Journal of Biomedical Informatics 42 (2009) 905–911

Contents lists available at ScienceDirect

Journal of Biomedical Informatics

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

Feature selection techniques for maximum entropy based biomedical named entity recognition

Sujan Kumar Saha *, Sudeshna Sarkar, Pabitra Mitra

Department of Computer Science and Engineering, Indian Institute of Technology, Kharagpur, West Bengal 721 302, India

ARTICLE INFO

Article history:
Received 27 May 2008
Available online 23 January 2009

Keywords:
Biomedical named entity recognition
Feature selection
Feature reduction
Maximum entropy classifier
Machine learning

ABSTRACT

Named entity recognition is an extremely important and fundamental task of biomedical text mining. Biomedical named entities include mentions of proteins, genes, DNA, RNA, etc which often have complex structures, but it is challenging to identify and classify such entities. Machine learning methods like CRF, MEMM and SVM have been widely used for learning to recognize such entities from an annotated corpus. The identification of appropriate feature templates and the selection of the important feature values play a very important role in the success of these methods. In this paper, we provide a study on word clustering and selection based feature reduction approaches for named entity recognition using a maximum entropy classifier. The identification and selection of features are largely done automatically without using domain knowledge. The performance of the system is found to be superior to existing systems which do not use domain knowledge.

1. Introduction

Named entities (NEs) are perhaps the most important indexing elements in biomedical text. The names of protein, DNA, RNA etc. are pivotal information for search and mining. Because of the complex nature of biomedical NEs, biomedical named entity recognition (NER) is still a challenging task.

In spite of the importance of the NER task in text mining in the biomedical domain, the recognition accuracy of current systems is significantly lower [14,27] compared to that of recognition accuracy of standard entities like person names and location names in the general domain or in the newswire domain. The complicated and ambiguous naming convention has been recognized as a source of difficulty [20,22] of this task. Biomedical NEs are often long and include common words, conjunctions, and prepositions. This makes the task of classification and boundary identification quite difficult. Spelling variation is another complicating factor. Further, the use of capitalization, parenthesis, hyphen and abbreviation does not follow a well-defined convention.

There are two main approaches to NER, namely rule based [7,9] and Machine Learning (ML) based. Rule based systems are difficult to develop for complex named entities. They require domain experts and it may be difficult to achieve high recall. Such systems are not portable to handle other NE types and domains. This is why ML based NER is a natural choice for complex domains. The success of a learning algorithm is crucially dependent on the features it uses. A supervised learning algorithm uses an annotated corpus. The training set derived from an annotated corpus represents the NEs in terms of the feature values.

A number of features have been used in the literature for NER in the general and the biomedical domain. The use of context features which include the words preceding and following the target word is quite common. Many NER systems use a word window size of five comprising of the current word and the two preceding and the two following words [22]. Certain suffixes and prefixes of words provide a good clue for classifying them as named entities, and NER systems often make use of these features. While domain specific NER systems use carefully crafted suffix lists that are known to be significant for the recognition task, a generic NER system makes use of all affixes of certain lengths as features.

Even though surrounding words and affixes are useful features, all of them are not equally important for the recognition task. This has motivated us to reduce the dimensionality of these features. We have used feature clustering and feature selection to achieve this reduction.

In our experiment, a Maximum Entropy (MaxEnt) classifier is trained using the JNLPBA 2004[1] data. We develop a baseline system using some general features including surrounding words and affixes. Subsequently we apply some feature reduction techniques to reduce the number of values of some of the features and use the reduced features for classification. This is found to improve the performance of the system.

* Corresponding author. Fax: +91 3222 278985.
E-mail addresses: sujan.kr.saha@gmail.com (S.K. Saha), shudeshna@gmail.com (S. Sarkar), pabitra@gmail.com (P. Mitra).

[1] http://research.nii.ac.jp/collier/workshops/JNLPBA04st.htm.

1532-0464/$ - see front matter © 2009 Elsevier Inc. All rights reserved.
doi:10.1016/j.jbi.2008.12.012

The paper is organized as follows. Section 2 contains a brief discussion on some of the previous work in biomedical NER as well as general approaches to feature reduction. The MaxEnt based NER system is described in Section 3. Various approaches for feature reduction are discussed in Section 4. Experimental results and discussions are given in Section 5. Finally Section 6 concludes the paper.

2. Background

Several ML algorithms have been used for biomedical NER development. Hidden Markov Model (HMM) [3,16,20,27], Maximum Entropy (MaxEnt) Lin et al., [14], Conditional Random Field (CRF) [13,19,22], Support Vector Machine (SVM) Kazama et al., [10] etc. are the commonly used techniques.

