Multiobjective Optimization and Unsupervised Lexical Acquisition for Information Extraction

A Thesis
Presented to
The Academic Faculty

by

Govind

In Partial Fulfillment
of the Requirements for the Degree of
Master of Technology
in
Mathematics and Computing

Indian Institute of Technology Patna
May 2014

Acknowledgement

I would like to express my sincere gratitude to my thesis advisors Dr. Asif Ekbal and Dr. Chris Biemann for their excellent guidance and advice. Their guidance helped me in all the time of research and writing of this thesis.

I would like to thank German academic exchange service(DAAD) for providing me scholarship to conduct my thesis work at Technische Universität Darmstadt, Germany.

My sincere thanks also go to the director of IIT Patna, Prof. Anil K. Bhowmick and all faculty members of Department of mathematics and Department of computer science and engineering. I thank all members of language technology(LT) group at TU Darmstadt for their kind support.

Last but not the least, I thank my family: my parents, my loving sisters who has always supported, loved and believed in me.

Govind

Certificate

This is to certify that the thesis entitled "Multiobjective Optimization and Unsupervised Lexical Acquisition for Information Extraction", submitted by Govind to Indian Institute of Technology Patna, is a record of bonafide research work under our joint supervision and we consider it worthy of consideration for the degree of Master of Technology of this Institute. This work or a part has not been submitted to any university/institution for the award of degree/diploma. The thesis is free from plagiarized material.

_____

Dr. Asif Ekbal

_____

Dr. Chris Biemann

Date: _____

Declaration

I certify that

a The work contained in this thesis is original and has been done by myself under the general supervision of my supervisors.

b The work has not been submitted to any other Institute for degree or diploma.

c I have followed the Institute norms and guidelines and abide by the regulation as given in the Ethical Code of Conduct of the Institute.

d Whenever I have used materials (data, theory and text) from other sources, I have given due credit to them by citing them in the text of the thesis and giving their details in the reference section.

e The thesis document has been thoroughly checked to exclude plagiarism.

Signature of Student
Roll No:1211MC03

Contents

List of Figures

List of Tables

Abstract

For many natural languages and domains, the number of training examples are often limited due to the costs associated with procuring, storing, and preparing the training examples. Unsupervised part of speech (PoS) tagging has been proved to be useful low–cost veritable alternative when PoS training data is none or very little and can be used to support various supervised task for resource poor languages. As it is not a priori clear that features from which of unsupervised lexical acquisition techniques are more useful for a particular task or language, so feature selection is desired. Optimization techniques are widely used in areas such as economics, engineering and have shown promise in human language technology (HLT). We treated feature selection problem as a multi-objective optimization problem. Appropriate feature combination can be computed with multi-objective optimization (MOO) rather than selecting heuristically as it becomes almost impossible when feature set size increases. We come up with a novel approach based on multi-objective optimization (MOO) and Machine Learning, and the approach is planned to solve the problems of a well-known information extraction problem, namely Named Entity Recognition and Classification (NERC) in multiple languages (namely Hindi, Bengali, German). Specifically, we explored various possibilities to bring together two key concepts viz. "Unsupervised Lexical Acquisition" and "Feature Selection" for the effective and efficient solution to the aforementioned problem. When experiments are performed on Hindi NERC dataset we got following $F_{\beta=1}$ measure values i.e. 63. 22 without using any unsupervised lexical acquisition features, 64. 54 after incorporating unsupervised PoS tags as a feature and 69. 68 after incorporating features from both unsupervised lexical acquisition techniques (i.e. Distributional thesaurus (DT) and unsupervised PoS tagging). In case of German dataset $F_{\beta=1}$ measure values are 71. 35 without any lexical acquisition features, 73. 31 after incorporating DT features and 78. 18 after simultaneously using both unsupervised PoS tag and DT features.

Chapter 1

Introduction

Penetration of Internet is increasing and electronic devices are becoming intrinsic part of our life. We are generating enormous amount of unstructured data with size beyond the ability of commonly used software tools to capture, manage, and process the data within a acceptable elapsed time. Machine learning is useful tool for prediction, based on known properties learned from the training data and discovering unknown valuable patterns.

