Expert Systems with Applications 38 (2011) 14760–14772

Contents lists available at ScienceDirect

Expert Systems with Applications

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

A multiobjective simulated annealing approach for classifier ensemble: Named entity recognition in Indian languages as case studies [☆]

Asif Ekbal [1], Sriparna Saha [*,1]

Department of Computer Science and Engineering, Indian Institute of Technology Patna, Patna 800 013, India

A R T I C L E   I N F O

Keywords:
Natural language processing
Named entity recognition
Maximum Entropy (ME)
Conditional Random Field (CRF)
Support Vector Machine (SVM)
Multiobjective optimization (MOO)
Simulated annealing (SA)
Classifier ensemble
Weighted voting

A B S T R A C T

In this paper, we propose a simulated annealing (SA) based multiobjective optimization (MOO) approach for classifier ensemble. Several different versions of the objective functions are exploited. We hypothesize that the reliability of prediction of each classifier differs among the various output classes. Thus, in an ensemble system, it is necessary to find out the appropriate weight of vote for each output class in each classifier. Diverse classification methods such as Maximum Entropy (ME), Conditional Random Field (CRF) and Support Vector Machine (SVM) are used to build different models depending upon the various representations of the available features. One most important characteristics of our system is that the *features are selected and developed mostly without using any deep domain knowledge and/or language dependent resources*. The proposed technique is evaluated for Named Entity Recognition (NER) in three resource-poor Indian languages, namely Bengali, Hindi and Telugu. Evaluation results yield the recall, precision and *F*-measure values of 93.95%, 95.15% and 94.55%, respectively for Bengali, 93.35%, 92.25% and 92.80%, respectively for Hindi and 84.02%, 96.56% and 89.85%, respectively for Telugu. Experiments also suggest that the classifier ensemble identified by the proposed MOO based approach optimizing the *F*-measure values of named entity (NE) boundary detection outperforms all the individual models, two conventional *baseline* models and three other MOO based ensembles.

© 2011 Elsevier Ltd. All rights reserved.

1. Introduction

Named entity recognition (NER) has important role in many natural language processing (NLP) application areas such as information extraction, information retrieval, machine translation, question answering and automatic summarization etc. The main task of NER can be thought of as a two-step procedure that involves identifying every word/term from the text and then classifying them into some predetermined categories like person name, location name, organization name, miscellaneous name (date, time, percentage and monetary expressions) and "none-of-the-above".

The existing approaches of NER can be grouped into three main categories, namely rule based, machine learning (ML) based and hybrid approach. Rule based approaches focus on extracting names using a number of handcrafted rules. These kinds of systems yield good results for restricted domains, and are capable of detecting complex entities that are difficult with machine learning models. However, these types of systems are often domain dependent, language specific and do not necessarily adapt well to new domains and/or languages. In contrast, ML approaches (Borthwick, 1999; Ekbal & Bandyopadhyay, 2008b, 2009a, 2009b; Ekbal, Naskar, & Bandyopadhyay, 2007; Florian, Ittycheriah, Jing, & Zhang, 2003; Li & McCallum, 2004) have gained more attention to the researchers for NER because these are easily trainable, adaptable to different domains and languages as well as their maintenance being also less expensive. The main shortcoming of learning algorithms (especially, supervised) is the requirement of a large annotated corpus to obtain the reasonable performance. But, this is often a great problem to deal with the resource poor languages as the creation of this resource is both time-consuming and cost-sensitive. In hybrid systems, more robust systems are constructed by exploiting the effectiveness of both rule and ML based methods. Hybrid approaches, in general, can achieve better results in comparison to others, but the major bottleneck is the re-design of handcrafted rules (for rule based) when there is a need to change the domain and/or language of data.

Besides these, there are lots of other existing works in the area of NER. Majority of the existing works in the area of NER involves languages such as English, most of the European languages and some of the Asian languages like Chinese, Japanese and Korean. India is a multilingual country with great linguistic and cultural diversities. People speak in 22 different official languages that are derived from almost all the dominant linguistic families. However,

☆ The work was carried out at Heidelberg University, Germany.
* Corresponding author.
E-mail addresses: asif.ekbal@gmail.com, asif@iitp.ac.in (A. Ekbal), sriparna.saha@gmail.com, sriparna@iitp.ac.in (S. Saha).
[1] Authors equally contributed for this work.

0957-4174/$ - see front matter © 2011 Elsevier Ltd. All rights reserved.
doi:10.1016/j.eswa.2011.05.004

the works related to NER in Indian languages have started to emerge only very recently. Named entity (NE) identification in Indian languages is more difficult and challenging compared to others due to a number of reasons such as:

(1) Missing of capitalization information that acts as a good indicator for NE identification in English;
(2) Indian names are more diverse and a lot of these appear in the dictionary as common nouns, which make it more difficult to distinguish;
(3) Indian languages are relatively free word order in nature;
(4) Indian languages are resource-constrained, i.e., corpus, annotated corpus, name dictionaries, morphological analyzers, part of speech (POS) taggers, etc. are not readily available in the web.

As part of the Indian languages, there are some existing works that cover a few languages like Bengali, Hindi and Telugu. For Bengali, the existing works are based on unsupervised lexical pattern learning (Ekbal & Bandyopadhyay, 2007), Hidden Markov Model (HMM) (Ekbal et al., 2007) that considers additional context information during emission probabilities, Conditional Random Field (CRF) (Ekbal & Bandyopadhyay, 2009b), Support Vector Machine (SVM) (Ekbal & Bandyopadhyay, 2008b) and voting (Ekbal & Bandyopadhyay, 2009a). The works on Hindi can be found in Li and McCallum (2004) with a CRF approach and in Patel, Ramakrishnan, and Bhattacharya (2009) using rules. Various systems on NER in Indian languages using different approaches are reported in the proceedings of the IJCNLP-08 Workshop on NER for South and South East Asian Languages (NERSSEAL)[2]. As part of this workshop, Gali, Sharma, Vaidya, Shisthla, and Sharma (2008) reported a CRF-based system using some commonly used features, post-processing technique based on some heuristics or rules for Bengali, Hindi, Oriya, Telugu, and Urdu. Srikanth and Murthy (2008) reported a system for Telugu with person, location and organization tags. Shishtla, Pingali, and Varma (2008) presented a CRF-based system for English, Telugu and Hindi, where they suggested that a character n-gram based approach is more effective than word based models to increase the recall of NER system.

Classifier ensemble[3] is a new direction of machine learning. In this paper, we assume that rather than selecting the best-fitting feature set, constructing an ensemble using several models, where each one is based on different feature representations and/or different learning strategies could be more effective in order to achieve better performance. In the literature (Ekbal & Bandyopadhyay, 2009a; Florian et al., 2003), it has been shown that the combination of multiple classifiers could be more effective compared to any individual one for NER. The main idea behind classifier ensemble is that ensembles are often much more accurate than the individual classifiers that make them up. Usually the members of an ensemble are generated either by applying a single learning algorithm (i.e., using homogeneous classifiers) (Dietterich, 2000) or using different learning algorithms over a dataset (i.e., using heterogeneous classifiers) (Wolpert, 1992). Two basic approaches to combine the outputs of several classifiers are majority voting and weighted voting (Dietterich, 2000). The other existing classifier ensemble techniques can be grouped into: sub-sampling the training examples like bagging (Breiman, 1996; Freund & Schapire, 1995); manipulating the input features (Cherkauer, 1996), manipulating the output target like Error Correcting Output Codes (ECOC) (Dietterich & Bakiri, 1995); and injecting randomness in the learning algorithm (Kolen & Pollack, 1991). A major factor in classifier ensemble is that the individual classifiers should be as much diverse as possible. But, it is to be noted that all these existing ensemble techniques must need a way of combining the decisions of a set of classifiers. Moreover, all the classifiers are not good to detect all types of output classes. For example, in NER, some classifiers are good to detect *person names* whereas some are good to detect *location names*. Thus, in case of weighted voting, weights of voting should vary among the various output classes in each classifier. The weight should be high for that particular output class for which the classifier performs good. Otherwise, weight should be low for the output class for which its output is not very reliable. So, it is a very crucial issue to select the appropriate weights of votes per classifier.

In this paper, we propose a simulated annealing (SA) (Kirkpatrick, Gelatt, & Vecchi, 1983) based multiobjective optimization (MOO) (Deb, 2001) approach for classifier ensemble. The proposed method provides some automatic ways of finding out the appropriate weights of votes for all the classes in each classifier using the search capability of MOO. Thereafter the decisions of all classifiers are combined together to form an ensemble using our proposed approach. The diverse classification methodologies such as ME, CRF and SVM are used as the base classifiers to generate several different models based on the various representations of the features and/or feature templates. Most of these features are *selected and developed without using any deep domain knowledge and/or language dependent resources*. These features are language independent in nature, and thus applicable for many languages. In addition to these, we also use few language dependent features that are extracted from the language specific resources and tools. Some optimization techniques like SA (Kirkpatrick et al., 1983) may be used to determine the appropriate weights of votes for all the classes in each classifier. But, these single objective optimization techniques can only optimize a single quality measure, e.g., recall, precision or *F*-measure at a time. But, in reality, sometimes a single measure like these cannot capture the quality of a good ensemble reliably. Any good ensemble for NER should have its all the parameters like recall, precision and *F*-measure be optimized simultaneously. In order to achieve this, in the present work, we propose a new classifier ensemble technique based on MOO (Deb, 2001) that has a rather different perspective. While in single objective optimization there is only one global optimum, in MOO there is a set of global optimum solutions called Pareto optimal set (Deb, 2001). All these solutions have equal importance. A single objective approximation of multiple objectives, in form of a weighted sum, unfortunately often fails to capture the full Pareto front. Over the past decade, a number of MOO algorithms (MOOs) have been suggested (Deb, 2001). Here, we use a recently developed multiobjective simulated annealing based technique, AMOSA (archived multiobjective simulated annealing based technique) (Bandyopadhyay, Saha, Maulik, & Deb, 2008) as the optimization algorithm. The main advantage of AMOSA over other MOO algorithms is that it is more effective in handling many objective functions. We experiment with several objective functions: (a) overall recall and precision values, (b) *F*-measure values of all the individual output classes, i.e. of person, location, organization and miscellaneous, (c) recall and precision of all the individual output classes and (d) *F*-measure values of NE boundary detection. The proposed approach is evaluated for three resource-poor languages like Bengali, Hindi and Telugu. Our proposed technique is general enough to be applicable for other domains as well as languages.

