Multi-Engine Approach for Named Entity Recognition in Bengali*

Asif Ekbal [a], and Sivaji Bandyopadhyay[a]
[a]Department of Computer Science and Enginneering, Jadavpur University, Kolkata-700032, India
asif.ekbal@gmail.com and sivaji_cse_ju@yahoo.com

Abstract. This paper reports about a multi-engine approach for the development of a 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 approach. 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 such as *Person*, *Location*, *Organization* and *Miscellaneous* tags. An appropriate tag conversion routine has been defined in order to convert the 122K wordforms of the IJCNLP-08 NER shared task[1], into the desired forms. The classifiers make use of the different contextual information of the words along with the variety of features that are helpful in predicting the various NE classes. Lexical context patterns, which are 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 have developed a number of techniques to post-process the output of each of the classifiers in order to reduce the errors and to improve the performance. Finally, we have applied weighted voting approach to combine the systems. Results show the effectiveness of the proposed approach with the overall average 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 of 18.36% in f-score over the least performing *baseline* ME based system. The proposed system also outperforms the other existing Bengali NER system.

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

1. Introduction

Named Entity Recognition (NER) is an important tool in almost all Natural Language Processing (NLP) application areas including machine translation, question answering, information retrieval, information extraction, automatic summarization etc. The current trend in NER is to use the machine-learning (ML) approach, which is more attractive in that it is trainable and adoptable and the maintenance of a ML based system is much cheaper than that of a rule-based one. The representative ML approaches used in NER are Hidden Markov Model (HMM) (BBN's IdentiFinder in (Bikel, 1999)), ME (New York University's MENE in (Borthwick, 1999)), CRFs (Lafferty et al., 2001) and SVM (Yamada et al., 2002). 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). Florian et al. (2003) tested different methods for combining the results of four systems and found that robust risk minimization worked best. The work reported in this paper differs from the existing works in the sense that here, we have conducted a number of experiments to improve the performance of the classifiers with the lexical context patterns, which are generated in a semi-automatic way from an unlabeled corpus of 3 million wordforms, and used several post-processing techniques to improve the performance of each classifier before applying weighted voting.

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

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

* This work is partially supported by the CLIA Project, funded by DIT, MCIT, Govt. of India, Vide Sanction letter no-14 (5)/2006- HCC (TDIL)/29.08.2006.

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

- Indian person names are more diverse and a lot of these words can be found in the dictionary with specific meanings.
- 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.
- Bengali is a relatively free order language.
- Bengali, like other Indian languages, is a resource poor language - annotated corpora, name dictionaries, good morphological analyzers, Part of Speech (POS) taggers etc. are not yet available in the required measure.
- Although Indian languages have a very old and rich literary history, technological developments are of recent origin.
- Web sources for name lists are available in English, but such lists are not available in 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 handling the unknown words. More recently, the related works in this area can be found in Ekbal et al. (2008a), Ekbal and Bandyopadhyay (2008b) with the CRF, and SVM approach, respectively. Other than Bengali, the works on Hindi can be found in Li and McCallum (2004) with CRF and Cucerzan and Yarowsky (1999) with a language independent method. As part of the IJCNLP-08 NER shared task, various works of NER in Indian languages using various approaches can be found in IJCNLP-08 NER Shared Task on South and South East Asian Languages (NERSSEAL)[2].

2. Named Entity Recognition in Bengali

Bengali is the seventh popular language in the world, second in India and the national language of Bangladesh. We have used a Bengali news corpus (Ekbal and Bandyopadhyay, 2008c), developed from the web-archive of a widely read Bengali newspaper for NER. Out of 34 million wordforms of this corpus, 200K wordforms have been manually annotated with the four NE tags namely, *Person, Location, Organization* and *Miscellaneous*. The annotation was carried out by a technical person and edited by a linguist. The data has been collected from the International, National, State and Sports domains. We have also used the annotated corpus of 122K wordforms, collected from the IJCNLP-08 NERSSEAL (http://ltrc.iiit.ac.in/ner-ssea-08). This data was a mixed one and dealt mainly with the literature, agriculture and scientific domains. Moreover, this data was originally annotated with a fine-grained NE tagset of twelve tags. We have defined an appropriate tag conversion routine in order to convert this data into the forms tagged with the four NE tags. The tagset mapping table is shown in Table 1.

