Development of a Hindi Named Entity Recognition System without Using Manually Annotated Training Corpus

Sujan Kumar Saha[1] and Mukta Majumder[2]
[1]Department of Computer Science and Engineering, Birla Institute of Technology, India
[2]Department of Computer Science and Application, University of North Bengal, India

Abstract: *Machine learning based approach for Named Entity Recognition (NER) requires sufficient annotated corpus to train the classifier. Other NER resources like gazetteers are also required to make the classifier more accurate. But in many languages and domains relevant NER resources are still not available. Creation of adequate and relevant resources is costly and time consuming. However a large amount of resources and several NER systems are available in resource-rich languages, like English. Suitable language adaptation techniques, NER resources of a resource-rich language and minimally supervised learning might help to overcome such scenarios. In this paper we have studied a few such techniques in order to develop a Hindi NER system. Without using any Hindi NE annotated corpus we have achieved a reasonable accuracy of F-Measure 73.87 in the developed system.*

Keywords: *Natural language processing, machine learning, named entity recognition, resource scarcity, language transfer, semi-supervised learning.*

Received July 22, 2015; accepted October 7, 2015

1. Introduction

Machine Learning (ML) techniques have been extensively used in various sequence labelling tasks in Natural Language Processing (NLP) like Parts-Of Speech (POS) tagging and Named Entity Recognition (NER) etc., these techniques require a considerable amount of resources, the most important of which is annotated data for training ML classifier. Preparation of NLP resources is costly and time consuming. For some resource-rich languages like English, sufficient resources are available for various NLP tasks. But relevant and adequate resources have not been developed or openly available in most other regional languages like Hindi, Bengali, Arabic, Oriya, Telugu, Chinese, and Japanese etc. Development of NLP systems in such resource-poor languages can be leveraged by using the resources of the resource-rich languages with proper use of language adaptation techniques.

To develop NER systems in various languages and domains requires an ample amount of resources. But we wanted to inspect whether we can build up a NER system without using these resources. With the help of available English resources, and minimally supervised learning we have tried to build a Hindi NER system without using any language specific training data.

This paper presents our study on development of a Hindi NER system with the help of existing English NER system. We have prepared the baseline system using gazetteer look-up based approach. But we have not used any readily available Hindi gazetteer. By using the English NER systems we have created some relevant lists and proposed a two-phase transliteration system and transliterated these lists into Hindi. We have used these lists to prepare the baseline system.

Gazetteer based name identification suffers from several limitations; fails to identify inflected names, cannot resolve ambiguity etc., Few pre-processing and post processing steps have been used to avoid these limitations. A set of context patterns have been extracted and used to improve the performance of the system. In our study we have used three different machine learning classifiers; Maximum Entropy model (MaxEnt), Conditional Random Field (CRF) and Support Vector Machine (SVM), in order to obtain the confidence measure during semi-supervised learning and our intention is not to compare the performance of various classifiers. The baseline system has achieved high precision but suffered from poor recall. To improve the recall first we have used bootstrapping where recall is enhanced but the precision is degraded.

Then we have used active learning where the uncertain samples are selected and queried by the system to human annotator. With these approaches we have been able to develop a Hindi NER system with moderate performance. The highest accuracy obtained in the system is F-Measure of 73.87 (in CRF classifier using active learning based query by committee) with 83.93% precision and 65.96% recall.

The rest of the paper is organised as follows. Related previous work is described in section 2. Next baseline NER system using gazetteer and context pattern is discussed in section 3. Section 4 presents the Bootstrapping based improvement of the NER system. Active Learning based enhancement of accuracy of the system is described in section 5. The conclusion is drawn in section 6.

2. Related Previous Work

In this section we have discussed some previous works which are related to our study.

2.1. NER Task: General and Indian Language Specific

In the last few years several research works have been carried out for developing NER systems in different languages and domains using ML algorithms. Hidden Markov Model (HMM) [4, 10, 19, 20], Maximum Entropy Model (MaxEnt) [6, 8, 43], Conditional Random Field (CRF) [14, 30, 31, 45], SVM [2, 24, 50, 52] etc., are the most commonly used ML classifier.

ML methods are also mainly used for NER system development in Indian languages. Due to several language specific issues like, absence of capitalization, free word order, inflection in NE, ambiguity in names and unavailability of sufficient resources, the task is quite difficult and NER systems in Indian language have not achieved the high accuracy as English.

A successful work on Hindi NER was done by Li and McCallum using CRF and feature induction [31].

After that several attempts have been taken for developing NER systems in Indian languages; Kumar and Bhattacharyya developed a MEMM based Hindi NER system [28]; Ekbal and Bandyopadhyay and Ekbal and Saha developed NER systems in Indian languages like Hindi and Bengali [16, 17]; Saha *et al*. performed several experiments with different ML techniques for developing Hindi NER systems [41, 42, 43]; Sharma and Goyal used CRF for designing Hindi NER system [46]; Gupta and Lehal developed a list based NER system in Punjabi [21]; and Kaur *et al*. developed a CRF based Punjabi NER system [23]. In IJCNLP 2008 a shared task[1] was organized on identification of NEs in five south and south-east Asian languages (Bengali, Hindi, Oriya, Telugu and Urdu). A number of systems have participated in that task.

Among these the best result, which is an F-Measure of 65.13 for Hindi and 65.96 for Bengali, was achieved by [43, 47]. All the systems mentioned here uses annotated training data and other language specific resources.

[1]More information on the shared task is available at: http://ltrc.iiit.net/ner-ssea08/index.cgi

2.2. Minimally Supervised Learning in NER

For NER system development, using ML based classifiers; training data plays a major role. When sufficient training data is not available, the performance of the system can be improved with semi-supervised learning (SSL). We have found some related works which used SSL for NER system development [1, 13, 18, 37].

