and SVM-B models, respectively.

It is to be noted that the difference in improvement between ME and other models in Table 11 varies from 2.49% to 4.45% over the *baseline* models. But, the difference in improvement between Table 11 and Table 12 varies between 2.27% and 3.28%. It is evident from Table 10 that language dependent features extracted from the gazetteers are effective to improve the performance in each of the models. Careful feature selection is essential in a ME framework and arbitrary inclusion of features may result in overfitting. On the other

Experiment Number	Feature	SVM-F	SVM-B	CRF	ME
1	Feature(26) of Table 8+$NESuf$	76.87	76.76	**76.53**	73.09
2	Feature(27) of Table 8+$NESuf$	75.59	75.52	75.07	**74.17**
3	Feature(28) of Table 8+$NESuf$	**77.23**	**77.15**	76.31	72.56
4	Feature (1) +$OrgSuf + PerPre$+ $DesG + ComLoc$	77.33	77.21	**77.12**	74.98
5	Feature (2) +$OrgSuf + PerPre$+ $DesG + ComLoc$	76.57	76.54	76.43	**75.11**
6	Feature (3) +$OrgSuf + PerPre$+ $DesG + ComLoc$	**78.39**	**78.26**	76.98	73.59
7	Feature (4) +$MidName$+ $SurName + ActVerb$	78.17	78.11	**78.09**	75.03
8	Feature (5) +$MidName$+ $SurName + ActVerb$	77.89	77.82	77.26	**75.14**
9	Feature (6) +$MidName$+ $SurName + ActVerb$	**78.91**	**78.82**	77.98	74.95
10	Feature (7) +$FirstName$+ $LocName + OrgName$	78.71	78.59	**78.62**	75.24
11	Feature (8) +$FirstName$+ $LocName + OrgName$	78.58	78.52	78.10	**75.38**
12	Feature (9) +$FirstName$+ $LocName + OrgName$	**79.55**	**79.46**	78.17	75.01
13	Feature (10) +$MonthName$+ $WeekDay$	79.91	79.81	**79.27**	75.23
14	Feature (11) +$MonthName$+ $WeekDay$	79.56	79.46	79.01	**75.67**
15	Feature (12) +$MonthName$+ $WeekDay$	**80.22**	**80.01**	79.11	75.13
16	Feature (13) +$MeasureMent$+ COM	80.52	80.13	**79.82**	75.51
17	Feature (14) +$MeasureMent$+ COM	80.29	80.14	79.51	**75.81**
18	Feature (15) +$MeasureMent$+ COM	**80.75**	**80.21**	79.62	75.21

Table 10: Results on the development set in terms of *F-Score* values including language dependent features

Model	Recall (in %)	Precision (in %)	F-Score (in %)
ME	76.09	75.53	75.81
CRF	79.03	80.62	79.82
SVM-F	81.37	80.14	**80.75**
SVM-B	81.29	79.16	80.21

Table 11: Overall evaluation results on the development set including language dependent features

Model	Recall (in %)	Precision (in %)	F-Score (in %)
ME	78.59	77.58	78.08
CRF	82.07	83.75	82.90
SVM-F	84.56	82.60	**83.57**
SVM-B	84.42	82.58	83.49

Table 12: Results on the development set including context features

hand, CRF and SVM can include arbitrary set of features and can avoid overfitting in an efficient way. Evaluation results of Table 10 show that the rate of performance improvement in the CRF and SVM based NER systems are better than that of the ME based NER system. Table 12 reports the results by considering the most frequently occurring context words as the features. These context words are basically the surrounding words around the NEs and almost equally important to improve the performance in each of the models. Thus, the difference in improvement between Table 11 and Table 12 are less compared to Table 11 and Table 9 (i.e., *baseline* model).

