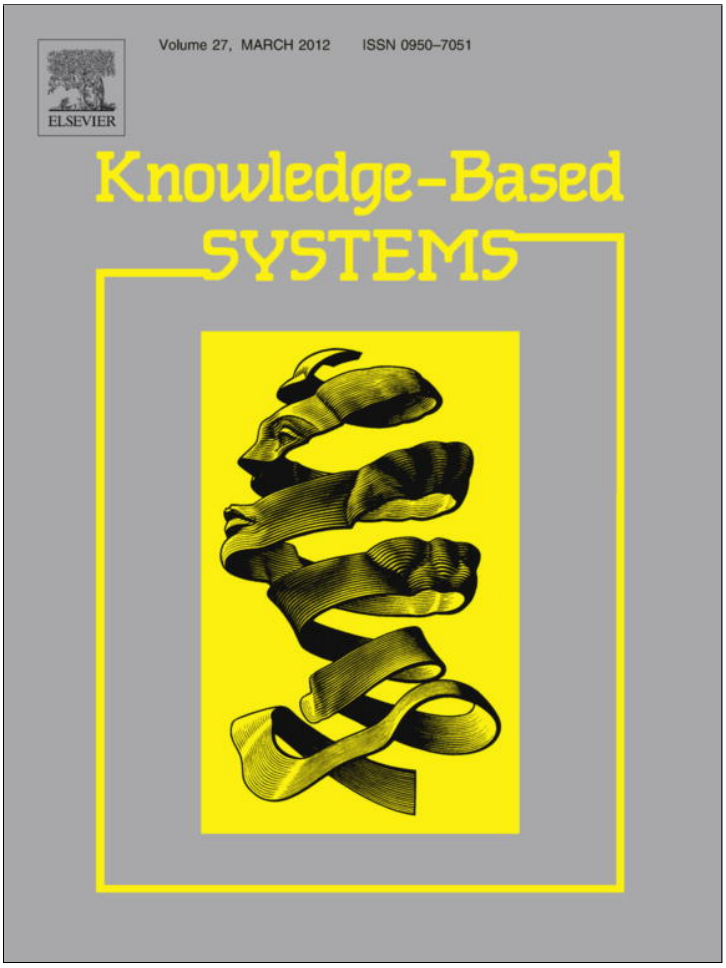
Volume 27, MARCH 2012
ISSN 0950-7051
ELSEVIER
Knowledge-Based
SYSTEMS

Knowledge-Based Systems 27 (2012) 322–332

Contents lists available at SciVerse ScienceDirect

Knowledge-Based Systems

journal homepage: www.elsevier.com/locate/knosys

A comparative study on feature reduction approaches in Hindi and Bengali named entity recognition

Sujan Kumar Saha [a,*], Pabitra Mitra [b], Sudeshna Sarkar [b]

[a] Dept. of CSE, Birla Institute of Technology Mesra, Ranchi 835215, India
[b] Dept. of CSE, Indian Institute of Technology Kharagpur, Kharagpur 721302, India

ARTICLE INFO

Article history:
Received 7 April 2011
Received in revised form 17 August 2011
Accepted 21 September 2011
Available online 14 October 2011

Keywords:
Named entity recognition
Dimension reduction
Feature selection
Feature clustering
Machine learning

ABSTRACT

Features used for named entity recognition (NER) are often high dimensional in nature. These cause overfitting when training data is not sufficient. Dimensionality reduction leads to performance enhancement in such situations. There are a number of approaches for dimensionality reduction based on feature selection and feature extraction. In this paper we perform a comprehensive and comparative study on different dimensionality reduction approaches applied to the NER task. To compare the performance of the various approaches we consider two Indian languages namely Hindi and Bengali. NER accuracies achieved in these languages are comparatively poor as yet, primarily due to scarcity of annotated corpus. For both the languages dimensionality reduction is found to improve performance of the classifiers. A Comparative study of the effectiveness of several dimensionality reduction techniques is presented in detail in this paper.

1. Introduction

Machine learning based approaches are now commonly used in various text processing tasks like parts-of-speech tagging, named entity recognition (NER) and text classification. These tasks are characterized by high dimensional feature sets, for example, surrounding words, suffixes, prefixes and so on. In machine learning approach a classifier is trained using annotated data (or, training data) with a suitable set of features. The performance of the classifier depends on the amount of annotated data and the effectiveness of the features used.

In many languages and domains getting sufficient annotated data is difficult. The use of a high dimensional feature set and insufficient training data in a classifier often causes overfitting and performance degradation of the classifier. This has been observed in NER and other tasks. A number of dimensionality reduction techniques have been proposed and used in order to reduce overfitting. In general the dimensionality reduction approaches can be broadly classified into two categories namely, *feature selection* and *feature extraction* [10]. Feature selection methods select a subset of the most effective features from the original set. Feature selection as a feature reduction approach is widely used in different text processing tasks [37,2,13,7,16] and performance improvements are observed when the reduced features are used. Feature extraction methods (e.g., linear discriminant analysis (LDA), principal component analysis (PCA)) transform the existing feature space to a lower dimensional feature space. Feature clustering is one type of feature extraction technique where similar features are merged into clusters and the dimensionality reduces. Feature clustering is widely used in various tasks in order to reduce the dimension of the feature space [5,28,35,9,1,26,24,4,36]. Feature selection and feature extraction is also important in other classification tasks like image classification [12], text clustering [21], character recognition [32], automatic abstracting [14] and efficient information retrieval [8].

In the literature we found in a few NER systems feature dimensionality is reduced to achieve performance improvement. Bender et al. [2] used count-based feature reduction in a maximum entropy (MaxEnt) based NER system. They selected only the features that have been observed in the training data set at least k times where the threshold k is user determined. Li and McCallum [23] developed a conditional random fields (CRF) based Hindi NER system. They observed that the use of all the features in a classifier caused overfitting. To overcome the problem they used *feature induction* aiming to create only those feature conjunctions that are found to significantly improve performance. They started with no feature at all and chosen new features iteratively. In each iteration, some set of candidates were evaluated, and the best ones were added to the model. In Saha et al. [33] we proposed word selection and word clustering based feature reduction techniques for the Hindi NER task. A few selection and clustering techniques were proposed and these were applied on the word feature only. We achieved performance improvement after using the reduced word features in a

[*] Corresponding author. Tel.: +91 9472711949.
E-mail addresses: sujan.kr.saha@gmail.com (S.K. Saha), pabitra@gmail.com (P. Mitra), shudeshna@gmail.com (S. Sarkar).

0950-7051/$ - see front matter © 2011 Elsevier B.V. All rights reserved.
doi:10.1016/j.knosys.2011.09.015

MaxEnt classifier. Ekbal and Saha [11] used a Genetic Algorithm based feature selection technique for developing Bengali NER system. They have identified a set of candidate feature categories (for example, previous and next words, first word, length of word, suffix, prefix) and among these they have selected the most important feature categories. For example, they have shown that, previous one word, next one word, prefix of length 3, suffix of length 4, position, previous tag and semantic features are important for the Bengali NER task. In the current article we have done a different type of feature reduction. A particular feature category contains a number of features. For example, 'previous word' is a particular feature category and it contains '*N*' (total number of unique words in the lexicon) number of different words. We have shown that all individual features of a particular feature category are not informative. For example, all the previous words are not important for identifying the NEs. Therefore we can ignore the useless features (e.g., removing the words which are not important in the identification task) and reduce the dimensionality of the feature space.