But when there is no availability of annotated data for preparing the initial classifier; the task becomes more complicated. The techniques like, learning rules or context patterns starting from a few seed entities or rules, learning through gazetteer, clustering, relevant query based web search etc. have been explored to build NER systems in such scenarios. A few NER systems which are developed using no (or very less) annotated data are discussed below.

Use of context patterns for finding the names from a text is quite popular. In this method initially a small set of patterns or rules are identified manually or, semi-automatically. Then the pattern set is enhanced using some pattern extraction techniques. Collins and Singer used only seven seed rules to develop an unsupervised NER system with a reasonable accuracy [11]. Cucchiarelli and Velardi presented a minimally supervised NER system, where they started with a small set of names and then used syntactic and semantic cues to develop a complete system [12].

Several other systems used context pattern based enhancement of classifiers [32, 39].

Preparation of classifier through gazetteers is another widely used approach, which is used in a number of minimally supervised NER systems [25, 36].

Use of web information is another option for building NER systems in resource scarce scenario. By using the web resources, Etzioni et al. developed an unsupervised NER system, KnowItAll [18]. Romcke and Johansson developed another system which also used web information [40]. We have found another system which identifies Turkish NEs using Wikipedia along with K-nearest neighbour algorithm [27].

But it is not easy to use such techniques in Hindi due to several language specific challenges. Use of gazetteers in Hindi is quite difficult due to the high ambiguity and rich morphology, also relevant and sufficient gazetteer lists are still not available. The context pattern learning techniques primarily use the capitalization and grammatical information like parse and parts-of-speech information etc. In Hindi deep parser with good accuracy is very rare. The capitalization information is not available in Hindi. So context pattern extraction is difficult. Since the word order is relatively free in Hindi, the context patterns might not be reliable and may identify false NEs. Use of Indian languages in the web is low, so it is not

possible to build an unsupervised NER system using only web information.

2.3. Language Adaptation

Cross language knowledge induction or transfer of technology from one language to another is an effective approach for preparing a NER system without using language specific resource. A NER system in a resource-poor language can be built with the help of a NER system of resource-rich language. Such few language adaptation approaches are discussed below.

Use of parallel text is a popular approach of language transfer. In this approach an existing model in a resource-rich language is applied to a bilingual corpus and the output is projected onto the target language via statistically derived word alignments. The annotation projection generally becomes erroneous and incomplete, so several pre-processing and post-processing modules have been used to improve the annotation. This technique has been used in various sequencing and labelling tasks [5, 15].

Language transfer can also be done without using any parallel or word aligned corpus. Carreras et al. developed a NER system in Catalan using the resources of a parallel language Spanish, where they have not used any training data for Catalan [7]. Solorio and Lopez used similar approaches for transferring Spanish NER system to Portuguese [48]. Maynard *et al*. [33] developed a NER system from English to Cebuano without any training data in Cebuano and using only four person-days effort they achieved an f-score of 69.1. Hana *et al*. [22] presented a method for transferring Spanish tagger to Portuguese with the help of cognates. Kim and Khudanpur [26] also studied a few statistics for cross lingual language model adaptation.

Pedersen *et al*. [38] presented another interesting study where they observed that the ambiguous name discrimination in resource-poor languages like Bulgarian, Romanian, and Spanish can be done by using resource-rich language like English. They named their technique as language salad.

From these studies we hypothesize that although we don't have any NE annotated data and other NLP resources for Hindi, we can build a Hindi NER system with the help of aforementioned techniques.

3. Baseline NER System using Gazetteer and Context Pattern

In this study we have attempted to develop a Hindi NER system without using any Hindi NE annotated data. To build the initial classifier, we plan to use the gazetteer look-up based approach. But in the study we have not used any readily available Hindi gazetteer list.

First we have attempted to prepare some relevant name lists with the help of existing English NER systems where we have used the Stanford[2] and the LingPipe[3] NER System. In our study we have considered two NE classes, person and location.

3.1. Preparation of Gazetteer List

Our basic idea is to run the English NER system on a large English corpus and extract a list of names; then use this list to identify names from Hindi documents.

Domain agreement plays an important role here. The English corpus from which the name list will be extracted and the Hindi corpus on which the list will be applied; should be from same domain. If the source and target domain differs, the prepared gazetteer lists will be unable to detect sufficient NEs. For this study we have taken the FIRE 2010[4] English and Hindi data.

We have taken an English raw corpus containing 5000K words and applied the English NER system on it. From this annotated corpus we have extracted the person and location NEs and compiled the corresponding gazetteer lists. These English gazetteer lists are required to transliterate into Hindi for using in the Hindi NER task; for this purpose we propose a two-phase transliteration process.

3.2. English-Hindi Transliteration System

We want to use the English gazetteer lists to identify the names from the Hindi corpus. As our objective is to make the decision that a particular Hindi string occurs in an English gazetteer list or not, we need not transliterate the English names into Hindi. Instead our idea is to define an intermediate phonetic alphabet.

Both the English and Hindi strings will be translated to the normalized form. For an English-Hindi string pair, if both the strings are translated to same normalized string then we can conclude that one string is the transliteration of the other. For this purpose first we need to decide the size of the intermediate alphabet. Preserving the phonetic properties we have defined our intermediate alphabet consisting of 34 characters.

3.2.1. Translation from English to Intermediate Alphabet

For translating the English strings into normalized form, we have built a phonetic map table. This table maps an English character n-gram into an intermediate character. A part of the map table is given in Table 1. The map table presents the mapping of an English character n-gram to an intermediate alphabet. In our table the length of the English character n-gram varies from 1 to 3. The procedure of translation using the map table is presented in Algorithm 1.

