World Academy of Science, Engineering and Technology
International Journal of Electrical and Computer Engineering
Vol:4, No:3, 2010

Named Entity Recognition using Support Vector Machine: A Language Independent Approach

Asif Ekbal and Sivaji Bandyopadhyay

Abstract—Named Entity Recognition (NER) aims to classify each word of a document into predefined target named entity classes and is now-a-days considered to be fundamental for many Natural Language Processing (NLP) tasks such as information retrieval, machine translation, information extraction, question answering systems and others. This paper reports about the development of a NER system for Bengali and Hindi using Support Vector Machine (SVM). Though this state of the art machine learning technique has been widely applied to NER in several well-studied languages, the use of this technique to Indian languages (ILs) is very new. The system makes use of the different contextual information of the words along with the variety of features that are helpful in predicting the four different named (NE) classes, such as *Person name*, *Location name*, *Organization name* and *Miscellaneous name*. We have used the annotated corpora of 122,467 tokens of Bengali and 502,974 tokens of Hindi tagged with the twelve different NE classes [1], defined as part of the IJCNLP-08 NER Shared Task for South and South East Asian Languages (SSEAL) [2]. In addition, we have manually annotated 150K wordforms of the Bengali news corpus, developed from the web-archive of a leading Bengali newspaper. We have also developed an unsupervised algorithm in order to generate the lexical context patterns from a part of the unlabeled Bengali news corpus. Lexical patterns have been used as the features of SVM in order to improve the system performance. The NER system has been tested with the gold standard test sets of 35K, and 60K tokens for Bengali, and Hindi, respectively. Evaluation results have demonstrated the recall, precision, and f-score values of 88.61%, 80.12%, and 84.15%, respectively, for Bengali and 80.23%, 74.34%, and 77.17%, respectively, for Hindi. Results show the improvement in the f-score by 5.13% with the use of context patterns. Statistical analysis, ANOVA is also performed to compare the performance of the proposed NER system with that of the existing HMM based system for both the languages.

Keywords—Named Entity (NE); Named Entity Recognition (NER); Support Vector Machine (SVM); Bengali; Hindi.

I. Introduction

NAMED Entity Recognition (NER) is an important tool in almost all Natural Language Processing (NLP) application areas such as 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, percentage and monetary expressions) and "none-of-the-above". The challenge in detection of named entities is that such expressions are hard to analyze using traditional NLP because they belong to the open class of expressions, i.e., there is an infinite variety and new expressions are constantly being invented.

The level of ambiguity in NER makes it difficult to attain human performance. There has been a considerable amount of work on NER problem, which aims to address many of these ambiguities, robustness and portability issues. There are two kinds of evidences that can be used in NER to solve the problems involved in NER. The first is the internal evidences found within the word and/or word string itself, while the second is the external evidence gathered from its context. During the last decade, NER has drawn more and more attention from the named entity (NE) tasks [1] [2] in Message Understanding Conferences (MUCs) [MUC6; MUC7]. The problem of correct identification of named entities (NEs) is specifically addressed and benchmarked by the developers of Information Extraction System, such as the GATE system [3]. NER also finds application in question-answering systems [4] and machine translation [5].

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. On the other hand, rule-based approaches lack the ability of coping with the problems of robustness and portability. Each new source of text requires significant tweaking of rules to maintain optimal performance and the maintenance costs could be quite steep.

The representative machine-learning approaches used in NER are HMM (BBN's IdentiFinder in [6] [7]), Maximum Entropy (ME)(New York University's MENE in [8] [9]), Decision Tree (New York University's system in [10] and SRA's system in [11]) and CRFs [12] [13]. There are other systems based on Support Vector Machine (SVM) [14], Nave Bayes [15], or the combination of the above [16]. As a ME model, MEME [9] makes use of diverse knowledge sources. Maximum Entropy conditional models like ME markov models [17] and CRFs [13] were reported to outperform the generative HMM models on several information extraction tasks.

NER can also be treated as a tagging problem, where each word in a sentence is assigned a label indicating whether it is part of a named entity and the entity type. Thus methods used for part of speech (POS) tagging can also be used for NER. The papers from the CoNLL-2002 shared task, which used such methods [18] [19] show results significantly lower than the best system [20]. However, Zhou and Su [21] have reported state of the art results on the MUC-6 and MUC-7 data using an HMM-based tagger.

Support Vector Machines (SVMs) based NER system was proposed by Yamada et al. [22] for Japanese. His system is an

Authors are with the Department of Computer Science and Engineering, Jadavpur University, Kolkata, West Bengal, India-700032, email: asif.ekbal@gmail.com, sivaji_cse_ju@yahoo.com

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

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

extension of Kudo's chunking system [23] that gave the best performance at CoNLL-2000 shared tasks. The other SVM-based NER systems can be found in [24] and [25]. Though this state of the art machine learning technique has been widely applied to a number of different languages and reported reasonably good accuracies, the use of this technique to the NER for Indian languages is very new.

India is a multilingual country with great cultural diversities. However, the works in the area of NER involving Indian languages have been started to appear very recently. Generally, named entity (NE) identification in Indian languages is difficult and challenging as:

1) Unlike English and most of the European languages, Indian languages lack capitalization information, which plays a very important role in identifying NEs.
2) Indian person names are more diverse compared to the other languages and a lot of these words can be found in the dictionary with specific meanings.
3) Indian languages are highly inflectional language providing one of the richest and most challenging sets of linguistic and statistical features resulting in long and complex word forms.
4) Indian languages are relatively free order.
5) Bengali and Hindi, like other Indian languages, are resource poor language - annotated corpora, name dictionaries, good morphological analyzers, 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 and Hindi forcing the use of transliteration.

There has been very little work in the area of NER in Indian languages. In Indian languages particularly in Bengali, the work in NER can be found in [26] [27] with the pattern directed shallow parsing approach. A semi-supervised technique based on lexical pattern learning from a newspaper corpus for NER in Bengali has been discussed in [26] that uses linguistic features. Two different NER systems, one using the lexical contextual patterns and the other using the linguistic features along with the same set of lexical contextual patterns, are reported in [27]. The statistical Hidden Markov Model(HMM)-based NER system has been reported in [28], where more contextual information has been considered during the emission probabilities. Recently, a SVM-based NER system has been reported in [29] for Bengali. Other than Bengali, a CRF-based Hindi NER system can be found in [30]. A HMM-based Hindi NER system has been described in [31].

In this paper, we have reported a NER system for the two most popular Indian languages, namely Bengali and Hindi. Bengali is the seventh most spoken language in the world, second in India and the national language of Bangladesh. Hindi is the third most spoken language in the world and the national language of India. The motivation behind the use of Support Vector Machine (SVM) framework is that it is more efficient than HMM or other conventional statistical methods to deal with the non-independent, diverse and overlapping features of the highly inflective Indian languages. The necessary tag conversion routine has been developed in order to map the fine-grained NE tagset , defined as part of the IJCNLP-08 NER Shared Task for SSEAL, to the four tags, namely *Person name*, *Location name*, *Organization name* and *Miscellaneous name*. The *Miscellaneous name* category includes the date, time, number, percentages, monetary expressions and measurement expressions. The system makes use of the different contextual information of the words along with the variety of orthographic word level features that are helpful in predicting the various NE classes. In this work, we have considered only the language independent features that are applicable to almost all the languages. Language independent features include the contextual words, prefix and suffix information of all the words in the training corpus, several digit features depending upon the presence and/or the number of digits in a token, length of the words and the frequency features of the words. Part of speech information of the words depends on some language specific phenomenon such as person, number, tense, gender etc. For example, gender information has a crucial role in Hindi but it is not an issue in Bengali. There are also some specific issues with the verbs in Bengali and Hindi. However, we consider the POS feature as a language independent feature as we have not used any language specific resources such as the lexicon, named entity recognizer or variable length word suffixes in order to handle the unknown word problems. Moreover, we have considered a coarse-grained POS tagger that has only three tags. A number of experiments have been carried out to find out the best-suited set of features for NER in Bengali and Hindi. An unsupervised algorithm has been used to generate the lexical context patterns from the unlabeled corpus of 10 million wordforms. These high ranked patterns have been used as the features of SVM and observed the significant improvement in the performance. We have also used these patterns to post-process the output of the SVM based system that helps to improve the recall values considerably, which in turn improve the overall performance of the system. The work reported in this paper differs from that of the work reported in [29] in terms of the following points:

1) This work deals with NER for two different Indian languages, namely Bengali and Hindi.
2) The system has been developed only with the help of language independent features. These features can be applied for NER in any language including the other Indian languages also.
3) An unsupervised technique has been developed in order to generate the lexical context patterns from the unlabeled Bengali news corpus. These patterns have been used as the features of SVM as well as a mean to post-process the output of the system. Experimental results show the effectiveness of these patterns with the impressive improvement in the overall performance of the system.
4) Only, 10-fold cross validation results were reported in [29]. But here, we have reported results with the gold standard test sets (open test) as well as on the 10-fold

cross validation tests.