There are a number of feature selection and feature extraction approaches that have been applied in various text processing tasks like text classification, statistical language modeling, root identification, parts-of-speech tagging and named entity recognition. Some of these approaches are class or *tag independent*, which do not make use of the class information. Another set of approaches use the class or tag information to reduce the feature set, these can be referred as *tag specific* approaches. Not all these methods have been applied to the NER task in the literature. We have also not come across any comprehensive study of the different approaches as applied to the NER task. This paper is a comprehensive and comparative study on the effectiveness of different feature dimension reduction approaches in the NER tasks. We have studied dimensionality reduction in Hindi, Bengali and Biomedical NER tasks. Our specific contributions in the paper are listed below:

- We have considered different types of feature reduction approaches, namely feature selection, feature extraction and feature clustering in the NER task. We have also studied different types of feature selection strategies, namely filter, wrapper and embedded. We have implemented many of these reduction techniques on a common task and reported their performance.
- We have proposed a set of tag specific feature reduction approaches and we perform a comparative study between the tag specific and tag independent approaches.
- We have applied the feature reduction techniques to different high-dimensional features, namely word, suffix, prefix and word *n*-gram.
- We have studied the performance of feature reduction in NER tasks in two different languages, namely Hindi and Bengali (in general domain). We have also proved the generalizability of the proposed feature reduction techniques by applying these in (English) biomedical NER task.
- We have studied the performance of feature reduction using different machine learning algorithms like maximum entropy, conditional random fields and decision tree.

2. Taxonomy of dimension reduction approaches

In this section we present an overview of the most common dimensionality reduction approaches. In general, feature dimension reduction techniques are broadly classified into two types: feature selection and feature extraction [10]. Feature selection algorithms reduce the dimension of the feature space by selecting a subset of the most effective features from the original set. In feature selection one scores each potential feature subset according to a particular feature evaluation metric, and then selects the best subset of the features by a search algorithm. Some feature selection methods make use of the class information, these are referred as tag specific; while the other methods that do not use the class information are referred as tag independent. Feature extraction transforms the existing feature space to a new feature space. Feature clustering is another type of feature reduction approach which is widely used in text processing tasks. As in feature selection, feature clustering approaches can also be divided in two categories namely, tag specific and tag independent.

A number of approaches to feature selection and feature extraction have been used in different tasks in the literature. Some of these approaches have been used in the NER task previously. We have experimented with several metrics of feature reduction on a common NER task. Fig. 1 presents the taxonomy of feature reduction techniques we use in this study. We have run our experiments on most of these approaches excluding the transformation based approaches. The individual techniques are discussed below.

2.1. Feature evaluation metrics

Feature selection requires metrics for evaluating the importance of the individual features. Several feature evaluation metrics have been used in different tasks. A few metrics widely used in text processing are mentioned below.

2.1.1. Term frequency (TF) – T1.1

Term frequency (TF) is a simple metric for feature selection which counts the number of times a particular term (i.e., feature) occurs. If a particular feature occurs more than *t* (a predefined value) times then it is selected in the reduced feature set.

2.1.2. TFiDF – T1.2

The metric *term frequency inverse document frequency* (TFiDF) is a commonly used metric for feature selection in text processing tasks. The relevance of a word is determined by a product of its total number of appearances and a inverse function of the number of different documents in which it appears. This metric has been widely used in tasks dealing with multiple documents like, text classification and text clustering.

2.1.3. Information gain (IG) – T1.3

IG is a commonly used metric for feature ranking [37,7]. IG of a word takes into account the belongingness of the word in a category as well as its absence in the category. IG of a word *t* is defined as,

$$IG(t) = P(t)\sum_{i=1}^{m} P(c_i/t)\log\frac{P(c_i/t)}{P(c_i)} + P(\bar{t})\sum_{i=1}^{m} P(c_i/\bar{t})\log\frac{P(c_i/\bar{t})}{P(c_i)}) \quad (1)$$

In this equation c_i denotes the NE classes (there is a total of *m* classes). In NER task the context words are used as feature. So $P(c_i/t)$ is calculated as, the probability of the current word as a NE of category c_i where the word *t* is present in the context.

2.1.4. Mutual information (MI) – T1.4

MI is another metric which is defined as,

$$MI(t,c) = \log\frac{P(t,c)}{P(t)\times P(c)} \quad (2)$$

and estimated as [6,37],

$$MI(t,c) = \log\frac{A\times N}{(A+C)\times(A+B)} \quad (3)$$

where *A* is the number of times *t* and *c* co-occur, *B* is the number of times *t* occurs without *c*, *C* is the number of times *c* occurs without *t* and *N* is the total number of instances. This equation measures the

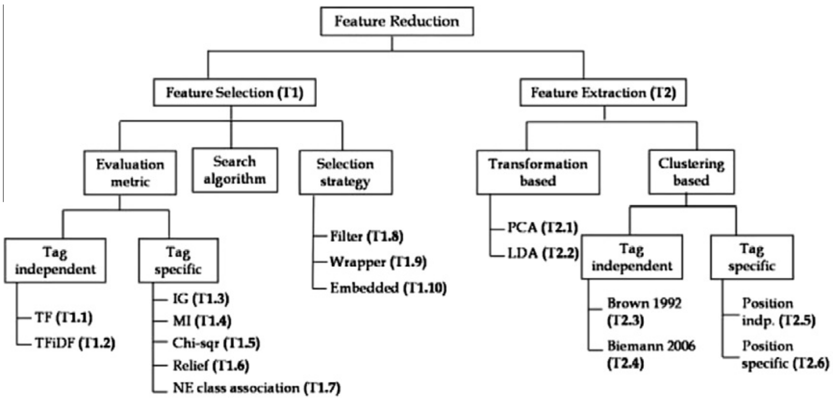

Fig. 1. A taxonomy of feature dimensionality reduction [the individual approaches are given unique-id which are used during discussion].

MI of the terms for a particular class. Then for selection maximum MI of a particular term over all NE categories can be used which is taken as,

$$MI_{max}(t) = max_{i=i}^{m} MI(t, c_i) \tag{4}$$

2.1.5. Chi-square statistic – T1.5

The (χ^2) measures the 'lack of independence' between term and class. This can be defined as [37],

$$\chi^2(t,c) = \frac{N \times (AD - BC)^2}{(A+C) \times (B+D) \times (A+B) \times (C+D)} \tag{5}$$

where D is the number of times neither t nor c occurs and A, B, C and N are defined earlier. Using this equation, the χ^2 value of the terms for each category is measured, and the maximum value (as Eq. (4)) is taken for selection.

2.1.6. Relief – T1.6

Relief [18,19] is another well known algorithm for feature selection. The key idea of Relief is to estimate the relevance of features according to how well their values distinguish between the instances of the same and different classes that are near to each other. Relief works by randomly sampling instances from the training data. For each sampled instance, the nearest instance of the same class (nearest hit) and opposite class (nearest miss) is found. An attributes weight is updated according to how well its values distinguish the sampled instance from its nearest hit and nearest miss. An attribute will receive a high weight if it differentiates between instances from different classes and has the same value for instances of the same class. The weight update in Relief is done by the following equation,

$$W_X = W_X - \frac{diff(X,R,H)^2}{m} + \frac{diff(X,R,M)^2}{m} \tag{6}$$

where W_X is the weight for attribute X, R is a randomly sampled instance, H is the nearest hit, M is the nearest miss and m is the number of randomly sampled instances. The function *diff* calculates the difference between two instances for a given attribute. For nominal attributes the difference is defined as either 1 (the values are different) or 0 (the values are the same), and for continuous attributes the difference is the actual difference normalized to the interval [0, 1].

2.1.7. NE class association metric – T1.7 (proposed by us)

In Saha et al. [33] we proposed a metric for word selection in the NER task. The metric uses the class association statistics of the words and it can be called as NE class association metric. Here we have applied the metric to reduce the word features as well as other high dimensional features like word *n*-gram, suffix and prefix.

The main principle of feature selection is to identify the features which have important role in the recognition task. Some words in the lexicon are typically associated with a particular NE category and hence are informative in the classification process. Similarly some bigrams and trigrams are informative as these provide some important clue regarding the class of a target word. But all the words and *n*-grams are not important. Among all the possible suffixes some are important in the sense that the presence of such suffixes help to predict the word category. Our aim is to select these informative words, *n*-grams and affixes. The feature selection method using the class association score is discussed below in the context of word features. Affix and *n*-gram selection follow a similar procedure.