Supervised classification algorithms require annotated data. Several publicly available annotated corpora are available for the biomedical NER task. These include GENIA (2002), JNLPBA (2004), BioCreative (2004) and BioInfer (2007) [11,12,25,17].

Kazama et al. developed a SVM based NER system which achieved a *f*-value of 54.4 on GENIA V1.1 corpus [10]. The HMM based system developed by Shen et al. achieved a *f*-value of 62.5 on GENIA V1.1 and 66.1 on GENIA V3.0 corpus [20]. One of their experiments which used simple deterministic features (capitalization, digit information, word formation etc.), morphological information, part-of-speech (POS) information, head noun and verb triggers as features, achieved the highest *f*-value of 63.0 on GENIA V3.0. In the next experiment they added two additional components, namely abbreviation recognition and rule-based cascaded entity identification, and were able to increase the *f*-value of the system to 66.1.

Several systems participated in JNLPBA 2004 shared task. Among these, the highest accuracy was achieved by the system developed by Zhou and Su which produced a *f*-value of 72.55 [27]. This system used HMM and SVM with some deep knowledge resources. Without the domain knowledge the reported *f*-value of the system was 60.3. The addition of in domain POS information increased the *f*-value to 64.1. Deep domain knowledge like name alias resolution, cascaded NE resolution, abbreviation detection and external name dictionaries, when integrated in the system raised the *f*-value to 72.55. The second highest accuracy in the JNLPBA 2004 task was achieved by the Maximum Entropy Markov Model (MEMM) based system developed by Finkel et al. This system used external resources (like British National Corpus, large gazetteer lists, web), deeper syntactic features, etc. to achieve a *f*-value of 70.06 [4]. Some other systems in the shared task that achieved good accuracy also used some amount of domain knowledge or external resources. For example, the CRF based system developed by Settle [19] used semantic domain knowledge in the form of 17 lexicons. Song et al. expanded the corpus using a set of virtual examples which require some domain knowledge on the training data. They achieved a final *f*-value of 66.28 using CRF, SVM, post-processing and virtual samples. The baseline system achieved a *f*-value of 63.85 using SVM and 63.06 using CRF [21].

Following the JNLPBA 2004 shared task, several systems were developed using the released data. Tsai et al. developed a CRF based system that integrated linguistic knowledge and rule based postprocessing to achieve a *f*-value of 70.2 on the JNLPBA 2004 data [22]. Ponomareva et al. developed a HMM based NER system which used only POS information as in-domain feature and achieved a *f*-value of 65.7 which is better than that of the system by Zhou and that used POS information as the only domain knowledge [16].

While using machine learning methods, the increase in features do not always give rise to performance enhancement. In fact, the curse of dimensionality is a major issue when using high dimensional features. Several approaches have been proposed and used for dimensionality reduction [5]. Feature selection involves selecting a subset of the original features, and is a widely used dimensionality reduction technique. Several approaches have been tried for feature subset selection [2,6]. It has been observed in several studies that the use of an effectively selected feature subset may achieve better performance than the use of the original high dimensional feature set. Feature extraction is another approach considered in the literature for dimensionality reduction. Clustering is a method to achieve this. Several approaches for word clustering is proposed and used in different NLP tasks, for example, hierarchical word clustering has been used in POS tagging and classification of multi-word compounds [23], automatic thesaurus construction [26], named entity recognition [15,18], machine translation [24] etc.

In the literature we have found very little work on the use of word clustering and word selection techniques in the biomedical NER task. Ganchev et al. used distributional word clustering for the improvement of the performance of their biomedical NER system [8].

3. Maximum entropy named entity recognition

Now we describe our biomedical NER system based on Maximum Entropy (MaxEnt). We initially developed a system that uses a set of general features. Some of the features are binary-valued and rest of them are the multi-valued features. These multi-valued features take a large number of possible values. The dimension of the feature set becomes very high due to the presence of these multi-valued features. We realized that all the possible attributes of these multi-valued features are not important for the NER task, and this has motivated us to reduce the dimension of some of the features. Subsequently the reduced features are used to train the MaxEnt based system.

3.1. Maximum entropy model

Maximum Entropy principle is a commonly used technique which provides the probability of belongingness of a token to a class. MaxEnt computes the probability $p(o|h)$ for any o from the space of all possible outcomes O, 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 of probability ($p(o|h)$) 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_i has a weight α_i. We can compute the conditional probability as [1]:

$$p(o|h) = \frac{1}{Z(h)} \prod_i \alpha_i^{f_i(h,o)} \quad (1)$$

$$Z(h) = \sum_o \prod_i \alpha_i^{f_i(h,o)} \quad (2)$$

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[2].

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

[2] http://sourceforge.net/projects/maxent/.

the decoding using MaxEnt, if the tag having the highest probability value is considered as the output tag, then some of the 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.