5) ANOVA statistical analysis is performed to compare the performance between the proposed SVM-based and an existing HMM-based system for each of the languages.

The rest of this paper is organized as follows. The NER problems in two Indian languages along with the NE tagset and SVM framework have been described in Section II. The unsupervised context pattern learning method has been described in Section III. Section IV gives the descriptions of the language independent features that are applicable to both Bengali and Hindi as well to the other languages. Detailed evaluation results of the system for the development sets, test sets and the 10-fold cross validation tests are reported in Section V. Finally, Section VI concludes the paper.

II. Named Entity Recognition in Bengali and Hindi

Named entity recognition (NER) in Indian languages is more difficult and challenging in comparison to English and European languages. Applying stochastic models to the NER problem requires large amount of annotated corpus in order to achieve reasonable performance. Stochastic models have been applied to English, German and other European languages, for which large labeled data are available. The problem is difficult for Indian languages (ILs) due to the lack of such annotated large corpus. Indian languages are morphologically very rich and contain one of the most challenging set of linguistic and statistical features.

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, a ME [9] or a CRF [13] or a SVM [22] based method can deal with the diverse and overlapping features of Indian languages. In this work, we have applied the SVM framework to identify and classify NEs in Indian languages, particularly in Bengali and Hindi.

A. Named Entity Tagset

We have used the IJCNLP-08 NER shared task data that were tagged with the twelve NE tags. The tagset consists of more tags than the four tags of CoNLL 2003 shared task on NER. The underlying reason to adopt this finer NE tagset is to use the NER system in various NLP applications, particularly in machine translation. The IJCNLP-08 NER shared task tagset is shown in Table I. 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 part 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 if *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. In fact, this the most difficult class to identify. Other ambiguous tags are 'NETE' and 'NETO' (NE title-objects). We have considered only those NE tags that denote person name, location name, organization name, number expression, time expression and measurement expressions. The number, time and measurement expressions are mapped to belong to the *Miscellaneous* tag. Other tags of the shared task have been mapped to the 'other-than-NE' category. Hence, the tagset now becomes as shown in Table II.

In order to properly denote the boundaries of the NEs, the four NE tags are further subdivided as shown in Table III. 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*.

B. Support Vector Machine

Support Vector Machines (SVMs), first introduced by Vapnik [32] [33], are relatively new machine learning approaches for solving two-class pattern recognition problems. SVMs are well-known for their good generalization performance, and have been applied to many pattern recognition problems. In the field of natural language processing, SVMs are applied to text categorization, and are reported to have achieved high accuracy without falling into over-fitting even though with a large number of words taken as the features [34] [35]. Suppose, we have a set of training data for a two-class problem: $\{(\mathbf{x}_1, y_1), \ldots, (\mathbf{x}_N, y_N)\}$, where $\mathbf{x}_i \in R^D$ is a feature vector of the i-th sample in the training data and $y \in \{+1, -1\}$ is the class to which $\mathbf{x}_i$ belongs. In their basic form, a SVM learns a linear hyperplane that separates the set of positive examples from the set of negative examples with *maximal margin* (the margin is defined as the distance of the hyperplane to the nearest of the positive and negative examples). In basic SVMs framework, we try to separate the positive and negative examples by hyperplane written as:

$$(\mathbf{w}.\mathbf{x}) + b = 0 \qquad \mathbf{w} \in \mathbf{R}^n, b \in \mathbf{R}.$$

SVMs find the "optimal" hyperplane (optimal parameter $\overline{w}, b)$ which separates the training data into two classes precisely. The linear separator is defined by two elements: a weight vector $\mathbf{w}$ (with one component for each feature), and a bias b which stands for the distance of the hyperplane to the origin. The classification rule of a SVM is:

$$sgn(f(\mathbf{x}, \mathbf{w}, b)) \tag{1}$$

$$f(\mathbf{x}, \mathbf{w}, b) = < \mathbf{w}.\mathbf{x} > + b \tag{2}$$

being $\mathbf{x}$ the example to be classified. In the linearly separable case, learning the maximal margin hyperplane $(\mathbf{w}, b)$ can be stated as a convex quadratic optimization problem with a

TABLE I
NAMED ENTITY TAGSET FOR INDIAN LANGUAGES (IJCNLP-08 NER 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

TABLE II
TAGSET USED IN THIS WORK

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

TABLE III
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

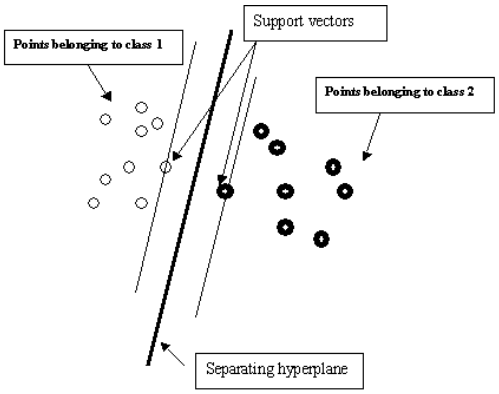

Fig. 1. Example of a 2-dimensional SVM

unique solution: *minimize* $||\mathbf{w}||$, *subject to the constraints* (one for each training example):

$$y_i(<\mathbf{w}.x_i> +b) \geq 1 \quad (3)$$

See an example of a 2-dimensional SVM in Figure 1.

The SVM model has an equivalent dual formulation, characterized by a weight vector α and a bias b. In this case, α contains one weight for each training vector, indicating the importance of this vector in the solution. Vectors with non null weights are called *support vectors*. The dual classification rule is:

$$f(\mathbf{x}, \alpha, b) = \sum_{i=1}^{N} y_i \alpha_i <\mathbf{x}_i.\mathbf{x}> +b \quad (4)$$

The α vector can be calculated also as a quadratic optimization problem. Given the optimal α^* vector of the dual quadratic optimization problem, the weight vector $\mathbf{w}^*$ that realizes the maximal margin hyperplane is calculated as:

$$\mathbf{w}^* = \sum_{i=1}^{N} y_i \alpha_i^* \mathbf{x}_i \quad (5)$$

The b^* has also a simple expression in terms of $\mathbf{w}^*$ and the training examples $(\mathbf{x}_i, y_i)_{i=1}^N$.

The advantage of the dual formulation is that efficient learning of non-linear SVM separators, by introducing *kernel functions*. Technically, a *kernel function* calculates a dot product between two vectors that have been (non linearly) mapped into a high dimensional feature space. Since there is no need to perform this mapping explicitly, the training is still feasible although the dimension of the real feature space can be very high or even infinite.

By simply substituting every dot product of $\mathbf{x}_i$ and $\mathbf{x}_j$ in dual form with any *kernel function* $K(\mathbf{x}_i, \mathbf{x}_j)$, SVMs can handle non-linear hypotheses. Among the many kinds of *kernel functions* available, we will focus on the d-th polynomial kernel:

$$K(\mathbf{x}_i, \mathbf{x}_j) = (\mathbf{x}_i.\mathbf{x}_j + 1)^d$$

Use of d-th polynomial kernel function allows us to build an optimal separating hyperplane which takes into account all combination of features up to d.

The SVMs have advantage over conventional statistical learning algorithms, such as Decision Tree, Hidden Markov Models, Maximum Entropy Models, from the following two aspects:

1) SVMs have high generalization performance independent of dimension of feature vectors. Conventional algorithms require careful feature selection, which is usually optimized heuristically, to avoid overfitting. So, it can more effectively handle the diverse, overlapping and morphologically complex Indian languages.
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, thus, we usually select important combinations heuristically with taking the trade-off between accuracy and computational complexity into consideration.