2.1.7.1. Important word selection. *Context words* are those which occur in the proximity of a NE, i.e., the words present in the w_{i-2}, w_{i-1}, w_{i+1} or w_{i+2} position if w_i is a NE. The ratio between the number of occurrences of the word as a context word and its total number of occurrences in the corpus can be used as a metric of NER task specific word selection. We call this metric as *NE association weight* (NE_wt) which is defined as follows.

$$NE\_wt(w_i) = \frac{Occurrence\ of\ w_i\ as\ context\ word}{Total\ occurrence\ of\ w_i\ in\ corpus} \tag{7}$$

The words in the lexicon are then ranked according to their *NE_wt*. From this ranked list top *N* words are selected as important words. To get the appropriate value of *N* a tuning process is followed where several values of *N* are considered using a validation set and the most suitable one is chosen.

Some words occur only once in the training corpus and as a context word; for these words the *NE_wt* is one. These words will have higher NE weight and may find place in the important word list. But these words which are not much frequent are removed for further reduction. Thus to make the selection more effective the number of occurrence of the words is also considered. The modified selection procedure becomes, if the word (w) has the *NE_wt* is greater than a *threshold* and the number of occurrence of w in the training corpus is greater than t (a predefined value) then the word is selected as important word.

Also in the NER task, it is obvious that a particular word which is important at the position previous to a NE might not be important for the next position also. For example, in the Hindi text *shrI*[1] (Mr.), *ballebAja* (batsman), *pradhAnmantrI* (prime-minister), etc. have high occurrence at the previous position of the person class but these

[1] All the Hindi and Bengali words are written using the 'Itrans' transliteration.

generally do not occur after the person names. Whereas another set of words like *kahA* (say), *jI* (a honorary term commonly used in Hindi), *dAdA* (brother), etc. occur frequently at the next position of the person class but not in the previous position. Further we note that a set of words which are important for the w_{i-1} position might not be equally important for the w_{i-2} position, for which a different set of words will be important.

In order to make the selection process more effective, the position specific important words are selected. To select the position based important words, Eq. (7) is modified as,

$$NE\_wt(w_i) = \frac{Occurrence\ of\ w_i\ at\ \pm pos\ position\ of\ a\ NE}{Total\ occurrence\ of\ w_i\ in\ corpus} \tag{8}$$

where ±*pos* denotes a particular position like (+1), (−2), etc. Four positions (−2, −1, +1 and +2) are considered and for each position different sets of words are selected.

2.2. Selection strategy: wrapper, filter and embedded

Feature selection approaches can be divided in three categories namely, *filter*, *wrapper* and *embedded*. Filter methods select the best features according to some prior knowledge (commonly, feature evaluation metric score) and use the selected features directly in the classifier. Filter methods select the features independent of the classifier and basically serve as a preprocessing step of feature pruning to ease the burden of classification. In general, filter methods are fast, since they do not incorporate learning.

Wrapper methods, on the other hand, do not rely only on prior knowledge, but these evaluate the feature subsets in a real classifier and evaluate their classification performance to select the features. Hence, wrapper based feature selection is classifier specific. Wrapper methods use a search algorithm along with evaluation measures to find the optimal reduced feature set. Wrapper methods are very computationally intensive, since they typically need to run and evaluate the feature subsets in the classifier at every iteration.

In embedded methods, the search for the optimal feature subset is built into the classifier construction, and can be seen as a search in the combined space of feature subsets and hypotheses. Like wrapper approaches, embedded methods are also classifier specific. As the embedded methods incorporate feature selection in the training of the classifier and enable efficient algorithms to reach the optimum, these are faster than the wrapper methods.

Filter method simply ranks the features using a selection metric and selects the top-ranked features as the reduced feature set. In tagging tasks like NER, filter based feature selection methods are mostly used. We have studied and compared the performance of different feature selection metrics discussed above in a common NER task using filter based selection. We could not find any work on wrapper based feature selection applied to tagging tasks. Embedded methods for feature selection are more popular than the wrapper methods, and different techniques are proposed and used in various tasks. For example, *grafting* [29], *L1 and L2 regularization* [27], *shrinkage method* [22,25], *SVM-RFE* [15] and *decision tree learning with pruning* ([30,31]). In this study we have used decision tree (C4.5) for embedded feature selection. Section 4.2 presents more discussion on different selection strategies in the NER task.

2.3. Search algorithm for feature selection

Several search algorithms can be used for selecting a subset of features. These algorithms can be grouped into three categories, which are sequential, exponential and randomized algorithms. Sequential algorithms add or remove features sequentially, starting from the full or empty feature set. Examples of such algorithm are sequential forward selection, sequential backward selection, plus-l minus-r selection, etc. These algorithms may get trapped in local minima. Exponential algorithms (example, branch and bound, beam search) evaluate a number of subsets that grow exponentially with the dimensionality of the search space. Randomized algorithms incorporate some randomness in their search procedure to avoid local minima.

In this paper we have not studied the comparative performance of different search algorithms. We have only tried the sequential forward selection algorithm during the wrapper based feature selection experiments.

2.4. Feature extraction

Feature extraction transforms the existing feature space to a lower dimensional feature space. Principal component analysis (PCA) and linear discriminant analysis (LDA) are the most commonly used techniques for feature extraction.

PCA (T2.1) transforms the data to a new coordinate system in such a way that the greatest variance by any projection of the data comes to lie on the first coordinate (in other words, the highest eigenvalue component), the second one on the second coordinate and so on. The dimensionality of the data can be reduced by ignoring the lower eigenvalue components. On the other hand, LDA (T2.2) uses the class information to perform a projection of the features which best separate two or more classes. So, PCA finds the directions that are efficient for representation of the data and LDA finds the directions that are efficient for discrimination.

2.5. Tag independent feature clustering

Clustering is another effective dimensionality reduction approach where several similar features are grouped together in order to reduce the feature dimension. During clustering based feature reduction the individual features are replaced by the corresponding feature clusters (e.g., word clusters). During training of the classifier, the clusters may be used instead of the individual features (e.g., words) and then during decoding the individual features are replaced by the corresponding clusters.

There are number of approaches for word clustering proposed and used by the researchers in the past few years (e.g., [5,28,35,24,4,36]). Some of these are general purpose word clustering, that take a large raw text as input from which some statistics are extracted to compute the distance between the words. NE annotation information is not required here. We have used two such clustering approaches in our study; these are the methods proposed by Brown et al. [5] and Biemann [4].

Brown et al. (T2.3) clustered the words based on the frequency of their co-occurrence with other words. From a large raw corpus, they extracted the *n*-gram statistics using which the individual words are clustered. The bottom-up agglomerative word clustering algorithm derives a hierarchical clustering of words. A particular number of clusters can be obtained from the dendrogram. Biemann (T2.4) proposed an efficient graph clustering algorithm, Chinese Whispers, which is applicable for clustering the words in a corpus. For clustering, they used a co-occurrence based similarity measure among the words. They applied the clustering approach in several NLP tasks like language separation, acquisition of syntactic word classes and word sense disambiguation. We have applied it in the NER task.

2.6. NE tag specific feature clustering (proposed by us)

Apart from the task independent clustering methods, we have proposed two NER task specific clustering approaches. To perform word clustering, we have represented the words as vectors and computed the similarity between the vectors. We have experimented with two different vector representations which are defined below.