4.2 Post-processing Techniques

The training set is divided into 10 equal subsets. One subset is withheld for testing while the remaining 9 subsets are used for training. This process is repeated 10 times to yield the average *Recall*, *Precision* and *F-Score* values. This is called the 10-fold cross validation test. We perform 10-fold cross validation test for each of the models with the following feature combinations:

1. With the language independent features only (Language independent version).
2. With the language independent features and language dependent features (Language dependent version).
3. With the language independent features, language dependent features and context features.

ANOVA statistical analyses (Anderson and Scolve 1978) have been performed on the evaluation parameters, *Recall*, *Precision* and *F-Score*. Statistical tests show that the performance improvement in the language dependent version (reported in Table 11) over the *baseline* system (reported in Table 9), and the system developed with the language independent, dependent and context features (reported in Table 12) over the language dependent version (reported in Table 11) are statistically significant as in all the cases the significance values are less than 0.05. Statistical tests, ANOVA, also show that the performance improvement in the CRF over ME, SVM-B over CRF and SVM-F over CRF are statistically significant.

We have conducted error analysis for all the classifiers with the help of confusion matrices. Several post-processing techniques have been adopted in order to improve the performance in each of the classifiers. It has been observed that the SVM models have the highest tendency of assigning NE tags to the words that are actually not NEs. Evaluation results of Table 12 represent this fact. Both the SVM models perform better than CRF with less than 1% in the *F-Score* values whereas the *Precision* of CRF is more than 1% higher compared to the SVM models. Though SVM performs better than ME with more than 5% *F-Score* value, the rate of improvement in the *Precision* value is less than the *Recall* value. In ME model, a lot of NEs are not identified at all. CRF model also suffers from this problem. The most confusing pairs of classes in these two models are LOC vs. NNE, MISC vs. NNE, PER vs. NNE, E-ORG vs. NNE and B-MISC vs. MISC. On the other hand, the most confusing pairs are LOC vs. NNE, PER vs. NNE, MISC vs. NNE and E-ORG vs. NNE. Depending upon the errors involved in the models, we propose the following techniques to improve the overall performance of the classifiers.

1. Class Decomposition Technique for SVM: Unlike ME or CRF, SVM does not predict the NE tags to the constituent words depending upon the sentence. SVM predicts the class depending upon the labeled word examples only. If target classes are equally distributed, the *pairwise* method can reduce the training cost. Here, we have a very unlabeled class distribution with a large number of samples belonging to the class 'NNE' (other than NEs) (Table 7). This leads to the same situation like the *one-vs-rest* strategy. One solution to this unbalanced class distribution is to decompose the 'NNE' class into several subclasses effectively. Here, we have splitted the 'NNE' class according to the POS information of the word. That is, given a POS tagset $POSTAG$, we produce new $|POSTAG|$ classes, 'NNE-C'$|C\varepsilon$ POSTAG. So, we have 26 sub-classes which correspond to non-NE regions such as 'NNE-NN' (common noun), 'NNE-VFM' (verb finite main) etc. Experimental results of the SVM based systems including the class decomposition technique have been presented in Table 13. In the table, SVM-F[*baseline*] and SVM-B[*baseline*] denote the *baseline* models of the SVM-F and SVM-B, respectively. Results show the *Recall*, *Precision* and *F-Score* values of 87.09%, 86.73% and 86.91%, respectively in the SVM-F system and 87.03%, 85.98% and 86.5%, respectively in SVM-B system. Thus, these are the improvement of *F-Score* values by 10.61% and 10.4% in the SVM-F and SVM-B model, respectively, over the corresponding *baseline* model.

2. Post-processing with the heuristics for ME: The output of ME based NER system

Model	Recall (in %)	Precision (in %)	F-Score (in %)
SVM-F[*baseline*]	77.14	75.48	76.30
SVM-B[*baseline*]	77.09	75.14	76.10
SVM-F	87.09	86.73	**86.91**
SVM-B	87.03	85.98	86.50

Table 13: Results on the development set with the class decomposition technique

developed with the language independent, language dependent and context features has been post-processed with a set of heuristics to improve the performance further. Some of the heuristics are useful to improve the recall values, whereas some are effective to increase the precisions. Many of the heuristics are also helpful to identify the boundaries properly. Following are the set of heuristics.