We have developed our system using SVM [34] [32], which perform classification by constructing an N-dimensional hyperplane that optimally separates data into two categories. Our general NER system includes two main phases: training and classification. The training has been carried out by YamCha [3] toolkit, an SVM based tool for detecting classes in documents and formulating the NER task as a sequential labeling problem. For classification, We have used TinySVM-0.07 [4] classifier that seems to be the best optimized among publicly available SVM toolkits. Here, the pairwise multi-class decision method and the *polynomial kernel function* have been used.

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 assigned to the words in a sentence 's' in a document 'D' is defined as follows:

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

where, $P(c_i|s, D)$ is determined by the SVM classifier.

III. Unsupervised Lexical Pattern Learning from the Unlabeled Bengali Corpus

We have developed a method to generate the lexical context patterns from a portion, containing 10 million wordforms, of the unlabeled Bengali news corpus [36] and annotated corpus of 272K wordforms in an unsupervised 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 [37], [38], and [39]. Using

[3]http://chasen-org/ taku/software/yamcha/
[4]http://cl.aist-nara.ac.jp/ taku-ku/software/TinySVM

a parsed corpus, the proper names that appear in certain syntactic contents were identified and classified in [37]. The procedures to identify and classify proper names in seven languages, learning character-based contextual, internal, and morphological patterns are reported in [39]. This algorithm does not strictly require capitalization but recall was much lower for the languages that do not have case distinctions. Others, such as [40] relied on structures such as appositives and compound nouns. Contextual patterns that predict the semantic class of the subject, direct object, or prepositional phrase object are reported in [41] and [42]. The techniques to use the windows of tokens to learn contextual and internal patterns without parsing are described in [43] and [44]. The technique reported in [44] enabled discovery of generalized names embedded in larger noun groups. An algorithm for unsupervised learning and semantic classification of names and terms is reported in [44]. 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 of the other categories. 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 set of lexical context patterns have been used as features of the classifier as well as to post-process the output of the system.

A. Seed list preparation

Three different seed lists of person names, location names and organization names have been created.

The most frequently words automatically extracted from the reporter, location and agency tags of the Bengali news corpus [36], developed from the web archive of a leading Bengali newspaper, are treated as the initial seed data and put into the appropriate seed lists. The location, reporter, and agency tags present in the web pages have been automatically NE tagged. These tags are useful to identify only the NEs that appear in some fixed places of the newspaper. These tags can not identify the NEs that appear in the actual news body. In addition to these extracted words, the NEs retrieved from the NE tagged corpus of 272,467 tokens are also added to the appropriate seed lists. There are 123, 87, and 32 entries in the person, location, and organization seed lists, respectively. We have not prepared any seed list for miscellaneous entities as these can be recognized by some fixed patterns.

B. Lexical pattern generation

The unlabeled corpus is tagged with the elements from the seed lists. For example,
<Person name> sonia gandhi < /Person name>,
<Location name> kolkata < /Location name>
<Organization name> jadavpur viswavidyalya < /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 31,986 entries, out of which 23,031 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 patterns. Finally, the set P has 25,233 distinct patterns.

C. 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. A CRF-based part of speech tagger [45] has been used to tag the unannotated corpus. A regular expression $NNPC^*NNP^+$ is used to detect the boundary of the NE, where NNPC denotes the initial words of a compound proper noun and NNP denotes the last word of the compound proper noun. The extracted entity may fall in one of the following categories:

1) *positive example*: The extracted entity is of the same NE *type* as that of the pattern.
2) *negative example*: The extracted entity is of the different NE *type* as that of the pattern.
3) *error example*: The extracted entity is not at all a NE.

D. 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 added to the appropriate seed lists. Then, we compute the pattern's *accuracy* as follows:
$accuracy(p) = |positive(p)|/[|positive(p)|+|negative(p)|+|error(p)|]$

A threshold value of *accuracy* has been chosen and the patterns below this threshold values 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*.

E. Generation of new patterns

All the *positive* and *negative* examples extracted by a pattern p in Step D 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 B-E 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. At the end of 17 iterations, there are 27,098 distinct patterns in the set P.

IV. Named Entity Features

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

F=$\{w_{i-m}, \ldots, w_{i-1}, w_i, w_{i+1}, \ldots w_{i+n}, |\text{prefix}| \leq n, |\text{suffix}| \leq n$, NE tag(s) of previous word(s), POS tag(s) of the current and/or the surrounding word(s), First word, Length of the word, Digit information, Rare word feature$\}$.

The set 'F' contains only the language independent features. These features are applicable for NER in any language including the Indian languages.

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

1) Context word feature: 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.
2) Word suffix: Word suffix information is helpful to identify NEs. This is based on the observation that the NEs share some common suffixes. A fixed length (say, n) word suffix of the current and/or the surrounding word(s) can be treated as feature. 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. We have defined this feature in this way due to the unavailability of stemmer or morphological analyzers.
3) Word prefix: Word prefixes are also helpful and based on the observation that NEs share some common prefix strings. This feature has been defined in a similar way as that of the fixed length suffixes. The use of stemmer or morphological analyzer may be more effective to extract this feature.
4) Named Entity Information: The NE tag(s) of the previous word(s) has been used as the only dynamic feature in the experiment.
5) First word: This is used to check whether the current token is the first word of the sentence or not. Though Bengali and Hindi are relatively free order languages, the first word of the sentence is most likely a NE as it appears in the subject position most of the time.
6) Digit features: Several binary valued digit features have been defined depending upon the presence and/or the number of digits in a token (e.g., ContainsDigit [token contains digits], FourDigit [token consists of four digits], TwoDigit [token consists of two digits]), combination of digits and punctuation symbols (e.g., ContainsDigitAndComma [token consists of digits and comma], ConatainsDigitAndPeriod [token consists of digits and periods]), combination of digits and symbols (e.g., ContainsDigitAndSlash [token consists of digit and slash], ContainsDigitAndHyphen [token consists of digits and hyphen], ContainsDigitAndPercentage [token consists of digits and percentages]). These binary valued features are helpful in recognizing miscellaneous NEs, such as time expressions, measurement expressions and numerical numbers etc.
7) 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 less than the cut off frequency in the training corpus. The cut off frequencies are set to 10, and 20 for Bengali, and Hindi, respectively. A binary valued feature 'RareWord' is defined to check whether the current token appears in this list or not. This feature has been included as the frequently occurring words are rarely named entities.
8) Length of a word: This binary valued feature is used to check whether the length of the current word is less than three or not. This is based on the observation that the very short words are rarely NEs.
9) Part of Speech information: For POS tagging, we have used a CRF-based POS tagger [45] that was originally developed with the help of a tagset of 26 different POS tags [5], defined for the Indian languages. This POS tagger had an accuracy of 90.2%. We have used a coarse-grained POS tagger that has only three tags, namely Nominal, PREP (Postposition) and Other. Postpositions have been considered as these often occur after the NEs. The language specific resources such as the variable length word suffixes, lexicon [46] and the NER system [29] were not considered for handling the unknown word

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

problems in POS tagging. The same POS tagger has been evaluated for Hindi. The POS tagger has been trained with the Bengali and Hindi data, obtained from the NLPAI_Contest06 [6] and SPSAL2007 competition [7].

The above set of language independent features along with their descriptions are shown in Table IV.

V. Experimental Results

We have used the annotated corpora of 122,467 tokens of Bengali and 502,974 tokens of Hindi tagged with the twelve different NE classes [8], defined for the IJCNLP-08 NER Shared Task for South and South East Asian Languages (SSEAL) [9]. An appropriate tag conversion routine has been developed in order to convert the twelve NE tagged corpora to the forms, tagged with the four NE tags. In addition to it, a Bengali news corpus [36], developed from the archive of a leading Bengali newspaper available in the web, has been also used. This corpus contains approximately 34 million wordforms. A portion of this corpus containing 150K wordforms have been manually annotated with four NE tags, namely *Person name*, *Location name*, *Organization name* and *Miscellaneous name*. This annotations were carried out with the help of *Sanchay Editor* [10], a text editor for the Indian languages. This manually tagged corpus is added to that of the 122,467 tokens and thus the resultant training set for Bengali consists of 272,467 tokens. A subset of each training set has been selected as the development set to identify the best set of features for each of the languages. Out of 272,467, and 502,974 tokens, 30K, and 50K tokens have been selected as the development sets for Bengali, and Hindi, respectively. The rest, i.e., 242,467, and 452,974 tokens are used as the training sets of Bengali, and Hindi, respectively. System has been tested with the gold standard test sets of 35K, and 60K tokens of Bengali, and Hindi, respectively.