3.2. Evaluation measures

The accuracies are measured in terms of the *f-measure*, which is the weighted harmonic mean of precision and recall. *Precision* is the percentage of the 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-value* is,

$$F_\beta = \frac{(1+\beta^2)\ (precision \times recall)}{(\beta^2 \times precision\ +\ recall)} \quad (3)$$

Here the value of β is taken as 1.

3.3. Feature set used by MaxEnt classifier

A MaxEnt model makes use of features for the recognition task. The features which we have used to develop the biomedical NER system are described below. These features are easy to derive and require no deep domain knowledge. Most of these features are general features and they are not specific to the biomedical domain.

3.3.1. Word feature

The current word and its context have been found to be very helpful in most recognition task. We have selected a word window of size 5 consisting of the current word, the previous two and the next two words.

3.3.2. Previous NE tags

NE tags of the previous words are helpful features. Our experiments confirmed that the use of up to two previous tags is found to be effective.

3.3.3. Capitalization and digit information

A few binary features are defined which use capitalization and digit information. The features are: initial capital, all capital, capital in inner, initial capital then mix, only digit, real number, digit with special character, initial digit then alphabetic, digit in inner, etc.

3.3.4. Special character

The presence of some special characters (e.g. ',','-','.',')') has proved to be helpful in the task. For example, the presence of '-' (hyphen) helps in identifying the NEs. This may not be so important in other domains, but in the biomedical domain, names often include '-' and the use of this feature helps. Some of the special characters are also found to be helpful in boundary detection.

3.3.5. Word normalization

Two types of normalization features are used. Firstly, the 'root' or lemma of the words are used as features. This helps us in handling plural forms, verb inflections, etc. The second type of normalization is based on word shape. For this, the capitalized characters are replaced by 'A', the small characters are replaced by 'a' and all the consecutive digits are replaced by '0'. For example, 'IL' is normalized to 'AA', 'IL-2' is normalized to 'AA-0' and 'IL-8' is also normalized to 'AA-0'.

3.3.6. Prefix and suffix information

Suffixes and prefixes provide useful clues for identifying NEs. During the experiments we have used all suffixes and prefixes with length up to 5 characters as features.

3.3.7. POS information

Part of Speech (POS) information is also important in the NER task. In our development we have used the POS values of the current, the previous two and the next two words as feature. To get the POS information we have used the GENIA tagger[3] V2.0.2, which is specially designed for the biomedical domain. The reported POS tagging accuracy of the tagger is 98.26% on GENIA corpus.

3.3.8. Trigger words

Two types of trigger words are used: head noun triggers and verb triggers. Head nouns are the main nouns or noun phrases (unigram or bigram) which occur very frequently in NEs. Some example of such nouns are receptor, protein and binding protein. Special verb triggers are the verbs which occur preceding to NEs and deliver useful information about the NE class. However, in the spirit of maintaining the domain independence of the system, we do not use a predefined list of trigger words. Instead these trigger words are extracted automatically from the training corpus based on their frequency of occurrence.

The MaxEnt classifier makes use of binary features that map from the class and its context to true or false. When a multi-valued feature is used, it is converted into several binary features. Thus if there are a total of N words in the training corpus, each of the word features corresponds to that many number of binary features. In our training corpus there are a total of ~22,000 unique words. Thus the surrounding word features of the two previous and the two next positions add as many as ~88,000 binary features. A similar increment in the dimension of the feature set occurs when the suffix and prefix information of the current word are used as feature templates. If the value of the maximum affix length (L) is large, then the total number of binary affix features becomes very large. So these high dimensional word and affix features make the overall dimension of the features very high.

But it is obvious that all the words in the corpus are not important for the recognition of the NEs, and only some of them play an important role in the recognition task. If only these informative words can be used, the feature dimension becomes smaller. For the affix features, the selection of L, the maximum affix length, is very crucial. For example, if L is selected as 5 then all the affixes of length up to 5 characters are used as features, yet some important affixes of length more than 5 characters (for example, peptide, vitamin) are left out. However taking a larger value for L is likely to include too many features, many of them unimportant.

The non-informative words and affixes inject noise and make the feature space unnecessarily high dimensional and may degrade the overall performance. To overcome this, we want to reduce the feature space so that the training become more effective. The details of our feature reduction approaches are given in the next section.

4. Feature selection techniques

If all feature values e.g. words, affixes etc. are used in constructing the binary features, the feature space becomes high dimensional. We describe below several techniques that we have used for feature reduction, with the aim of enhancing performance.