- The NNE tag of a particular word is replaced by the appropriate NE tag, if that word appears somewhere in the output with that NE tag.
- If any word is tagged as B-XXX/I-XXX/E-XXX (XXX: PER/LOC/ORG/MISC) and the previous and next words are tagged as NNE then that word is assigned the NE tag of type XXX.
- The NNE tag of a word is replaced by the E-XXX if the previous word is already tagged as B-XXX.
- NNE tag of a word is replaced by B-XXX, if the next word is already tagged as E-XXX.
- If there is sequence B-XXX/I-XXX followed by XXX in the output, then the tag XXX is replaced by the E-XXX.
- If the sequence of tags is of the form XXX B-XXX1/I-XXX1/E-XXX1 NNE (XXXXXX1) for three consecutive words in the output, then the tag B-XXX1/ I-XXX1/ E-XXX1 is replaced by the XXX1.
- If current word is not tagged as B-XXX/ I-XXX/NNE but the following word is tagged as B-XXX/I-XXX/E-XXX then the current word is assigned the tag B-XXX.
- If the words, tagged as NNE, contain the variable length NE suffixes then the words are assigned the NE tags. The types of the NE tags are determined by the types of the suffixes (e.g., *Person name* tag is assigned if the suffix matches with the person name suffix.

  Evaluation results using the heuristics are shown in Table 14. In the table, ME [*baseline*] denotes the *baseline*. Results show the *Recall*, *Precision*, and *F-Score* values of 84.32%, 81.31% and 82.72%, respectively. This is the overall improvement of 9.40% *F-Score* value over the *baseline* model.

Model	Recall (in %)	Precision (in %)	F-Score (in %)
ME [*Baseline*]	73.57	73.07	73.32
CRF [*Baseline*]	75.97	75.45	75.71
ME	84.32	81.31	82.72
CRF	86.75	85.91	86.33

Table 14: Results on the development set with the n-best output strategy for CRF and heuristics for ME

- Post-processing with the n-best output for CRF: There are some inconsistent results in the CRF model. We have performed a post-processing step to correct these errors. The post-processing tries to assign the correct tag according to the n-best results for every sentence of the test set. We have considered the top 10 labeled sequences for each sentence with the confidence scores. Initially, we collect the NEs from the high confident results and then we reassign the tags for low confident results using the NE list. The procedure is given below:

  S is the set of sentences in the test set, i.e.,
  $S = s_1, s_2, \ldots, s_n$.
  R is set of n-best result (n=10) of S, i.e.,
  $R = r_1, r_2, \ldots, r_n$, where r_i is a set of n-best results of s_i.
  c_{ij} is the confidence score of r_{ij} , that is the j^{th} result in r_i.

  Creation of NE set from the high confident tags:
  for i = 1 to n
  {
  if ($r_{i0} >= 0.6$) then collect all NEs from r_{i0} and add to the set NESet;
  }
  Replacement:
  for i=1 to n
  {
  if ($r_{io} >= 0.6$) then Result(s_i) = r_{i0};
  else
  {
  TempResult(s_i) = r_{i0} ;
  for j=1 to m
  {
  if (NEs of r_{ij} are included in NESet)
  {
  Replace the NE tags of TempResult with these new tags;
  }
  Result(s_i)=TempResult(s_i);

Model	Recall (in %)	Precision (in %)	F-Score (in %)
ME	88.53	80.52	84.33
CRF	89.64	85.03	87.27
SVM-F	90.82	86.01	**88.35**
SVM-B	90.63	85.73	88.11

Table 15: Evaluation results using second confident tags

```
    }
  }
}
```

Evaluation results with the n-best output strategy have been presented in Table 14. In the table, CRF [*baseline*] specifies the CRF based *baseline* model. Results have demonstrated the *Recall*, *Precision*, and *F-Score* values of 86.75%, 85.91% and 86.33%, respectively. This is actually the improvement of 10.62% *F-Score* value over the corresponding *baseline* model.