2.6.1. Similarity based on proximal words – T2.5

A target word may be represented as a vector of other words in its proximity. If all the words in the lexicon are used during vector representation, the vector dimension will be very high and the computation will be difficult. For efficient implementation, we consider only the words which occur in the context of the NEs. *List_Prev* contains the most frequent (top 2000) words that occur as w_{i-1} or w_{i-2} if w_i is the beginning word of a NE, and *List_Next* contains the 2000 most frequent next words in positions w_{i+1} or w_{i+2} if w_i is the last word of a NE. The *List_Prev* and *List_Next* can be replaced by position specific word lists (e.g., *List_*w_{i-1} and *List_*w_{i-2} can replace *List_Prev*).

Assume a particular word w occurs n times in the corpus. For each occurrence w_k of w, its previous words (w_{k-1} and w_{k-2}) are checked if these match any element of *List_Prev*. If there is a match, then we set to one the corresponding position of the vector and set to zero to the other positions related to *List_Prev*. Similarly we check the next word (w_{k+1} or w_{k+2}) in *List_Next* and find the values of the corresponding positions. The final word vector $\overrightarrow{W}_k$ is obtained by taking the average of the n vectors corresponding to the n occurrences of w. This measures the similarity of the contexts of the occurrences of the word w in terms of the proximal words.

2.6.2. Similarity based on proximity to NE categories – T2.6

We have already discussed that in the NER task the position of a context word is very important. Like in word selection, position specific clusters can be defined using this approach. The two preceding and two following positions ($i-1, i-2, i+1, i+2$) of a word are considered, and corresponding to these positions, four different word vectors are defined.

Each vector is of dimension $m+1$ corresponding to m NE classes (C_j), and one for the not-name class. For a particular word w_k, we measure the fraction ($P_j(w_k)$) of the total occurrences of the word in a particular position belonging to a class C_j (similar to Eq. (8)). The component of the word vector $\overrightarrow{W}_k$ for the position corresponding to C_j is $P_j(w_k)$.

2.6.3. Clustering the Vectors

Once the vectors are obtained, these are clustered using the K-means clustering algorithm. The value of K (the number of clusters) is chosen using a tuning process on a validation set during experiments. The cluster seeds are chosen randomly. We have used two types of vector similarity representation measures for clustering the vectors; these are cosine similarity and Euclidean distance.

The clustering approach defined above (similarity based on proximity to NE categories) can also be applied on other high-dimensional features apart from the words. In this study we have also experimented with affix and n-gram clustering using the task specific clustering approach.

3. Experimental setup

In this section we discuss the training and the test data, and the collection of the features used for the NER task, the classifiers used in our experiments and the performance evaluation metric.

3.1. Training and test corpus

The training data for the Hindi NER task is composed of about 200*K* (we have used *K* to represent *multiplied by* 10^3) words which are collected from the popular daily Hindi newspaper "Dainik Jagaran". Here three types of NEs are considered, namely, *person*, *location* and *organization*. The corpus has been manually annotated and contains about 5400 person, 4400 location and 2700 organization entities. The Hindi test corpus contains 25K words, which is distinct from the training corpus. The test corpus contains 678 person, 480 location and 322 organization entities. To preserve the boundary information of the NEs, the corpus is annotated using *BIO* format where *B-ne* denotes the beginning of a NE, *I-ne* refers to the other terms if the NE contains more than one word and *O* refers to the not-name words.

Table 1
Features used in the NER system.

Feature type	Features
Word unigram	w_i, w_{i-1}, w_{i-2}, w_{i-3}, w_{i+1}, w_{i+2}, w_{i+3} [w_i: current word]
Word n-gram ($n = 2, 3$)	$\langle w_i\ w_{i+1}\rangle$, $\langle w_{i-1}\ w_i\rangle$, $\langle w_i\ w_{i+1}\ w_{i+2}\rangle$, $\langle w_{i-2}\ w_{i-1}\ w_i\rangle$
NE tag	t_{i-1}, t_{i-2}, t_{i-3}
Suffix information	Suf(w_i), Suf(w_{i-1}), Suf(w_{i+1}), Suffix list
Prefix information	Pref(w_i), Pref(w_{i-1}), Pref(w_{i+1}), etc.
Digit information	contains_digit, numerical_word, etc.
POS information	POS(w_i), POS(w_{i-1}), POS(w_{i+1}) [POS: parts of speech]

For the Bengali NER task we have used the corpus published in the IJCNLP 2008 shared task [34] on NER in south and south-east Asian languages.[2] The corpus is annotated using 12 NE classes, but in our development we have considered only three of the NE classes namely, person, location and organization. The corpus contains ~110K words with about 1300 person names, 700 location names and 300 organization names. The shared task included the identification of nested NEs also, so the test corpus of the task is annotated considering the nested NEs. But in this study our objective is to identify the single level NEs only, so we have not considered the embedded or nested NEs. The test corpus contain ~38K words containing 755 person, 224 location and 25 organization names.

3.2. Features used in NER

We present in this section a list of candidate features useful for NER. The complete feature set is used as baseline in our study. Several techniques are described henceforth to reduce the feature set. The reduced features are then used to build classifiers. The features are listed in Table 1. The details of the features can be found in Saha et al. [33].

Note that gazetteer lists, context patterns etc. are some commonly used resources that help to improve the performance of NER systems. As our main objective here is to evaluate the feature reduction techniques in NER; we have not used such additional or external resources in our experiment. We have defined a set of simple and easily derivable common features using which we have conducted the desired experiments.

3.3. Classifiers used

Maximum entropy and Conditional random field classifiers are used in the study. These are briefly described below.

3.3.1. Maximum entropy classifier

MaxEnt principle is a commonly used technique which provides the probability of belongingness of a token to a class. MaxEnt computes the probability $p(y|h)$ for any y from the space of all possible outcomes Y, and for every h from the space of all possible histories H. In NER, history can be viewed as all information derivable from the training corpus relative to the current token. The computation

[2] The data can be found in http://ltrc.iiit.ac.in/ner-ssea-08/index.cgi?topic=5.

of probability ($p(y|h)$) of an outcome for a token in MaxEnt depends on a set of features that are helpful in making the predictions about the outcome. Given a set of features and a training corpus, the MaxEnt estimation process produces a model in which every feature f_k has a weight λ_k. We can compute the conditional probability as [3]:

$$p(y|h) = \frac{1}{Z(h)} \prod_k \lambda_k^{f_k(h,y)} \tag{9}$$

$$Z(h) = \sum_y \prod_k \lambda_k^{f_k(h,y)} \tag{10}$$

The conditional probability of the outcome is the product of the weights of all active features, normalized over the products of all the features. For our development we have used a Java based MaxEnt toolkit.[3]

The used corpus is annotated using BIO format, where 'B-ne' refers to the words which are the beginning word of a NE of type 'ne', 'I-ne' indicates rest of the words (if the NE contains more than one words) and 'O' refers to the not-name words. Some tag sequences can never happen. For example, 'I-ne' should not occur after a 'O' tag. Also 'I-ne2' should not occur after a 'B-ne1' or 'I-ne1' where 'ne1' and 'ne2' are two different NE classes. During the decoding using MaxEnt, if the tag having the highest probability value is considered as the output tag, then some of these inadmissible tag sequences might occur. To eliminate these inadmissible sequences, we have used a beam search algorithm for decoding with some restrictions to get the most probable NE category.

3.3.2. Conditional random field classifier

CRF [20] is an undirected graphical model trained to maximize the conditional probability of the output sequence given the inputs, or, in the case of token based NLP tasks, the conditional probability of the sequence of labels y given a sequence of tokens x. The number of previous labels taken into account defines the order of the CRF model. More formally,

$$P(y|x) = \frac{1}{Z(x)} exp\left\{ \sum_t \sum_k \lambda_k f_k(y, x_t) \right\} \tag{11}$$

In Eq. (11), $Z(x)$ is a normalization factor computed over all possible label sequences, f_k is a feature function and λ_k its respective weight. y represents the labels taken into account as context and it is defined by the order of the CRF. For a nth order model, y becomes $y_t, y_{t-1}, \ldots, y_{t-n}$. x_t is the feature representation of the token in position t, which can include features extracted by taking the whole input sequence into account, not just the target token. For our development we have used a CRF toolkit.[4]

3.4. NER evaluation measures

In the NER task, the accuracies are measured in terms of *f-measure*, which is the weighted harmonic mean of precision and recall. *Precision* is the percentage of correct annotations and *recall* is the percentage of the total NEs that are successfully annotated. The general expression for measuring the *f-measure* or *f-score* is,

$$F_\beta = \frac{(1+\beta^2)(precision \times recall)}{(\beta^2 \times precision + recall)} \tag{12}$$

Here the value of β is taken as 1.