The main contributions of our work are listed below:

- A multiobjective simulated annealing based technique is used for selecting best weights to form a classifier ensemble. We tried to establish that such ensemble is capable to increase the classification quality by a large margin compared to the conventional ensemble methods.

[2] http://ltrc.iiit.ac.in/ner-ssea-08.
[3] We use 'ensemble classifier' and 'classifier ensemble' interchangeably.

- To the best of our knowledge, use of multiobjective simulated annealing approach to select appropriate weights for voting is a novel contribution, especially in the area of NLP. Here we have used several different versions of objective functions.
- ME, CRF and SVM are used as the base classifiers. However, the proposed method will work for any set of classifiers, i.e. either homogeneous or heterogeneous. The proposed technique is a very general approach and its performance may further improve depending upon the choice and/or the number of classification models as well as the use of more diverse set of features.
- The proposed technique could be language independent and can be replicated for any resource-poor language very easily. We evaluated our proposed technique for three resource-poor languages, namely Bengali, Hindi and Telugu.
- The proposed framework is applicable for any type of classification problems like NER, Part of Speech (PoS)-tagging, question–answering, etc.
- Note, that our work proposes a novel way of combining the available classifiers. Thus, the accuracies of the existing ensemble works (e.g., (Ekbal & Bandyopadhyay, 2009a; Florian et al., 2003)) can be further improved with our framework.

The rest of the paper is organized as follows. Basic principles of simulated annealing (SA) are described in Section 2. Preliminaries of multiobjective optimization along with a brief description of AMOSA are presented in Section 3. The weighted vote based classifier ensemble selection problem is formulated in Section 4. Section 5 describes the details of the proposed algorithm. Section 6 depicts the set of NE features that we have used for NER in three leading Indian languages, namely Bengali, Hindi and Telugu. The base classifiers used for experiments are briefly described in Section 7. Datasets and detailed experimental results are reported in Section 8. Finally, we conclude in Section 9.

2. Simulated annealing: basic principles

Application of techniques having physical or natural correspondence for solving difficult optimization problems has been receiving widespread attention for the last two decades. It has been found that these techniques consistently outperform classical methods like gradient descent search when the search space is large, complex and multimodal. Simulated annealing (SA) (Kirkpatrick et al., 1983) is one such paradigm having its foundation in statistical mechanics, which studies the behavior of a very large system of interacting components in thermal equilibrium.

In statistical mechanics, if the system is in thermal equilibrium, the probability $\pi_T(s)$ that the system is in state s, $s \in S$, S being the state space, at temperature T, is given by

$$\pi_T(s) = \frac{e^{\frac{-E(s)}{kT}}}{\sum_{w \in S} e^{\frac{-E(w)}{kT}}}$$

where k is the Boltzmann's constant and $E(s)$ is the energy of the system in state s.

Metropolis, Rosenbluth, Rosenbloth, Teller, and Teller (1953) developed a technique to simulate the behavior of the system in thermal equilibrium at temperature T as follows: Let the system be in state q at time t. Then the probability p that it will be in state s at time $t + 1$ is given by the equation

$$p = \frac{\pi_T(s)}{\pi_T(q)} = e^{\frac{-(E(s)-E(q))}{kT}}$$

If the energy of the system in state s is less than that in state q, then $p > 1$ and the state s is automatically accepted. Otherwise it is accepted with probability p. Thus it is also possible to attain higher energy values. It can be shown that for $T \to \infty$, the probability that the system is in state s is given by $\pi_T(s)$ irrespective of the starting configuration (Geman & Geman, 1984).

When dealing with a system of particles, it is important to investigate very low energy states, which predominate at extremely low temperatures. To achieve such states, it is not sufficient to lower the temperature. An annealing schedule is used, where the temperature is first increased and then decreased gradually, spending enough time at each temperature in order to reach thermal equilibrium.

In AMOSA (Bandyopadhyay et al., 2008), the annealing process of the Boltzmann machine is used, which is a variant of the Metropolis algorithm. Here, at a given temperature T, the new state is chosen with a probability

$$p_{qs} = \frac{1}{1 + e^{\frac{-(E(q,T)-E(s,T))}{T}}} \tag{1}$$

The parameters of the search space are usually encoded in the form of strings of fixed length. The objective value associated with the string is computed and mapped to its energy. The string with the minimum energy value provides the solution to the problem. The initial string (say q) of 0s and 1s is generated randomly and its energy value is computed. Keeping the initial temperature high (say $T = T_{max}$), a neighbor of the string (say s) is generated by randomly flipping one bit. The energy of the new string is computed and it is accepted in favor of q with a probability p_{qs} mentioned earlier. This process is repeated a number of times (say k) keeping the temperature constant. Then the temperature is decreased using the equation $T = rT$, where $0 < r < 1$, and the k loops, as earlier, are executed. This process is continued till a minimum temperature (say T_{min}) is attained. The simulated annealing steps are shown in Fig. 1.

3. Multiobjective optimization

In many real world situations there may be several objectives that must be optimized simultaneously in order to solve a certain problem. This is in contrast to the problems tackled by conventional simulated annealing (SA) which involve optimization of just a single criterion. The main difficulty in considering MOO is that there is no accepted definition of optimum in this case, and therefore it is difficult to compare one solution with another one. In general, these problems admit multiple solutions, each of which is considered acceptable and equivalent when the relative importance of the objectives is unknown. The best solution is subjective

```
Begin
  generate the initial state $q$
  $T = T_{max}$
  Let $E(q,T)$ be the associated energy
  while $(T \geq T_{min})$
    for $i = 1$ to $k$
      Perturb $q$ to yield $s$
      Let $E(s,T)$ be the associated energy
      Set $q \leftarrow s$ with probability $\frac{1}{1+e^{-(E(q,T)-E(s,T))/T}}$
    end for
    $T = rT$
  end while
Decode $q$ to provide the solution of the problem.
End
```

Fig. 1. Steps of simulated annealing.

and depends on the need of the designer or decision maker. The MOO can be formally stated as (Coello Coello, 1999; Deb, 2001):

Find the vector $\bar{x}^* = [x_1^*, x_2^*, \ldots, x_n^*]^T$ of decision variables which will satisfy the m inequality constraints:

$$g_i(\bar{x}) \geqslant 0, \quad i = 1, 2, \ldots, m \tag{2}$$

the p equality constraints

$$h_i(\bar{x}) = 0, \quad i = 1, 2, \ldots, p \tag{3}$$

and optimize the vector function

$$\bar{f}(\bar{x}) = [\overline{f_1}(\bar{x}), \overline{f_2}(\bar{x}), \ldots, \overline{f_k}(\bar{x})]^T \tag{4}$$

The constraints given in Eqs. (2) and (3) define the feasible region F which contains all the admissible solutions. Any solution outside this region is inadmissible since it violates one or more constraints. The vector $\bar{x}^*$ denotes an optimal solution in F. In the context of MOO, the difficulty lies in the definition of optimality, since it is only rarely that we will find a situation where a single vector $\bar{x}^*$ represents the optimum solution to all the objective functions. The concept of *Pareto optimality* comes handy in the domain of MOO. A formal definition of Pareto optimality from the point of view of a minimization problem may be given as follows:

A decision vector $\bar{x}^*$ is called Pareto optimal if and only if there is no $\bar{x}$ that dominates $\bar{x}^*$, i.e., there is no $\bar{x}$ such that

$$\forall i \in 1, 2, \ldots, k, f_i(\bar{x}) \leqslant f_i(\bar{x}^*) \text{ and } \exists i \in 1, 2, \ldots, k, \ f_i(\bar{x}) < f_i(\bar{x}^*)$$

In other words, $\bar{x}^*$ is Pareto optimal if there exists no feasible vector $\bar{x}$ which causes a reduction on some criterion without a simultaneous increase in at least one other.

There are different approaches for solving MOO problems, e.g., aggregating, population based non-Pareto and Pareto-based techniques (Coello Coello, 1999; Deb, 2001). In aggregating techniques, the different objectives are generally combined into one using a weighting or a goal based method. Vector evaluated genetic algorithm (VEGA) is a technique in the population based non-Pareto approach in which different subpopulations are used for the different objectives. Evolutionary computation techniques have been heavily used for Pareto approaches, because of their population based nature, making them conducive for providing multiple solutions. Although simulated annealing (Kirkpatrick et al., 1983), another popular search technique based on the principle of statistical mechanics, has a well-founded theoretical framework (Geman & Geman, 1984), its efficient multiobjective versions have started appearing only recently (Bandyopadhyay et al., 2008), primarily because of its search from a point nature. A review of several earlier multi-objective simulated annealing algorithms and their comparative performance analysis can be found in Suman and Kumar (2006). In this article, the multiobjective simulated annealing technique called AMOSA (Bandyopadhyay et al., 2008) has been utilized that overcomes several limitations of some earlier SA based MOO techniques. Its comparative performance with several state-of-art evolutionary multiobjective approaches demonstrates its effectiveness for a wide variety of MOO problems (Bandyopadhyay et al., 2008).

3.1. AMOSA

In AMOSA (archived multiobjective simulated annealing) (Bandyopadhyay et al., 2008), which is an multiobjective version of SA, several concepts have been newly integrated. It utilizes the concept of an archive where the non-dominated solutions seen so far are stored. Two limits are kept on the size of the archive: a hard or strict limit denoted by *HL*, and a larger, soft limit denoted by *SL*, where *SL* > *HL*. The non-dominated solutions are stored in the archive as and when they are generated. In the process, if some members of the archive get dominated by the new solutions, then these are removed. If at some point of time, the size of the archive exceeds a specified value, then the clustering process, described below, is invoked.