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

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

In this model, each word in the test data is assigned the NE tag which occurred most frequently for that word in the training data.

The performance of the system has been evaluated in terms of the standard recall, precision and f-score parameters as defined below:

$$\text{Recall} = \frac{\text{NEs retrieved by the system}}{\text{NEs present in the test set}} \times 100$$

$$\text{Precision} = \frac{\text{NEs correctly retrieved by the system}}{\text{NEs retrieved by the system}} \times 100$$

$$\text{F-Score} = \frac{2 \times \text{Recall} \times \text{Precision}}{\text{Recall} + \text{Precision}} \times 100$$

[6] http://ltrc.iiitnet/nlpai_contest06/
[7] http://shiva.iiit.ac.in/SPSAL2007/
[8] http://ltrc.iiit.ac.in/ner-ssea-08/index.cgi?topic=3
[9] http://ltrc.iiit.ac.in/ner-ssea-08
[10] Sourceforge.net/project/nlp-sanchay

A. Experimental results on the development sets

Forty nine and thirty five different experiments were conducted taking the different combinations from the set of features that are applicable to both Bengali and Hindi. From our empirical analysis, we found that the following combination gives the best result for the development sets of Bengali and Hindi.

F=[$w_{i-3}w_{i-2}w_{i-1}w_iw_{i+1}w_{i+2}$, |prefix| ≤ 3, |suffix| ≤ 3, NE information of the previous two words, POS information of the window $[-1, -1]$, FirstWord, Length, RareWord, Digit features]

The meanings of the notations, used in the experimental results, are defined below:
cw, pw, nw: Current, previous and the next word; pwi, nwi: Previous and the next ith word; pre, suf: Prefix and suffix of the current word; pt: NE tag of the previous word; pti: NE tag of the previous ith word; psuf, ppre: Suffix and prefix of the previous word; nsuf, npre: Suffix and prefix of the next word.

Evaluation results of the system for the development set in terms of f-score (FS) are presented in Table V for Bengali. It is observed from Table V (2nd-6th rows) that word window $[-3, +2]$ gives the best result (5th row) with the 'FirstWord' and 'Length' features. The use of 'RareWord' feature increases the f-score value by 0.84% (5th and 7th rows). The intuition of using this feature is that the most frequently occurring words in the training corpus have the possibility of not being the NEs. The NE information of the word is the only dynamic feature in the experiment and it has the important role in the overall performance of the system. Evaluation results (7th and 8th rows) show that the performance of the system increases by 2.29% with the addition of NE information of the previous word only. It is also evident (8th-10th rows) that the NE information of the previous two words are the most effective and have demonstrated the f-score value of 71.92%. It is indicative from the results (9th and 11th rows) that inclusion of prefix and suffix features of all the words are very effective in improving the performance of the system. Clearly, the results (11th-18th rows) indicate that the prefixes and suffixes of length up to three characters (12th row) of the current word only are more effective than the others. Evaluation results (13th -16th rows) also show the fact that surrounding word suffixes and/or prefixes do not increase the f-score value. It is also observed (17th -18 rows) that inclusion of the surrounding word suffixes and/or prefixes along with the suffixes and prefixes of the current word reduce the f-score value. The overall f-score value is further improved by 1.81% (12th and 19th rows) with the various digit features.

Results (20th-23rd rows) show that POS information of the current and/or the surrounding word(s) are very helpful and plays an important role in the overall performance improvement of the system. Results also suggest that the POS information of the window $[-1, +1]$ is more effective (f-score= 79.82%) than the POS information of the windows $[-1, 0]$, $[0, +1]$ or the current word alone. We also conducted experiments by considering the POS information of the windows [-2, +2], [-2, +1], $[-2, 0]$, $[0, +2]$, $[-2, -1]$ and $[+1, +2]$ and observed that the f-score value of the system get reduced

TABLE IV
DESCRIPTIONS OF THE LANGUAGE INDEPENDENT FEATURES FOR BENGALI AND HINDI. HERE, i REPRESENTS THE POSITION OF THE CURRENT WORD AND w_i REPRESENTS THE CURRENT WORD

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	$\text{FirstWord}_i = \begin{cases} 1, \text{if } w_i \text{ is the first word of a sentence} \\ 0, \text{Otherwise} \end{cases}$
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	$\text{CntDgtPrctg}_i = \begin{cases} 1, \text{if } w_i \text{ contains digit} \\ \quad \text{and percentage} \\ 0, \text{otherwise} \end{cases}$
RareWord	$\text{RareWord}_i = I_{\{\text{Infrequent word list}\}}(w_i)$
Length	$\text{Length}_i = \begin{cases} 1, \text{if } w_i \geq 3 \\ 0, \text{otherwise} \end{cases}$

TABLE V
EXPERIMENTAL RESULTS ON THE DEVELOPMENT SET FOR BENGALI

Feature (word, tag)	F-Score (in %)
pw, cw, nw, FirstWord, Length	67.01
pw, cw, nw, FirstWord, Length	67.82
pw2, pw, cw, nw, nw2, FirstWord, Length	67.11
pw3, pw2, pw, cw, nw, nw2, FirstWord, Length	68.28
pw3, pw2, pw, cw, nw, nw2, nw3, FirstWord, Length	68.11
pw3, pw2, pw, cw, nw, nw2, FirstWord, Length, RareWord	69.12
pw3, pw2, pw, cw, nw, nw2, FirstWord, Length, RareWord, pt	71.41
pw3, pw2, pw, cw, nw, nw2, FirstWord, Length, RareWord, pt, pt2	71.92
pw3, pw2, pw, cw, nw, nw2, FirstWord, Length, RareWord, pt, pt2, pt3	71.53
pw3, pw2, pw, cw, nw, nw2, FirstWord, Length, RareWord, pt, pt2, |suf| $\leq$4, |pre| $\leq$4	74.56
pw3, pw2, pw, cw, nw, nw2, FirstWord, Length, RareWord, pt, pt2, |suf| $\leq$3, |pre| $\leq$ 3	75.96
pw3, pw2, pw, cw, nw, nw2, FirstWord, Length, RareWord, pt, pt2, |psuf| $\leq$3, |ppre| $\leq$ 3	74.57
pw3, pw2, pw, cw, nw, nw2, FirstWord, Length, RareWord, pt, pt2, |nsuf| $\leq$3, |npre| $\leq$ 3	73.98
pw3, pw2, pw, cw, nw, nw2, FirstWord, Length, RareWord, pt, pt2, |npre| $\leq$3, |ppre| $\leq$3	73.52
pw3, pw2, pw, cw, nw, nw2, FirstWord, Length, RareWord, pt, pt2, |psuf| $\leq$3, |nsuf| $\leq$3	73.95
pw3, pw2, pw, cw, nw, nw2, FirstWord, Length, RareWord, pt, pt2, |suf| $\leq$3, |pre| $\leq$ 3, |psuf| $\leq$3, |nsuf| $\leq$3	73.28
pw3, pw2, pw, cw, nw, nw2, FirstWord, Length, RareWord, pt, pt2, |suf| $\leq$3, |pre| $\leq$ 3, |npre| $\leq$3, |nsuf| $\leq$3	73.16
pw3, pw2, pw, cw, nw, nw2, FirstWord, Length, RareWord, pt, pt2, |suf| $\leq$3, |pre| $\leq$ 3, Digit	77.77
pw3, pw2, pw, cw, nw, nw2, FirstWord, Length, RareWord, pt, pt2, |suf| $\leq$3, |pre| $\leq$ 3, Digit, pp, cp, np	79.82
pw3, pw2, pw, cw, nw, nw2, FirstWord, Length, RareWord, pt, pt2, |suf| $\leq$3, |pre| $\leq$ 3, Digit, cp	77.79
pw3, pw2, pw, cw, nw, nw2, FirstWord, RareWord, pt, pt2, |suf| $\leq$3, |pre| $\leq$3, Digit, pp, cp	78.89
pw3, pw2, pw, cw, nw, nw2, FirstWord, RareWord, pt, pt2, |suf| $\leq$3, |pre| $\leq$3, Digit, cp, np	78.12

compared to the POS information of the window $[-1, +1]$ in each of the cases. It is also indicative from the results (22nd and 23rd rows) that the POS information of the window $[-1, 0]$ is more helpful than the POS information of the window $[0, +1]$.