During the evaluation we have followed *exact match* strategy; which means a detected NE is assumed as correct if it matches exactly with the corresponding test data entity in terms of both the NE category and boundary.

4. Experimental results and discussion: Hindi NER

We present below the details of the experiments conducted on the Hindi NER task. Effectiveness of different feature reduction techniques are studied by the following comparative approach.

First a set of candidate features is chosen for the task. A feature template containing all the features is used to train the baseline system using the available training data. Then the feature reduction techniques are applied to reduce the dimension of the feature template. The performance of the system using the reduced feature set is compared with the baseline system.

4.1. Baseline system

The performance of the Hindi NER using the features described in Section 3.2 is reported here as a baseline. In Table 2 we have presented the precision (Pre), recall (Rec) and f-scores (Fm) of different feature sets. During the baseline experiments all the available feature values (e.g., all the words in the corpus in case of word feature) are used. Using MaxEnt we have achieved the highest f-score of 75.52 with 85.23% precision and 67.8% recall. In CRF the highest f-score is 83.39 with 90.63% precision and 77.22% recall. These values are obtained using previous one word and next one word (i.e., word window of length three), affixes of length up to four characters, NE tag of the previous word and parts-of-speech information of the words.

When a wider word window or higher affix length is used, the accuracy reduces in both the MaxEnt and CRF classifiers. But when the only word features are used then we observe that a word window of length five gives higher f-score than a word window of length three (first two rows in the table) in both the MaxEnt and CRF classifiers. But when other features are used along with the high-dimensional word features then a shorter word window works better. Also we observe that the addition of the word n-grams causes performance degradation, although we found that in the corpus there are a number of bigrams and trigrams that provide important clue for identifying NEs.

As the amount of training data is not sufficient, when a large number of features are used in the baseline classifier, it suffers from overfitting and the performance degrades. In the classifiers when the word feature for a particular position is used, it is transformed into N binary features where N is the size of the lexicon (total number of unique words in the corpus). The used Hindi corpus contains about 17K unique words. So the word feature for a particular position (say, previous position or -1) corresponds to 17K binary features. When we use a wider window, say, window length is increased from three to five, a total of 34K features are included in the feature template. This increase in dimensionality of the feature space causes overfitting here. It is not just in the word features, many of the other features are also high dimensional; for example, suffix, prefix and n-gram. Similar to the word features, increment of the maximum affix length from l to $l+1$ corresponds to the addition of thousands of binary features (total number of affixes of length $l+1$) in the feature space. Use of all words, affixes, n-grams as feature makes the overall feature space very high dimensional where many non-informative features are present. These non-informative features inject noise and degrade the overall performance. To overcome this, we have reduced the high dimensional features using various dimension reduction techniques.

4.2. Feature reduction approaches used in the NER task

In Section 2 we have discussed several feature dimension reduction approaches. Many of these are implemented and used in the NER task, however we are not able to implement and use

[3] http://sourceforge.net/projects/maxent/.
[4] http://crfpp.sourceforge.net/.

Table 2
Hindi baseline results using MaxEnt and CRF classifiers.

Feature	MaxEnt			CRF		
	Pre	Rec	Fm	Pre	Rec	Fm
$[w_{-1} \cdots w_{+1}]$, t_{-1}	77.97	58.46	66.82	89.11	72.83	80.15
$[w_{-2} \cdots w_{+2}]$, t_{-2}, t_{-1}	79.76	60.33	68.7	89.09	73.42	80.5
$[w_{-1} \cdots w_{+1}]$, t_{-1}, suf	82.67	65.63	73.17	89.75	74.99	81.71
$[w_{-2} \cdots w_{+2}]$, t_{-2}, t_{-1}, suf	80.54	62.69	70.94	89.04	74.46	81.1
$[w_{-1} \cdots w_{+1}]$, t_{-1}, suf, pref	82.75	65.77	73.29	89.47	75.41	81.84
$[w_{-1} \cdots w_{+1}]$, t_{-1}, suf, pref, *n*-Gram	80.39	64.84	71.78	88.9	75.37	81.58
$[w_{-1} \cdots w_{+1}]$, t_{-1}, suf, pref, POS information	85.23	67.8	**75.52**	90.63	77.22	**83.39**
$[w_{-2} \cdots w_{+2}]$, t_{-2}, t_{-1}, suf, pref, POS information	82.05	65.72	72.98	89.71	76.6	82.64

all the methods. Here we discuss our observations on using the feature reduction approaches in our NER task.

We have used the filter (T1.7) based feature selection strategy to evaluate the performance of the metrics mentioned in Section 2.1. Term frequency (T1.1) is used in the NER task as a simple count i.e., occurrence of the feature in the training corpus. It is applicable in the high-dimensional word feature as well as suffix, prefix and *n*-gram features. In the NER task TFiDF (T1.2) can not be used as multiple documents are not involved here. IG (T1.3), MI (T1.4) and χ^2 (T1.5) use the NE class information from the training corpus and are applicable in the NER task. In Relief both nominal and numeric features can be used simultaneously. Although the Relief defined by Kira and Rendell [18] operates on two classes only, Kononenko [19] enhanced it to cope with multi-class domains. Hence Relief is also applicable in the NER task.

We have also tried to use the wrapper (T1.8) based feature selection in the NER task. To use the wrapper like technique, we choose a small development data (50K words, randomly selected from the original training corpus) on which the feature subsets are evaluated and validated for including the features in the final reduced feature set. We run *sequential forward selection algorithm* to select the feature subsets. In sequential forward selection algorithm initially an empty set is taken as the reduced feature set and the 'good' features are added iteratively to the reduced set. We found that the method is too time consuming. In our feature template, where we have considered surrounding words, suffixes, prefixes, previous tag information, parts-of-speech information, etc. (see features in Section 3.2), a total of ~100K features are present. The search process starts with no feature in the reduced set and sequentially adds the features to the reduced set. To include a feature (or a group of features) in the reduced set, a classifier (MaxEnt or CRF) is trained using the validation set and the reduced feature set (containing the new feature). Then the performance of the classifier is evaluated using the test data. The process of training the classifier, generating predictions on the test set and the performance evaluation requires a few minutes in a system with moderate processing capability. So, it is very difficult to perform the repetitive process to select the final feature set. Again, we found that in the NER task the performance of the feature subsets on the validation data does not guarantee a similar performance in the actual training data. In our experiment using wrapper feature selection, the selected feature set did not achieve good results. Therefore, we have not further studied the wrapper based feature selection in the NER task.

Decision tree (C4.5) [31] is a widely used machine learning technique. The C4.5 algorithm contains an internal feature selection as a part of the algorithm. Generally, when the dataset contains a large number of features, a subset of the features is included into a decision tree. Such approaches for classifier training are said to contain an embedded feature selection mechanism. In our study we have used the C4.5 classifier to study the performance of embedded feature selection in the Hindi NER task. The experimental results using C4.5 are presented in the next subsection.

We did not find any work in the literature where PCA (2.1) and LDA (2.2) are used for dimension reduction in tagging tasks like NER where several types of features are involved. These feature extraction techniques reduce the feature dimensionality but transform the whole feature space to a new feature space. In this study we have not considered the transformation based feature extraction like PCA and LDA.