In AMOSA, the initial temperature is set to T_{max}. Then, one of the points is randomly selected from the archive. This is taken as the *current-pt*, or the initial solution. The *current-pt* is perturbed to generate a new solution called *new-pt*, and its objective functions are computed. The domination status of the *new-pt* is checked with respect to the *current-pt* and the solutions in the archive. A new quantity called the amount of domination, $\Delta dom(a, b)$, between two solutions a and b is defined as follows:

$$\Delta dom_{a,b} = \prod_{i=1, f_i(a) \neq f_i(b)}^{M} \frac{|f_i(a) - f_i(b)|}{R_i} \tag{5}$$

where $f_i(a)$ and $f_i(b)$ are the ith objective values of the two solutions and R_i is the corresponding range of the objective function computed from the individuals in the population. M is the number of objectives. Based on the domination status between the *new-pt*, *current-pt* and the points in the archive, different cases may arise. These are discussed in details in Bandyopadhyay et al. (2008), and are briefly mentioned here for the sake of completeness.

Case 1: *new-pt* is either dominated by the *current-pt* or it is non-dominating with respect to the *current-pt*, but some points in the archive dominate the *new-pt*. Suppose *new-pt* is dominated by a total of k points (including *current-pt* and points in the archive). This case is demonstrated in Fig. 2 (the points D, E, F, G and H in the figure signify the content of the archive at any instant, while the other points illustrate different cases that may arise with respect to the archive) where F represents the *current-pt* and B represents the *new-pt*. Then a quantity Δdom_{avg} is computed as $\left(\sum_{i=1}^{k} (\Delta dom_{i,new-pt}) + \Delta dom_{current-pt,new-pt}\right)/(k+1)$. The *new-pt* is accepted as *current-pt* with a probability as defined in Eq. (1) with $-(E(q, T) - E(s, T)$ replaced by Δdom_{avg}. Note that Δdom_{avg} denotes the average amount of domination of the *new-pt* by $(k+1)$ points, namely, the *current-pt* and k points of the archive. Also, as k increases, Δdom_{avg} will increase since here the dominating points that are farther away from the *new-pt* are contributing to its value.

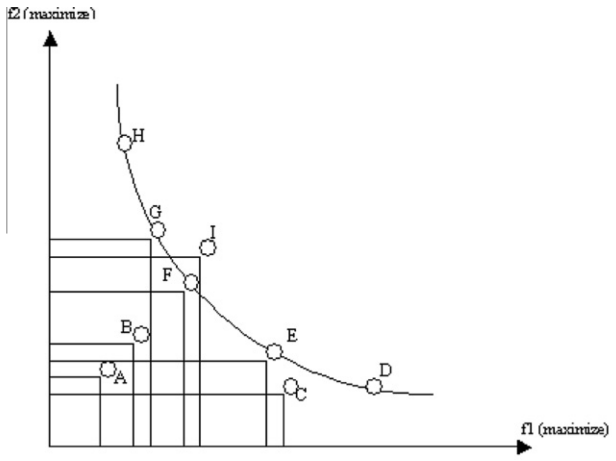

Fig. 2. Pareto-optimal front and different domination examples.

Case 2: Neither the *current-pt* nor the points in the archive dominate the *new-pt*. This can be demonstrated with different examples shown in Fig. 2, e.g., F represents the *current-pt* and E represents the *new-pt*, G represents the *current-pt* and I represents the *new-pt*, F represents the *current-pt* and I represents the *new-pt*. For all these cases, accept the *new-pt* as the *current-pt*. If there are any points in the archive which are dominated by *new-pt*, remove them from the archive. Add *new-pt* in the archive. If archive size crosses the *SL*, apply single linkage clustering to reduce its size to *HL*.

Case 3: *new-pt* dominates the *current-pt* but *k* points in the archive dominate the *new-pt*. This case can be demonstrated using Fig. 2 where A represents the *current-pt* and B represents the *new-pt*. Here the minimum of the differences of domination amounts between the *new-pt* and the *k* points, denoted by Δdom_{min} of the archive is computed. The point from the archive that corresponds to the minimum difference is selected as the *current-pt* with probability

$$probability = \frac{1}{1+\exp(\Delta dom_{min})} \tag{6}$$

Otherwise the *new-pt* is selected as the *current-pt*. This may be considered as an informed reseeding of the annealer only if the archive point is accepted.

The above process is repeated *iter* times for each temperature (*temp*). Temperature is reduced to $\alpha \times temp$, using the cooling rate of α till the minimum temperature, T_{min}, is attained. The process thereafter stops, and the archive contains the final non-dominated solutions.

It has been demonstrated in Bandyopadhyay et al. (2008) that the performance of AMOSA is better than that of NSGA-II (Deb, Pratap, Agarwal, & Meyarivan, 2002) and some other well-known MOO algorithms.

4. Problem formulation for classifier ensemble

Suppose, the N number of available classifiers be denoted by $C_1,\ldots,C_N$. Let, $\mathscr{A} = \{C_i :\ i = 1;N\}$ and there are M number of output classes. The weighted vote based classifier ensemble problem is then stated as follows:

Find the weights of votes V per classifier which will optimize a function $F(V)$. Here, V is a real array of size $N \times M$. $V(i,j)$ denotes the weight of vote of the ith classifier for the jth class. More weight is assigned for that particular class for which the classifier is more confident; whereas the output class for which the classifier is less confident is given less weight. $V(i,j) \in s[0,1]$ denotes the degree of confidence of the i^{th} classifier for the jth class. These weights are used while combining the outputs of classifiers using weighted voting. Here, F is a classification quality measure of the combined classifiers. The particular type of problem like NER has mainly three different kinds of classification quality measures, namely recall, precision and F-measure. Thus, $F \in$ {recall, precision, F-measure}.

The objective functions could be of different variations such as the overall recall and precision values, F-measure values of the individual classes, and F-measure values of NE boundary detection.

5. Proposed approach

A multiobjective SA, along the lines of AMOSA (Bandyopadhyay et al., 2008), is now developed for solving the weighted vote based classifier ensemble problem. Note, that although the proposed approach has some similarity in steps with AMOSA, any other existing MOO techniques could have been also used as the underlying MOO technique.

5.1. String representation and archive initialization

If the total number of available classifiers is M and total number of output classes is O, then the length of the string is $M \times O$. Each string encodes the weights of votes for possible O classes in each classifier. As an example, the encoding of a particular string is represented in Fig. 3. Here, M = 3 and O = 3 (i.e., total nine votes can be possible). The string represents the following voting ensemble:

The weights of votes for three different output classes for classifier 1 are 0.59, 0.12 and 0.56, respectively. Similarly, weights of votes for three different output classes are 0.09, 0.91 and 0.02, respectively for classifier 2 and 0.76, 0.5 and 0.21, respectively for classifier 3.

In the present work, we use real encoding. The entries of each string are randomly initialized to a real value (r) between 0 and 1. Here, $r = \frac{rand()}{RAND\_MAX+1}$. If the archive size is P then all the P number of strings of this archive are initialized in the above way.

5.2. Objective functions computation

Initially, the F-measure values of all the available classifiers (or, models) for each of the output classes are calculated based on the development data. Thereafter, we execute the following steps to compute the objective values.

(1) Suppose, there are total M number of classifiers. Let, the overall F-measure values of these M classifiers on the development data be F_i, $i = 1,\ldots,M$.
(2) We have M classes, each from a different classifier, for each word in the development data. Now for the ensemble classifier, the output class label for each word in the development data is determined using the weighted voting of these M classifiers' outputs. The weight of the output class provided by the ith classifier is equal to F_i. The combined score of a particular class for a particular word w is:

$$f(c_i) = \sum F_m \times I(m,i),$$
$$\forall m = 1 \text{ to } M \text{ and } op(w,m) = c_i$$

Here, $I(m, i)$ is the entry of the chromosome corresponding to the mth classifier and ith class; and $op(w, m)$ denotes the output class provided by the classifier m for the word w. The class receiving the maximum combined score is selected as the joint decision.
(3) We use the different variations of the objective functions as below:
   (a) *MOO1*: Overall recall and precision: (i) recall and (ii) precision.

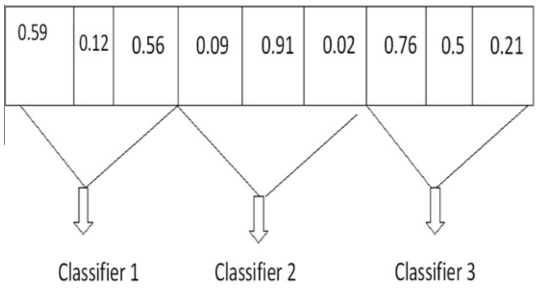

Fig. 3. String representation.

(b) *MOO2*: *F*-measure values of the individual output classes: (i) $F\text{-measure}_{PER}$, (ii) $F\text{-measure}_{LOC}$, (iii) $F\text{-measure}_{ORG}$ and (iv) $F\text{-measure}_{MISC}$.
(c) *MOO3*: recall and precision values of each individual output classes: (i) precision_{PER}, (ii) recall_{PER} (iii) precision_{LOC}, (iv) recall_{LOC}, (v) precision_{ORG}, (vi) recall_{ORG}, (vii) precision_{MISC} and (viii) recall_{MISC}.
(d) *MOO4*: *F*-measure values of each individual NE boundaries: (i) $F\text{-measure}_{B-PER}$, (ii) $F\text{-measure}_{I-PER}$, (iii) $F\text{-measure}_{B-LOC}$, (iv) $F\text{-measure}_{I-LOC}$, (v) $F\text{-measure}_{B-ORG}$, (vii) $F\text{-measure}_{I-ORG}$, (viii) $F\text{-measure}_{B-MISC}$ and (ix) $F\text{-measure}_{I-MISC}$.

In each of the above cases, AMOSA (Bandyopadhyay et al., 2008) is used to simultaneously maximize the above mentioned objective functions.

5.3. Mutation operation

Here, each position in a string is mutated with probability μ_m in the following way. The value is replaced with a random variable drawn from a *Laplacian distribution*, $p(\epsilon) \propto e^{-\frac{|\epsilon-\mu|}{\delta}}$, where the scaling factor δ sets the magnitude of perturbation. Here, μ is the value at the position which is to be perturbed. The scaling factor δ is chosen equal to 0.1. The old value at the position is replaced with the newly generated value. By generating a random variable using *Laplacian distribution*, there is a non-zero probability of generating any valid position from any other valid position while probability of generating a value near the old value is more.