Evaluation results of the system for the development set are presented in Table VI for Hindi. Results (2nd-6th rows) show that the system performs the best for the window $[-3, +2]$ with the 'FirstWord' and 'Length' features only. An overall f-score value of 70.53% is obtained by including the NE information of the window $[-2, -1]$ and the 'RareWord' feature (7th-10th rows). The prefixes and suffixes of length up to three characters of the current word only increases the f-score value to 74.95% (11th- 18th rows). It is also observed (13th-18th rows) that the surrounding word suffixes and/or prefixes do not increase the f-score value. The introduction of digit features improve the f-score value by 1.91% (12th and 19th rows). Evaluation results (20th row) show an improvement of 1.22% in the overall f-score value with the POS information of the current, previous and next words. It is also evident from the experimental results (20th-23rd rows) that POS information of the window $[-1, +1]$ is more effective than the others.

One important observation is that the system performs better for Bengali though the size of the Hindi training corpus is approximately twice to that of the Bengali. One possible reason is the presence of less number of NEs in the Hindi data in comparison to Bengali. In order to achieve the reasonable performance, the training corpus should have enough evidences of the target NE classes. Model performance can degrade if the distribution of non-names is very high in the training corpus. Thus, it can be decided that appropriate training data that contains the evenly distributed names and non-names is very essential.

1) Use of context patterns as features for Bengali: High ranked patterns of the *Accept Pattern set* (discussed in section III) can be used as the features of SVM. 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 have demonstrated the f-score value of 82.93%, which is an improvement of **3.11**% with the use of context features.

B. Experimental results on the test sets

The best set of features for NER in Bengali and Hindi are identified by training the SVM based NER system with 242,467, and 452,974 tokens and testing with the development sets of 30K, and 50K tokens, respectively. Now, the development sets are included as part of the training sets and the resultant training sets thus consist of 272,467 and 502,974 tokens for Bengali and Hindi, respectively. Now, the gold standard test sets of 35K, and 60K tokens have been used in order to report the experimental results for Bengali, and Hindi, respectively . There are approximately 17%, and 15% unknown NEs in the test sets of Bengali and Hindi, respectively. Evaluation results of the system have been presented in Table VII for the test sets. Table also shows the results of the *baseline* models.

1) Use of context patterns in Bengali: The high ranked patterns of the *Accept Pattern set* (discussed in section III) are used as the features of SVM. The inclusion of this feature has demonstrated the recall, precision, and f-score values of 83.78%, 80.35%, and 82.03%, respectively. So, it can be decided that context patterns are very useful to improve the performance of the system. Results show an improvement of 3.06% f-score with the use of context features. Results also show that the context patterns are more effective to improve the precision (an improvement of 4.28%) value in comparison to the recall (an improvement of 1.57%) value.

A detailed look to the evaluation results reveals that the errors are mostly concerned with the unseen words. Some NEs are also tagged as NNE by the system. To deal with these two particular problems, the high ranked patterns contained in the *Accept Pattern set* set are used to post-process the output generated by the SVM based system. Each pattern is looked for a match in the output file of the form (word, tag), generated by the classifier. Lexical pattern induction component is then used to change and/or assign the NE tag to the current token if it is either unknown or tagged as NNE by the system. Evaluation results have demonstrated the recall, precision, and f-score values of 88.61%, 80.12%, and 84.15%, respectively. Hence, this is a significant improvement in the recall value (approximately, 5%) by including this module. However, the recall value drops by 0.23%. Thus, it results in the overall f-score gain by 2.12%.

C. Evaluation results of the 10-fold cross validation tests

The resultant training sets of Bengali and Hindi consist of 272,467 tokens and 502,974 tokens. Each of these training sets has been distributed into 10 equal subsets. One of the subsets is kept as the test set and the remaining nine subsets are used as the training sets of the system. This process is repeated 10 times to yield an average result, which is called the 10-fold cross validation test.

During all these 10 experiments in Bengali, the system is initially trained with all the language independent features. The context features are then incorporated into the system. After each test, the high ranked patterns are used to change and/or assign the NE tag(s) in the output of the SVM based system. For Hindi, we have conducted the 10-fold cross validation

TABLE VI
EXPERIMENTAL RESULTS ON THE DEVELOPMENT SET FOR HINDI

Feature (word, tag)	F-Score (in %)
pw, cw, nw, FirstWord, Length	65.91
pw, cw, nw, FirstWord, Length	66.23
pw2, pw, cw, nw, nw2, FirstWord, Length	66.81
pw3, pw2, pw, cw, nw, nw2, FirstWord, Length	67.37
pw3, pw2, pw, cw, nw, nw2, nw3, FirstWord, Length	67.02
pw3, pw2, pw, cw, nw, nw2, FirstWord, Length, RareWord	67.58
pw3, pw2, pw, cw, nw, nw2, FirstWord, Length, RareWord, pt	69.91
pw3, pw2, pw, cw, nw, nw2, FirstWord, Length, RareWord, pt, pt2	70.53
pw3, pw2, pw, cw, nw, nw2, FirstWord, Length, RareWord, pt, pt2, pt3	69.81
pw3, pw2, pw, cw, nw, nw2, FirstWord, Length, RareWord, pt, pt2, |suf| $\leq$4, |pre| $\leq$4	73.74
pw3, pw2, pw, cw, nw, nw2, FirstWord, Length, RareWord, pt, pt2, |suf| $\leq$3, |pre| $\leq$ 3	74.95
pw3, pw2, pw, cw, nw, nw2, FirstWord, Length, RareWord, pt, pt2, |psuf| $\leq$3, |ppre| $\leq$ 3	72.91
pw3, pw2, pw, cw, nw, nw2, FirstWord, Length, RareWord, pt, pt2, |nsuf| $\leq$3, |npre| $\leq$ 3	71.79
pw3, pw2, pw, cw, nw, nw2, FirstWord, Length, RareWord, pt, pt2, |npre| $\leq$3, |ppre| $\leq$3	72.47
pw3, pw2, pw, cw, nw, nw2, FirstWord, Length, RareWord, pt, pt2, |psuf| $\leq$3, |nsuf| $\leq$3	72.92
pw3, pw2, pw, cw, nw, nw2, FirstWord, Length, RareWord, pt, pt2, |suf| $\leq$3, |pre| $\leq$ 3, |psuf| $\leq$3, |nsuf| $\leq$3	73.16
pw3, pw2, pw, cw, nw, nw2, FirstWord, Length, RareWord, pt, pt2, |suf| $\leq$3, |pre| $\leq$ 3, |npre| $\leq$3, |nsuf| $\leq$3	73.11
pw3, pw2, pw, cw, nw, nw2, FirstWord, Length, RareWord, pt, pt2, |suf| $\leq$3, |pre| $\leq$ 3, Digit	76.86
pw3, pw2, pw, cw, nw, nw2, FirstWord, Length, RareWord, pt, pt2, |suf| $\leq$3, |pre| $\leq$ 3, Digit, pp, cp, np	78.08
pw3, pw2, pw, cw, nw, nw2, FirstWord, Length, RareWord, pt, pt2, |suf| $\leq$3, |pre| $\leq$ 3, Digit, cp	77.79
pw3, pw2, pw, cw, nw, nw2, FirstWord, RareWord, pt, pt2, |suf| $\leq$3, |pre| $\leq$3, Digit, pp, cp	77.92
pw3, pw2, pw, cw, nw, nw2, FirstWord, RareWord, pt, pt2, |suf| $\leq$3, |pre| $\leq$3, Digit, cp, np	77.59

TABLE VII
EXPERIMENTAL RESULTS ON THE TEST SETS

Language	Model	Recall	Precision	F-Score
Bengali	Baseline	64.73	52.21	57.80
Hindi	Baseline	64.67	52.01	57.65
Bengali	SVM	82.21	76.07	79.02
Hindi	SVM	80.23	74.34	77.17

test only with the language independent features. Evaluation results of the 10-fold cross validation tests are reported in Table VIII for Bengali and in Table IX for Hindi. The system has demonstrated the overall average recall, precision, and f-score values of 88.69%, 80.35%, and 84.31%, respectively, for Bengali. The system has shown the overall average recall, precision, and f-Score values of 80.48%, 74.54%, and 77.39%, respectively, for Hindi.