- Second confident tags: If a word is tagged as NNE by any model and the confidence of the second best tag is greater than a threshold value then the second best tag is considered as the correct tag. We have conducted a number of experiments to find out the threshold values for the confidence. The threshold values are set to 0.45, 0.5, and 0.34 in the ME, CRF and SVM-F/B based systems, respectively. This post-processing technique is executed after the techniques 1 and 2. Evaluation results are presented in Table 15. It shows the loss in *Precision* value by less than 1% and the gain in *Recall* by more than 2.5% in each case. This results in the overall performance improvement for each of the models.
- Use of gazetteers and lexicon for handling unknown words: We have used the person, location and organization name gazetteers to handle the unknown words. These gazetteers do not include the ambiguous NEs, i.e., words that may or may not be NEs (e.g., *kamal*, *dhar* etc.). Lexicon of 128,000 entries has been also used to handle the unknown words. If the confidence of the unknown word is less than a predefined threshold value then its output tag is determined by checking the gazetteers. The NE tag is determined only if the threshold value of the second best tag is also below some threshold value (i.e., only when technique 3 fails). In some cases, an unknown word that is assigned the NE tag can also appear in the lexicon. The NE tag of the unseen word is changed to NNE, if its confidence is below a predetermined threshold value and the word is not found in the 'Common word' gazetteer list (contains the words that may be NEs as well as having the valid dictionary meanings). Appropriate threshold values have been determined within the range [0.3, 0.6] by observing the effects on the evaluation results. There are approximately 21% unknown words in the development set. Experimental results are presented in Table 16. Evaluation results show the effectiveness of the

Model	Recall (in %)	Precision (in %)	F-Score (in %)
ME	89.46	81.92	85.44
CRF	90.03	86.18	88.06
SVM-F	91.01	87.23	**89.08**
SVM-B	90.99	86.97	88.95

Table 16: Evaluation results using second confident tags

technique with the improvement in *Recall*, *Precision* and *F-Score* values in each of the models.

4.3 Voting Techniques

Voting is a very widely used term and generally used to combine a set of classifiers together into a final system. It is better to give importance to the outputs of all the classifiers rather than giving importance to the output of any particular classifier. It is possible to assign varying weights to the models in order to give more priority to one model than the others. The voting scheme becomes effective in order to improve the overall performance of any system.

In our experiments, in order to obtain higher performance, we have applied weighted voting to the four systems. But before applying weighted voting, we need to decide the weights to be given to the individual systems. We can obtain the best weights if we could obtain the accuracy for the 'true' test data. However, it is impossible to estimate them. Following four weighting methods have been used in the present experiments:

(a) Uniform weights (Majority voting): The same voting weight is assigned to all the systems. The combined system selects the classifications, which are proposed by the majority of the models. If four outputs are different then the output of the SVM-F system is selected.

(b) Cross validation *F-Score*: The training data is divided in to N portions. We employ the training by using N-1 portions, and then evaluate the remaining portion. This is repeated N times. In each iteration, we have evaluated the individual system following the similar methodology, i.e., by including the language independent features, language dependent features and context features followed by the same set of post-processing techniques. At the end, we get N *F-Score* values for each of the system. Final voting weight for a system is given by the average of these N *F-Score* values. Here, we have considered the value of N to be 10. Average *F-Score* values of the four systems are shown in Table 17. Table represents the overall average *F-Score* value (column 6) as well as the average *F-Scores* for person, location, organization and miscellaneous names (columns 2-5).