Research in the area of machine learning is helping us to design more flexible and intuitive digital systems employing sophisticated human–computer interfaces, where the machines can understand and "guess" the intended action without the user having to input complex instructions. The research in the area of Natural Language Processing (NLP), Machine Learning (ML) and more broadly Artificial Intelligence (AI) is moving us closer to Alan Turing's realization of a machine that can answer questions, as if it was a human being.

Language technology specifically for western languages such as English, German, etc. is constantly improving. An enabling factor to do so, as availability of language specific resources, which are at hand in abundance for the various NLP related tasks for these languages.

1.1 Natural language processing

Natural language processing (NLP) is a branch of linguistic and computer science focused on developing systems that allow computers to interact with people using everyday human (i.e. natural) languages. NLP has several

mutual goals and challenges with the area of human-computer interaction. Many of the challenges in NLP involve understanding natural language, i. e., enduing computers to infer meaning from natural language input. Natural language generation is also a challenge in NLP that involves generating computer's response in natural language.

There are various applications that utilizes methods from natural language processing. In fact, any application that employs text is a candidate for NLP. The most frequent applications utilizing NLP include the following: Information retrieval and extraction, Machine translation, Question answering, Summarization, and Dialogue systems and many more.

Problem of processing natural language like human beings is an AI-complete problem. Which means, solving the central artificial intelligence problem "making computers as intelligent as people", or strong AI. As natural language understanding techniques will improve, ability of computers to learn from information continuously and applying it in taking decisions in real world will also improve. A computer endued with better natural language understanding and generation techniques will become more and more capable of receiving and giving instructions. In our work we focus on improving the accuracy of NLP a application namely information extraction for several Indian languages and German.

1.2 Research Objectives and Hypothesis

Named entity recognition and classification (NERC) is also known by entity chunking, entity identification and entity extraction. The objective of NERC is to find and assign tokens in unstructured text to pre-defined classes such as the names of organizations, persons' names, locations' names, miscellaneous names which represents date-times, quantities, monetary expression etc. and "none-of-the-above". We hypothesized that we can improve upon the results of the Named entity recognition and classification (NERC) system for various Indian languages (namely Hindi, Bengali) and German by incorporating the techniques of unsupervised lexical acquisition. We try to explore how NERC task can benefit from lexical expansion of text using a distributional thesaurus and unsupervised PoS tagging . Feature selection technique will be proposed based on the concept of multi-objective optimization (MOO)[10] and various possibilities of incorporating features from unsupervised lexical acquisition resource will be investigated.

Hypotheses

- Unsupervised lexical acquisition techniques like Lexical expansion and unsupervised PoS tagging can be useful to tackle many of the critical issues related to NERC in resource-poor languages.
- MOO based feature selection, and parameter selection methods could be useful to handle the NERC problems in resource-poor languages.
- The combination of unsupervised lexical acquisition and multi-objective optimization can improve upon existing system.

Our main focus is to develop a framework integrating multi-objective optimization (MOO) and unsupervised lexical acquisition methods and study the feasibility of the proposed approach for solving the information extraction problems, particularly NERC in multiple languages (more specifically Hindi, Bengali, German). Various specific modules such as feature selection and parameter optimization using MOO are to be developed.

This thesis is structured as follows: in Chapter 2 we have discussed the related work and literature survey that is relevant to our work. We explain research methods and techniques employed in this work in Chapter 3. Feature selection using MOO and incorporation of features from distributional thesaurus and unsupervised PoS tag is discussed there in detail. In Chapter 4 experimental results are reported and discussed. In this chapter we discuss various details and specifications of experiments, results of various experiments are analysed and discussed in detail. Chapter 5 concludes the present work and outline the further possibilities of improvement.

Chapter 2

Literature and Research Review

In this chapter we review literature on related work to our project. Our main focus of this work is evolutionary optimization in conjunction with machine learning [15, 17] and role of lexical expansion [23] and unsupervised PoS tagging [6]. The works reported in the papers [15, 17] outline the use of a combination of individual classifiers and choice of appropriate features. In this chapter, mathematical optimization and it's application in Human language technology is discussed. Sequence tagging and its importance in other NLP tasks is explained. NERC and PoS tagging are explained as special cases of NLP sequence tagging.