D. Comparisons with other systems

The existing Bengali NER systems reported in [27] and in [28] have been trained and tested under the same experimental setup. Two models, namely A and B have been reported in [27]. These models are based on a pattern directed shallow parsing approach. An unsupervised algorithm was developed to tag the unlabeled corpus with the seed entities of *Person name*, *Location name* and *Organization name*. Model A uses only the seed lists to tag the training corpus whereas in model B, we have used the various gazetteers along with the seed entities for tagging. The lexical context patterns generated in such way are used to generate further patterns in a bootstrapped manner. The algorithm terminates until no new patterns can be generated. During testing, model A can not deal with the *NE classification disambiguation* problem (i.e, can not handle the situation when a particular word is tagged with more than one NE type) but model B can handle with this problem with the help of gazetteers and various language dependent features.

A HMM-based NER system has been reported in [28], where more context information has been considered during emission probabilities and the word suffixes have been used for handling the unknown words. Comparative evaluation results for the test set are presented in Table X. Result show that the proposed system outperforms the least performing model A by 15.83% and the existing HMM-based system by 6.57%. One reason behind the rise in recall, precision and f-score values in the SVM based NER system is its ability to handle the non-independent, diverse and overlapping features of the morphologically rich Indian languages more efficiently than the HMM.

A HMM-based Hindi NER system has been reported in [31]. This system has been trained and tested with the same Hindi training and test sets. Evaluation results of the test set have demonstrated the recall, precision, and f-score values of 76.37%, 74.99%, and %, 75.99, respectively.

Experimental results of the 10-fold cross validation test for the HMM-based NER system [28] are presented in Table XI. The system has demonstrated the overall average recall, precision, and f-score values of 78.81%, 76.64%, and 77.62%, respectively. Results of the 10-fold cross validation tests of the HMM-based Hindi NER system [31] are shown in Table XII. Results show the overall average recall, precision, and f-score

TABLE VIII
RESULTS OF 10-FOLD CROSS VALIDATION TEST FOR BENGALI USING SVM AND CONTEXT PATTERN

Test Set No	Recall	Precision	F-Score
1	87.89	79.97	83.74
2	88.54	80.32	84.23
3	88.73	80.21	84.26
4	88.69	80.27	84.27
5	88.98	80.31	84.42
6	88.45	80.54	84.31
7	88.91	80.33	84.40
8	88.93	80.31	84.41
9	88.89	80.61	84.55
10	88.87	80.65	84.57
Average	88.69	80.35	84.31

TABLE IX
RESULTS OF 10-FOLD CROSS VALIDATION TEST FOR HINDI USING SVM

Test Set No	Recall	Precision	F-Score
1	80.21	74.30	77.14
2	80.17	74.80	77.39
3	80.22	74.33	77.16
4	80.34	74.31	77.21
5	80.71	74.39	77.42
6	80.72	75.11	77.81
7	80.54	75.12	77.74
8	80.51	74.19	77.22
9	80.44	74.25	77.21
10	80.93	74.61	77.64
Average	80.48	74.54	77.39

TABLE X
COMPARATIVE EVALUATION RESULTS (A: PATTERN DIRECTED SHALLOW PARING APPROACH WITHOUT LINGUISTIC KNOWLEDGE, B: PATTERN DIRECTED SHALLOW PARSING APPROACH WITH LINGUISTIC KNOWLEDGE)

Model	Recall	Precision	F-Score
A	69.57	67.12	68.32
B	72.17	70.09	71.11
HMM	78.81	76.39	77.58
SVM	88.61	80.12	84.15

TABLE XI
RESULTS OF 10-FOLD CROSS VALIDATION TEST FOR BENGALI USING HMM

Test Set No.	Recall	Precision	F-Score
1	78.75	76.33	77.52
2	78.83	76.41	77.6
3	78.81	76.51	77.64
4	78.49	76.41	77.44
5	78.92	76.53	77.71
6	78.87	76.69	77.73
7	78.89	76.22	77.53
8	78.88	76.41	77.63
9	78.79	76.67	77.72
10	78.87	76.68	77.76
Average	78.81	76.64	77.62

TABLE XII
RESULTS OF 10-FOLD CROSS VALIDATION TEST FOR HINDI USING HMM

Test Set No.	Recall	Precision	F-Score
1	76.34	74.47	75.39
2	76.55	74.92	75.72
3	76.31	75.19	75.75
4	76.52	75.22	75.86
5	76.84	74.97	75.89
6	76.12	74.11	75.10
7	76.03	74.23	75.11
8	76.29	75.16	75.72
9	76.41	75.01	75.70
10	76.44	74.98	75.71
Average	76.39	74.83	75.59

values of 76.39%, 74.83%, and 75.59%, respectively.

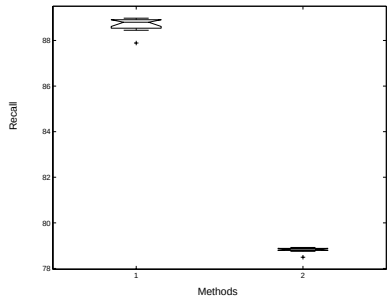

Fig. 2. Boxplot of the Recall values obtained by SVM (1) and HMM (2) based NER approach for Bengali

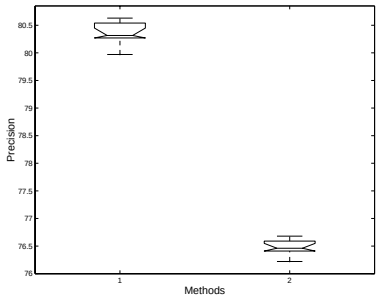

Fig. 3. Boxplot of the Precision values obtained by SVM (1) and HMM (2) based NER approach for Bengali

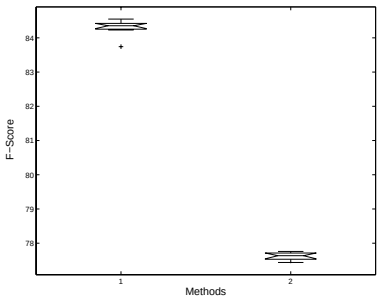

Fig. 4. Boxplot of the F-Score values obtained by SVM (1) and HMM (2) based NER approach for Bengali

ANOVA [47] statistical analysis is performed on the combined results of the two NER systems (HMM based and SVM based) for Bengali and Hindi both. In statistics, ANOVA is short for analysis of variance. Analysis of variance is a collection of statistical models, and their associated procedures, in which the observed variance is partitioned into components due to different explanatory variable. The initial techniques of the analysis of variance were developed by the statistician and geneticist R.A. Fisher in the 1920s and 1930s, and is sometimes known as Fisher's ANOVA or Fisher's analysis of variance, due to the use of Fisher's F-distribution as part of the test of statistical significance. There are three conceptual classes of such models:

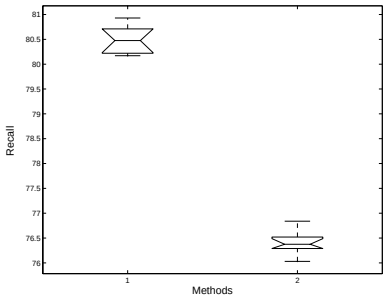

Fig. 5. Boxplot of the Recall values obtained by SVM (1) and HMM (2) based NER approach for Hindi

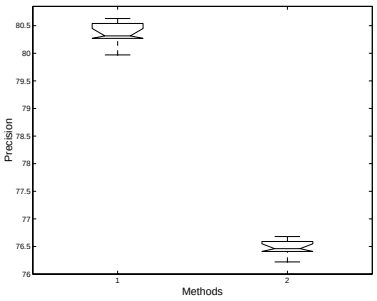

Fig. 6. Boxplot of the Precision values obtained by SVM (1) and HMM (2) based NER approach for Hindi

1) Fixed-effects models assumes that the data came from normal populations which may differ only in their means.
2) Random effects models assume that the data describe a hierarchy of different populations whose differences are constrained by the hierarchy.
3) Mixed-effect models describe situations where both fixed and random effects are present.