[2]http://nlp.stanford.edu/ner/index.shtml
[3]http://alias-i.com/lingpipe/web/models.html
[4]http://www.isical.ac.in/fire/data_download.html

Table 1. Partial map table of english to intermediate.

English	Intermediate	English	Intermediate
a	â	e	ê
ee, i, ii	î	o	ô
oo, u	û	ou	ॊ
k	क	kh, ksh	क्ष
g	ग	gh	घ
ch	च	j	ज
b, w	ब	bh, v	भ
r, rh	र	sh, s	श

Algorithm 1: English to intermediate translation Source (S)- English, Output (T)- Intermediate

1. Scan the source string (S) from left to right
2. Extract the first n-gram (G) from the string (n = 3)
3. Find if it is in the map-table
4. If yes, insert the corresponding normalized character into the target string T
Remove the n-gram from S, S = S-G
Go to step 2
5. Else, set n = n-1
Go to step 3

3.2.2. Translation from Hindi to Intermediate

For this translation we have taken the help of 'itrans transliteration'. Itrans is representation of Indian language alphabets in ASCII. Since Indian text is composed of syllabic units rather than individual alphabetic letters, itrans uses combinations of two or more letters of English alphabet to represent an Indian language syllable. However, there being multiple sounds in Indian languages corresponding to the same English letter, not all Indian syllables can be represented by logical combinations of English alphabet. Hence, itrans also uses some non-alphabetic special characters for such syllables. First the Hindi strings are translated into itrans using the itrans map table[5]. After that the itrans strings are translated into the intermediate state using the similar procedure as followed by the English to intermediate transliteration.

3.2.3. Evaluation of the Transliteration System

The two-phase transliteration system presented above is designed using a phonetic intermediate alphabet. The system is unable to produce the Hindi transliteration of any English name or the reverse. We have developed the system only for using the English gazetteer lists in the Hindi name recognition task. Additionally, the system is able to handle various spelling variations of a particular name. For example, an Indian name 'surabhi' may have several variations (while written in English) like 'suravi', 'shuravi', 'surabhee', 'shurabhi' etc. Our transliteration system converts the Hindi 'surabhi' and all its English variations into the intermediate string '*suravi*'. The system has a few limitations, like, sometimes two different strings can be mapped in to a same intermediate alphabet string (e.g., '*ghAnA*[6].' is a location entity and '*ghana*' is a not-name word, both are mapped to a same intermediate string), it cannot handle translation in transliteration (e.g., 'India' in English and '*bhAratabarSha*' in Indian language).

For the evaluation of the transliteration system we have created a bi-lingual test set containing 520 English-Hindi word pairs of person and location names and applied it on the test set. 496 of these NEs are transliterated correctly by the system. Therefore the accuracy of the system is 95.38%.

3.3. Gazetteer Based Identification

We have taken a Hindi raw corpus containing ~2000K words, annotated using the prepared name lists. But the gazetteer based NE identification is not so simple; due to several language specific issues; it suffers from several shortcoming and requires special attention.

Our observations on the gazetteer look-up based NE identification in Hindi are summarized below.

- As the gazetteer lists are not sufficiently large, many NEs are not detected.
- The English NER system has misclassified several names of the raw text. These erroneous NEs will take place in the lists and can lead to false-identification. So we have tried to make the prepared gazetteer lists error free by using two NER systems (Stanford and LingPipe NER system) on the same text and considering a name as correct only if both the systems have identified it.
- A lot of Indian names are ambiguous. Many common words are used as names. Ambiguity between nouns and NEs is observed in many languages. But in Hindi ambiguity occurs between names and adjectives, verbs and other parts-of-speech categories also. For example, neela (blue), sambhaba (possible), nabIna (new) etc., are adjectives but also used as person NEs. The gazetteer based NE identification identifies a number of such common words as name. It is difficult to resolve these completely due to the absence of capitalization in Hindi. We use a Hindi POS tagger to minimize the common word ambiguities. When the POS category of a word is not noun but detected as name, then we mark this as 'uncertain' identification. A number of ambiguities can be resolved with the help of the POS information but this cannot detect the noun-NE ambiguities (as the names are also noun).
- Noun-NE ambiguities also occur in the person first name, like, AkAsha (sky), sandhyA (evening), mAyA (kindness), kiraNa (light/ray). We observe that a number of person NEs contain some clue information which can help to detect the

[5]The map table is available at www.aczoom.com/itrans

[6]In the paper all Hindi words are written in italics using the Itrans transliteration.

ambiguities. For example, designation or title words often precede the person names; the honorary terms like jI (a widely used honorary term), bAbu (sir) etc., or relation words like bhAiyA (brother), chAchA (uncle) etc., occur after the person names; and the surnames follow the first names. If a gazetteer based identified NE contains any such clue then we mark this as a 'confident' identification. Similarly the clue lists like common location terms, common location suffixes (nagar, pur, gram etc.,) help to detect ambiguities in the location class.

- Apart from the ambiguity between names and common words, there are the ambiguities between inter NE classes. The clue lists help to resolve some of these ambiguities, but a number of annotation errors still remain.

3.4. Extraction of Training Data and Learning Classifier

Now we plan to build a ML based classifier using the gazetteer based annotated training data. But in the corpus many annotations are erroneous. Hence if we use these total corpora for training the classifier, it will be of poor quality. To make a better classifier we select a confident portion from this corpus. In this way we select a corpus containing ~280K words that contains ~4900 person and ~7400 location NEs.

3.4.1. Classifiers Used

For these experiments we have used three different classifiers, maximum entropy model, conditional random field and support vector machine [3, 29, 51]. In SVM we have used the polynomial kernel function.