Model	F_P (in %)	F_L (in %)	F_O (in %)	F_M (in %)	F_{OV} (in %)
ME	87.12	81.05	79.16	95.51	86.11
CRF	92.51	84.17	83.05	98.71	89.27
SVM-F	93.28	85.09	84.14	98.98	90.59
SVM-B	93.06	85.05	84.06	98.92	89.98

Table 17: Results of the 10-fold cross validation test (F_P: Avg. *F-Score* for person, F_L: Avg. *F-Score* for location, F_O: Avg. *F-Score* for organization, F_M: Avg. *F-Score* for miscellaneous, F_{OV}: Overall average *F-Score*)

- Total F-Score: In the first method, we have assigned the overall average *F-Scores* (6^{th} column of Table 17), i.e., F_{OV}, of the 10-fold cross validation test as the weight for each system. The classification of any word is determined by the following function:
  $C(w) = \sum_{i=1}^{n} a_i \times Out(Model)$,
  where, $C(w)$ is the voted output tag to be assigned to the word w, a_i is the overall average *F-Score* of the i^{th} system (ME/CRF/SVM-F/SVM-B) and $Out(Model)$ is the output tag (one of the NE tags or NNE tag) predicted by the i^{th} system for the word w. Finally, the tag with the highest coefficient value (i.e., the largest value of a_i) is selected as the final output of the voted system.

- Tag F-Score: In the second method, we have assigned the average *F-Score* value of the individual tag as the weight. For example, if a particular word is assigned the tag 'PER' by any system then we assign the average *F-Score* value of person name as the weight. So, the output tag of any word w is determined by:
  $C(w) = \sum_{i=1}^{n} a_{ij} \times Out(Model)$,
  J=*Person name*/*Location name*/*Organization name*/*Miscellaneous name*/NNE
  where, $C(w)$ is the voted output tag to be assigned to the word w, a_{ij} is the average *F-Score* of the j^{th} tag of the i^{th} system.
  This actually depends on the $Out(Model)$, the output tag predicted by the i^{th} system for the word w. The weight of the NNE tag is set to (1-average of the *F-Scores* for person, location, organization and miscellaneous). Finally, the tag with the highest coefficient value is selected as the value of $C(w)$.

Now, the experimental results of the voted system are presented in Table 18. Evaluation results show that the system achieves the highest *Recall*, *Precision* and *F-Score* values for the voting scheme "Tag F-Score" which considers the individual tag *F-Score* value as the weight of the corresponding system. Effectiveness of voting can be observed by comparing the results between the Table 16 and Table 18. This shows an overall improvement of 6.56% over the least performing ME based system and 2.92% over the best performing SVM-F system in terms of *F-Score* values.

Voting Scheme	Recall	Precision	F-Score
Majority	93.15	89.33	91.20
Total F-Score	93.78	89.91	91.80
Tag F-Score	93.82	90.24	92.00

Table 18: Results of the voted system for the development set

Model	Baseline			Baseline + LD		
	Recall	Precision	F-Score	Recall	Precision	F-Score
ME	74.76	73.09	73.92	77.69	75.98	76.83
CRF	76.65	76.05	76.35	80.11	81.57	80.83
SVM-F	78.43	76.32	77.36	82.08	81.33	81.70
SVM-B	78.21	76.27	77.23	82.03	80.29	81.15

Table 19: Results on the test set

4.4 Experimental Results on the Test Set

The four systems are tested with a gold standard test set of 35K wordforms. Approximately, 25% of the NEs are unknown in the test set. Experimental results of the test set for the *baseline* models that use only language independent features and the models that use language independent as well as language dependent features are shown in Table 19. In the table, LD denotes the language dependent versions of the models. Evaluation results show the improvement in *Recall*, *Precision* and *F-Score* values with the use of various language dependent features extracted from the gazetteers. It is observed that the improvement in the ME model is less compared to the other models. This is because ME model cannot include arbitrary set of features like CRF and SVM.