2.1 Optimization

Optimization refers to finding best among all feasible solutions or in other words finding one or more feasible solution which correspond to extreme value of objectives. Optimal solutions are highly significant in taking crucial and critical decisions in various disciplines like engineering, economics, operational research etc. Optimization problems can be continuous or discrete. Definition of a standard continuous optimization problem is given in Equation 2.1. Various computational techniques has been devised to solve optimization problem be computers. Among these techniques are finitely terminating algorithms and convergent iterative methods, heuristics algorithms that can provide approximate solutions to some optimization problems and

genetic algorithms which also approximate the optimal solution.

$$\begin{aligned} \underset{x}{\text{minimize}} \quad & f(x) \\ \text{subject to} \quad & g_i(x) \leq 0,\ i = 1, \ldots, p(\text{inequality constraints}) \\ & h_j(x) = 0,\ j = 1, \ldots, q(\text{equality constraints}) \\ & \text{where } f(x) : R^n \to R \text{ and } p, q \text{ are some integers} \end{aligned} \tag{2.1}$$

Here in equation 2.1 $g_i(x)$ are set of inequality constraints and $f_j(x)$ are set of equality constraints.

2.2 NLP Sequence tagging

In natural language processing (NLP), sequence tagging concerns the prediction of labels for a sequence of observed outputs. Typical sequence tagging problem can be defined as:

$$\begin{aligned} &\text{Given a word sequence} && w_1, w_2, \ldots, w_n, \\ &\text{Determine the corresponding tag sequence} && t_1, t_2, \ldots, t_n. \end{aligned} \tag{2.2}$$

Part of speech tagging and Named entity recognition are considered as two of the highly important tasks in Natural language processing research domain. We are also targeting these two tasks for Hindi, Bengali, German languages. More about PoS tagging and Named Entity recognition is explained in further subsections.

2.2.1 Part of speech tagging

Part of speech (PoS) tagging is the process of classifying words into their parts of speech and labeling them accordingly. Parts of speech are also known as word classes or lexical categories or syntactic categories. Assigning syntactic categories (i.e. PoS tags) to words is a crucial pre-processing step for many higher level NLP tasks. Machine translation, parsing, anaphora resolution, named entity recognition and information extraction (NERC), and many other NLP tasks utilize PoS tags. Hence PoS tagging is one of the fundamental task in NLP that affect the accuracy of many higher level NLP tasks which indicates that it is very important to have PoS techniques that produce highly accurate tagged data. Set of all available PoS tags for part of speech tagging is called tagset.

In schools we commonly learn that there are 9 parts of speech in English: article, adjective, preposition, noun, verb, pronoun, adverb, interjection, and conjunction. But there exist many more fine grained classes of part–of–speech as aforementioned categories can be further divided into subcategories. Verbs can be further categorised in transitive and intransitive and nouns can be distinguished as plural, possessive, and singular forms. Also syntactic categories may differ from one language to other. Words are also categorised on basis of their "case", grammatical gender, and in name of days and name of months if we categorise to more fine degree. Degree of categorisation subjects to the type of tagging system and it changes from one tagging system to another.

Part-of-speech tagging is not a easy task as some words can represent more than one part of speech at different times of their use in corpus, and because some parts of speech are inherently complex. Natural languages as opposed to artificial languages are highly ambiguous in nature. For example, the word "boats", which is usually conceived as just a plural noun, can also act as a verb:

"He boats the timber down the lake. "

In this sentence, correct grammatical tagging will reflect "boats" is here used as a verb, not as the more common plural noun. Grammatical context is one way to determine the correct tag in such situation but also; semantic analysis can also be used to infer the actual part-of-speech tag.

The identification of the parts of speech tags was originally performed manually, in course of time the process has been increasingly automated with the help of computational linguistics methods. These computational linguistics methods are further categorised in two types, one is supervised and second is unsupervised methods. In supervised PoS tagging classification models like Hidden Markov Models, Conditional Random Fields (CRF), Maximum Entropy (ME) and decision trees are used and any part of speech tags come from a so-called predefined tag set.