5.4. Selection of a solution from the final pareto optimal front in MOO approach

In MOO, the algorithms produce a large number of non-dominated solutions (Deb, 2001) on the final Pareto optimal front. Each of these solutions provides a way of combining the available classifiers. All the solutions are equally important from the algorithmic point of view. But, sometimes the user may want only a single solution. Consequently, in this paper a method of selecting a single solution from the set of solutions is now developed.

For every solution on the final Pareto optimal front, the overall *F*-measure value of the vote based classifier ensemble is computed on the development data. We select the best solution as the one having the maximum *F*-measure value. Final results on the test data are reported using the classifier ensemble corresponding to this best solution. There can be many other different approaches of selecting a solution from the final Pareto optimal front.

6. Named entity features

The main features for the NER task are identified based on the different possible combinations of available word and tag contexts. We use the following features for constructing the various models of ME, CRF and SVM classifiers. Most of these features are *identified and generated without using any deep domain knowledge and/or language specific resources*. Due to this language independent nature, the features can be easily obtained for almost all the languages. However, in order to improve the overall system performance, we use few language dependent features that are extracted from the language specific resources or tools like gazetteers for Bengali.

(1) *Context words*: These are the preceding and succeeding words of the current word. This is based on the observation that surrounding words carry effective information for the identification of NEs.
(2) *Word suffix and prefix*: Fixed length (say, n) word suffixes and prefixes can be effectively used to identify NEs for the highly inflective languages. Actually, these are the fixed length character strings stripped either from the rightmost (for suffix) or from the leftmost (for prefix) positions of the words. For example, the suffixes of length up to 3 characters of the word "*ObAmA*" [Obama] are "*A*", "*mA*" and "*AmA*". Henceforth, all the Bengali glosses are written in ITRANS notation.[4] The prefixes of length up to three characters of the word "*ObAmA*" [Obama] are "*O*", "*Ob*" and "*ObA*". If the length of the corresponding word is less than or equal to $n-1$ then the feature values are not defined and denoted by ND. The feature value is also not defined (ND) if the token itself is a punctuation symbol or contains any special symbol or digit. This feature is included with the observation that NEs share some common suffixes and/or prefixes. Morphological analyzers or stemmers could be more effective to extract the meaningful affixes of the wordforms. But, we had to define features in this manner as morphological analyzers and/or stemmers are not readily available in Indian languages.
(3) *First word*: This is a binary valued feature that checks whether the current token is the first word of the sentence or not. We consider this feature with the observation that the first word of the sentence is most likely a NE. This is the most useful feature for Bengali as NEs generally appear in the first position of the sentence in news-wire data.
(4) *Length of the word*: This binary valued feature checks whether the number of characters in a token is less than a predetermined threshold value (here, set to 5). This feature is defined with the observation that very short words are most probably not the NEs.
(5) *Infrequent word*: This is a binary valued feature that checks whether the current word appears in the training set very frequently or not. We compile a list of most frequently occurring words from the training set by defining an appropriate threshold value. This threshold value does vary depending upon the size of the training set. In the present work, we consider the words to be infrequent if they have less than 10, 15 and 5 occurrences in the training sets of Bengali, Hindi and Telugu, respectively. A binary valued feature is then defined that fires if and only if the word does not appear in the list of frequently occurring words. We include this feature as the frequently occurring words are most likely not the NEs.
(6) *Position of word in sentence*: This feature checks whether the word is the last word of a sentence or not. In Indian languages, verbs generally appear in the last position of the sentence. Indian languages follow *subject–object–verb* structure. This feature distinguishes NEs from the verbs.
(7) *Part-of-Speech (PoS) information*: PoS information of the current and/or the surrounding tokens (s) are effective for NE identification. We use a SVM-based PoS tagger (Ekbal & Bandyopadhyay, 2008a) for Bengali, Hindi and Telugu. This PoS tagger was developed with a tagset[5] of 27 PoS tags, defined as part of the Indian languages. The PoS tagger has been trained with the Bengali, Hindi and Telugu data, obtained through our participations in the NLPAI_Contest06[6] and SPSAL2007[7] competitions. In this particular work, we evaluate the SVM-based PoS tagger with a coarse-grained tagset that

[4] http://www.aczone.com/itrans/.
[5] http://shiva.iiit.ac.in/SPSAL2007/iiit_tagset_guidelines.pdf.
[6] http://ltrc.iiitnet/nlpaicontest06/.
[7] http://shiva.iiit.ac.in/SPSAL2007/.

contains only three tags, namely Nominal, PREP (Postpositions) and Other. Postpositions are considered as they often appear after the NEs.

(8) *Digit features*: Several digit features are defined depending upon the presence and/or the number of digits and/or symbols in a token. These features are digitComma (token contains digit and comma), digitPercentage (token contains digit and percentage), digitPeriod (token contains digit and period), digitSlash (token contains digit and slash), digitHyphen (token contains digit and hyphen) and digitFour (token consists of four digits only).

(9) *Dynamic NE information*: This is the output label(s) of the previous token(s). The value of this feature is determined dynamically at run time.

(10) *Semantic feature*: This feature is semantically motivated. We consider all unigrams in contexts $w_{i-3}^{i+3} = w_{i-3} \cdots w_{i+3}$ of w_i (crossing sentence boundaries) for the entire training data. We convert tokens to lower case, remove stopwords, numbers and punctuation symbols. We define a feature vector of length 10 using the 10 most frequent content words. Given a classification instance, the feature corresponding to token t is set to 1 iff the context w_{i-3}^{i+3} of w_i contains t.

(11) *Features extracted from the gazetteers for Bengali*: We use different gazetteers that were prepared either manually or semi-automatically from the Bengali news corpus (Ekbal & Bandyopadhyay, 2008c). We use the lists of person, location, and organization names; lists (designations, common location, NE suffixes, action verbs etc.) of entities that are helpful to predict the appearance of NEs; and the lists of month names, weekdays etc. Detailed descriptions of these gazetteers can be found in Ekbal & Bandyopadhyay (2009b).

7. Base classifiers for NER

We use Maximum Entropy (ME), Conditional Random Field (CRF) and Support Vector Machine (SVM) as the base classifiers to construct the ensemble system based on weighted voting. Brief descriptions of these classifiers are given below.

7.1. Maximum Entropy Framework for NER

The ME framework estimates probabilities based on the principle of making as few assumptions as possible, other than the constraints imposed. Such constraints are derived from the training data, expressing some relationships between features and outcome. The probability distribution that satisfies the above property is the one with the highest entropy. It is unique, agrees with the maximum likelihood distribution, and has the exponential form

$$P(t|h) = \frac{1}{Z(h)} exp\left(\sum_{j=1}^{n} \lambda_j f_j(h,t)\right) \qquad (7)$$

where t is the NE tag, h is the context (or history), $f_j(h, t)$ are the features with associated weight λ_j and $Z(h)$ is a normalization function.

The problem of NER can be formally stated as follows. Given a sequence of words $w_1,\ldots,w_n$, we want to find the corresponding sequence of NE tags $t_1,\ldots,t_n$, drawn from a set of tags T, which satisfies:

$$P(t_1,\ldots,t_n|w_1,\ldots,w_n) = \prod_{i=1,2\ldots,n} P(t_i|h_i) \qquad (8)$$

where h_i is the context for the word w_i.

The features are, in general, binary valued functions, which associate a NE tag with various elements of the context. For example:

$$f_j(h,t) = 1 \text{ if word } (h) = \text{sachIn and } t = \text{Person name}$$
$$= 0 \text{ otherwise}$$

We use the OpenNLP Java based MaxEnt package[8] for the computation of the values of the parameters λ_j. This allows to concentrate on selecting the features, which best characterize the problem instead of worrying about assigning the relative weights to the features. We use the Generalized Iterative Scaling (Darroch & Ratcliff, 1972) algorithm to estimate the MaxEnt parameters.

7.2. Conditional Random Field Framework for NER

Conditional Random Fields (CRFs) (Lafferty, McCallum, & Pereira, 2001) are undirected graphical models, a special case of which corresponds to conditionally trained probabilistic finite state automata. Being conditionally trained, these CRFs can easily incorporate a large number of arbitrary, non-independent features while still having efficient procedures for non-greedy finite-state inference and training.

CRF is used to calculate the conditional probability of values on designated output nodes given values on other designated input nodes. The conditional probability of a state sequence $s = \langle s_1, s_2, \ldots, s_T \rangle$ given an observation sequence $o = \langle o_1, o_2, \ldots, o_T \rangle$ is calculated as:

$$P_\wedge(s|o) = \frac{1}{Z_o} \exp\left(\sum_{t=1}^{T}\sum_{k=1}^{K} \lambda_k \times f_k(s_{t-1}, s_t, o, t)\right)$$

where $f_k(s_{t-1}, s_t, o, t)$ is a feature function whose weight λ_k, is to be learned via training. The values of the feature functions may range between $-\infty, \ldots, +\infty$, but typically they are binary. To make all conditional probabilities sum up to 1, we must calculate the normalization factor,

$$Z_o = \sum_s \exp\left(\sum_{t=1}^{T}\sum_{k=1}^{K} \lambda_k \times f_k(s_{t-1}, s_t, o, t)\right)$$

which as in HMMs, can be obtained efficiently by dynamic programming.

To train a CRF, the objective function to be maximized is the penalized log-likelihood of the state sequences given the observation sequences:

$$L_\wedge = \sum_{i=1}^{N} \log(P_\wedge(s^{(i)}|o^{(i)})) - \sum_{k=1}^{K} \frac{\lambda_k^2}{2\sigma^2}$$

where $\{\langle o^{(i)}, s^{(i)} \rangle\}$ is the labeled training data. The second sum corresponds to a zero-mean, σ^2-variance Gaussian prior over parameters, which facilitates optimization by making the likelihood surface strictly convex. Here, we set parameters λ to maximize the penalized log-likelihood using Limited-memory BFGS (Sha & Pereira, 2003), a quasi-Newton method that is significantly more efficient, and which results in only minor changes in accuracy due to changes in λ.

When applying CRFs to the NER problem, an observation sequence is a token of a sentence or document of text and the state sequence is its corresponding label sequence. A feature function $f_k(s_{t-1}, s_t, o, t)$ has a value of 0 for most cases and is only set to be 1, when s_{t-1}, s_t are certain states and the observation has certain properties. We have used the C^{++} based CRF^{++} package,[9] a simple, customizable, and open source implementation of CRF for segmenting or labeling sequential data.