The systems are now trained by including the contextual information and their results

Model	Recall	Precision	F-Score
ME	78.77	77.93	78.35
CRF	82.78	84.85	83.80
SVM-F	84.99	83.04	84.01
SVM-B	84.73	83.01	83.86

Table 20: Results on the test set by including context information

Model	Recall	Precision	F-Score
ME	89.67	82.09	85.71
CRF	90.54	86.79	88.63
SVM-F	91.55	87.75	89.61
SVM-B	91.37	87.56	89.42

Table 21: Results on the test set for the post-processed models

are presented in Table 20. Effectiveness of the contextual information is evident with the significant improvement in the overall performance over the language dependent versions. Approximately, an improvement of 3% (Table 19-Table 20) in the overall *F-Score* value is observed for each of the models.

Output of each system is then post-processed by a number of techniques. Results are presented in Table 21. The post-processing techniques are also very effective in order to increase the *Recall*, *Precision* as well as the *F-Score* values. Results of Table 21 show the improvement of 7.36%, 4.83%, 5.6% and 5.56% in the ME, CRF, SVM-F and SVM-B models, respectively, with the inclusion of different post-processing modules.

Now, the post-processed systems are combined together into a final system by applying three weighted voting approaches. Experimental results are presented in Table 22. Results show that the voting scheme that considers the *F-Score* value of the individual NE tag as the weight of a particular classifier, i.e., 'Tag F-Score' yields the best result among the three voting methods. The system shows the improvement of 6.57%, 3.65%, 2.67% and 2.86% *F-Score* values over the ME, CRF, SVM-F and SVM-B models, respectively. Evaluation results of Table 19 to Table 21 also demonstrate the fact that language dependent features extracted from the gazetteers, context features and post-processing can improve the performance significantly in each of the individual models. The improvement of 11.79%, 12.28%, 12.25% and 12.19% *F-Scores* values are observed in the ME, CRF, SVM-F and SVM-B models, respectively, over the corresponding *baseline* model. The multi-engine NER system has demonstrated the highest *Recall*, *Precision* and *F-Score* values of 93.98%, 90.63% and 92.28%, respectively. The *Recall*, *Precision* and *F-Score* values of the individual NE tag in the voted system are presented in Table 23. The voted system performs best for the *Miscellaneous name* tag followed by *Person name*, *Location name* and *Organization name* tags. Evaluation results suggest that the combination of several systems attain higher performance compared to any individual system.

4.5 Comparison with other Systems

Some of the existing Bengali NER systems, i.e., Ekbal et al. (2007b), Ekbal et al. (2007a), Ekbal et al. (2008) and Ekbal and Bandyopadhyay (2008a) have been trained

Voting Scheme	Recall	Precision	F-Score
Majority	93.21	89.75	91.45
Total F-Score	93.92	90.11	91.98
Tag F-Score	93.98	90.63	92.28

Table 22: Results of the voted system for the test set

NE tag	Recall	Precision	F-Score
Person name	96.12	93.26	94.67
Location name	89.03	87.62	88.32
Organization name	88.12	85.97	87.03
Miscellaneous name	99.15	98.89	99.02

Table 23: Results of the individual NE tag in the voted system (Tag F-Score)

and tested with the same datasets. Evaluation results are presented in Table 24. Results show the effectiveness of the proposed multi-engine NER system that outperforms the other existing Bengali NER systems based on HMM, CRF and SVM by the impressive margins of **19**%, **12.13**% and **11.99**% *F-Scores*, respectively. Thus, it can be decided that purely statistical approaches cannot yield very good performance always. Comparative evaluation results suggest that the contextual words along with their information and several post-processing methods can yield reasonably good performance in each of the individual models. Results also suggest that combination of several classifiers is more effective than the single classifier.