We have considered two tag independent clustering approaches following the methods proposed at Brown et al. [5] and Biemann [4]. These methods are applied for clustering the Hindi words using a large raw corpus and the word clusters are used as reduced features by replacing the word features. We use two tag specific clustering techniques which are discussed in Section 2.6. The clustering technique which computes the similarity between the features based on the proximity to NE categories (T2.6) can be applied to cluster the *n*-grams and affixes apart from the words.

The details of the performance of the reduced features are discussed below.

4.3. Feature reduction: word feature

In this section the details of the performance of the feature reduction approaches are discussed when applied on the word features. In Table 3 we have summarized the results. From this table we can also compare the different approaches used in this study.

From the table it is observed that all the feature reduction techniques are able to improve the baseline accuracy. In this table for each feature reduction technique we mention the highest accuracy obtained in our experiments. It is previously mentioned that the used Hindi corpus contains 17K unique words; all these are used during the baseline experiments. Feature selection approaches select a subset of these words which are most informative. Feature clustering approaches cluster these words into a number of groups. The performance of a feature reduction approach depends on the number of features (number of words or number of clusters) in the reduced set. To obtain the suitable size of the reduced feature set we have performed several experiments for each feature reduction technique.

With feature selection the highest accuracy is obtained when the words are selected using the NE class association position specific measure (T1.7). With word selection the highest accuracy of the CRF based system is a f-score of 84.52 (the highest accuracy is shown in bold font in table). The corresponding precision and recall values are 91.94% and 78.2% respectively. This value is obtained when total number of selected words is 2800 in the four positions. The corresponding f-score in MaxEnt classifier is 79.17 with 88.67% precision and 71.51% recall.

All the other tag specific word selection approaches which use class information of the words (e.g., MI, IG and χ^2) outperform the baseline MaxEnt and CRF classifiers. Among these the performance of Relief is lower compared to the others. In the MaxEnt classifier the selection using IG marginally outperforms the NE

Table 3
Performance of different feature reduction techniques applied to word features.

	Feature reduction approach	CRF	MaxEnt
No reduction	Baseline classifier	83.39	75.52
Selection	Frequency of occurrence (>1) – T1.1	83.54	76.2
	Frequency of occurrence (>4) – T1.1	83.05	75.93
	Information gain – T1.3	84.28	**79.34**
	Mutual information – T1.4	84.21	78.89
	CHI-square – T1.5	84.4	79.1
	Relief – T1.6	83.61	77.88
	Class association: position indep. – T1.7	84.14	78.54
	Class association: position specific – T1.7	**84.52**	79.17
Clustering	Brown word clustering – T2.3	84.6	78.87
	Biemann word clustering – T2.4	84.42	**78.95**
	Tag specific: position independent – T2.5	84.35	78.63
	Tag specific: position specific – T2.6	**84.66**	78.94

class association based word selection. In MaxEnt IG based word selection achieves a f-score of 79.34.

In the NER task, word frequency (T1.1) based reduction does not appear to be a good approach for feature selection compared to the other approaches. When the value of '*k*' (number of occurrences of a feature) is small (1 or 2) then little performance improvement over baseline is observed. But the use of a higher *k* value degrades the performance in both the MaxEnt and CRF classifiers.

We have also conducted experiments for evaluating performance of embedded feature selection using decision tree (C4.5). As embedded feature selection is classifier specific, the features selected using decision tree can not be used in MaxEnt or CRF classifiers. Hence we can not compare the performance of the C4.5 based embedded feature selection with the results presented in Table 3. In the experiments we have examined different levels of tree pruning to obtain decision trees of various sizes. In our experiments using C4.5 we have achieved the highest f-score of 78.33 with 80.1% precision and 76.63% recall. This performance is better than the baseline MaxEnt classifier but with filter based feature selection the MaxEnt classifier performs better. In these experiments we have made an interesting observation that the tree with no pruning has much lower accuracy (68.9, can be taken as baseline for C4.5) and the best accuracy is obtained using a smaller tree. This proves that, a larger tree uses many features and causes overfitting, and when the embedded feature selection removes the unnecessary features the performance improves. Hence, we can conclude that the embedded feature selection is effective in Hindi NER.

Similar to the word selection, word clustering based feature reduction approaches are also effective and improve the baseline accuracy. We have used two types of clustering approaches namely, tag specific and tag independent.

In the tag specific category, two types of clustering are used, position specific and position independent. In our experiments position specific clustering (T2.6) proved the better option. Using position specific word clustering we achieve f-scores of 84.66 in CRF and 78.94 in MaxEnt. In our experiments the tag independent word clustering approaches also perform well. The word clusters with Brown (T2.3), Biemann (T2.4) and position specific (T2.6) approaches achieve comparable performance in our experiments.

The overall performance of selection and clustering based word feature reduction are comparable. In our experiments selection techniques perform better than the clusters in the MaxEnt classifier but in CRF the clusters marginally outperform the selection based reduced features. From the table it is also observed that the overall performance of the MaxEnt classifier is poor compared to the CRF classifier but the accuracy increment after feature selection in MaxEnt is higher than in CRF.

4.4. Feature reduction to all high-dimensional features

The feature reduction techniques are now applied on all the high dimensional features (words, *n*-grams, affixes). Table 4 summarizes the results of individual feature category and the combined result. By applying feature reduction on the suffix, prefix and *n*-gram features also we achieve performance improvement over the baseline. In the baseline system addition of all *n*-grams (in the used Hindi corpus ∼95K bigrams and ∼150K trigrams are available) caused performance degradation. But when the selected *n*-grams are used, the baseline accuracy is improved to 83.78 and 76.85 respectively. Using the reduced word, affix and *n*-gram features by replacing the baseline features the system achieves a f-score of 85.05 in CRF and 79.8 in MaxEnt.

Similarly, by the use of cluster features by replacing the high-dimensional word, *n*-gram and affix features, we achieve a f-score of 84.81 with 91.28% precision and 79.19% recall in CRF. The MaxEnt classifier achieves a f-score of 79.16 with 87.36% precision and 72.37% recall with feature clustering. For the affix and *n*-gram features, selection performs better than the clustering.

4.5. Accuracy using selection and clustering combined

The individual performance of feature selection and clustering is discussed earlier. We next combine the feature selection and feature clustering. The combined word features are obtained as,

If {the surrounding word (say, w_{i+1}) belongs to the important word list} *then* {use the word as feature} *else* {use the *cluster_id* of the cluster in which the word (w_{i+1}) belongs as feature}.

Similar to word feature, other features are also modified using both the clustering and selection. In Table 5 we summarize the overall results obtained in the experiments on Hindi NER. When both the clustering and selection are used for feature reduction the system accuracy increases to a f-score of 85.31 in CRF and f-score of 80.2 in MaxEnt. This is the highest accuracy of the Hindi NER system.

4.6. Tag specific and independent reduction approaches with limited data

In our previous experiments we found that both the tag specific and tag independent feature reduction approaches perform well and improve the baseline accuracy. The tag specific approaches perform better than the tag independent approaches in most of the cases. Tag specific approaches are dependent on the class information of the training data. For good performance of the tag specific approaches, a reasonable amount of training data is required which should be sufficient to represent the NE classes. On the other

Table 4
Feature reduction on word, suffix, prefix and word *n*-gram features.

	Feature	CRF	MaxEnt
No Reduction	Baseline classifier	83.39	75.52
Selection	Baseline with word selection	84.52	79.17
	Baseline with suffix selection	84.06	77.69
	Baseline with prefix selection	83.48	76.2
	Baseline with *n*-gram selection	83.78	76.85
	Baseline with word, *n*-gram and affix selection	85.05	79.8
Clustering	Baseline with word clustering	84.66	78.94
	Baseline with affix clustering	83.9	76.11
	Baseline with word, *n*-gram and affix clustering	84.81	79.16

Table 5
Performance of Hindi NER after combining feature selection and clustering.