3.4.2. Feature

To train the classifiers we have selected a list of features following the feature set defined in [43].

- *Word feature*: Word feature is widely used to develop NER system. Current words along with preceding and next words are used in this system. We have used word window of size 3, 5 and 7.
- *Context information*: These are the words which occur frequently in a target word window. For example, *bAbu* (sir), *chAchA* (uncle), *bhAiyA* (brother), *rAjadhAnI* (capital), *deshA* (country) etc.
- *Affix Feature*: Affix feature is extremely significant to identify NEs. We have used prefix and suffix of variable length (2 and 3) for training purpose of our NER system.
- *Parts-of-speech information (POS) feature*: For NER system development Part-Of-Speech (POS) information is also an important feature. Mainly the POS of the target word and its surrounding words are used in our system.
- *Numeric feature*: We have used numerical information based feature, like word contains any number or denotes numeric value.
- *Surrounding NE tag*: Named entity tags of the previous and next words are used as features.

3.5. Context Pattern Extraction and Integration

Next we extract a set of context patterns for each name category using the pattern extraction methodology discussed in [43]. The extracted patterns are searched in the Hindi raw corpus. These patterns identify a number of NEs which were not identified by the gazetteers.

A particular NE might have several occurrences; if any of these matches a context pattern then the others can also be identified. Hence to improve the identification, these new NEs are included in the gazetteer lists. Then the gazetteer and pattern based outputs are combined. From the combined corpus we select the confident portion as we did previously. The selected corpus now contains ~360K words including ~7500 person and ~9900 location names.

We observe that in a number of entities the patterns predict a different class label compared to the gazetteer based label. Sometimes it is due to the inter-class ambiguity. The patterns generally predict the correct category as the identification is based on the context information. We also observe that for a number of instances the gazetteer identified entity is a subset of the pattern identified entity. In such scenarios we consider the pattern based output as the correct label.

3.6. Experimental Results: Baseline System

We present the results obtained in the Hindi baseline NER system using gazetteers and patterns based approaches, discussed above. For the performance evaluation of the system we have manually annotated a test corpus containing ~30K words. The test corpus contains 792 person and 768 location NEs. Among these 598 location NEs are single words and 438 person NEs are multi words.

In Table 2 we have presented the accuracies obtained in different stages of baseline classifier preparation. The accuracy is measured in terms of f-measure, which is the harmonic mean of precision and recall. $F= (1+\beta^2)$ (precision×recall)/(β^2×precision+ recall) Precision is the percentage of the correct annotations and recall is the percentage of the total NEs that are successfully annotated. The value of β is taken as 1. The accuracies are measured using the 'exact match' strategy, that is, if there is a match in both the category and boundary then only it is considered as correct identification. Due to the use of insufficient resources, several NEs are not identified

completely. Such identifications are considered as incorrect. First we have computed the accuracy of the gazetteer based identification. This achieves an f-measure of 36.81 with 96.21% precision and 22.76% recall. We get better performance by using POS information. The accuracy is better for location class than person NE. The location class achieves an f-measure of 45.16 with 97% precision and 29.43% recall. In the person class several NEs are partially identified which are not considered during the accuracy computation. The f-measure of the person class is 27.8 with 94.85% precision and 26.29% recall. A high portion of the location NEs are single word entity, so the boundary mismatch is not a major problem.

The gazetteer based annotated corpus is then used to learn the classifiers. We have already mentioned that from a large gazetteer based annotated corpus we have selected the high confidence sentences for the training purpose. We have experimented with different combinations of the features defined in Section 3.4.2 and selected the best feature set. These experiments are similar to the baseline experiments conducted in [42]. In our experiments we found that a feature set containing a word window of length three, NE tag of the previous word, suffixes of length four, prefixes of length three and parts-of-speech information gives the best accuracy. When this feature set is used the CRF classifier achieves an f-score of 46.15 with 85.94% precision and 31.54% recall. The MaxEnt and SVM classifiers achieve f-scores of 42.6 and 42.48 respectively. The ML classifiers achieve high precision but recall is poor. In our baseline experiments we have observed that CRF outperforms MaxEnt and SVM.

Table 2. Performance of baseline classifiers.

Classifier	Precision	Recall	F-Score
Gazetteer based identification	96.21	22.76	36.81
Gazetteer with POS	95.97	26.41	41.46
Classifier using gazetteer based corpus: CRF	85.94	31.54	46.15
Classifier using gazetteer based corpus: MaxEnt	84.62	28.46	42.60
Classifier using gazetteer based corpus: SVM	86.70	28.14	42.48
Pattern based identification	98.75	25.26	40.22
Classifier using pattern integrated corpus: CRF	88.69	34.68	49.86
Classifier using pattern integrated corpus: MaxEnt	86.28	32.25	46.94
Classifier using pattern integrated corpus: SVM	86.52	32.0	47.25

The patterns which identify NEs with higher precision are used to improve the training corpus. From our test corpus the patterns recognize the NEs with 98.75% precision and 25.26% recall. When this corpus is used to train the classifiers, we get better accuracy.

The CRF classifier identifies a total of 610 names, out of 1560 in the test corpus, among which 541 entities are correctly identified. Therefore the CRF classifier achieves an f-score of 49.86 with 88.69% precision and 34.68% recall. The MaxEnt and SVM classifiers identify 583 and 586 names respectively with f-measure 46.94 and 47.25 respectively (see Table 5).

4. Bootstrapping the Initial Classifier

Next we plan to use Semi-Supervised Learning (SSL) to improve the baseline classifier. SSL is much cheaper compared to the supervised learning in terms of labelled training data. Recently ML researchers have paid more attention to these SSL techniques like, bootstrapping, active learning etc. to reduce the need for huge annotated training data.