In computational methods for part-of-speech tagging, typically tagset of size 50 to 150 is employed to distinguish separate parts of speech for English. For example, NN denotes singular common nouns, NNS denotes plural common nouns, NP denotes singular proper nouns. The Brown Corpus for English language distinguishes 87 simple tags and allows the formation of more compound tags. For German language, the Stuttgart-Tübingen-Tagset

(STTS)[1] is often used which consists of a set of 54 PoS tags. In unsupervised learning methods, the tag set is not fixed in advance, but it results from a clustering process. A sentence in German language with PoS tagged words is shown below.

Es|PPER werde|VAFIN wieder|ADV eine|ART Dividende|NN geben|VVINF, |$, die|PRELS zuletzt|ADV ausfiel|VVFIN. |$.

In the sentence above, behind each word or punctuation mark is the PoS tag after a vertical bar. To tag a word with its correct category information from context play a vital role. In supervised learning, a PoS tagging model is built by observing various pattern on manually tagged training data. These pattern is often extracted with the help of various probabilistic models like Hidden Markov Models, Conditional random field etc. Unsupervised learning methods do not require prior training but require a corpus of significant size to induce word categories accurately. General methodology for unsupervised PoS induction can be summarized in two steps. First is, collection of global context vectors of words to be tagged (i. e target words) by counting how often feature words (i.e. words used to describe syntactic contexts) appear in neighbouring positions and and second step is applying a clustering algorithm on these context vectors to obtain target word classes.

2.2.2 Named Entity Recognition and Classification

Named Entity Recognition and Classification (NERC)[24] is a subtask of information extraction that has great importance in many Natural language processing (NLP)application areas such as Machine Translation [2], Automatic Question Answering [26], Automatic Summarization [25], Information Retrieval [22] etc. NERC is also known by entity chunking, entity identification and entity extraction. The objective of NERC is to find and assign tokens in unstructured text to pre-defined classes such as the names of organizations, persons' names, locations' names, miscellaneous names which represents date-times, quantities, monetary expression etc. and "none-of-the-above". These categories are application domain dependent and hence subjected to change like for biomedical domain these can be drugs' names,

[1]http://www. ims. uni-stuttgart. de/forschung/ressourcen/lexika/TagSets/stts-table. html

product's names or genes etc.

$$[John]_{Person} \text{ worked for } [Acme\ Corp.]_{Organization} \text{ in } [Zigcity]_{Location}.$$

In this sentence there are three named entities (NE), first is John that is categorised as "person name", second is Acme Corp. which categorised as "organization name" and third is Zigcity which is a "location name". Other words in this sentence come under the category of "none-of-the-above".

Basically three kind of approaches exist to solve NERC task and these categories are rule-based approach, machine-learning based approach and combination of both i.e. hybrid approach. In rule-based approach handcrafted rules are devised to extract name entities. Rule-based NERC system consists of dictionaries along with set of patterns using various kind of features like grammatical, orthographic and syntactic features. Advantage of rule based system is their ability to detect complex named entities that can't be easily detected by a machine learning based NERC system. On the other hand rule based systems are highly domain and language specific and lack robustness. Furthermore, maintaining database of rules involves high costs and rules are not domain adaptive in nature. Rule based NERC systems don't inevitably adapt well to new languages and domains.

There has been a good number of research works in named entity recognition and classification area but it is limited to English, majority of European languages and a few of the Asian languages like Japanese, Chinese and Korean.

India is a multilingual country with great cultural diversity and a very high linguistic diversity index (LDI). According to the Constitution of India [1] there are 22 official languages spoken in India with many more dialects. This fact emphasizes the variety, and the diversity of natural languages and linguistics that exist in India. Research in NLP relating to the Indian languages is still evolving and poses some interesting problems. Some of the problems outlined previously in [15] with reference to a specific NERC task are listed below :

- Absence of capitalization which is a useful cue for identifying proper names in English.
- Many proper names could also appear in the dictionary with other word classes.