[8] http://maxent.sourceforge.net/.
[9] http://crfpp.sourceforge.net.

7.3. Support Vector Machine Framework for NER

In the field of NLP, Support Vector Machines (SVMs) (Vapnik, 1995) are applied to text categorization, and are reported to have achieved high accuracy without falling into over-fitting even though with a large number of words taken as the features (Joachims, 1999; Taira & Haruno, 1999). Suppose, we have a set of training data for a two-class problem: $\{(\mathbf{x}_1, y_1), \ldots, (\mathbf{x}_N, y_N)\}$, where $\mathbf{x}_i \in R^D$ is a feature vector of the ith sample in the training data and $y \in \{+1, -1\}$ is the class to which $\mathbf{x}_i$ belongs. In their basic form, a SVM learns a linear hyperplane that separates the set of positive examples from the set of negative examples with *maximal margin* (the margin is defined as the distance of the hyperplane to the nearest of the positive and negative examples). In basic SVMs framework, we try to separate the positive and negative examples by the hyperplane written as:

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

SVMs find the "optimal" hyperplane (optimal parameter $\overline{w}, b$) which separates the training data into two classes precisely.

The linear separator is defined by two elements: a weight vector $\mathbf{w}$ (with one component for each feature), and a bias b which stands for the distance of the hyperplane to the origin. The classification rule of a SVM is:

$$sgn(f(\mathbf{x}, \mathbf{w}, b)) \tag{9}$$

$$f(\mathbf{x}, \mathbf{w}, b) = \langle \mathbf{w} \cdot \mathbf{x} \rangle + b \tag{10}$$

being $\mathbf{x}$ the example to be classified. In the linearly separable case, learning the maximal margin hyperplane ($\mathbf{w}$, b) can be stated as a convex quadratic optimization problem with a unique solution: *minimize* $\|\mathbf{w}\|$, *subject to the constraints* (one for each training example):

$$y_i(\langle \mathbf{w} \cdot x_i \rangle + b) \geqslant 1 \tag{11}$$

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

$$f(\mathbf{x}, \alpha, b) = \sum_{i=1}^{N} y_i \alpha_i \langle \mathbf{x}_i \cdot \mathbf{x} \rangle + b \tag{12}$$

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

$$\mathbf{w}^* = \sum_{i=1}^{N} y_i \alpha_i^* \mathbf{x}_i \tag{13}$$

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

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

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

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

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

Support Vector Machines have advantages over conventional statistical learning algorithms from the following two aspects:

(1) SVMs have high generalization performance independent of dimension of feature vectors.
(2) SVMs can carry out their learning with all combinations of given features without increasing computational complexity by introducing the *Kernel function*.

We develop our system using SVM (Joachims, 1999, Vapnik, 1995) which perform classification by constructing an N-dimensional hyperplane that optimally separates data into two categories. We have used YamCha[10] toolkit, an SVM based tool for detecting classes in documents and formulating the NER task as a sequential labeling problem. Here, the *pairwise multi-class decision* method and the *polynomial kernel function* are used. We use TinySVM-0.07[11] classifier.

8. Datasets, experimental setup and evaluation results

In this section, we report the details of datasets used for experiment, experimental setup and evaluation results with necessary discussion.

8.1. Datasets for NER

Indian languages are resource-constrained in nature. For NER, we use a Bengali news corpus (Ekbal & Bandyopadhyay, 2008c), developed from the archive of a leading Bengali newspaper available in the web. We manually annotate a portion containing approximately 250 K wordforms with a coarse-grained NE tagset of four tags namely, PER (*Person name*), LOC (*Location name*), ORG (*Organization name*) and MISC (*Miscellaneous name*). The *Miscellaneous name* includes date, time, number, percentages, monetary expressions and measurement expressions. The data is collected mostly from the *national*, *states*, *sports* domains and the various sub-domains of *district* of the particular newspaper. This annotation was carried out by one of the authors and verified by an expert. We also use the IJCNLP-08 NER on South and South East Asian Languages (NERSSEAL)[12] Shared Task data of around 100 K wordforms that were originally annotated with a fine-grained tagset of twelve tags. This data is mostly from the *agriculture* and *scientific* domains. For Hindi and Telugu, we use the datasets obtained from the NERSSEAL shared task. The underlying reason to adopt the finer NE tagset in the shared task was to use the NER system in various NLP applications, particularly in machine translation. The IJCNLP-08 NERSSEAL shared task tagset is shown in Table 1. One important aspect of the shared task was to identify and classify the maximal NEs as well as the nested NEs, i.e. the constituent parts of a larger NE. But, the training data were provided with the type of the maximal NE only. For example, *mahatmA gAndhi roDa*(Mahatma Gandhi Road) was annotated as location and assigned the tag 'NEL' even if *mahatmA* (Mahatma) and *gAndhi* (Gandhi) are NE title person (NETP) and person name (NEP), respectively. The task was to identify *mahatmA gAndhi roDa* as a NE and classify it as NEL. In addition, *mahatmA* and *gAndhi* had to be recognized as NEs of categories NETP (Title person) and NEP (Person name), respectively.

[10] http://chasen-org/taku/software/yamcha/.
[11] http://cl.aist-nara.ac.jp/taku-ku/software/TinySVM.
[12] http://ltrc.iiit.ac.in/ner-ssea-08.

Table 1
NE tagset for Indian languages (IJCNLP-08 NERSSEAL Shared Task Tagset).

NE tag	Meaning	Example
NEP	Person name	*sachIna*/NEP, *sachIna ramesha tenDUlkara*/ NEP
NEL	Location name	*kolkAtA*/NEL, *mahatmA gAndhi roDa*/NEL
NEO	Organization name	*yadabpUra bishVbidyAlYa*/NEO,*bhAbA eytOmika risArcha sentAra*/ NEO
NED	Designation	*cheYArmAn*/NED, *sA.msada*/NED
NEA	Abbreviation	*bi e*/NEA, *ci em di a*/NEA,*bi je pi*/NEA, *Ai.bi.em*/ NEA
NEB	Brand	*fYAntA*/NEB
NETP	Title-person	*shrImAna*/NED, *shrI*/NED, *shrImati*/NED
NETO	Title-object	*AmericAn biUti*/NETO
NEN	Number	*10*/NEN, *dasha*/NEN
NEM	Measure	*tina dina*/NEM, *p.NAch keji*/NEM
NETE	Terms	*hidena markbha madela*/NETE, *kemikYAla riYYAkchYAna*/NETE
NETI	Time	10 *i mAgha 1402*/ NETI, 10 *ema*/NETI

In the present work, we consider only the tags that denote person names (NEP), location names (NEL), organization names (NEO), number expressions (NEN), time expressions (NETI) and measurement expressions (NEM). The NEN, NETI and NEM tags are mapped to the MISC tag that denotes miscellaneous entities. Other tags of the shared task are mapped to the 'other-than-NE' category denoted by 'O'. Hence, the tagset mapping now becomes as shown in Table 2.

In order to properly denote the boundaries of NEs, four basic NE tags are further divided into the format I-TYPE (TYPE → PER/LOC/ORG/MISC) which means that the word is inside a NE of type TYPE. Only if two NEs of the same type immediately follow each other, the first word of the second NE will have tag B-TYPE to show that it starts a new NE. For example, the name *mahatmA gAndhi*[Mahatma Gandhi] is tagged as *mahatmA*[Mahatma]/I-PER *gAndhi*[Gandhi]/I-PER. But, the names *mahatmA gAndhi*[Mahatma Gandhi] *rabIndrAnAth thAkur* [Rabindranath Tagore] are to be tagged as: *mahatmA*[Mahatma]/I-PER *gAndhi*[Gandhi]/I-PER *rabIndrAnAth*[Rabindranath]/B-PER *thAkur*[Tagore]/I-PER, if they appear sequentially in the text. This is the standard IOB format that was followed in the CoNLL-2003 shared task (Tjong Kim Sang & De Meulder, 2003). In order to report the evaluation results, we randomly partition each dataset into training, development and test sets as shown in Table 3.

8.2. Experimental setup

We use ME, CRF and SVM as the base classifiers to generate a number of models based on the features and/or feature templates.

Table 2
Tagset mapping table.

IJCNLP-08 shared task tag	Coarse-grained tag	Meaning
NEP	PER	Person name
NEL	LOC	Location name
NEO	ORG	Organization name
NEN, NEM, NETI	MISC	Miscellaneous name
NED, NEA, NEB, NETP, NETE	O	Other than NEs

Table 3
Statistics of the datasets.

Language	# Tokens in training	# Tokens in development	# Tokens in test
Bengali	277,947	35,009	37,053
Hindi	394,231	50,000	58,682
Telugu	47,179	10,000	6847

The parameters of AMOSA based ensemble technique are as follows: $Tmax = 100$, $Tmin = 0.00001$, $\alpha = 0.8$, $SL = 200$, $HL = 100$ and *iter* (number of iterations) = 50.

We define two different *baseline* classifier ensemble techniques as below:

- *Baseline 1*: In this *baseline* model, all the individual classifiers are combined together into a final system based on the majority voting of the output class labels. The choice of output is random if all the outputs differ to each other.
- *Baseline 2*: All the individual classifiers are combined with the help of a weighted voting approach. In each classifier, weight is calculated based on the *F*-measure value on the development data. The final output label is selected based on the highest weighted vote.

8.3. Evaluation results and discussion

We build a number of different ME, CRF and SVM models for Bengali by considering the various combinations of the available NE features and/or feature templates. In this particular work, we construct several models by considering the various subsets of the following set of features:

various context window within the previous three and next three words, i.e. within $w_{i-3}^{i+3} = w_{i-3} \cdots w_{i+3}$ of w_i, word suffixes and prefixes of length up to three (3 + 3 different features) or four (4 + 4 different features) characters, first word in the sentence, length of the word, infrequent word, last word in the sentence, several digit features, semantic feature, dynamic NE information and Gazetteer based feature.