Table 1: Tagset mapping table

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

In order to properly denote the boundaries of NEs, the four NE tags are further divided into the forms:

B-XXX: Beginning of a multiword NE, I-XXX: Internal of a multiword NE consisting of more than two words, E-XXX: End of a multiword NE, XXX→PER/LOC/ORG/MISC. For example,

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

the name *sachin ramesh tendulkar* is tagged as *sachin*/B-PER *ramesh*/I-PER *tendulkar*/E-PER. The single word NE is tagged as, PER: Person name, LOC: Location name, ORG: Organization name and MISC: Miscellaneous name. In the output, sixteen NE tags are mapped to the four NE tags with some simple rules.

2.1. Approaches of NER in Bengali

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.

In this paper, we have used ME, CRF and SVM frameworks in order to identify NEs from a Bengali text and to classify them into *Person, Location, Organization* and *Miscellaneous*. We have developed two different systems with the SVM model, one using **forward parsing** that parses from left to right and other using **backward parsing** that parses from right to left. Lexical patterns, generated from an unlabeled corpus of 3 million wordforms, have been used to improve the performance of each of the classifiers. In addition, a number of post-processing techniques have been adopted in order to improve the performance of the classifiers. Finally, the classifiers are combined together with the three different weighted voting schemes.

We have used the C++ based ME package (http://homepages.inf.ed.ac.uk/s0450736/software/maxent/maxent-20061005.tar.bz2) and C^{++} based CRF++ package (http://crfpp.sourceforge.net) for NER. The SVM system has been developed based on (Jochims, 1999; Valdimir, 1995), which perform classification by constructing an N-dimensional hyperplane that optimally separates data into two categories. We have used *YamCha* toolkit (http://chasen-org/~taku/software/yamcha), an SVM based tool for detecting classes in documents and formulating the NER task as a sequential labeling problem. Here, the *pair wise* multi-class decision method and *polynomial kernel function* have been used. We have used TinySVM-0.07 (http://cl.aist-nara.ac.jp/~taku-ku/software/TinySVM) classifier that seems to be the best optimized among publicly available SVM toolkits.

2.2. Named Entity Features

- Context words: Preceding and following words of a particular word. This is based on the observation that the surrounding words are very effective in the identification of NEs.
- Word suffix and prefix: Word suffix and prefix information are helpful to identify NEs. A fixed length (say, *n*) word suffix/prefix of the current and/or the surrounding word(s) can be treated as feature(s). 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. Another way to use the suffix information is to modify the feature as binary valued. Variable length suffixes of a word can be matched with predefined lists of useful suffixes (e.g., *-babu*, *-da*, *-di* etc. for persons and *-land*, *-pur*, *-lia* etc. for locations). These features are useful to handle the highly inflective Indian languages as like Bengali.
- Named Entity Information (dynamic feature): NE tag(s) of the previous word(s).
- First word (binary valued): Current token is the first word of the sentence or not. First word is most likely a NE.
- Length of the word (binary valued): Length of the token is less than three or not. This is based on the observation that very short words are rarely NEs.
- Infrequent word (binary valued): A cut off frequency has been chosen in order to consider the infrequent words in the training corpus. 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. These binary valued features are helpful in recognizing miscellaneous NEs such as time, monetary and date expressions, percentages, numerical numbers etc.

•Position of the word (binary valued): Position of the word in a sentence is a good indicator of NEs. Generally, verbs occur at the last position of the sentence. This feature is used to check whether the word is the last word in the sentence.

•Part of Speech (POS) Information: We have used a CRF-based POS tagger (Ekbal et al., 2007c) that was originally developed with the 26 POS tags, defined for the Indian languages. For SVM based systems, we have used this POS tagger. However, for CRF and ME models, we have considered a coarse-grained POS tagset that has the following tags: Nominal, PREP (Postpositions) and Other. Postpositions are considered as these often appear after the NEs.

•Gazetteer Lists: Gazetteer lists, developed from the news corpus (Ekbal and Bandyopadhyay, 2008c), have been used as the features in each of the classifiers. These features can improve the performance of the classifiers when used as the features or used to post-process the outputs. Any particular gazetteer does not include the ambiguous entries, i.e., those that can appear in more than one gazetteer list. If the current token is in a particular list, then the corresponding feature is set to 1 for the current and/or the surrounding word(s); otherwise, it is set to 0. Following is the list of gazetteers along with the number of entries:

(1). Organization clue word (e.g., *kong, limited* etc): 94, Person prefixes (e.g., *sriman, sreemati* etc.): 245, Middle names: 1,491, Surnames: 5,288, Common location (e.g., *sarani, road* etc.): 547, Action verb (e.g., *balen*, *ballen* etc.): 241, Function words: 743, Designation words (e.g., *neta*, *sangsad* etc.): 947, First names: 72,206, Location names: 7,870, Organization names: 2,225, Month name (English and Bengali calendars): 24, Weekdays (English and Bengali calendars): 14

(2). Common word (521 entries): Most of the Indian language NEs appears in the dictionary with some other valid meanings. For example, the word *kamol* may be the name of a person but also appears in the dictionary with another meaning *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 prepared a list, containing the words that can be NEs as well as valid dictionary words.