4.1. Informative word selection: surrounding words

NEs in biomedical domain are often much longer than those in the general domain. To accommodate for this, the use of context words have been given special importance. We have selected two

[3] http://www-tsujii.is.s.u-tokyo.ac.jp/GENIA/tagger/.

types of informative surrounding words: *intra NE words* and *extra NE words*.

4.1.1. Intra NE words

In the biomedical domain, many NEs are long and contain common words (which are generally not NE words) inside them. In the corpus there are about 9550 words which occur inside one or more NEs. If all these words were to be considered as informative words, many non-important words would be included in the list. For example, 'and' occurs 1074 times inside the NEs in the corpus, but it is not very useful for NE recognition. Similarly 'of' occurs 212 times, 'normal' occurs 137 times, 'active' occurs 24 times and 'low' occurs 10 times, but all these are words that do not play an important role in identifying the NEs. Our method of selection is described below.

To select the *intra NE words*, we first compiled a list comprising of all the words which are present inside the NEs. The words which contain no alphabetic characters (e.g. digits, real numbers) are removed from the list. Then for each word (w_i) in the list, *intraNeWeight* is calculated as:

$$intraNeWeight(w_i) = \frac{\text{number of occurrence of } w_i \text{ as part of a NE}}{\text{total occurrence of } w_i \text{ in the training corpus}} \quad (4)$$

Now the *intra NE words* are selected based on the *intraNeWeight* and number of occurrences. We have conducted some experiments to get the suitable threshold of the *interNeWeight* and the number of occurrence of the word, based on which we have selected the *intra NE words*.

The words which occur once or twice inside the NEs are not considered as informative. The rest of the words are divided into four Categories as follows.

- Category 1 includes the words that occur more than 100 times.
- Category 2 includes the words having occurrence $\geqslant$10 and <100.
- Category 3 includes the words having occurrence $\geqslant$5 and <10.
- Category 4 includes the words having occurrence <5.

We select a *Category* 1 word as informative word, if the *intraNeWeight* is greater than 0.5. A total of 130 words are selected from *Category* 1. For *Category* 2, 3 and 4 the minimum values of *intraNeWeight* are taken to be 0.75, 0.8 and 1.0, respectively. The numbers of selected words for these categories are 725, 810 and 775, respectively.

Using this procedure, a total of 2440 words are selected as *intra NE words*. Note that the category division is done only for the selection purpose. These categories and weights are not further used during the development of the system, but only the list of informative words are used.

4.1.2. Extra NE words

The words which are highly probable to occur at the preceding or following positions of the NEs, are selected as *extra NE words*. We define *context words* as those that occur in the proximity of a NE. We consider the previous two and the next two words which are denoted as w_{i-1}, w_{i+1}, w_{i-2} and w_{i+2} positions in the context of a NE w_i. Similar to the *intraNeWeight*, the *extraNeWeight* is defined as:

$$extraNeWeight(w_i) = \frac{\text{number of occurrence of } w_i \text{ as context word}}{\text{total occurrence of } w_i \text{ in the training corpus}} \quad (5)$$

A similar approach as in *intra NE words* is followed to select the *extra NE words*. The threshold values of *extraNeWeight* for the categories are taken to be 0.4 for *Category* 1, 0.6 for *Category* 2, 0.7 for *Category* 3 and 0.8 for *Category* 4. Using the procedure a total of 900 words have been selected as *extra NE words*.

To make the selection more effective, we have selected position specific informative words. We define *left context words* as the words which are present at the preceding positions of a NE. Similarly, *right context words* refer to the words that follow the occurrence of a NE. Now *leftNeWeight* and *rightNeWeight* are defined in a similar manner as the *extraNeWeight* using the *left context words* and *right context words*. The words having higher *leftNeWeight* are considered as informative words at preceding position (*left NE words*) and those with higher *rightNeWeight* are considered as informative words at the following positions (*right NE words*).

Certain words are highly probable to occur at both the preceding and following positions of NEs, i.e., *leftNeWeight* and *rightNeWeight* are not high, but the total weight as occurring at the surrounding positions of the NEs is high. To handle such words we have considered a third weight, called *lrNeWeight*, which is the sum of the *leftNeWeight* and *rightNeWeight*.

4.1.3. Reduced features

Now the word features are modified with the help of the selected informative words. There are four types of surrounding word features corresponding to two previous and two next positions. Previously these features considered the words present at $i-1$, $i-2$, $i+1$, $i+2$ positions (w_i is the target word). Now the features take the corresponding word as feature value if the word is an informative word. Consider an example sentence:

"Number of glucocorticoid receptors..."