A feature vector consisting of the features as described above is extracted for each word in the NE tagged corpus. Now, we have a training data in the form (W_i, T_i), where, W_i is the *i*th word and its feature vector and T_i is its corresponding output class. For CRF, we consider various combinations from the set of feature templates as given by,

$F_1 = \{w_{i-m}, \ldots, w_{i-1}, w_i, w_{i+1}, \ldots, w_{i+n}$; Combination of w_{i-1} and w_i; Combination of w_i and w_{i+1}; Feature vector consisting of all other features of w_i; B (bigram feature template)}.

Please note that in CRF, the 'bigram feature template' represents the combination of current and preceding output labels. For Bengali, we generate 152 different models based on ME classifier by varying the available features. Some (21) of these classifiers are shown in Table 4. Varying the available features and/or feature templates, we construct many CRF and SVM-based models, out of which 9 CRF-based and 8 SVM-based models are shown in Table 4. The CRF-based model exhibits the best performance with overall recall, precision and *F*-measure values of 91.42%, 92.55% and 91.98%, respectively. Overall evaluation results of several different versions of our proposed ensemble technique along with the best individual model and two different *baseline* ensembles are reported in Table 5.

Results show that the overall performance attained by the first version of our proposed algorithm, i.e. MOO1 is better than the best performing individual classifier with the increments of 1.45%, 1.12% and 1.28% recall, precision and *F*-measure points, respectively. MOO1 performs reasonably better than the two *baseline* models. It demonstrates the overall performance increments of 5.58% and 4.73% *F*-measure points over *Baseline 1* and *Baseline 2*, respectively. The relatively lower performance in each of the *baseline* ensembles is due to the (i) majority-voted combination of all the available classifiers (in first baseline) and (ii) mere assignment of the overall *F*-measure value as the weight of the classifier (in second baseline). This fact also votes in favor of our underlying assumption that determination of voting weights for all the classes in each classifier is very important. The MOO2 attains the overall

Table 4
Evaluation results with various feature combinations for the ME, CRF and SVM based models for Bengali. Here, the following abbreviations are used: ‘CW’: Context words, ‘PS’: Size of the prefix, ‘SS’: Size of the suffix, ‘WL’: Word length, ‘IW’: Infrequent word, ‘PW’: Position of the word in sentence, ‘FW’: First word, ‘DI’: ’Digit-Information’, ‘PoS’: ‘PoS information’ ‘NE’: Dynamic NE information, ‘FT’: ’Feature template for CRF’, ‘Sem’: Semantic feature, ‘Gaz’: Gazetteer based feature, ‘P’, ‘C’ and ‘N’: Previous, current and next tokens, ‘B’: ‘Bigram feature template’ $-i, j$: Words spanning from the ith left position to the jth right position, Current token is at 0th position, r’: recall, ‘p’: precision, ‘F’: F-measure, ‘X: Denotes the presence of the corresponding feature (we report in percentages).

Model	CW	FW	PS	SS	WL	IW	PW	DI	PoS	NE/FT	Sem	Gaz	r	p	F
M_9	−2, 2	X	3					X	X	X	X	X	85.69	89.92	87.75
M_{10}	−2, 1	X	3					X	X	X	X	X	85.92	89.87	87.85
M_{12}	−1, 1	X	3					X	X	X	X	X	86.05	88.96	87.48
M_{13}	−1, 2	X	3					X	X	X	X	X	86.03	89.70	87.83
M_{17}	−2, 2	X	3	3				X	X	X	X	X	86.87	90.09	88.45
M_{18}	−2, 1	X	3	3				X	X	X	X	X	86.82	90.28	88.52
M_{19}	−2, 0	X	3	3				X	X	X	X	X	85.92	89.36	87.60
M_{19}	−2, 0	X	3	3				X	X	X	X	X	85.92	89.36	87.60
M_{20}	−1, 1	X	3	3				X	X	X	X	X	86.48	88.63	87.54
M_{21}	−1, 2	X	3	3				X	X	X	X	X	87.12	89.52	88.30
M_{22}	0, 2	X	3	3				X	X	X	X	X	86.76	88.69	87.71
M_{24}	−3, 3	X	3	3				X	X	X	X	X	86.12	90.10	88.07
M_{57}	−2, 2	X	4	3				X	X	X	X	X	85.44	90.23	87.77
M_{58}	−2, 1	X	4	3				X	X	X	X	X	85.62	90.15	87.83
M_{60}	−1, 1	X	4	3				X	X	X	X	X	85.71	89.09	87.37
M_{61}	−1, 2	X	4	3				X	X	X	X	X	85.71	89.84	87.73
M_{65}	−2, 2	X	3	4				X	X	X	X	X	85.80	89.89	87.80
M_{66}	−2, 1	X	3	4				X	X	X	X	X	86.21	89.87	88.00
M_{67}	−2, 0	X	3	4				X	X	X	X	X	85.46	89.05	87.22
M_{68}	−1, 1	X	3	4				X	X	X	X	X	85.78	88.56	87.15
M_{69}	−1, 2	X	3	4				X	X	X	X	X	86.17	89.52	87.81
M_{72}	−3, 3	X	3	4				X	X	X	X	X	85.19	89.74	87.41
CRF_1	−2, 2	X	4	4	X	X	X	X	X	B	X	X	90.67	91.91	91.29
CRF_2	−3, 3	X	4	4	X	X	X	X	X	B	X	X	90.49	91.71	91.10
CRF_3	−3, 2	X	4	4	X	X	X	X	X	B	X	X	90.51	91.79	91.15
CRF_4	−1, 1	X	4	4	X	X	X	X	X	B	X	X	90.49	91.26	90.87
CRF_5	−2, 2	X	4	4	X	X	X	X	X	B		X	76.85	85.03	80.73
CRF_6	−2, 2	X	3 (P & C)	3 (P & C)	X	X	X	X	X	B	X	X	91.42	92.55	91.98
CRF_7	−2, 2	X	3	3	X	X	X	X	X	B	X	X	91.06	92.10	91.58
CRF_8	−2, 2	X	3 (C & N)	3 (C & N)	X	X	X	X	X	B	X	X	90.46	91.75	91.10
CRF_9	−1, 1	X	3	3	X	X	X	X	X	B	X	X	89.85	91.25	90.54
SVM_1	−1, 1	X	4	4	X	X	X	X	X	−1	X	X	89.58	89.31	89.44
SVM_2	−2, 2	X	4	4	X	X	X	X	X	−1	X	X	89.49	89.59	89.54
SVM_3	−2, 2	X	4	4	X	X	X	X	X	−2	X	X	89.13	89.19	89.16
SVM_4	−2, 2	X	3	3	X	X	X	X	X	−2	X	X	89.29	89.39	89.34
SVM_5	−2, 2	X	2	2	X	X	X	X	X	−2	X	X	88.52	88.95	88.73
SVM_6	−2, 2	X	2	2	X		X	X	X	−2	X	X	88.63	88.95	88.79
SVM_7	−2, 2	X	4	4	X	X	X	X	X	−2	X	X	75.93	83.32	79.45
SVM_8	−2, 2	X	3	3	X					−2		X	67.65	78.71	72.76

Table 5
Overall results for Bengali (we report in percentages).

Model	Recall	Precision	F-measure
Best individual classifier	91.42	92.55	91.98
Baseline 1	87.22	88.15	87.68
Baseline 2	88.01	89.05	88.53
MOO1	92.87	93.67	93.26
MOO2	93.23	94.21	93.71
MOO3	93.82	94.95	94.38
MOO4	93.95	95.15	94.55

recall, precision and F-measure values of 93.23%, 94.21% and 93.71%, respectively. These are actually the increments of 1.73%, 6.03%, 5.18% and 0.45% F-measure points over the best individual classifier, *Baseline 1*, *Baseline 2* and MOO1, respectively. The proposed MOO3 based technique performs better than both MOO1 and MOO2 with the increments of 1.12% and 0.67% F-measure points, respectively. Its effectiveness over the previous two models might be due to simultaneous optimization of both the metrics, recall and precision. The proposed MOO4 based technique that optimizes the F-measure values of NE boundary identification shows the best performance with the overall recall, precision and F-measure values of 93.95%, 95.15% and 94.55%, respectively. These are actually the increments of 2.57%, 6.87% and 6.02% F-measure points over the best individual classifier, *Baseline 1* and *Baseline 2*, respectively. It also shows better performance over MOO1, MOO2 and MOO3 with the improvements of 1.29, 0.84 and 0.17 F-measure points, respectively. The highest performance of this model reveals the fact that proper boundary identification of NEs is a crucial issue.

In order to show that the proposed MOO version really outperforms the best individual classifier and two *baseline* ensembles, statistical analysis of variance (ANOVA) (Anderson & Scolve, 1978) is performed, when each is executed ten times. ANOVA tests showed that the differences in mean recall, precision and F-measure are statistically significant as p value is less than 0.05 in each of the cases.

Thereafter, the proposed system is evaluated on Hindi data. We use the same set of features as Bengali except the gazetteer based features. Several different versions of ME, CRF and SVM based classifiers are built. Based on the performance, 39 model (ME:22, CRF:9, SVM:8), which are selected to construct the ensemble, are shown in Table 6. Like Bengali, CRF-based approach yields the best individual performance with the overall recall, precision and F-measure values of 88.72%, 90.10% and 89.40%, respectively. Thereafter, *baselines*, and MOO based ensemble techniques are executed on these available classifiers. Overall evaluation results are presented in Table 7. Results for Hindi shows that the first version of our proposed algorithm, i.e. MOO1 performs better than the best

Table 6
Evaluation results with various feature combinations for the ME, CRF and SVM based models for Hindi. Here, the abbreviations are same as Bengali (we report in percentages).