(3). Lexicon (128,000 entries): We have used a lexicon that has been developed from the Bengali news corpus in an unsupervised way. The feature 'LEX' has value 0 for those words that appear in the lexicon; otherwise, the value is 1. This feature has been included as the words that appear in the lexicon are rarely NEs.

3. Unsupervised Lexical Pattern Learning from the Unlabeled Corpus

We have developed a method to generate the lexical context patterns from a portion of the unlabeled Bengali news corpus (Ekbal and Bandyopadhyay, 2008c) containing 3 million wordforms. 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.

(1). **Seed list preparation:** We have collected the frequently occurring words from a part of this Bengali news corpus 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*> *sonia gandhi* </*Person*>, <*Location*> *kolkata* </*Location*> and <*Organization*> *jadavpur viswavidyalya* </*Organization*>. 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> ...</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* (pattern), pattern *type* (*Person/Location/Organization*) 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 of patterns generated of that *type*). This table has 28,986 entries, out of which 17,031 patterns are distinct. We have also generated the context patterns by extracting the examples from the labeled training data of 272K wordforms and it yields 15,488 patterns. Finally, the set P has 21,233 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 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:

- *positive example*: The extracted entity is of the same NE *type* as that of the pattern.
- *negative example:* The extracted entity is of the different NE *type* as that of the pattern.
- *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. We then 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*.

(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 2-4 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 is 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 Set P are updated accordingly. Otherwise, the newly generated pattern is added to the set with the *type* and *relative frequency* fields set properly. The algorithm terminates after the 17 iterations and there are 27,098 distinct entries in the set P.

4. Evaluation Results and Discussions

We have manually annotated approximately 200K wordforms of the Bengali news corpus (Ekbal and Bandyopadhyay, 2008c) with *Person*, *Location*, *Organization* and *Miscellaneous* NE tags with the help of *Sanchay Editor*[3], a text editor for the Indian languages. Out of 200K wordforms, 150K wordforms along with the IJCNLP-08 shared task data has been used for training the models. Out of 200K wordforms, 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 2. A number of experiments have been carried out taking the different combinations of the 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**

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

parsing, and **backward parsing** are denoted by SVM-F, and SVM-B, respectively. The systems developed with these features are defined as the *baseline* models.

Table 2: Training, development and test set statistics

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

Evaluation results of the development set for the *baseline* models are presented in Table 3 in terms of recall (R), precision (P) and F-Score (FS). Evaluation results of the development set 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 word), 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 other features. The *baseline* CRF model has shown best performance (f-score=**75.71%**) for the context window of size five, POS information of the current and previous words along with the other set of features like ME. The SVM-F based *baseline* system has performed best among the three models and has demonstrated the f-score value of **76.3%** for the context window of size six, NE information of the previous two words, POS information of the current, previous and the next words along with the other set of features as like ME and CRF. The SVM-B has shown the f-score value of 76.1% with the same set of features used in SVM-F. In SVM models, we have conducted experiments with the different *polynomial kernel* functions and observed the highest f-score value with degree 2.

Evaluation results are reported in Table 3 by including the gazetteers to the *baseline* models. We have observed that all the gazetteers are not equally important to improve the performance of the classifiers. The use of gazetteers increases the performance by 2.49%, 4.11%, 4.45%, and 4.11% in the ME, CRF, SVM-F, and SVM-B classifiers, respectively. Evaluation results suggest that adding all the available features may not be always helpful to improve the performance in a ME framework as careful feature selection has an important role. On the other hand, CRF and SVM can avoid overfitting more efficiently and this fact is established by their performance improvement.

Table 3: Results of the development set for *baseline* models and by adding gazetteers

Baseline models				Baseline + Gazetteers		
Model	R (in %)	P (in %)	FS (in %)	R (in %)	P (in %)	FS (in %)
ME	73.57	73.07	73.32	76.09	75.53	75.81
CRF	75.97	75.45	75.71	79.03	80.62	79.82
SVM-F	77.14	75.48	76.30	81.37	80.14	80.75
SVM-B	77.09	75.14	76.10	81.29	79.16	80.21

4.1. Use of Context Patterns as Features

Patterns in the *Accept Pattern* set (discussed in section 3) can be used as the features of the individual classifier. Words in the left and/or the right contexts of person, location and organization names carry effective information that could be helpful in their identification. High ranked patterns, which are generated in a bootstrapped manner from the unlabeled corpus, contain these types context words. These words are used as the *trigger words* and very useful to identify the NEs. A particular *trigger* word may appear in more than one pattern *type*. A feature 'ContextInformation' is defined by observing the three preceding and following words of the current word. The feature is set depending upon the type of the *trigger* word. Experimental results of the system for the development set are presented in Table 4 by including the context features. Results show the effectiveness of context features with improvement of f-scores by **2.27%, 3.08%, 2.82%**, and **3.28%** in the ME, CRF, SVM-F, and SVM-B models, respectively.