In practice, there are several types of ANOVA depending on the number of treatments and the way they are applied to the subjects in the experiment. One-way ANOVA is used to test for differences among two or more independent groups. Typically, however, the One-way ANOVA is used to test for differences among at least three groups, since the two-group case can be covered by a T-test ([48]). When there are only two means to compare, the T-test and the F-test are equivalent; the relation between ANOVA and t is given by $F = t^2$. One-way ANOVA for repeated measures is used when the subjects are subjected to repeated measures; this means that the same subjects are used for each treatment. The One-Way ANOVA procedure produces a one-way analysis of variance for a quantitative dependent variable (here, it is three different evaluation criterion) by a single independent variable (here, it is the algorithm, SVM and HMM). Analysis of variance is used to test the hypothesis that several means are equal. Here, ANOVA analysis is carried out on the results of 10-fold cross validation tests obtained by SVM and HMM for Bengali and Hindi both. The One-Way ANOVA procedure produces a one-way analysis of variance for a quantitative dependent variable (here, it is 3 different evaluation criterion) by a single

TABLE XIII
ESTIMATED MARGINAL MEANS AND PAIRWISE COMPARISON OF NER TECHNIQUES USING SVM AND HMM ON RECALL, PRECISION AND F-SCORE VALUES OBTAINED BY ANOVA TESTING FOR BENGALI

Evaluation criterion	Algo. (I)	Mean of I	Algo. (J)	Mean	Mean Diff. (I-J)	Significance value
Recall	SVM	88.6880 ± 0.1094	HMM	78.8100 ± 0.0153	9.8780 ± 0.1016	0
Precision	SVM	80.3500 ± 0.0397	HMM	76.4760 ± 0.0218	3.8740 ± 0.0186	0
F-Score	SVM	84.3132 ± 0.0529	HMM	77.6254 ± 0.0108	6.6878 ± 0.0366	0

TABLE XIV
ESTIMATED MARGINAL MEANS AND PAIRWISE COMPARISON OF NER TECHNIQUES USING SVM AND HMM ON RECALL, PRECISION AND F-SCORE VALUES OBTAINED BY ANOVA TESTING FOR HINDI

Evaluation criterion	Algo. (I)	Mean of I	Algo. (J)	Mean	Mean Diff. (I-J)	Significance value
Recall	SVM	80.4790 ± 0.0639	HMM	76.3850 ± 0.0521	4.0940 ± 0.1155	0
Precision	SVM	74.5410 ± 0.1242	HMM	74.8260 ± 0.1646	-0.2850 ± 0.5092	0.1108
F-Score	SVM	77.3959 ± 0.0630	HMM	75.5971 ± 0.0835	1.7987 ± 0.2376	$1.5030e-011$

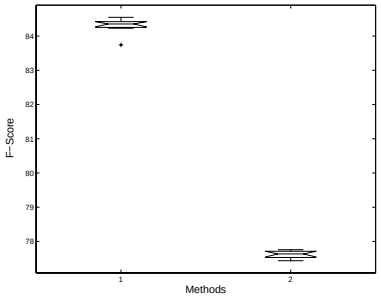

Fig. 7. Boxplot of the F-Score values obtained by SVM (1) and HMM (2) based NER approach for Hindi

independent variable (here, it is the algorithm (SVM based method and HMM based method)). Analysis of variance is used to test the hypothesis that several means are equal. The results for Bengali, and Hindi are reported in Tables XIII, and XIV, respectively.

From the statistical test ANOVA, it is found that the difference in the mean recall, precision and f-score values obtained by the SVM based method with those obtained by the HMM based method for Bengali and Hindi are statistically significant as in all the cases the significance value < 0.05.

In a descriptive statistics, box plot is a convenient way of graphically depicting groups of numerical data through their five no. summaries:the smallest observation, lower quartile (Q1), median (Q2), upper quartile (Q3) and largest observation. A box plot may also indicate which observations can be considered as 'outliers'. Box plot can be useful to display differences between populations without making any assumptions of the underlying statistical distribution. The spacing between the different parts of the box help to indicate the degree of dispersion (spread) and skewness in the data and identify 'outliers'. A quartile is any of the three values which divide the sorted data into 4 equal parts, so that each part represents 1/4 of the sampled population. The box plots (showing the mean and the variances) of the three evaluation criterion for these two approaches of Bengali are also shown in Figures 2-4. Similarly, the box plots (showing the mean and the variances) of the three evaluation criterion for these two approaches of Hindi are also shown in Figures 5-7.

VI. CONCLUSION

In this paper, we have presented a NER system for two most popular Indian languages (ILs), namely Bengali and Hindi with the SVM framework. The NER system uses only the language independent features. These features can be applied for NER in any language including the Indian languages. A number of experiments have been conducted to find out the best set of features for NER in Bengali and Hindi. The system has demonstrated the overall f-score values of 84.15%, and 77.17% for Bengali, and Hindi, respectively. We have generated the lexical context patterns from the unlabeled Bengali news corpus. These patterns are used as the features of SVM as well as the means to post-process the output of SVM. Evaluation results have demonstrated the overall performance gain by 5.13% f-score with the use of context patterns. The 10-fold validation tests have demonstrated the recall, precision, and f-score values of 88.69%, 80.35%, and 84.31%, respectively, for Bengali and 80.48%, 74.54%, and 77.39%, respectively, for Hindi. We have also compared the performance between the proposed SVM based systems and the existing HMM based systems and shown that the performance improvement are statistically significant for both of the languages.

A detailed look to the evaluation results of the system for Bengali reveals that the system performs poorly for some specific situations, such as for the common words that can appear in the dictionary with other valid meanings, can not efficiently handle the short names and the variation of NEs. Also, the uneven ratio between the names and non-names is the another reason behind the poor performance for Hindi.

Future works include development of the NER system for the other Indian languages, particularly for Telugu, Oriya and Urdu using the same set of language independent features. We would like to incorporate the language specific features of Bengali and Hindi in order to achieve better performance.

REFERENCES

[1] N. Chinchor, "MUC-6 Named Entity Task Definition (Version 2.1)," in *MUC-6*, 1995.