Model	Context	FW	PS	SS	WL	IW	PW	DI	PoS	NE/FT	Sem	r	p	F
ME_1	−2, 2	X	3					X	X	−1	X	81.73	89.09	85.25
ME_2	−2, 1	X	3					X	X	−1	X	82.13	88.71	85.29
ME_3	−1, 1	X	3					X	X	−1	X	82.99	89.02	85.90
ME_4	−1, 2	X	3					X	X	−1	X	82.33	89.52	85.78
ME_5	0, 2	X	3					X	X	−1	X	82.59	89.34	85.84
ME_6	0, 1	X	3					X	X	−1	X	85.19	90.67	87.85
ME_7	−2, 2	X	3	3				X	X	−1	X	84.16	92.21	86.00
ME_8	−2, 1	X	3	3				X	X	−1	X	83.03	90.00	86.37
ME_9	−2, 0	X	3	3				X	X	−1	X	82.03	86.79	85.28
ME_{10}	−1, 1	X	3	3				X	X	−1	X	82.63	89.66	86.00
ME_{11}	−1, 2	X	3	3				X	X	−1	X	82.53	90.03	86.12
ME_{12}	0, 2	X	3	3				X	X	−1	X	82.43	89.73	85.93
ME_{13}	0, 1	X	3	3				X	X	−1	X	83.72	89.52	86.53
ME_{14}	−3, 3	X	3	3				X	X	−1	X	81.23	90.91	85.80
ME_{15}	−2, 2	X	3	4				X	X	−1	X	81.99	90.35	85.97
ME_{16}	−2, 1	X	3	4				X	X	−1	X	82.69	89.73	86.07
ME_{17}	−2, 0	X	3	4				X	X	−1	X	82.03	88.91	85.33
ME_{18}	−1, 1	X	3	4				X	X	−1	X	83.03	89.65	86.21
ME_{19}	−1, 2	X	3	4				X	X	−1	X	82.63	90.27	86.28
ME_{20}	0, 2	X	3	4				X	X	−1	X	82.36	89.56	85.81
ME_{21}	0, 1	X	3	4				X	X	−1	X	83.29	89.43	86.26
ME_{22}	−3, 3	X	3	4				X	X	−1	X	81.19	90.74	85.71
CRF_1	−2, 2	X	4	4	X	X	X	X	X	B	X	88.02	89.36	88.68
CRF_2	−3, 3	X	4	4	X	X	X	X	X	B	X	87.42	88.90	88.15
CRF_3	−3, 2	X	4	4	X	X	X	X	X	B	X	87.92	89.35	88.63
CRF_4	−1, 1	X	4	4	X	X	X	X	X	B	X	88.62	89.78	89.20
CRF_5	−2, 2	X	4	4	X	X	X	X	X	B		66.82	80.32	72.95
CRF_6	−2, 2	X	3 (P & C)	3 (P & C)	X	X	X	X	X	B	X	87.72	89.17	88.44
CRF_7	−2, 2	X	3	3	X	X	X	X	X	B	X	88.72	90.10	89.40
CRF_8	−2, 2	X	3 (C & N)	3 (C & N)	X	X	X	X	X	B	X	87.59	89.07	88.32
CRF_9	−1, 1	X	3	3	X	X	X	X	X	B	X	87.45	89.08	88.26
SVM_1	−1, 1	X	4	4	X	X	X	X	X	−1	X	87.95	88.33	88.14
SVM_2	−2, 2	X	4	4	X	X	X	X	x	−1	X	88.38	89.24	88.81
SVM_3	−2, 2	X	4	4	X	X	X	X	X	−2	X	85.55	86.27	85.91
SVM_4	−2, 2	X	3	3	X	X	X	X	X	−2	X	84.89	85.63	85.26
SVM_5	−2, 2	X	2	2	X	X	X	X	X	−2	X	83.72	84.65	84.18
SVM_6	−2, 2	X	2	2	X	X	X	X	X	−2	X	83.22	84.09	83.65
SVM_7	−2, 2	X	4	4	X	X	X	X	X	−2	X	83.79	84.46	84.12
SVM_8	−2, 2	X	3	3					X	−2		64.65	75.71	69.74

performing individual model with the increments of 1.94% *F*-measure points. Results again show that MOO1 performs reasonably better in comparison to *Baseline 1* and *Baseline 2* with the increments of 16.65% and 7.70% *F*-measure points, respectively. The MOO2 based ensemble attains the overall recall, precision and *F*-measure values of 92.45%, 90.79%, and 91.62%, respectively, which is an increment of 0.28% *F*-measure points over MOO1. This result reveals the fact that optimization of *F*-measure values of individual output classes is more worthy than optimizing the overall recall and precision values. The MOO3 based technique performs better than both MOO1 and MOO2 with the increments of 0.64% and 0.36% *F*-measure points, respectively. The proposed MOO4 based technique that optimizes the *F*-measure values of NE boundary identification shows the best performance with the overall recall, precision and *F*-measure values of 93.35%, 92.25% and 92.80%, respectively. These are actually the increments of 3.40%, 18.11% and 9.16% *F*-measure points over the best individual classifier, *Baseline 1* and *Baseline 2*, respectively. It also shows better performance over MOO1, MOO2 and MOO3 with the improvements of 1.46, 1.18 and 0.82 *F*-measure points, respectively. The relatively poor performance of *baselines* again suggests that rather than combining all the classifiers blindly or eliminating some classifiers completely, it is more effective to quantify the amount of voting of each class in any classifier.

Finally, we apply our proposed method for Telugu, another popular language widely spoken in the southern part of India. Initially, several different versions of ME, CRF and SVM based classifiers are developed using the same set of features as of Hindi. Among them, based on the performance, 29 ME-based, 9 CRF-based and 8 SVM-based models are selected to construct an ensemble. Results of these models are reported in Table 8. The best individual model corresponds to a SVM-based classifier which yields recall, precision and *F*-measure values of 77.42%, 77.99% and 77.70%, respectively. Overall evaluation results are presented in Table 9. It shows that all the proposed MOO based ensemble techniques perform superior to the best individual model and the two *baseline* ensembles. Evaluation results show the recall, precision and *F*-measure values of 82.79%, 95.18% and 88.55%, respectively for MOO1, 82.92%, 95.29% and 88.68%, respectively for MOO2, 83.67%, 96.01% and 89.42%, respectively for MOO3 and 84.02%, 96.56% and 89.85%, respectively for MOO4. The highest performance attained by MOO4 is actually the increment of 12.15%, 18.62%, 9.02%, 1.30%, 1.17% and 0.43% *F*-measure points over the best individual model, *Baseline 1*, *Baseline 2*, MOO1, MOO2 and MOO3, respectively.

Table 7
Overall results for Hindi (we report in percentages).

Model	Recall	Precision	*F*-measure
Best individual classifier	88.72	90.10	89.40
Baseline 1	63.32	90.99	74.69
Baseline 2	74.67	94.73	83.64
MOO1	92.07	90.63	91.34
MOO2	92.45	90.79	91.62
MOO3	92.82	91.15	91.98
MOO4	93.35	92.25	92.80

Table 8
Evaluation results with various feature combinations for the ME, CRF and SVM based models for Telugu. Here, the abbreviations are same as Bengali (we report in percentages).

Model	Context	FW	PS	SS	WL	IW	PW	DI	PoS	NE/FT	Sem	*r*	*p*	*F*
ME_1	−2, 2	X	3	–				X	X	−1	X	67.40	80.94	73.55
ME_2	−2, 1	X	3	–				X	X	−1	X	67.40	80.52	73.38
ME_3	−1, 1	X	3	–				X	X	−1	X	69.01	80.52	74.32
ME_4	−1, 2	X	3	–				X	X	−1	X	67.40	80.11	73.20
ME_5	0, 2	X	3	–				X	X	−1	X	67.54	80.84	73.59
ME_6	0, 1	X	3	–				X	X	−1	X	69.16	79.50	73.97
ME_7	−2, 2	X	3	3				X	X	−1	X	69.01	84.79	76.09
ME_8	−2, 1	X	3	3				X	X	−1	X	69.01	84.79	76.09
ME_9	−2, 0	X	3	3				X	X	−1	X	69.45	84.73	76.33
ME_{10}	−1, 1	X	3	3				X	X	−1	X	71.35	84.98	77.57
ME_{11}	−1, 2	X	3	3				X	X	−1	X	70.48	85.24	77.16
ME_{12}	0, 2	X	3	3				X	X	−1	X	70.33	85.07	77.00
ME_{13}	0, 1	X	3	3				X	X	−1	X	70.97	83.30	76.64
ME_{14}	−3, 3	X	3	3				X	X	−1	X	66.52	83.94	74.22
ME_{15}	−2, 2	X	4	3				X	X	−1	X	69.01	83.61	75.61
ME_{16}	−2, 1	X	4	3				X	X	−1	X	70.33	84.04	76.58
ME_{17}	−2, 0	X	4	3				X	X	−1	X	69.30	83.10	75.57
ME_{18}	−1, 1	X	4	3				X	X	−1	X	71.79	83.37	77.15
ME_{19}	−1, 2	X	4	3				X	X	−1	X	70.92	83.46	76.68
ME_{20}	0, 2	X	4	3				X	X	−1	X	70.92	84.02	76.92
ME_{21}	0, 1	X	4	3				X	X	−1	X	71.53	82.94	76.81
ME_{22}	−3, 3	X	4	3				X	X	−1	X	67.10	83.02	74.22
ME_{23}	−1, 1	X	3	4				X	X	−1	X	69.74	82.07	75.41
ME_{24}	−1, 2	X	3	4				X	X	−1	X	68.72	81.96	74.76
ME_{25}	0, 2	X	3	4				X	X	−1	X	68.13	83.12	74.88
ME_{26}	0, 1	X	3	4				X	X	−1	X	70.48	82.10	75.85
ME_{27}	−1, 1	X	4	4				X	X	−1	X	69.45	79.97	74.34
ME_{28}	0, 2	X	4	4				X	X	−1	X	68.28	80.75	73.99
ME_{29}	0, 1	X	4	4				X	X	−1	X	70.33	80.58	75.11
CRF_1	−2, 2	X	4	4	X	X	X	X	X	B	X	74.63	75.18	74.91
CRF_2	−3, 3	X	4	4	X	X	X	X	X	B	X	73.75	74.30	74.02
CRF_3	−3, 2	X	4	4	X	X	X	X	X	B	X	74.34	74.89	74.61
CRF_4	−1, 1	X	4	4	X	X	X	X	X	B	X	76.39	76.96	76.67
CRF_5	−2, 2	X	4	4	X	X	X	X	X	B		37.98	94.19	54.13
CRF_6	−2, 2	X	3 (P & C)	3 (P & C)	X	X	X	X	X	B	X	73.31	75.08	74.18
CRF_7	−2, 2	X	3	3	X	X	X	X	X	B	X	75.66	77.36	76.50
CRF_8	−2, 2	X	3 (C & N)	3 (C & N)	X	X	X	X	X	B	X	73.61	75.04	74.32
CRF_9	−1, 1	X	3	3	X	X	X	X	X	B	X	73.17	74.70	73.93
SVM_1	−1, 1	X	4	4	X	X	X	X	X	−1	X	77.42	77.99	77.70
SVM_2	−2, 2	X	4	4	X	X	X	X	X	−1	X	77.42	77.99	77.70
SVM_3	−2, 2	X	4	4	X	X	X	X	X	−2	X	74.34	74.89	74.61
SVM_4	−2, 2	X	3	3	X	X	X	X	X	−2	X	73.46	74.00	73.73
SVM_5	−2, 2	X	2	2	X	X	X	X	X	−2	X	71.99	72.53	72.26
SVM_6	−2, 2	X	2	2	X	X	X	X	X	−2	X	73.61	74.15	73.88
SVM_7	−2, 2	X	4	4	X	X	X	X	X	−2	X	71.55	76.85	74.12
SVM_8	−2, 2	X	3	3					X	−2	X	73.90	74.45	74.17

Table 9
Overall results for Telugu (we report in percentages).