- Indian languages are free word order languages. So, the position of the word in a sentence is not much helpful for NE identification.
- Non–availability of various NLP resources and preprocessing technology for Indian languages such as, annotated corpus, name dictionaries, morphological analyzers, part of speech (PoS) taggers, etc.

In our work, we investigated some novel methods based on machine learning and multi-objective optimization (MOO) for solving the problems of NERC for several resource poor languages.

2.3 Unsupervised Lexical Acquisition

One of the major problems in applying machine learning algorithms for solving information extraction problems is the availability of large annotated corpus.

We now summarize two methods for unsupervised lexical acquisition and their use as features for NERC. Namely, we explored possibilities arising from the use of unsupervised PoS induction [6] and lexical expansion [23]. Unsupervised PoS induction [6] is a technique that induces lexical-syntactic categories through the statistical analysis of large, raw text corpora. As compared to linguistically motivated PoS-tags, the categories are usually more fine-grained, i.e. the linguistic class of nouns is split into several induced classes that also carry semantic information, such as days of the week, professions, mass nouns etc. As shown in [5], using these induced categories as features results in improved accuracies for a variety of NLP tasks, including NERC. Since the induction of PoS is entirely language independent and sensitive to domain vocabulary as shown in [6].

Also, as shown in [6], it might be advantageous to combine several approaches to PoS induction, e. g. not only using the graph-based method of [6] but also the word clustering approach described in [8]. While unsupervised PoS tagging primarily targets syntactic categories, the second method we want to utilize is of a more semantic nature. Lexical expansion [23] is also an unsupervised technique that needs a large corpus for the induction, and is based on the computation of a distributional thesaurus [21] (DT).

Lexical expansion has already proven useful in semantic text similarity evaluations [3], which is a task related to matching sense definitions to contexts. The author of the paper [23] used distributional thesaurus, or DT

[21], expanded the lexical representations of the context and sense definition with additional terms. They showed significant performance gain for word sense disambiguation (WSD). The same technique can be used for other text processing application including NERC. In machine learning, one of the major challenges is the handling of unseen words, and this lexical expansion technique can be useful for classifying the unseen instances.

As expected, using both lexical acquisitions as features help in the NERC task, since both features perform a generalization of the training and test data-unsupervised PoS tags as categories, and lexical expansions as alternative vocabulary. This is especially apparent for situations where training data is scarce, and other available features from preprocessing steps are little, i.e. for Indian languages.

Chapter 3

Research Methods

Multi-objective optimization is a technique for solving optimization problems with more than one objectives. Conditional random field (CRF) is classification model. Unsupervised part of speech (PoS) tagging is a method of inducing PoS categories without any training data. Distributional thesaurus (DT) groups the words together according to some predefined distribution similarity index. In this chapter, multi-objective optimization (MOO), Conditional random field (CRF), Unsupervised part of speech (PoS) tagging and distributional thesaurus (DT) are discussed in detail and their significance in out purposed is also highlighted.

3.1 Multi-objective optimization-MOO

In real life, we encounter majority of optimization problems which have more than one objectives to be optimized at the same time. when a optimization problem modeling a system involves more than one objective function, the task of finding one or more optimal solution is called multi-objective optimization (MOO). Traditional optimization techniques are not compatible and suitable for multi-objective optimization problems (MOOP) as these techniques were developed by taking in view one objective function only in contrast to MOOP where more than one objective are present. Multi-objective optimization technique are developed to deal with such situations.