Feature	CRF	MaxEnt
Baseline classifier: no feature reduction	83.39	75.52
Selection based feature reduction	85.05	79.8
Clustering based feature reduction	84.81	79.16
Feature reduction using selection and clustering	85.31	80.2

hand the tag independent approaches are not dependent on the annotation information. Thus they can use additional data and are not limited to use the training data. For example, the tag independent word clustering approaches like Brown clustering are applied on a large raw corpus to obtain the clusters. So we hypothesize that, if the training data is not sufficient then the tag specific approaches will suffer from information scarcity but the tag independent approaches might perform well. To show this we perform a set of experiments on feature reduction using limited training data.

For these experiments we have selected a smaller training corpus of size 50K word. In these experiments the tag independent approaches perform better than the tag specific approaches. The experimental results on reduction of word features using the smaller training corpus are summarized in Table 6.

When the training data is 50K, where there are about 8K unique words, the task independent feature reduction approaches perform better. Even the simple frequency count based word feature reduction (T1.1) performs well. The tag specific word selection techniques have improved the baseline accuracy in both the MaxEnt and CRF classifiers but these perform poorer than the word frequency. Similarly in clustering also, the Brown clustering approach performs much better than the tag specific clustering approach. When the amount of corpus is too less, the vector representation of the words used in the tag specific clustering is not good enough to represent the classes. But the tag independent clustering approaches are not dependent on the training data, same clusters used in larger training data experiments are used here also, and better performance is achieved.

Table 6
Feature (word) reduction performance comparison with 50K training corpus.

	Feature	CRF	MaxEnt
	Baseline classifier	73.69	61.75
Selection	Word selection: frequency (T1.1)	74.41	63.1
	Word selection: IG (T1.3)	74.24	62.47
	Word selection: MI (T1.4)	73.8	62.04
	Word selection: class association (T1.7)	73.98	62.63
Clustering	Word clustering: brown (T2.3)	74.83	64.82
	Word clustering: tag specific (T2.6)	74.06	62.37

5. Experimental results: Bengali NER

In the previous section we have reported the details of the Hindi NER results. We obtain similar results in the Bengali NER also. This section summarizes the experimental results of the Bengali NER task. In Table 7 the results of the Bengali NER system using both the MaxEnt and CRF classifiers are presented. The baseline accuracy of Bengali is poorer in both the classifiers as compared to Hindi. Hence the overall accuracy of the Bengali NER system is also not very high even after feature reduction.

Bengali is morphologically richer than Hindi where the NEs are also highly inflected. For example, a person name *sachina* (Sachin) is inflected in Bengali as *sachinake* (to Sachin), *sachinarA* (Sachin and others), *sachiner* (of Sachin), *sachinader* (of Sachin and others), etc. Location names, organization names and other words are inflected similarly. These inflections add difficulty in the name identification task. A good stemmer can extract the roots from the inflected words; but it is not used in our experiments due to unavailability of a reliable Bengali stemmer. Also training data available is less in Bengali compared to Hindi.

The baseline accuracy of the Bengali NER system is a f-score of 68.32 using CRF. The corresponding precision is 73.87% and recall is 63.55%. The accuracy is obtained using a feature set containing previous and next words, NE tag of the previous word, suffixes of length up to four characters, prefixes of length up to three characters and parts-of-speech tags of the current and surrounding words. The feature reduction techniques are then applied to reduce the word, affix and word *n*-gram features. The f-score is increased to 70.25 using feature selection and 69.2 using clustering. After combining selection and clustering, the f-score is further increased to 70.75 in CRF. These results are obtained when the class association (T1.7) based feature selection and the tag specific (T2.6) clustering approaches are used.

As in the Hindi NER task, the overall performance of MaxEnt is lower compared to CRF. Here the baseline accuracy is a f-score of 65.48. Using selection and clustering based feature reduction the f-score is increased to 66.87 and 66.59 respectively. Using a feature reduction which combines the selection and clustering a f-score of 67.54 is obtained.

Thus we conclude that both the selection and clustering based feature reduction approaches are able to increase the NER system accuracy in Hindi as well as Bengali languages.

Table 7
Performance (f-score) of the Bengali NER system with feature reduction.

Feature	CRF	MaxEnt
Baseline classifier: no feature reduction	68.32	65.48
Selection based feature reduction	70.25	66.87
Clustering based feature reduction	69.2	66.59
Feature reduction using both selection and clustering	70.75	67.54

6. Experimental results: biomedical NER

The feature reduction approaches are also tested in the NER task in biomedical (English) domain. This section presents our experiments on biomedical NER. We like to mention here that we have chosen the biomedical NER task in our study only to show that the proposed tag specific feature reduction approaches are very general and is expected to work well in all domains.

For the task we have used the JNLPBA 2004 data [17]. This corpus is extracted from the GENIA corpus Version 3.02. The training set consists of 2000 abstracts (about 500K words) and the test set contains 404 abstracts (about 100K words). In this data five NE classes are considered: DNA, RNA, protein, cell-type and cell-line.

For the biomedical NER task we have chosen a set of features that are easy to derive and require no deep domain knowledge. Most of the features are general features and not specific to the biomedical domain. The features we have used are, word features (current and surrounding words), NE tags of the previous words, capitalization and digit information based features, special characters (e.g., hyphen), word normalization (e.g., root of the word), prefix and suffix information and parts-of-speech information (extracted using the GENIA Tagger V2.0.2).

We first used the complete feature set (that contains all the words and affixes) to develop the baseline classifier. The MaxEnt based baseline classifier achieves a f-score of 64.82 with 63.66% precision and 66.01% recall. This value is obtained when the word and parts-of-speech information in window five, previous NE tags, capitalization and digit information based features, word normalization, suffix and prefix information are used. Using a similar feature set in CRF we obtain a f-score of 70.62 with 71.82% precision and 69.47% recall.

Then we have applied the class association (T1.7) based feature selection and the tag specific (T2.6) clustering techniques to reduce the feature set. For selecting the word features in the biomedical domain, the technique is modified a little. NEs in the biomedical domain are often much longer than those in general domain. Here the NEs contain common words also. Hence the word selection procedure is modified accordingly. Here we select two types of words, namely, *intra NE words* and *extra NE words*.

In the biomedical NE corpus, there are 9550 words that occur inside one or more NEs. If we consider all these words as informative word, many non-important words would be included in the list. For example, in the corpus 'and' occurs 1074 times inside the NEs, but it is not much useful in recognition. Similarly 'of' occurs 212 times, 'normal' occurs 137 times, 'expression' occurs 48 times, 'active' occurs 24 times, 'low' occurs 10 times and all these words do not play important role in finding the NEs. So we select the important intra NE words from these. To select the intra NE words, Eq. (7) is modified as,

$$intraNE\_wt(w_i) = \frac{Number\ of\ occurrence\ of\ w_i\ as\ part\ of\ a\ NE}{Total\ occurrence\ of\ w_i\ in\ the\ training\ corpus} \quad (13)$$

The intra NE words are selected based on the *intraNE_wt* and number of occurrence (as in the Hindi NER task). Here a total of 2440 words are selected as intra NE words. The words which are highly probable to occur at preceding or following positions of the NEs are selected as extra NE words. Extra NE word selection uses the NE weight as defined in Eq. (7). A total of 900 words are selected as extra NE words for the task.

In the biomedical NER task we achieve performance improvement when the reduced features are used. When the selected features are used in the baseline MaxEnt classifier (f-score 64.82), it achieves a f-score of 66.87; and with feature clustering the f-score becomes 65.75. Finally by using the combined selection and clustering based feature reduction we obtain a f-score of 67.24 in the MaxEnt classifier. In the CRF classifier (baseline f-score is 70.62, which is much higher than MaxEnt) also we achieve performance improvement using feature reduction. With feature reduction in CRF the f-score is increased to 71.56 (see Table 8).