Assume 'glucocorticoid' is the target word. To get the word feature value for the $(i+1)th$ position, the word 'receptor' is checked for its belongingness to the *intra NE word* or *lr NE word* or *right NE word* lists. The word 'receptor' is used as feature if it belongs to any of these categories, otherwise the feature value is taken to be 'null'. In this example, 'receptor' is an *intra NE word*, and the $word\_feature\_value_{+1}$ (*glucocorticoid*) = *receptor*.

4.2. Word clustering

The word selection procedure extracts the informative words from the corpus. The JNLPBA 2004 training corpus contains total ~22,000 different words and from these about 2440 words are selected as intra NE words and 900 words are selected as extra NE words. We will like to point out that among the top 10 most frequent words in the corpus, only one word (*cell*) is selected as informative word. During the feature definition we have assumed that if a context word belongs to the informative word list, then it is used as feature, otherwise its value is set to *null*. Thus there will be many cases of target words (w_i) for which all the surrounding word feature values are *null*. For these words no information is gained from the surrounding words. This results in some amount of information loss. To reduce the information loss, we have clustered the non-informative words and the clusters are used as features.

Clustering is the process of grouping together objects based on their similarity. To perform word clustering we should represent the words as vectors. We have experimented with two different vector representations with two different similarity measure approach. These approaches are similar to the word vector representation approaches defined by Saha et al. [18]. The approaches for representing the words as vectors are defined in the following.

4.2.1. Similarity based on proximal words

A word is represented as a vector based on the words in its proximity. The dimension of this vector is equal to the lexicon size (number of unique words in the corpus). For efficiency of implementation, we consider only 2×1000 words comprising of the 1000 most frequent preceding and 1000 most frequent following

words of a token word. *List_Prev* contains the most frequent (top 1000) 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 1000 most frequent next words in positions (w_{i+1} or w_{i+2} if w_i is the last word of a NE.

Suppose a particular word w occurs n times in the corpus. For each occurrence w_k of w, we find if its previous word (w_{k-1} or w_{k-2}) matches any element of *List_Prev*. If there is a match, then we set 1 to the corresponding position of the vector and set 0 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 $\vec{W}_k$ 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.

4.2.2. Similarity based on proximity to NE categories

This type of clustering finds position specific clusters. We consider two preceding and two following positions ($i-1$, $i-2$, $i+1$, $i+2$) of a word, and corresponding to these positions, we define four word vectors. Each vector is of dimension nine corresponding to five NE classes (C_j), one for the occurrence of the word as *intra NE word*, two for *left and right NE words* 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 belonging to a class C_j. The component of the word vector $\vec{W}_k$ for the position corresponding to C_j is $P_j(w_k)$. Measure the fraction of occurrence of w_k as *intra NE word*, *left NE word* and *right NE word* to get corresponding components of $\vec{W}_k$.

4.2.3. Clustering the word vectors

Once the word vectors are obtained, we cluster the words using the K-means clustering algorithm. The value of K (the number of clusters) is chosen as 200 after tuning during experiments. The seeds are chosen randomly. For the first type of vector representation, we have used cosine similarity, and for the second one, Euclidean distance measure. The cosine similarity between two word vectors ($\vec{X}$ and $\vec{Y}$) with dimension d is calculated as:

$$CosSim(\vec{X},\vec{Y}) = \frac{\sum_d X_d Y_d}{(\sum_d X_d^2)^{\frac{1}{2}} \times (\sum_d Y_d^2)^{\frac{1}{2}}} \quad (6)$$

The Euclidean distance is calculated as:

$$\text{Euclidean}(\vec{X},\vec{Y}) = \sqrt{\sum_d (X_d - Y_d)^2} \quad (7)$$

4.2.4. Surrounding word feature space construction

Now the word clusters are used to modify the surrounding word features by the following procedure:

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

Two types of clusters are obtained: one is position independent and the other is position specific. The position independent clusters are obtained using the vectors prepared using the *similarity based on proximal words* and the position specific clusters are obtained using the *similarity based on proximity to NE categories*. Corresponding to the two types of clustering, two different types of modifications are considered. For position independent clusters, the clusters containing the non-informative words need to be identified. But for position specific cluster based modification, the position of the word is considered and we look for the word from only the clusters prepared for that particular position.

As an illustration of position specific cluster based modification, consider the sentence:

"Number of glucocorticoid receptors..."

where 'glucocorticoid' is the target word (w_k). To get the word feature value for the $(k-1)th$ position, the word 'of' is checked to see if it is an *informative word*. As it does not happen to be an informative word, we search for 'of' in the clusters for the previous position (k − 1). The cluster-id of the cluster in which the word 'of' is found is used as the feature value for the previous (k − 1) word feature of 'glucocorticoid'.

During the experiments we have observed that the position specific clusters perform better than the position independent clusters. So, in the final system we have used the position specific clusters.