Model	Recall	Precision	*F*-measure
Best individual classifier	77.42	77.99	77.70
Baseline 1	60.12	87.39	71.23
Baseline 2	71.87	92.33	80.83
MOO1	82.79	95.18	88.55
MOO2	82.92	95.29	88.68
MOO3	83.67	96.01	89.42
MOO4	84.02	96.56	89.85

8.4. Summary of results

Evaluation results on all the languages show that, in general, the proposed MOO based approach optimizing the *F*-measure values of NE boundary identification, i.e. MOO4 performs superior to all the other approaches. This confirms that proper boundary identification plays an important role in NE identification. Results on three datasets show that MOO approach optimizing *F*-measure values of individual output classes (i.e., MOO2) performs better than MOO approach optimizing the overall recall and precision values (i.e., MOO1). The superiority of MOO3 over MOO2 suggests that optimization of both recall and precision metrics of only the individual output classes works better than the optimization of only *F*-measure metric. For all the languages, all the proposed models perform better than the conventional *baseline* ensembles. This shows the importance of finding proper weights of each class in each model rather than combing all the models' decisions together either by majority voting or simply by considering the overall *F*-measure value as the weight of each classifier. Sometimes, the *baselines* even show inferior performance in comparison to the best individual model. Evaluation also confirms the fact that eliminating some classifiers completely may not be a good idea to construct an ensemble.

9. Conclusion

In this paper, we have proposed a MOO based classifier ensemble technique that makes use of a new multiobjective simulated annealing technique, AMOSA. Here, we present the problem of selecting the appropriate votes for each class per classifier in NER as an optimization problem. Our underlying assumption is that instead of eliminating some classifiers completely, it is better to quantify the amount of votes for all the classes in each classifier. We have experimented with the several different variations of the objective functions and evaluated our proposed ensemble technique for three Indian languages, namely Bengali, Hindi and Telugu. For each of these languages, evaluation shows that the overall performance attained by the proposed MOO based techniques outperform the best individual classifier and two different *baseline* ensembles. Results show that AMOSA optimizing *F*-measure values of NE boundary detection performs better than three other MOO versions.

In future we plan to evaluate our proposed algorithm for other Indian languages as well as for English and some European languages. We would also like to evaluate our proposed technique for NER in biomedical textual data and for other application areas like PoS tagging, question–answering etc.

References

Anderson, T. W., & Scolve, S. (1978). *Introduction to the statistical analysis of data*. Mifflin: Houghton.

Bandyopadhyay, S., Saha, S., Maulik, U., & Deb, K. (2008). A simulated annealing based multi-objective optimization algorithm: AMOSA. *IEEE Transactions on Evolutionary Computation, 12*(3), 269–283.

Borthwick, A. (1999). *Maximum entropy approach to named entity recognition*. Ph.D. dissertation, New York University.

Breiman, L. (1996). Bagging predictors. *Machine Learning, 24*(2), 123–140.

Cherkauer, K. (1996). Human expert-level performance on a scientific image analysis task by a system using combined artificial neural networks. In *Working notes of the AAAI workshop on integrating multiple learned models* (pp. 15–21).

Coello Coello, C. A. (1999). A comprehensive survey of evolutionary-based multiobjective optimization techniques. *Knowledge and Information Systems, 1*(3), 129–156.

Darroch, J., & Ratcliff, D. (1972). Generalized iterative scaling for log-linear models. *Annals of Mathematical Statistics, 43*, 1470–1480.

Deb, K. (2001). *Multi-objective optimization using evolutionary algorithms*. England: John Wiley and Sons, Ltd.

Deb, K., Pratap, A., Agarwal, S., & Meyarivan, T. (2002). A fast and elitist multiobjective genetic algorithm: NSGA-II. *IEEE Transactions on Evolutionary Computation, 6*(2), 181–197.

Dietterich, T. G. (2000). Ensemble methods in machine learning. In J. Kittler & F. Roli (Eds.), *Multiple Classifiers Systems: First International Workshop, Proceedings/ MCS 2000*. Springer.

Dietterich, T. G., & Bakiri, G. (1995). Solving multiclass learning problems via error correcting output codes. *Journal of Artificial Intelligence Research, 2*, 263–286.

Ekbal, A., & Bandyopadhyay, S. (2008a). *Web-based Bengali news corpus for lexicon development and POS tagging, POLIBITS* (Vol. 37, pp. 20–29). ISSN 1870-9044.

Ekbal, A., & Bandyopadhyay, S. (2007). Lexical pattern learning from corpus data for named entity recognition. In *Proceedings of the 5th international conference on natural language processing (ICON), India* (pp. 123–128).

Ekbal, A., & Bandyopadhyay, S. (2008b). Bengali named entity recognition using support vector machine. In *Proceedings of workshop on NER for South and South East Asian languages, 3rd international joint conference on natural language processing (IJCNLP), India* (pp. 51–58).

Ekbal, A., & Bandyopadhyay, S. (2009). Voted NER system using appropriate unlabeled data. In *Proceedings of the 2009 named entities workshop: Shared task on transliteration (NEWS 2009), ACL-IJCNLP 2009* (pp. 202–210).

Ekbal, A., & Bandyopadhyay, S. (2008c). A web-based Bengali news corpus for named entity recognition. *Language Resources and Evaluation Journal, 42*(2), 173–182.

Ekbal, A., & Bandyopadhyay, S. (2009b). A conditional random field approach for named entity recognition in Bengali and Hindi. *Linguistic Issues in Language Technology (LiLT), 2*(1), 1–44.

Ekbal, A., Naskar, S., & Bandyopadhyay, S. (2007). Named Entity Recognition and Transliteration in Bengali. *Named Entities: Recognition, Classification and Use, Special Issue of Lingvisticae Investigationes Journal, 30*(1), 95–114.

Florian, R., Ittycheriah, A., Jing, H., & Zhang, T. (2003). Named entity recognition through classifier combination. In *Proceedings of the seventh conference on natural language learning at HLT-NAACL.*

Freund, Y., & Schapire, R. (1995). A decision-theoretic generalization of on-line learning and an application to boosting. In *Proceedings of the second European conference on computational learning theory, Taipei, Taiwan* (pp. 23–37).

Gali, K., Sharma, H., Vaidya, A., Shisthla, P., & Sharma, D. M. (2008). Aggregrating machine learning and rule-based heuristics for named entity recognition. In *Proceedings of the IJCNLP-08 workshop on NER for South and South East Asian languages* (pp. 25–32).

Geman, S., & Geman, D. (1984). Stochastic relaxation, gibbs distributions and the Bayesian restoration of images. *IEEE Transactions on Pattern Analysis and Machine Intelligence, 6*(6), 721–741.

Joachims, T. (1999). *Making large scale SVM learning practical*. Cambridge, MA, USA: MIT Press.

Kirkpatrick, S., Gelatt, C., & Vecchi, M. (1983). Optimization by simulated annealing. *Science, 220*, 671–680.

Kolen, J. F., & Pollack, J. B. (1991). Back propagation is sensitive to initial conditions. *Advances in Neural Information Processing Systems*, 860–867.

Lafferty, J. D., McCallum, A., & Pereira, F. C. N. (2001). Conditional random fields: Probabilistic models for segmenting and labeling sequence data. *ICML*, 282–289.

Li, W., & McCallum, A. (2004). Rapid development of Hindi named entity recognition using conditional random fields and feature induction. *ACM Transactions on Asian Languages Information Processing, 2*(3), 290–294.

Metropolis, N., Rosenbluth, A. W., Rosenbloth, M. N., Teller, A. H., & Teller, E. (1953). Equation of state calculation by fast computing machines. *Journal of Chemical Physics, 21*.

Patel, A., Ramakrishnan, G., & Bhattacharya, P. (2009). Relational learning assisted construction of rule base for indian language NER. In *Proceedings of ICON 2009: 7th international conference on natural language processing, India.*

Sha, F., & Pereira, F. (2003). Shallow parsing with conditional random fields. In *Proceedings of NAACL '03, Canada* (pp. 134–141).

Shishtla, P. M., Pingali, P., & Varma, V. (2008). A character n-gram based approach for improved recall in Indian language NER. In *Proceedings of the IJCNLP-08 workshop on NER for South and South East Asian languages* (pp. 101–108).

Srikanth, P., & Murthy, K. N. (2008). Named entity recognition for Telugu. In *Proceedings of the IJCNLP-08 workshop on NER for South and South East Asian languages* (pp. 41–50).

Suman, B., & Kumar, P. (2006). A survey of simulated annealing as a tool for single and multiobjective optimization. *Journal of the Operational Research Society, 57*(10), 1143–1160.

Taira, H., & Haruno, M. (1999). Feature selection in SVM text categorization. In *Proceedings of AAAI-99.*

E.F. Tjong Kim Sang, F. De Meulder, Introduction to the Conll-2003 shared task: Language independent named entity recognition. In *Proceedings of the seventh conference on natural language learning at HLT-NAACL 2003* (pp. 142–147).

Vapnik, V. N. (1995). *The nature of statistical learning theory*. New York, NY, USA: Springer-Verlag New York, Inc.

Wolpert, D. (1992). Stacked generalization. *Neural Networks, 5*, 241–259.