Table 4: Results of the development set by adding context 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

4.2. Post-processing Techniques

We have performed error analysis for all the classifiers with the help of confusion matrices. In order to improve the performance of the classifiers, several post-processing techniques have been adopted depending upon the nature of errors involved. It has been observed that SVM models suffer most with the tendency of assigning NE tags to the non-NEs. Though SVM models perform better than CRF in terms of f-score, their precisions suffer. SVM performs better than ME with more than 5% f-score value but the rate of improvement of precision is less compared to recall. 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, B-PER 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 in the SVM models. Depending upon the errors involved in the models, we have adopted various mechanisms to improve the recall and precision values of the classifiers.

(1). **Class splitting technique for SVM**: Unlike CRF or ME, SVM model 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 2). This leads to the same situation like the *one-vs-rest* strategy. One solution to this unbalanced class distribution is to split 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 *POS*, we produce new |*POS*| classes, 'NNE-C'|C∈*POS* . 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 have shown 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 the SVM-B system.

(2). **Post-processing with the n-best outputs for CRF and ME**: There are inconsistent results in the CRF and ME models. 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 15 labeled sequences for each sentence with the confidence scores. Initially, we collect the NEs from the high confident results and then we re-assign 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, $S = \{s_1, s_2, ..., s_n\}$; R is set of n-best result (n=15) of S, i.e, $R = \{r_1, r_2, ..., r_n\}$, 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 jth 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_{i0} >=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) then Replace the NE tags of TempResult with these new tags}.
Result(s_i)=TempResult(s_i)}}.

Evaluation results have demonstrated the recall, precision, and f-score values of 84.32%, 81.31%, and 82.72%, respectively, in ME and 86.75%, 85.91%, and 86.33%, respectively, in the CRF model. Thus, these are the improvement of **4.64%,** and **4.43%** f-score in the ME, and CRF models, respectively.

(3). **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 predetermined threshold value then the second best tag is considered as the correct tag. We have heuristically determined the threshold values of confidence in each model by observing their effects on the evaluation results. This post-processing technique is executed after the techniques 1 and 2. Evaluation results are presented in Table 5. Results show 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 improvement of the performance in each of the models.

(4). **Use of gazetteers and lexicon for handling unknown words**: We have used person, location and organization name lists along with the lexicon to deal with the unknown words. If the confidence of the tag assigned to an unknown word is less than a predefined threshold value then its tag is determined by checking the gazetteers. This approach is followed only when the confidence of the second best tag is also below some threshold value (i.e., only when less than or equal to the thresholds used in technique 3). In some cases, an unknown word that is assigned the NE tag can also appear in the lexicon. The NE tag of such 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 as discussed in Section 2.2. Appropriate threshold values have been determined by observing the effects on the evaluation results in each of the models. Approximately, there are 21% unknown words in the development set. Experimental results are presented in Table 5.

Table 5: Evaluation results using second confident tags and unknown word handling

	With second confident tags			Unknown word handling		
Model	R (in %)	P (in %)	FS (in %)	R (in %)	P (in %)	FS (in %)
ME	88.53	80.52	84.33	89.46	81.92	85.44
CRF	89.64	85.03	87.27	90.03	86.18	88.06
SVM-F	90.82	86.01	88.35	91.01	87.23	89.08
SVM-B	90.63	85.73	88.11	90.99	86.97	88.95

4.3. Voting Techniques

Voting scheme becomes effective in order to improve the overall performance of any system. Here, we have combined four systems using three different voting mechanisms. In our experiments, 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 system. We can obtain the best weights if we could obtain the accuracy for the 'true' test data. However, it is impossible to estimate them. Thus, we have used following weighting methods in our experiments:

(1). **Uniform weights (Majority voting)**: We have assigned the same voting weight 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.

(2). **Cross validation f-score values**: 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 various gazetteers and 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. We have defined two different types of weights depending on the cross validation f-score as follows:

(a). Total F-Score: In the first method, we have assigned the overall average f-score of any classifier as the weight for it.