4.3. Modification of the affix features

Affix features are modified by selection and clustering in a manner similar to the process of modification of surrounding word features. The suffix and prefix features are handled separately but the same procedure is used for both.

Firstly, the list of all affixes of length up to eight characters is compiled. Then for each affix in the list, *affixWeight* is measured using the expression,

$$affixWeight(a_i) = \frac{\text{number of } intraNEwords \text{ contain } a_i}{\text{total occurrence of } a_i} \quad (8)$$

In Eq. (8), the *intra NE words* are considered to reduce the effect caused by the common words which occur as part of a NE in the training corpus. We have already mentioned that biomedical NEs often contain common words. So in this equation if we consider 'all words which occur as part of the NEs', then the *affixWeight* of some unimportant affixes become high. For example, 'and' occurs 1074 times inside the NEs in the training corpus. So the *affixWeight* of the suffix 'nd' will then include these 1074 occurrences and the weight becomes higher. To reduce this effect we have used *intra NE words*.

For a affix a_i, if the $affixWeight(a_i)$ is greater than 0.7, then a_i is considered as an informative affix; if $affixWeight(a_i)$ is less than 0.2 then it is considered as non-informative for the task. In order to reduce the feature space and information loss, the affixes having *affixWeight* between 0.2 and 0.7 are clustered.

For affix clustering, the defined affix vectors are of dimension six, corresponding to the five NE classes and one not-NE class. For a particular affix a_k, we estimate the fraction $P_j(a_k)$ of the total occurrences of the affix belonging to a class C_j. The component of the affix vector for the position corresponding to C_j is $P_j(a_k)$. The K-means clustering algorithm with Euclidean distance is used to cluster the affix vectors.

The affix features are then defined using the informative affixes and the affix clusters as done for the reduced word features.

5. Experimental results and discussion

In this section, we discuss the corpus and report the performance of the MaxEnt based NER system using the general features and the reduced features.

5.1. Training corpus

We have used the JNLPBA 2004 data for our experiments. 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 5 NE classes are considered: DNA, RNA, protein, cell-line and cell-type. To detect the NE boundaries, the corpus is annotated using BIO format, where 'B-ne' refers to the words which are the beginning word of a NE of

Table 1
Performance of the MaxEnt based NER system using the features mentioned in Section 3.3 [Pre, Precision; Rec, Recall; Fm, *F*-measure; CapDig, Capitalization & Digit Info.; Norm, Word normalization features].

F-id	Features	Pre	Rec	Fm
F1	Word (window 5)	54.89	57.34	56.09
F2	Word (window 7)	53	54.85	53.91
F3	Word (window 5), NE Tag	58.24	60.84	59.51
F4	Word, NE Tag, CapDig	60.54	61.2	60.87
F5	Word, NE Tag, CapDig, Spl. Char, Norm	59.92	63.02	61.43
F6	Word, NE Tag, CapDig, Spl. Char, Norm, Suffix & Prefix	61.89	64.29	63.07
F7	Word, NE Tag, CapDig, Spl. Char, Norm, Affix, POS	63.66	66.01	64.82
F8	Word, NE Tag, CapDig, Spl. Char, Norm, Affix, POS, Trigger words	64.76	66.85	65.79

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.

5.2. Performance using complete feature set

In Table 1 we have summarized the results when the general features are used (as defined in Section 3.3). In the table we have shown the contribution of each feature category in the recognition task. Using the defined features we have achieved the highest *f*-value of 65.79 (F8 in Table 1) where word and POS information of window 5, the previous NE tags, capitalization and digit information based features, word normalization, affix information and head noun triggers are used.

From the table it is observed that all the defined feature categories are able to increase the performance. We observe that when the window size is increased from 5 to 7, i.e. the three previous and three next words are considered, the *f*-value has decreased (F2 in table). This shows that the increment of the word window size (after a optimum value) increases the dimensionality and causes overfitting and the performance degrades. In F6 we have used suffix and prefix information where affixes of length up to five characters are considered. We have conducted other experiments (which are not shown in the table) to arrive at an optimum affix length of 5. We observed that when the affixes of length 6 and 7 are considered the performance is degraded.

5.3. Performance of the reduced features

The effectiveness of the reduced features (as defined in Section 4) are shown here. In the feature sets mentioned in Table 1, 'Word' refers to four sets of word features corresponding to the two previous and the two next positions. Each set contains N features where N is the number of unique words in the training corpus. During feature reduction we have considered M features from the original set of N features, where $M < N$. Similarly 'Affix' refers to all possible suffixes and prefixes up to a length of five characters, and the dimension of this feature is also reduced.