First we use the bootstrapping technique, which is presented in Algorithm 2. It is well-motivated in many modern ML problems where sufficient labelled training data is not available. In this learning process, a model is trained on a previously annotated data set, and then it classifies an unlabelled set of data to get self-labelled data. Then the confidence of labelling is measured and the high confidence portion is extracted and added to the original training data set. That is why bootstrapping is also called "self-training," procedure. Bootstrapping is applicable when the existing supervised model fails to produce moderated result and complicated or hard to modify. We have found some related works of bootstrapping technique in newsier [11] and biomedical domain [34, 49].

Algorithm 2: The bootstrapping procedure

(1) Run a classifier (C) using available training data (L)
(2) Apply C on a raw corpus (U) to obtain labelled data U1
(3) Find confidence of annotation of each sample in U1
(4) Find the high confident samples (H)
(5) Add H to L : L1 = L + H
(6) Train a new classifier C1 using L1
(7) Repeat steps 2 to 6 while C1 (new) is better than C (old)

For bootstrapping we need to calculate the confidence of the annotations for selecting the samples to be added with the training data. A new classifier is trained using the extended training corpus. This process is performed repeatedly until the classifier converges. For preparing the Hindi NER system we have used three different ML techniques, MaxEnt, CRF and SVM. If, for a word all these classifiers predict same class, then we consider the prediction as confident.

As our training data is not created manually, the data itself contains several annotation errors. Many NEs are not annotated or partially annotated. Due to the poor quality of the training data we have not achieved good accuracy in the baseline system in spite of using a corpus of reasonable size. To improve the training corpus, first we have applied all the three classifiers to predict the training data itself by using 10-fold cross validation technique. The training corpus is partitioned into ten subsets, a classifier is trained using nine subsets and the rest one is predicted by the classifier. We observe that a number of additional NEs

are identified in the process. From these the confident NEs are marked to improve the training data. The classifiers are then retrained using the improved training data.

Then we have applied bootstrapping using a large raw corpus. We mark an annotation as uncertain if any of the classifiers disagree on it or this is in conflict with the POS category or pattern. From this corpus we have selected the sentences that contain at least one NE and any word is not marked as uncertain. These sentences are then added to the training corpus and the classifiers are retrained. The process is repeated while the performance of the classifier is increasing.

4.1. Experimental Results: Bootstrapping

In Table 3 we have summarized the results obtained using the CRF based bootstrapping. When the corpus is extended through self prediction, the f-score is improved to 53.81 in CRF. To apply bootstrapping, we take a large raw corpus which is partitioned into a number of subsets of ~200K words. In this process we annotate one of these subsets with the classifier and extract the confident names to extend the training data and build a new classifier and the process is repeated.

Table 3. Performance of bootstrapping.

Classifier	Precision	Recall	F-Measure
Baseline CRF classifier	88.69	34.68	49.86
Self improved corpus	86.28	39.10	53.81
Bootstrapping 1 iteration	84.31	41.35	55.49
Bootstrapping 2 iterations	80.48	43.33	56.33
Bootstrapping 3 iterations	79.8	45.06	57.6
Bootstrapping 16 iterations	73.94	53.06	61.78
Bootstrapping 17 iterations	73.39	53.81	62.09
Bootstrapping 18 iterations	73.23	54.36	62.45
Bootstrapping 19 iterations	72.97	54.49	62.39

After the first iteration we extract a confident portion containing ~24K words which are added to the training data. The CRF classifier with this extended corpus achieves an f-score of 55.49 with 84.31% precision and 41.35% recall. Running bootstrapping repeatedly we have achieved the highest f-score of 62.45 with 73.23% precision and 54.36% recall in eighteen iterations. Here a total of 1158 entities are identified among which 848 are actual NEs. In the next few iterations, we observed that although the recall values are increasing the f-measure decreases due to much decrease in precision.

Similarly using the MaxEnt and SVM classifiers we achieve performance improvement with bootstrapping. In these classifiers also after a few iterations the accuracy degrades due to the fall in precision. In MaxEnt 1062 entities are identified in which 773 are correct. Hence the precision, recall and f-score are 72.78%, 49.55% and 58.97 respectively. With bootstrapping the highest f-score achieved in SVM classifier is 60.13 with 72.28% precision and 51.48% recall (see Table 5 for statistics).

It is obvious that an error free corpus cannot be prepared using the approaches we have followed where manual annotation is not used. As it is not possible to create a name list of all the names, the gazetteer based identification has suffered from poor accuracy but the precision is quite high. So we have used the bootstrapping approach to enhance the corpus and some additional names are recognized but the precision is gradually degraded. As the precision becomes poor, bootstrapping like techniques cannot be continued as the erroneous entities will lead high identification error.

5. Improving Classifier By Active Learning

Then we have used Active Learning, another popular SSL technique which reduces the annotation effort by effective sampling of unlabeled data. In active learning the most uncertain samples from the raw data are selected and given to the human teacher to annotate and added to the existing training corpus. It is important in active learning to select the samples to be annotated by the human teacher. To minimize the human effort, the selected samples should be 'good' and should meet certain criteria. A number of approaches have been used for this purpose, for example, uncertainty sampling [9], query by committee [44], redundant view [35] etc.

5.1. Refining Training Corpus through Active Learning

To remove the annotation errors from training data we have used the active learning framework with 10 fold cross validation where the training corpus is partitioned into ten equal subsets and the classifier is trained using nine of these and the remaining one is kept for testing. The uncertain samples from the test portion are selected and annotated manually. Algorithm 3 presents the Active Learning process.