Model	Recall	Precision	F-Score
HMM (Ekbal et al. 2007b)	74.02	72.55	73.28
CRF (Ekbal et al. 2008)	80.02	80.21	80.15
SVM (Ekbal and Bandyopadhyay 2008a)	81.57	79.05	80.29
Voted System (proposed)	93.98	90.63	92.28

Table 24: Comparisons with other Bengali NER systems

5 Conclusion and Future Works

In this paper, we have reported a multi-engine NER system for Bengali by combining the outputs of the classifiers such as ME, CRF and SVM. Two different systems have been developed with the SVM approach based on the forward and backward parsing directions. Performance of the individual classifier has been improved significantly with the use of context patterns learned from an unlabeled corpus of 3 million wordforms and the various post-processing methodologies developed by observing the different kinds of errors involved in each classifier. All the four systems are then combined together into a final system by the three different weighted voting techniques. The voted system has demonstrated the overall *Recall*, *Precision* and *F-Score* values of 93.98%, 90.63% and 92.28%, respectively. This is actually an improvement of 18.63% in *F-Score* over the least performing *baseline* ME system and 14.92% in *F-Score* over the best performing *baseline* SVM based system.

Future works include investigating the methods that will be able to reduce the errors that still exist because of the abbreviated names and short names. We would also like to experiment with the other weighted voting methods. The system needs to be tested with the test data of other than newspaper domain. The appropriate unlabeled data selection may be also effective in order to improve the performance of the system.

References

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

Bikel, Daniel M., Richard L. Schwartz, and Ralph M. Weischedel. 1999. An Algorithm that Learns What's in a Name. *Machine Learning* 34(1-3):211–231.

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

Collins, M. and Y. Singer. 1999. Unsupervised Models for Named Entity Classification. In *Proceedings of the Joint SIGDAT Conference on Empirical Methods in Natural Language Processing and Very Large Corpora*.

Cucerzan, S. and David Yarowsky. 1999. Language Independent Named Entity Recognition Combining Morphological and Contextual Evidence. In *Proceedings of the 1999 Joint SIGDAT conference on EMNLP and VLC*. Washington, D.C.

Cucerzan, S. and D. Yarowsky. 2002. Language Independent NER using a Unified Model of Internal and Contextual Evidence. In *Proceedings of CoNLL 2002*, pages 171–175.

Ekbal, A. and S. Bandyopadhyay. 2007a. Lexical Pattern Learning from Corpus Data for Named Entity Recognition. In *Proceedings of 5th International Conference on Natural Language Processing (ICON)*, pages 123–128. India.

Ekbal, A. and S. Bandyopadhyay. 2007b. Pattern Based Bootstrapping Method for Named Entity Recognition. In *Proceedings of the 6th International Conference on Advances in Pattern Recognition (ICAPR)*, pages 349–355. World Scientific.

Ekbal, Asif and S. Bandyopadhyay. 2008a. Bengali Named Entity Recognition using Support Vector Machine. In *Proceedings of NERSSEAL, IJCNLP-08*, pages 51–58.

Ekbal, A. and S. Bandyopadhyay. 2008b. A Web-based Bengali News Corpus for Named Entity Recognition. *Language Resources and Evaluation Journal* 42(2):173–182.

Ekbal, Asif, Rejwanul Haque, and Sivaji Bandyopadhyay. 2007a. Bengali Part of Speech Tagging using Conditional Random Field. In *Proceedings of Seventh International Symposium on Natural Language Processing (SNLP-2007)*. Thailand.

Ekbal, Asif, R Haque, and S. Bandyopadhyay. 2008. Named Entity Recognition in Bengali: A Conditional Random Field Approach. In *Proceedings of the 3rd International Joint Conference on Natural Language Processing (IJCNLP08)*, pages 589–594.

Ekbal, A., S.K. Naskar, and S. Bandyopadhyay. 2007b. Named Entity Recognition and Transliteration in Bengali. *Named Entities: Recognition, Classification and Use, Special Issue of Lingvisticae Investigationes Journal* 30(1):95–114.