A general multi-objective optimization problem is shown in Equation 3.1.

$$\begin{aligned} &\text{Minimize/Maximize} && f_m(x), m = 1, 2 \ldots, M; \\ &\text{subject to} && g_j(x) \geq 0,\ j = 1, \ldots, J; (\text{ inequality constraints}) \\ & && h_k(x) = 0,\ k = 1, \ldots, K; (\text{ equality constraints}) \\ & && x_i^{(L)} \leq x_i \leq x_i^{(U)}, i = 1, \ldots, n. \end{aligned} \tag{3.1}$$

A solution x is a vector $(x_1, x_2, \ldots, x_n)^T$. of size n, where $x_i; i = 1, \ldots, n$ denotes decision variables. The last set of constraints are called variable bounds, restricting each decision variable x_i to take a value within a lower $x_i^{(L)}$ and upper $x_i^{(U)}$ bound. Optimization problem specified above has J number of inequality constraints and K number of equality constraints. The terms $g_i(x)$ and $h_k(x)$ are called constraint functions. *Infeasible solution* is a solution x that does not satisfy all of the $(J + K)$ number of constraints and all of the $2N$ variable bounds. On the contrary, *feasible solution* is a solution y which satisfies all variable bounds and constraints. The set of all possible feasible solutions is called the feasible region and denoted by S.

Optimization problem in Equation 3.1 differs from optimization problem in Equation 2.1 in the number of objective functions to be optimized simultaneously. In optimization problem 3.1 there may be more than one objective function i.e. $f_m(x)$ are possible. Another important difference between single-objective and multi-objective optimization problems is that in multi-objective optimization the objective functions form a multi-dimensional space, in addition to the usual decision variable space that is common in both type of optimization. Here mapping take place between an n-dimensional solution vector and M-dimensional objective vector.

3.1.1 MOO Algorithms

The multiobjective optimization problem (MOOP) is formally stated in equation 3.1, In other terms it can be stated as follows find the vectors x of decision variables that simultaneously optimize the M objective values $f_1(x), f_2(x), ..., f_M(x)$, while satisfying the constraints, if any. An important concept of MOO is that of domination. In the context of a maximization problem, a solution x_i is said to dominate x_j if $\forall k \in 1, 2, \ldots, M, f_k(x_i) \geq f_k(x_j)$ and $\exists k \in 1, 2, \ldots, M$, such that $f_k(x_i) > f_k(x_j)$. Among a set of solutions P, the non-dominated set of solutions P are those that are not dominated by any member of the set

P. The non-dominated set of the entire search space S is the globally Pareto optimal set. In general, a MOO algorithm usually admits a set of solutions that are not dominated by any solution encountered by it. Pareto optimal front for a two-objective optimization problem having a set of non-dominated solutions is shown in Fig. 4.2.

MOO performance measures: In MOO, basically two functionalities must be achieved regarding the obtained solution set [10]. It should converge as close to the true Pareto optimal front as possible, and it should maintain as diverse a solution set as possible. The first condition clearly ensures that the obtained solutions are near optimal, and the second condition ensures that solutions with a wide range of trade-off objectives are obtained. Clearly, these two tasks cannot be measured with one performance measure adequately. A number of performance measures for MOO algorithm have been suggested in the past. Here, we mainly use one such performance measure. The measure named Minimal Spacing reflects the uniformity of the solutions over the non-dominated front. Smaller values of Minimal Spacing for a particular MOO algorithm indicate better performance. A large number of approaches exist in the literature to solve MOOPs [10]. These are aggregating, population based non-Pareto and Pareto based techniques. In case of aggregating techniques, the different objectives are generally combined into one using weighting or goal based methods. One of the techniques in the population based non-Pareto approach is Vector evaluated genetic algorithm (VEGA). Pareto based approaches include Multiple objective GA (MOGA), non-dominated sorting GA (NSGA), and niched Pareto GA.

Non-dominated Sorting genetic algorithm-II (NSGA-II) We are using NSGA–II [11] for the purpose of multi-objective optimization in our experiments. The decision of choosing NSGA–II is taken on the basis of several advantages it provides. These are low computation complexity for non-dominated sorting, high elitism and no need to specify any parameter manually for ensuing diversity in population. The overall complexity of the algorithm is $O(MN^2)$ (where is M the number of objectives and N is the population size). The pseuodcode of NSGA-II is given in Algo. 1.