Algorithm 3: Active learning for corpus refining

(1) Split training data (L) into n parts (L_1, L_2, ..., L_n)
(2) Repeat steps 3 to 8 for n times (i = 1 to n)
(3) Select L_i as test data and combine rest n-1 parts as training data (L_{rest})
(4) Learn classifier (C_i) using L_{rest}
(5) Apply C_i on L_i
(6) Find the most uncertain samples (S_i) from L_i
(7) Ask the teacher (human) for the labels of S_i
(8) Add S_i to L_i with their new labels by replacing the old
(9) Merge L_1, L_2,.., L_n to form new training corpus L'

To select the uncertain samples we have used query by committee principle and three different classifiers, MaxEnt, CRF and SVM. These three classifiers predict three class labels for a particular word. Apart from this we have another class label, obtained from baseline system. If these four labels are not same then we consider the label as uncertain. We query the uncertain samples along with their context (previous

and next words) for annotation. Sometimes it becomes difficult to predict the class of a NE by observing a word-window three due to ambiguities and complex NEs. In such cases a larger word-window is provided.

Next we compare query by committee with other sample selection strategies, random sampling and selection based on classifier confidence. In classifier confidence sampling we have considered the confidence score of CRF as in our experiments it performed better than others. It gives the conditional probability of the most likely label sequence for an observation sequence from where we select the least confident one from the most likely label sequence.

5.2. Experimental Results: Active Learning

We have previously mentioned that the Hindi baseline corpus contains ~360K words with ~7500 person and ~9900 location NEs. During bootstrapping we have added more data to the original corpus.

We have applied active learning based ten-fold cross filtering on the total corpus. From each fold we have selected the uncertain samples based on classifier committee and manually annotated these. The active learning algorithm has selected a total of 2600 queries which are manually checked and the annotation errors are corrected using only two person-days effort. This improved corpus is now used to build the classifiers.

A CRF classifier, when trained using this new corpus, achieves an f-score of 73.87 with 83.93% precision and 65.96% recall. This is the highest accuracy we have achieved. We have also compared the query by committee based active learning with random sampling and selection based on classifier confidence. In our experiments random sampling is not performing well. The classifier confidence based strategy works better. The details result of the CRF based active learning strategies are shown in Table 4.

Table 4. Performance of the active learning using CRF.

Classifier	Precision	Recall	F-Measure
Query by committee (After 1st fold)	74.11	55.27	63.32
Query by committee (After 9th fold)	83.07	64.88	72.86
Query by committee (After 10th fold)	83.93	65.96	73.87
Random selection (After 10th fold)	79.22	59.61	68.03
Classifier confidence (After 10th fold)	84.03	64.42	72.93

In the MaxEnt and SVM classifiers we also achieve good accuracy using active learning. The final f-score in MaxEnt classifier is 69.29 with 80.25% precision and 60.96% recall. In the SVM classifier the f-score is 70.47 with 80.9% precision and 62.43% recall. In Table 5 we have shown the detail results of the NER system for each classifier in various stages of the development.

Table 5. NER Result in various stages of development.

Classifier	Total	True	F-Measure
Gazetteer based identification	360	355	36.81
Pattern based identification	399	394	40.22
CRF Baseline	610	541	49.86
MaxEnt Baseline	583	503	46.94
SVM Baseline	586	507	47.25
CRF Bootstrapping	1158	848	62.45
MaxEnt Bootstrapping	1062	773	58.97
SVM Bootstrapping	1111	803	60.13
CRF Active learning	1226	1029	73.87
MaxEnt Active learning	1185	951	69.29
SVM Active learning	1204	974	70.47

Table 6. Comparison with other hindi NER systems.

System	Technique Used	Training Data Size	Resources Used	F-Measure
Saha et al. [42]	MaxEnt, CRF with feature reduction	200000 tokens	Name lists, clues, POS	85.31
Sharma and Goyal [46]	CRF with feature selection	503179 tokens	Several gazetteer lists	70.45
Ekbal and Saha [17]	GA, classifier group & feature selection	503179 tokens	7 gazetteer (total 168K entities), chunk, POS	89.65
Singh [47]	Hybrid technique	503179 tokens	Gazetteers, rules, clues, patterns	65.13
Proposed System	Gazetteer, pattern, 3 ML Technique, SSL	NIL	Only a few clues and POS	73.87

6. Conclusions

We have developed a Hindi NER system without using any manually annotated training corpus. We have primarily used language transfer and minimally supervised learning. The baseline system is prepared using gazetteer and context patterns. Gazetteers for Hindi are prepared using English NER system, large raw corpus and transliteration. For the transliteration we have proposed a two-phase transliteration technique. The baseline classifier achieves high precision but suffers from poor recall. To improve the recall we have used the bootstrapping and active learning techniques. In CRF classifier using active learning based query by committee approach we have obtained the highest accuracy of F-Measure of 73.87.

It will not be fair to compare the performance of the proposed system with state-of-the-art Hindi NER systems; those using large training corpus and other language specific resources. In Table 6 we have listed a few Hindi NER systems with their usage of language specific resources. As these systems used different NE classes, different datasets and various language specific resources, this comparison is just to show that the proposed methodology is effective. The effectiveness of the proposed approach encourages us to develop NER systems in other resource poor languages; we have planning to work on such systems in future.

References

[1] Becker M., Hachey B., Alex B., and Grover C., "Optimising Selective Sampling for Bootstrapping Named Entity Recognition," *in Proceedings of ICML Workshop on Learning with Multiple Views*, pp. 5-11, 2005.

[2] Benajiba Y., Diab M., and Rosso P., "Using Language Independent and Language Specific Features to Enhance Arabic Named Entity Recognition," *The International Arab Journal of Information Technology*, vol. 6, no. 5, pp. 464-473, 2009.