Florian, R., A. Ittycheriah, H. Jing, and T. Zhang. 2003. Named Entity Recognition through Classifier Combination. In *Proceedings of CoNLL-2003*, pages 168–171. Edmonton, Canada.

Joachims, T. 1999. *Making Large Scale SVM Learning Practical*, pages 169–184. MIT Press.

Krebel, Ulrich H.G. 1999. Pairwise Classification and Support Vector Machine. *In Advances in Kernel Methods* .

Kudo, Taku and Yuji Matsumoto. 2001. Chunking with Support Vector Machines. In *NAACL '01: Second Meeting of the North American Chapter of the Association for Computational Linguistics on Language Technologies*, pages 1–8. Association for Computational Linguistics.

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

Lafferty, John D., Andrew McCallum, and Fernando C. N. Pereira. 2001. Conditional Random Fields: Probabilistic Models for Segmenting and Labeling Sequence Data. In *Proceedings of the 18th International Conference on Machine Learning (ICML)*, pages 282–289.

Li, Wei and Andrew McCallum. 2004. Rapid Development of Hindi Named Entity Recognition using Conditional Random Fields and Feature Induction. *ACM Transactions on Asian Languages Information Processing* 2(3):290–294.

Malouf, R. 2002. A Comparison of Algorithms for Maximum Entropy Parameter Estimation. In *Proceedings of Sixth Conference on Natural Language Learning*, pages 49–55.

Munro, Robert, Daren Ler, and Jon Patrick. 2003. Meta-learning Orthographic and Contextual Models for Language Independent Named Entity Recognition. In *Proceedings of the Seventh Conference on Natural Language Learning at HLT-NAACL 2003*, pages 192–195. Association for Computational Linguistics.

Phillips, William and Ellen Riloff. 2002. Exploiting Strong Syntactic Heuristics and Co-training to Learn Semantic Lexicons. In *EMNLP '02: Proceedings of the ACL-02 Conference on Empirical Methods in Natural Language Processing*, pages 125–132. Association for Computational Linguistics.

Riloff, Ellen and Rosie Jones. 1999. Learning Dictionaries for Information Extraction by Multi-level Bootstrapping. In *AAAI '99/IAAI '99: Proceedings of the Sixteenth National Conference on Artificial Intelligence and the Eleventh Innovative Applications of Artificial Intelligence Conference*, pages 474–479. American Association for Artificial Intelligence. ISBN 0-262-51106-1.

Sha, Fei and Fernando Pereira. 2003. Shallow Parsing with Conditional Random Fields. In *Proceedings of NAACL '03*, pages 134–141. Canada.

Strzalkowski, Tomek and Jin Wang. 1996. A Self-learning Universal Concept Spotter. In *Proceedings of the 16th Conference on Computational Linguistics*, pages 931–936.

Thelen, Michael and Ellen Riloff. 2002. A Bootstrapping Method for Learning Semantic Lexicons using Extraction Pattern Contexts. In *EMNLP '02: Proceedings of the ACL-02 conference on Empirical Methods in Natural Language Processing*, pages 214–221.

Vapnik, Vladimir N. 1995. *The Nature of Statistical Learning Theory*. New York, NY, USA: Springer-Verlag New York, Inc. ISBN 0-387-94559-8.

Wu, Dekai, Grace Ngai, and Marine Carpuat. 2003. A Stacked, Voted, Stacked Model for Named Entity Recognition. In *Proceedings of the Seventh Conference on Natural Language Learning at HLT-NAACL 2003*, pages 200–203. Association for Computational Linguistics.

Yamada, Hiroyasu, Taku Kudo, and Yuji Matsumoto. 2001. Japanese Named Entity Extraction using Support Vector Machine. *In Transactions of IPSJ* 43(1):44–53.

Yangarber, Roman, Winston Lin, and Ralph Grishman. 2002. Unsupervised learning of generalized names. In *Proceedings of the 19th International Conference on Computational Linguistics*, pages 1–7.