[2] N. Chinchor, "MUC-7 Named Entity Task Definition (Version 3.5)," in *MUC-7*, 1998.
[3] H. Cunningham, "GATE, a General Architecture for Text Engineering," *Computers and the Humanities*, vol. 36, pp. 223–254, 2002.
[4] D. Moldovan, S. Harabagiu, R. Girju, P. Morarescu, F. Lacatusu, A. Novischi, A. Badulescu, and O. Bolohan, "LCC Tools for Question Answering," in *Text REtrieval Conference (TREC) 2002*, 2002.
[5] B. Babych and A. Hartley, "Improving Machine Translation Quality with Automatic Named Entity Recognition," in *Proceedings of EAMT/EACL 2003 Workshop on MT and other Language Technology Tools*, pp. 1–8, 2003.
[6] S. Miller, M. Crystal, H. Fox, L. Ramshaw, R. Schawartz, R. Stone, R. Weischedel, and the Annotation Group, "BBN: Description of the SIFT System as Used for MUC-7," in *MUC-7*, (Fairfax, Virginia), 1998.
[7] D. M. Bikel, R. L. Schwartz, and R. M. Weischedel, "An Algorithm that Learns What's in a Name," *Machine Learning*, vol. 34, no. 1-3, pp. 211–231, 1999.
[8] A. Borthwick, *Maximum Entropy Approach to Named Entity Recognition*. PhD thesis, New York University, 1999.
[9] A. Borthwick, J. Sterling, E. Agichtein, and R. Grishman, "NYU:Description of the MENE Named Entity System as Used in MUC-7," in *MUC-7*, 1998.
[10] S. Sekine, "Description of the Japanese NE System used for MET-2," in *MUC-7*, (Fairfax, Virginia), 1998.
[11] S. W. Bennet, C. Aone, and C. Lovell, "Learning to Tag Multilingual Texts Through Observation," in *Proceedings of Empirical Methods of Natural Language Processing*, (Providence, Rhode Island), pp. 109–116, 1997.
[12] A. McCallum and W. Li, "Early results for Named Entity Recognition with Conditional Random Fields, Feature Induction and Web-enhanced Lexicons," in *Proceedings of CoNLL*, (Canada), pp. 188–191, 2003.
[13] J. D. Lafferty, A. McCallum, and F. C. N. Pereira, "Conditional Random Fields: Probabilistic Models for Segmenting and Labeling Sequence Data," in *Proceedings of the 18th International Conference on Machine Learning (ICML)*, pp. 282–289, 2001.
[14] A. Sun, "Using Support Vector Machine for Terrorism Information Extraction," in *Proceedings of the 1st NSF/NIJ Symposium on Intelligence and Security*, 2003.
[15] A. De Sitter and W. Daelemans, "Information Extraction via Double Classification," in *Proceedings of International Workshop on Adaptive Text Extraction and Mining*, (Dubrovnik), 2003.
[16] N. Kushmerick, E. Johnston, and S. McGuinness, "Information Extraction by Text Classification," in *Proceedings of IJCAI-01 Workshop on Adaptive Text Extraction and Mining*, (Seattle, WA), 2001.
[17] A. McCallum, D. Freitag, and F. Pereira, "Maximum Entropy Markov Models for Information Extraction and Segmentation," in *Proceedings of the 17th International Conference on Machine Learning (ICML)*, pp. 591–598, 2000.
[18] R. Malouf, "Markov Models for Language Independent Named Entity Recognition," in *Proceedings of the 6th Conference on Natural Language Learning (CoNLL-2002)*, (Taipei, Taiwan), pp. 187–190, 2002.
[19] J. D. Burger, J. C. Henderson, and T. Morgan, "Statistical Named Entity Recognizer Adaption," in *Proceedings of the CoNLL Workshop*, (Taipei, Taiwan), pp. 163–166, 2002.
[20] X. Carrears, L. Marquez, and L. Padro, "Named Entity Recognition using AdaBoost," in *Proceedings of the CoNLL Workshop*, (Taipei, Taiwan), pp. 167–170, 2002.
[21] G. Zhou and J. Su, "Named Entity Recognition using an HMM-based Chunk Tagger," in *Proceedings of ACL*, (Philadelphia), pp. 473–480, 2002.
[22] H. Yamada, T. Kudo, and Y. Matsumoto, "Japanese Named Entity Extraction using Support Vector Machine," *In Transactions of IPSJ*, vol. 43, no. 1, pp. 44–53, 2001.
[23] T. Kudo and Y. Matsumoto, "Chunking with Support Vector Machines," in *Proceed-ings of NAACL*, pp. 192–199, 2001.
[24] K. Takeuchi and N. Collier, "Use of Support Vector Machines in Extended Named Entity Recognition," in *Proceedings of the 6th Conference on Natural Language Learning (CoNLL-2002)*, pp. 119–125, 2002.
[25] A. Masayuki and Y. Matsumoto, "Japanese Named Entity Extraction with Redundant Morphological Analysis," in *NAACL '03: Proceedings of the 2003 Conference of the North American Chapter of the Association for Computational Linguistics on Human Language Technology*, (Morristown, NJ, USA), pp. 8–15, Association for Computational Linguistics, 2003.
[26] A. Ekbal and S. Bandyopadhyay, "Pattern Based Bootstrapping Method for Named Entity Recognition," in *Proceedings of the 6th International Conference on Advances in Pattern Recognition (ICAPR)*, pp. 349–355, World Scientific, 2007.
[27] A. Ekbal and S. Bandyopadhyay, "Lexical Pattern Learning from Corpus Data for Named Entity Recognition," in *Proceedings of 5th International Conference on Natural Language Processing (ICON)*, (India), pp. 123–128, 2007.
[28] A. Ekbal, S. Naskar, and S. Bandyopadhyay, "Named Entity Recognition and Transliteration in Bengali," *Named Entities: Recognition, Classification and Use, Special Issue of Lingvisticae Investigationes Journal*, vol. 30, no. 1, pp. 95–114, 2007.
[29] A. Ekbal and S. Bandyopadhyay, "Bengali Named Entity Recognition using Support Vector Machine," in *Proceedings of Workshop on NER for South and South East Asian Languages, 3rd International Joint Conference on Natural Languge Processing (IJCNLP)*, (India), pp. 51–58, 2008.
[30] W. Li and A. McCallum, "Rapid Development of Hindi Named Entity Recognition using Conditional Random Fields and Feature Induction," *ACM Transactions on Asian Languages Information Processing*, vol. 2, no. 3, pp. 290–294, 2004.
[31] A. Ekbal and S. Bandyopadhyay, "A Hidden Markov Model Based Named Entity Recognition System: Bengali and Hindi as Case Studies," in *Proceedings of the 2nd International Conference on Pattern Recognition and Machine Intelligence (PReMI 2007)*, pp. 545–552, Springer Verlag, 2007.
[32] V. N. Vapnik, *The nature of statistical learning theory*. New York, NY, USA: Springer-Verlag New York, Inc., 1995.
[33] C. C and V. N. Vapnik, "Support Vector Networks," *Machine Learning*, vol. 20, pp. 273–297, 1995.
[34] T. Joachims, "Making large-scale support vector machine learning practical," pp. 169–184, 1999.
[35] H. Taira and M. Haruno, "Feature Selection in SVM Text Categorization," in *Proceedings of AAAI-99*, 1999.
[36] A. Ekbal and S. Bandyopadhyay, "A Web-based Bengali News Corpus for Named Entity Recognition," *Language Resources and Evaluation Journal*, vol. 42, no. 2, 2008.
[37] M. Collins and Y. Singer, "Unsupervised models for named entity classification," in *Proceedings of the Joint SIGDAT Conference on Empirical Methods in Natural Language Processing and Very Large Corpora*, 1999.
[38] S. Cucerzon and D. Yarowsky, "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.), 1999.
[39] S. Cucerzan and D. Yarowsky, "Language Independent NER using a Unified Model of Internal and Contextual Evidence," in *Proceedings of CoNLL 2002*, pp. 171–175, 2002.
[40] W. Phillips and E. Riloff, "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*, (Morristown, NJ, USA), pp. 125–132, Association for Computational Linguistics, 2002.
[41] E. Riloff and R. Jones, "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 innovative applications of artificial intelligence*, (Menlo Park, CA, USA), pp. 474–479, American Association for Artificial Intelligence, 1999.
[42] M. Thelen and E. Riloff, "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*, (Morristown, NJ, USA), pp. 214–221, Association for Computational Linguistics, 2002.
[43] T. Strzalkowski and J. Wang, "A Self-learning Universal Concept Spotter," in *Proceedings of the 16th conference on Computational linguistics*, (Morristown, NJ, USA), pp. 931–936, Association for Computational Linguistics, 1996.
[44] R. Yangarber, W. Lin, and R. Grishman, "Unsupervised Learning of Generalized Names," in *Proceedings of the 19th international conference on Computational linguistics*, (Morristown, NJ, USA), pp. 1–7, Association for Computational Linguistics, 2002.
[45] A. Ekbal, R. Haque, and S. Bandyopadhyay, "Bengali Part of Speech Tagging using Conditional Random Field," in *Proceedings of Seventh International Symposium on Natural Language Processing (SNLP2007)*, 2007.
[46] A. Ekbal and S. Bandyopadhyay, "Lexicon Development and POS Tagging using a Tagged Bengali News Corpus," in *Proceedings of the*

20th International Florida AI Research Society Conference (FLAIRS-2007), (Florida), pp. 261–263, 2007.
[47] T. W. Anderson and S. Scolve, *Introduction to the Statistical Analysis of Data*. Houghton Mifflin, 1978.
[48] W. S. Gosset, "The Probable Error of a Mean," in *Biometrika*, vol. 6, pp. 1–25, 1908.