[3] Berger A., Pietra V., and Pietra S., "A Maximum Entropy Approach to Natural Language Processing," *Computational Linguistic*, vol. 22, no. 1, pp. 39-71, 1996.

[4] Bikel D., Miller S., Schwartz R., and Weischedel R., "Nymble: High-Performance Learning Name-Finder," *in Proceedings of 5th Conference on Applied Natural Language Processing*, Washington, pp. 194-201, 1997.

[5] Borin L., "Briefly noted: Parallel Corpora, Parallel Worlds," *Computational Linguistic*, vol. 29, no. 1, pp. 149-151, 2003.

[6] Borthwick A., A Maximum Entropy Approach to Named Entity Recognition, Thesis, New York University 1999.

[7] Carreras X., Màrquez L., and Padró L., "Named Entity Recognition for Catalan Using Spanish Resources," *in Proceedings of 10th Conference on European Chapter of the Association for Computational Linguistics*, Budapest, pp. 43-50, 2003.

[8] Chieu H. and Ng H., "Named Entity Recognition: a Maximum Entropy Approach using Global Information," *in Proceedings of 19th International Conference on Computational Linguistics*, Taipei, pp. 1-7, 2002.

[9] Cohn D., Atlas L., and Ladner R., "Improving Generalization with Active Learning," *Machine Learning*, vol. 15, pp. 201-221, 1994.

[10] Collier N., Nobata C., and Tsujii J., "Extracting the Names of Genes and Gene Products with a Hidden Markov Model," *in Proceedings of 18th Conference on Computational Linguistics*, Saarbrücken, pp. 201-207, 2000.

[11] Collins M. and Singer Y., "Unsupervised Models for Named Entity Classification," *in Proceedings of Joint SIGDAT Conference on Empirical Methods in Natural Language Processing and Very Large Corpora*, pp. 100-110, 1999.

[12] Cucchiarelli A. and Velardi P., "Unsupervised Named Entity Recognition using Syntactic and Semantic Contextual Evidence," *Computational Linguistics*, vol. 27, no. 1, pp. 123-131, 2001.

[13] Cucerzan S. and Yarowsky D., "Language Independent Named Entity Recognition Combining Morphological and Contextual Evidence," *in Proceedings Joint SIGDAT Conference on Empirical Methods in Natural Language Processing and Very Large Corpora*, pp. 90-99, 1999.

[14] Das A. and Garain U. "CRF-based Named Entity Recognition@ ICON 2013," Carnell University Library, arXiv preprint arXiv:1409.8008, 2014.

[15] Dien D. and Kiem H., "POS-Tagger for English-Vietnamese Bilingual Corpus," *in Proceedings of the HLT-NAACL*, Edmonton, pp. 88-95, 2003.

[16] Ekbal A. and Bandyopadhyay S., "A Hidden Markov Model based Named Entity Recognition System: Bengali and Hindi as Case Studies," *in Proceedings of International Conference on Pattern Recognition and Machine Intelligence*, Kolkata, pp. 545-552, 2007.

[17] Ekbal A. and Saha S., "Classifier Ensemble Selection Using Genetic Algorithm for Named Entity Recognition," *Research on Language and Computation*, vol. 8, no. 1, pp. 73-99, 2010.

[18] Etzioni O., Cafarella M., Downey D., Popescu A., Shaked T., Soderland S., Weld D., and Yates A., "Unsupervised Named Entity Extraction from the Web: An Experimental Study," *Artificial Intelligence*, vol. 165, no. 1, pp. 91-134, 2005.

[19] Gayen V. and Sarkar, K. "An HMM based Named Entity Recognition System for Indian Languages: the JU System at ICON 2013," Carnell University Library, arXiv preprint arXiv:1405.7397, 2014.

[20] GuoDong Z. and Jian S., "Exploring Deep Knowledge Resources in Biomedical Name Recognition," *in Proceedings of the International Joint Workshop on Natural Language Processing in Biomedicine and its Applications*, Geneva, pp. 96-99, 2004.

[21] Gupta V. and Lehal G., "Named Entity Recognition for Punjabi Language Text Summarization," *International Journal of Computer Applications*, vol. 33, no. 3, pp. 28-32, 2011.

[22] Hana J., Feldman A., Brew C., and Amaral L., "Tagging Portuguese with a Spanish Tagger using Cognates," *in Proceedings of International Workshop on Cross-Language Knowledge Induction*, Trento, pp. 33-40, 2006.

[23] Kaur A., Josan G., and Kaur J., "Named Entity Recognition for Punjabi: A Conditional Random Field Approach," *in Proceedings of 7th International Conference on Natural Language Processing*, 2009.

[24] Kazama J., Makino T., Ohta Y., and Tsujii J., "Tuning Support Vector Machines for Biomedical Named Entity Recognition," *in Proceedings of ACL Workshop Natural*

Language Processing in the Biomedical Domain, Phildadelphia, pp. 1-8, 2002.

[25] Kim J., Kang I., and Choi K., "Unsupervised Named Entity Classification Models and Their Ensembles," *in Proceedings of the 19th International Conference on Computational Linguistics*, Taipei, pp. 1-7, 2002.

[26] Kim W. and Khudanpur S., "Lexical Triggers and Latent Semantic Analysis for Cross-Lingual Language Model Adaptation," *ACM Transactions on Asian Language Information Processing*, vol. 3, no. 2, pp. 94-112, 2004.

[27] Küçük D., "Automatic Compilation of Language Resources for Named Entity Recognition in Turkish by Utilizing Wikipedia Article Titles," *Computer Standards and Interfaces*, vol. 41, no. c, pp. 1-9, 2015.

[28] Kumar N. and Bhattacharya P., "Named Entity Recognition in Hindi using MEMM," Technical Report, 2006.