These reduced features now replace the corresponding full features. The best feature set from Table 1 is chosen and the reduced features are applied in the feature set replacing the corresponding general features. The improvement in accuracy demonstrates the effectiveness of the reduced features. In Table 2 we have shown the results using the reduced features.

From Table 2 it is observed that the reduced features improve the system accuracy in terms *f*-value from 65.79 to 67.41, which is the highest accuracy of our system. The highest improvement is achieved by the word selection and word clustering based reduced surrounding word features. Here the improvement of *f*-value is 1.42%. We also observe that the position specific clusters perform better than the position independent clusters. Here we like to mention that the results shown in Table 2 are obtained using the position specific clusters. Using the position independent clusters we have achieved the highest *f*-value of 67.33.

5.4. Comparison with existing biomedical NER systems

Now we compare the accuracy of our system with some other biomedical NER systems. As our system has not used deep domain knowledge (except the POS information), we compare its performance with those systems that do not use deep domain knowledge. In Table 3 we present a few comparisons.

We have used the MaxEnt classifier to build the system. The JNLPBA 2004 training corpus is used to train the classifier and the JNLPBA 2004 test data is used for evaluation. To make the comparison meaningful and fair we have selected only those NER systems which are built using the JNLPBA data. So all the systems mentioned in Table 3 have used the JNLPBA 2004 data for both training and testing.

Table 2
Performance of the reduced features in the MaxEnt based NER system.

Features	Pre	Rec	Fm
Word, NE Tag, CapDig, Spl. Char, Norm, Affix, POS, Trigger words	64.76	66.85	65.79
Reduced word, NE Tag, CapDig, Spl. Char, Norm, Affix, POS, Trigger words	67.52	66.9	67.21
Reduced word, NE Tag, CapDig, Spl. Char, Norm, Reduced affix, POS, Trigger words	67.86	66.94	67.41

Table 3
Comparison with other systems which use JNLPBA 2004 data.

System	ML approach	Domain knowledge	Fm
Our system	MaxEnt	POS information	67.41
Zhou & Su (2004) Final	HMM, SVM	Resolution of Name alias, Cascaded NEs, Abbreviations; Dictionary; POS	72.55
Zhou & Su (2004)	HMM, SVM	POS information	64.1
Song et al. (2004) Final	SVM, CRF	POS Information, Phrase, Virtual Sample	66.28
Song et al. (2004) Base	SVM	POS Information, Phrase	63.85
Ponomareva et al. (2007)	HMM	POS information	65.7

Zhou and Su developed the best system in the JNLPBA 2004 shared task. This system achieved a *f*-value of 72.55 with several deep domain knowledge [27]. But when this system used only POS information as domain knowledge, the *f*-value is 64.1, which is lower compared to our system. Song et al. developed the system using CRF, SVM and virtual samples and achieved the highest *f*-value of 66.28, which is lower compared to our system [21]. The baseline performance of the system is a *f*-value of 63.85 where only one classifier is used (SVM) and the virtual samples are not used. The HMM based system developed by Ponomareva et al. used POS information as domain knowledge and achieved a *f*-value of 65.7 [16].

Some other systems like, Finkel et al. (*f*-value—70.06), Settles B (*f*-value—69.8), Tsai et al. (*f*-value—70.2) etc. have used several domain knowledge, postprocessing, dictionaries and other external resources to achieve their best accuracies. As the accuracies of the systems without domain knowledge or external resources are not reported, we are not able to compare our system with these systems.

6. Conclusion

NER in biomedical texts is a complex task. We study a MaxEnt based machine learning approach in this paper. The performance of such approaches depends on the suitability of the features. We present a comparative study of different features that may be used. As domain dependent features are difficult to build they are not considered.

We show that the use of dimensionality reduction techniques can improve performance substantially. Two approaches to dimensionality reduction namely, informative word/affix selection, and word/affix clustering are used. The system has provided better performance than existing biomedical NER systems that do not use deep domain knowledge.

References

[1] Berger AL, Pietra SD, Pietra VD. A maximum entropy approach to natural language processing. Comput. Linguistic 1996;22(1):39–71.

[2] Chen Y, Li Y, Cheng XQ, Guo L. Survey and taxonomy of feature selection algorithms in intrusion detection system. In: Lipmaa H, Yung M, Lin D, editors. Inscrypt 2006, 4318. LNCS; 2006. p. 153–67.

[3] Collier N, Nobata C, Tsujii J. 2000. Extracting the names of genes and gene products with a hidden Markov model. In: Proceedings of COLING; 2000: p. 201–7.

[4] Finkel J, Dingare S, Nguyen H, Nissim M, Manning C. 2004. Exploiting context for biomedical entity recognition: from syntax to the Web. In: Proceedings of the international joint workshop on natural language processing in biomedicine and its applications at COLING; 2004 (JNLPBA 2004).