(b). Tag F-Score: In the second method, we have assigned the average f-score value of the individual tag as the weight.

Experimental results of the voted system are presented in Table 6. Results show that the system achieves the highest performance for the voting scheme 'Tag F-Score', which considers the individual tag f-score value as the weight of the corresponding system. Voting shows (tables 5-6) an overall improvement of the f-scores of **6.76%** over the least performing ME based system and **3.12%** over the best performing SVM-F system.

Table 6: Results of the voted system for the development set

Voting Scheme	R (in %)	P (in %)	FS (in %)
Majority	93.15	89.33	91.2
Total F-Score	93.78	89.91	91.8
Tag F-Score	93.82	90.24	92.2

4.4. Experimental Results of the Test Set

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 have shown the f-score values of 73.92%, 76.35%, 77.36%, and 77.23% in the ME, CRF, SVM-F, and SVM-B based systems, respectively. Evaluation results have demonstrated the fact that the use of gazetteers, context features and post-processing can improve the performance of each individual system by the impressive margins of 11.79%, 12.28%, 12.25%, and 12.19% in f-scores over the *baseline* ME, CRF, SVM-F, and SVM-B base systems, respectively. These post-processed systems are then combined together into a final system by applying three weighted voting approaches. Experimental results are presented in Table 7. 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' gives the best result among the three voting methods. The multi-engine system has demonstrated the improvement in the f-scores by 6.57%, 3.65%, 2.67%, and 2.86% in the ME, CRF, SVM-F, and SVM-B systems, respectively.

Table 7: Results of the voted system for the test set

Voting Scheme	R (in %)	P (in %)	FS (in %)
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

The most recent existing Bengali NER systems have been trained and tested with the same datasets. Evaluation results are presented in Table 8. Results have shown the effectiveness of the proposed multi-engine NER system that outperforms the other existing Bengali NER systems by the impressive margins. Thus, it can be decided that purely statistical approaches cannot yield very good performance always. Evaluation results also suggest that the contextual words along with their information and several post-processing methods can yield a reasonably good performance for each of the classifiers. Results also suggest that combination of several classifiers is more effective than the single classifier.

Table 8: Comparisons with other Bengali NER systems

Model	R (in %)	P (in %)	FS (in %)
HMM, Ekbal et al. (2007c)	74.02	72.55	73.28
CRF, Ekbal et al. (2008a)	80.02	80.21	80.15
SVM, Ekbal and Bandyopadhyay (2008b)	81.57	79.05	80.29
Voted System (proposed)	93.98	90.63	92.28

5. Conclusion

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. 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 incorporated 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 methods. The voted system has exhibited the improvement in f-scores by 18.63% over the least performing *baseline* ME system and 14.92% over the best performing *baseline* SVM based system. Future works include investigating the methods that will enable to reduce the errors that still exist because of the abbreviated names and short names. Also, we would like to conduct experiments with the other weighted voting methods.

References

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

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

Cucerzan, S., Yarowsky, D. 2002. Language Independent NER using a Unified Model of Internal and Contextual Evidence. In *Proceedings of the Sixth Conference on Natural Language Learning,* San Francisco, Morgan Kaufmann.

Ekbal, Asif, and S. Bandyopadhyay. 2007a. Lexical Pattern Learning from Corpus Data for Named Entity Recognition. In *Proc. of 5th ICON*, India, 123-128.

Ekbal, Asif, Naskar, Sudip 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 (2007), 95-114.*

Ekbal, Asif, R. Haque and S. Bandyopadhyay. 2007c. Bengali Part of Speech Tagging using Conditional Random Field. In *Proc. of 7th SNLP*, Thailand.

Ekbal, Asif, Haque, R and S. Bandyopadhyay. 2008a. Named Entity Recognition in Bengali: A Conditional Random Field Approach. In *Proc. of 3rd IJCNLP-08*, 589-594.

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

Ekbal, Asif, and S. Bandyopadhyay. 2008c. A Web-based Bengali News Corpus for Named Entity Recognition. *Language Resources and Evaluation Journal*, Vol. (40).

Florian, Radu, Ittycheriah, A., Jing, H. and Zhang, T. 2003. Named Entity Recognition through Classifier Combination. In *Proc. of CoNLL-2003*.

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

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

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

Munro, R., Ler, D., and Patrick, J. 2003. Meta-learning Orthographic and Contextual Models for Language Independent Named Entity Recognition. In *Proc. of CoNLL-2003*.

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

Wu, D., Ngai, G., and Carpuat, M. 2003. A Stacked, Voted, Stacked Model for Named Entity Recognition. In *Proc. of CoNLL-2003*.

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