[29] Lafferty J., McCallum A., and Pereira F., "Conditional Random Fields: Probabilistic Models for Segmenting and Labeling Sequence Data," *in Proceedings of 18th International Conference on Machine Learning*, pp. 282-289, 2001.

[30] Leaman R. and Gonzalez G., "BANNER: An Executable Survey of Advances in Biomedical Named Entity Recognition," *in Proceedings of Pacific Symposium on Bio computing*, Chicago, pp. 652-663, 2008.

[31] Li W. and McCallum A., "Rapid Development of Hindi Named Entity Recognition Using Conditional Random Fields and Feature Induction," *ACM Transactions on Asian Language Information Processing*, vol. 2, no. 3, pp. 290-294, 2003.

[32] Lin W., Yangarber R., and Grishman R., "Bootstrapped Learning of Semantic Classes from Positive and Negative Examples," *in Proceedings of the 20th International Conference on Machine Learning Workshop on The Continuum from Labeled to Unlabeled Data in Machine Learning and Data Mining*, Washington, 2003.

[33] Maynard D., Tablan V., and Cunningham H., "NE Recognition without Training Data on a Language you don't Speak," *in Proceedings of ACL*, Sapporo, pp. 33-40, 2003.

[34] Morgan A., Hirschman L., Colosimo M., Yeh A., and Colombe J., "Gene Name Identification and Normalization Using a Model Organism Database," *Biomedical Informatics*, vol. 37, no. 6, pp. 396-410, 2004.

[35] Muslea I., Minton S., and Knoblock C., "Selective Sampling with Redundant Views," *in Proceedings of 7th National Conference on Artificial Intelligence*, pp. 621-626, 2000.

[36] Nadeau D., Semi-Supervised NER: Learning to Recognize 100 Entity Types with Little Supervision, Thesis, University of Ottawa, 2007.

[37] Olsson F., Bootstrapping Named Entity Anotation by Means of Active Machine Learning, thesis, University of Gothenburg, 2008.

[38] Pedersen T., Kulkarni A., Kozareva Z., Angheluta R., and Solorio T., "Improving Name Discrimination: A Language Salad Approach," *in Proceedings of Workshop on Cross-Language Knowledge Induction*, pp. 25-32, 2006.

[39] Riloff E. and Jones R., "Learning Dictionaries for Information Extraction by Multi-Level Bootstrapping," *in Proceedings of the National Conference on Artificial Intelligence*, Menlo Park, pp. 474-479, 1999.

[40] Rømcke A. and Johansson C., "Named Entity Recognition using the Web," *in Proceedings of Workshop on Anaphora Resolution*, pp. 83-90, 2008.

[41] Saha S., Mitra P., and Sarkar S., "A Semi-Supervised Approach for Maximum Entropy based Hindi Named Entity Recognition," *in Proceedings of International Conference on Pattern Recognition and Machine Intelligence*, New Delhi, pp. 225-230, 2009.

[42] Saha S., Mitra P., and Sarkar S., "A Comparative Study on Feature Reduction Approaches in Hindi and Bengali Named Entity Recognition," *Knowledge-Based Systems*, vol. 27, pp. 322-332, 2012.

[43] Saha S., Sarkar S., and Mitra P., "A hybrid Feature set based Maximum Entropy Hindi Named Entity Recognition," *in Proceedings of the 3rd International Joint Conference on Natural Language Processing*, pp. 343-349, 2008.

[44] Seung H., Opper M., and Sompolinsky H., "Query by Committee," *in Proceedings of 5th Annual ACM Conference on Computational Learning Theory*, Pennsylvania, pp. 287-294, 1992.

[45] Sharma P., Sharma U., and Kalita J., "Named Entity Recognition in Assamese using CRFS and Rules," *in Proceedings International Conference on Asian Language Processing*, Kuching, pp. 15-18, 2014.

[46] Sharma R. and Goyal V., "Name Entity Recognition Systems for Hindi using CRF Approach," *International Conference on Information Systems for Indian Languages*, Patiala, pp. 31-35, 2011.

[47] Singh A., "Named Entity Recognition for South and South East Asian Languages: Taking Stock," *in Proceedings of International Joint Conference on Natural Language Processing*, Hyderabad, pp. 5-16, 2008.

[48] Solorio T. and López A., "Learning Named Entity Recognition in Portuguese from Spanish," *in Proceedings of Computational Linguistics and Intelligent Text Processing*, Mexico, pp. 762-768, 2005.

[49] Summerfield N., Zhang Z., and Chen H., "Disease Named Entity Recognition using Semi-Supervised Learning and Conditional Random Fields," *Journal of American Society for Information Science and Technology*, vol. 62, no. 4, pp. 727-737, 2011.

[50] Takeuchi K. and Collier N., "Use of Support Vector Machines in Extended Named Entity Recognition," *in Proceedings of 6th Conference on Natural language learning*, Stroudsburg, pp. 1-7, 2002.

[51] Vapnik V., *The Nature of Statistical Learning Theory*, Springer-Verlag, 1995.

[52] Yamada H., Kudo T., and Matsumoto Y., "Japanese Named Entity Extraction Using Support Vector Machine," *Transactions of IPSJ*, vol. 43, no. 1, pp. 44-53, 2002.

Sujan Kumar Saha is an Assistant Professor in Department of Computer Science and Engineering, Birla Institute of Technology Mesra, Ranchi, India. His main research interests include Natural Language Processing, Machine Learning, and Educational Technologies.

Mukta Majumder is an Assistant Professor in Department of Computer Science and Application, University of North Bengal, Siliguri, India. Prior to this he served Vidyasagar University as an Assistant Professor for almost three years. His main research interests include Text Processing, Machine Learning, Micro-fluidic System, and Biochip.