[5] Fodor IK. a survey of dimension reduction techniques. Technical report, Lawrence Livermore Nat Laboratory, Center for Applied Scientific Computing; 2002.

[6] Forman G. An extensive empirical study of feature selection metrics for text classification. J Machine Learning Res 2003;3:1289–305.

[7] Fukuda K, Tsunoda T, Tamura A, Takagi T. Toward information extraction: identifying protein names from biological papers. In: Proceedings of the pacific symposium on biocomputing; 1998: p. 707–18.

[8] Ganchev K, Crammer K, Pereira F, Mann G, Bellare K, McCallum A, et al. Penn/UMass/CHOP Biocreative II systems. In: Proceedings of the second biocreative challenge evaluation workshop; 2007.

[9] Grishman R. The New York University System MUC-6 or Where's the Syntax? In: Proceedings of the sixth message understanding conference; 1995.

[10] Kazama J, Makino T, Ohta Y, Tsujii J. Tuning support vector machines for biomedical named entity recognition. In: Proceedings of the workshop on natural language processing in the bio-medical domain at ACL; 2002: p. 1–8.

[11] Kim J, Ohta T, Tateisi Y, Tsujii J. Genia Corpus—a semantically annotated corpus for bio-textmining. Bioinformatics (Supplement: Eleventh International Conference on Intelligent Systems for Molecular Biology) 2003; 19: 180–2.

[12] Kim J, Ohta T, Tsuruoka Y, Tateisi Y, Collier N. Introduction to the bio-entity recognition task at JNLPBA. Nazarenko, editors, proceedings of the International Joint Workshop on Natural; 2004.

[13] Leaman R, Gonzalez G. Banner: an executable survey of advances in biomedical named entity recognition. Pacific Symp Biocomput 2008;13:652–63.

[14] Lin YF, Tsai TH, Chou WC, Wu KP, Sung TY, Hsu WL. A maximum entropy approach to biomedical named entity recognition. In: Proceedings of 4th workshop on data mining in bioinformatics 2004; pages.

[15] Miller S, Guinness J, Zamanian A. Name tagging with word clusters and discriminative training. In: Proceedings of the HLT-NAACL; 2004.

[16] Ponomareva N, Pla F, Molina A, Rosso P. Biomedical named entity recognition: a poor knowledge HMM-based approach. LNCS 2007;4592:382–7.

[17] Pyysalo S, Ginter F, Heimonen J, Bjorne J, Boberg J, Jarvinen J, et al. BioInfer: a corpus for information extraction in the biomedical domain. BMC Bioinfo 2007;8(50).

[18] Saha SK, Mitra P, Sarkar S. Word clustering and word selection based feature reduction for MaxEnt based Hindi NER. In: Proceedings of ACL-08: HLT; 2008: p. 488–95.

[19] Settles B. Biomedical named entity recognition using conditional random fields and rich feature sets. In: Proceedings of joint workshop on natural language processing in biomedicine and its applications (JNLPBA-2004); 2004.

[20] Shen D, Zhang J, Zhou GD, Su J, Tan CL. Effective adaptation of a hidden markov model-based named entity recognizer for biomedical domain. In: Proceedings of ACL 2003 workshop on natural language processing in biomedicine; 2003: p. 49–56.

[21] Song Y, Kim E, Lee GG, Yi BK. POSBIOTM-NER in the shared task of BioNLP/NLPBA. In: Proceedings of the joint workshop on natural language processing in biomedicine and its applications (JNLPBA-2004); 2004.

[22] Tsai T, Chou WC, Wu SH, Sung TY, Hsiang J, Hsu WL. Integrating linguistic knowledge into a conditional random field framework to identify biomedical named entities. Expert Syst Appl 2006;30(1):117–28.

[23] Ushioda A. Hierarchical clustering of words and application to NLP tasks. In: Proceedings of fourth workshop on very large corpora; 1996.

[24] Uszkoreit J, Brants T. Distributed word clustering for large scale class-based language modeling in machine translation. In: Proceedings of ACL-08: HLT; 2008: p. 755–62.

[25] Yeh A, Morgan A, Colosimo M, Hirschman L. BioCreAtIvE task 1A: gene mention finding evaluation. BMC Bioinfo 2005;6(Suppl. 1):S2.

[26] You JM, Chen KJ. Improving context vector models by feature clustering for automatic thesaurus construction. In: Proceedings of the fifth SIGHAN workshop on chinese language processing; 2006: p. 1–8.

[27] Zhou GD, Su J. Exploring deep knowledge resources in biomedical name recognition. In: Proceedings of joint workshop on natural language processing in biomedicine and its applications (JNLPBA-2004); 2004: p. 96–9.