Table 8
Performance (f-score) of the biomedical NER system with feature reduction.

Feature	CRF	MaxEnt
Baseline classifier: no feature reduction	70.62	64.82
Selection based feature reduction	71.37	66.87
Clustering based feature reduction	71.1	65.75
Feature reduction using both selection and clustering	**71.56**	**67.24**

This performance improvement demonstrates the effectiveness and generalizability of the feature reduction techniques.

7. Conclusion

The performance of a machine learning based classifier is largely dependent on the feature set. Identification of a suitable feature set is very important to yield the best performance from the available training data. Generally the context and affix information are useful in identification of the named entities from a text. But these features are high-dimensional which may lead overfitting if the available training data is not sufficient. Thus the dimensionality of these features needs to be reduced to achieve a better performance. We have studied the effectiveness of several feature reduction approaches. The performance of both the tag specific and tag independent approaches of feature reduction are shown in the Hindi NER tasks using MaxEnt and CRF classifiers. We have also studied the effectiveness of the feature reduction approaches in Bengali NER and biomedical domain NER tasks. In our experiments we found that the reduced features perform better than the corresponding complete feature sets.

References

[1] R. Bekkerman, R. El-Yaniv, N. Tishby, Y. Winter, Distributional word clusters vs. words for text categorization, Journal of Machine Learning Research 3 (2003) 1183–1208.

[2] O. Bender, F.J. Och, H. Ney, Maximum entropy models for named entity recognition, in: Proceedings of the CoNLL-2003, 2003, pp. 148–151.

[3] A.L. Berger, S.D. Pietra, V.D. Pietra, A maximum entropy approach to natural language processing, Computational Linguistic 22 (1) (1996) 39–71.

[4] C. Biemann, Chinese Whispers – an efficient graph clustering algorithm and its application to natural language processing problems, in: Proceedings of the HLT-NAACL-06 Workshop on Textgraphs-06, 2006.

[5] P.F. Brown, V.J.D. Pietra, P.V. deSouza, J.C. Lai, R.L. Mercer, Class-based *n*-gram models of natural language, Computational Linguistics 18 (4) (1992) 467–479.

[6] K.W. Church, P. Hanks, Word association norms, mutual information, and lexicography, Computational Linguistic 16 (1) (1990) 22–29.

[7] E.F. Combarro, E. Montanes, I. Diaz, J. Ranilla, R. Mones, Introducing a family of linear measures for feature selection in text categorization, IEEE Transactions on Knowledge and Data Engineering 17 (9) (2005) 1223–1232.

[8] R. Dhanapal, An intelligent information retrieval agent, Knowledge-Based Systems 21 (6) (2008) 466–470.

[9] I.S. Dhillon, S. Mallela, R. Kumar, A divisive information-theoretic feature clustering algorithm for text classification, Journal of Machine Learning Research 3 (2003) 1265–1287.

[10] R.O. Duda, P.E. Hart, D.G. Stork, Pattern Classification, second ed., John Wiley & Sons Inc, 1999.

[11] A. Ekbal, S. Saha, Multiobjective Optimization for classifier ensemble and feature selection: an application to named entity recognition, International Journal on Document Analysis and Recognition, in press. doi:10.1007/s10032-011-0155-7.

[12] A. Elsayed, F. Coenen, C. Jiang, M. Garca-Fiana, V. Sluming, Corpus callosum MR image classification, Knowledge-Based Systems 23 (4) (2010) 330–336.

[13] G. Forman, An extensive empirical study of feature selection metrics for text classification, Journal of Machine Learning Research 3 (2003) 1289–1305.

[14] Q. Guo, M. Zhang, Multi-documents automatic abstracting based on text clustering and semantic analysis, Knowledge-Based Systems 22 (6) (2009) 482–485.

[15] I. Guyon, J. Weston, S. Barnhill, V. Vapnik, Gene selection for cancer classification using support vector machines, Machine Learning 46 (1–3) (2002) 389–422.

[16] Uguz Harun, A two-stage feature selection method for text categorization by using information gain, principal component analysis and genetic algorithm, Knowledge-Based Systems 24 (2011) 1024–1032.

[17] J. Kim, T. Ohta, Y. Tsuruoka, Y. Tateisi, N. Collier, Introduction to the Bio-entity Recognition Task at JNLPBA, in: Proceedings of the PBA-2004, 2004, pp.70–75.

[18] K. Kira, L.A. Rendell, A practical approach to feature selection, in: Proceedings of the International Conference on Machine Learning (ICML-1992), 1992, pp. 249–256.

[19] I. Kononenko, Estimating attributes: analysis and extensions of relief, in: Proceedings of the European Conference on Machine Learning, 1994.

[20] J. Lafferty, A. McCallum, F. Pereira, Conditional random fields: probabilistic models for segmenting and labeling sequence data, in: Proceedings of the International Conference on Machine Learning (ICML-2001), 2001, pp. 282–289.

[21] M. Li, L. Zhang, Multinomial mixture model with feature selection for text clustering, Knowledge-Based Systems 21 (7) (2008) 704–708.

[22] W.S. Lu, Approximate Bayesian Shrinkage Estimation, Annals of the Institute of Statistical Mathematics 46 (3) (1994) 497–507.

[23] W. Li, A. McCallum, Rapid development of Hindi named entity recognition using conditional random fields and feature induction, ACM Transactions on Asian Language Information Processing (TALIP) 2 (3) (2003) 290–294.

[24] Y. Matsuo, K. Uchiyama, Graph-based word clustering using web search engine, in: Proceedings of the EMNLP-2006, 2006.

[25] A. McCallum, R. Rosenfeld, T.M. Mitchell, A.Y. Ng, Improving text classification by shrinkage in a hierarchy of classes, in: Proceedings of the Fifteenth International Conference on Machine Learning, 1998.

[26] S. Miller, J. Guinness, A. Zamanian, Name tagging with word clusters and discriminative training, in: Proceedings of the HLT-NAACL-2004, 2004, pp. 337–342.

[27] A.Y. Ng, Feature selection, L1 vs. L2 regularization, and rotational invariance, in: Proceedings of the ICML-2004, 2004.

[28] F. Pereira, N. Tishby, L. Lee, Distributional clustering of English words, in: Proceedings of the ACL-1993, 1993, pp. 183–190.

[29] S. Perkins, K. Lacker, J. Theiler, Grafting: fast, incremental feature selection by gradient descent in function space, Journal of Machine Learning Research 3 (2003) 1333–1356.

[30] J.R. Quinlan, Induction of decision trees, Machine Learning 1 (1) (1986) 81–106.

[31] J.R. Quinlan, C4.5: Programs for Machine Learning, Morgan Kaufmann, San Francisco, 1993.

[32] M.I. Razzak, F. Anwar, S.A. Husain, A. Belaid, M. Sher, HMM and Fuzzy Logic: A Hybrid Approach for Online Urdu Script-based Languages' Character Recognition, Knowledge-Based Systems 23 (8) (2010) 914–923.

[33] S.K. Saha, P. Mitra, S. Sarkar, Word clustering and word selection based feature reduction for MaxEnt based Hindi NER, in: Proceedings of the ACL-2008:HLT, 2008, pages 488–495.

[34] A.K. Singh, Named entity recognition for south and south east Asian languages: taking stock, in: Proceedings of the IJCNLP-08 Workshop on NER for South and South East Asian Languages, 2008, pp. 5–16.

[35] A. Ushioda, Hierarchical clustering of words, in: Proceedings of the COLING-1996, 1996, pp. 1159–1162.

[36] J. Uszkoreit, T. Brants, Distributed word clustering for large scale class-based language modeling in machine translation, in: Proceedings of the ACL-08: HLT, 2008, pp. 755–762.

[37] Y. Yang and J.O. Pedersen, A comparative study on feature selection in text categorization, in: Proceedings of the ICML-1997, 1997, pp. 412–420.