Genetic algorithms (GAs) are known to be more effective than classical methods such as weighted metrics, goal programming [10], for solving multiobjective problems primarily because of their population-based nature. NSGA-II [11] is widely used in this regard, where initially a random parent population P_0 is created and the population is sorted based on the partial order defined by the non-domination relation. This results in a sequence of

Algorithm 1 Non-dominated Sorting Genetic algorithm-II (NSGA-II)

$R_t = P_t \cup Q_t$
$F = fast - non - dominated - sort(R_t)$
$P_{t+1} = \phi$ and $i = 1$
while $|P_{t+1}| + |F_i| \leq N$ **do**
    $crowding - distance - assignment(F_i)$
    $P_{t+1} = P_{t+1} \cup F_i$
    $i = i + 1$
end while
$Sort(F_i, <_n)$
$P_{t+1} = P_{t+1} \cup F_i[1 : (N - |P_{t+1}|)]$
$Q_{t+1} = make - new - pop(P_{t+1})$
$t = t + 1$

non-dominated fronts. Each solution of the population is assigned a fitness value that is equal to its non-domination level in the partial order.

Here, functions are minimized using the search capability of GA, i. e., the algorithm is trying to find those solutions that are non-dominated with respect to the minimization of the objective values. The authors have assumed the minimization of fitness. A child population Q_0 of size N is created from the parent population P_0 by using binary tournament selection, recombination, and mutation operators. According to this algorithm, in the t^{th} iteration, a combined population $R_t = P_t + Q_t$ is formed. The size of R_t is $2N$, as size of both P_t and Q_t is N. All the solutions of R_t are sorted according to non-domination. If the total number of solutions belonging to the best non-dominated set F_1 is smaller than N, then F_1 is totally included in $P_{(t+1)}$. The remaining members of the population $P_{(t+1)}$ are chosen from the subsequent non-dominated fronts in the order of their ranking. To choose exactly N solutions, the solutions of the last included front are sorted using the crowded comparison operator[11] and the best among them (i. e., those with lower crowding distance) are selected to fill in the available slots in $P_{(t+1)}$. The new population $P_{(t+1)}$is then used for selection, crossover, and mutation to create a population $Q_{(t+1)}$of size N.

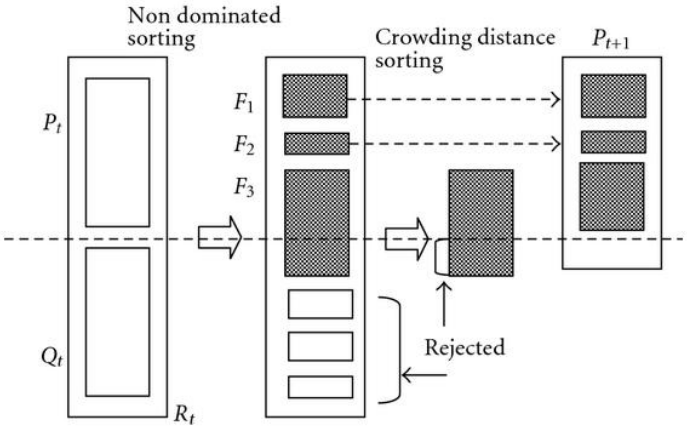

Figure 3.1: Non-dominated sorting algorithm–II Procedure, Figure adopted from [10]

3.2 Conditional Random field-CRF

Conditional random field (CRF) [20] is a statistical modelling method for building probabilistic models to segment and label sequence data. Advantages of conditional random fields over hidden Markov models (HMM) and stochastic grammars for sequence tagging tasks is the ability to relax strong independence assumptions made in HMM and stochastic grammar models. A fundamental limitation of maximum entropy Markov models (MEMMs) and other discriminative Markov models which are based on directed graphical models is that these models can be biased towards states with few successor states. Conditional random fields avoids this limitation also. Figure 3.2 depicts the graphical model of simple hidden Markov models, maximum entropy Markov models and conditional random fields, where an open circle indicates that the variable is not generated by the model.

A CRF has a single exponential model for the joint probability of the entire sequence of labels given the observation sequence and hence weights of different features at different states can be traded off against each other, while a MEMM uses per-state exponential models for the conditional probabilities of next states given the current state. A conditional random field is defined formally in definition 1, where X is a random variable over data sequences (i.e. word sequences) to be labelled, and Y is another random variable over

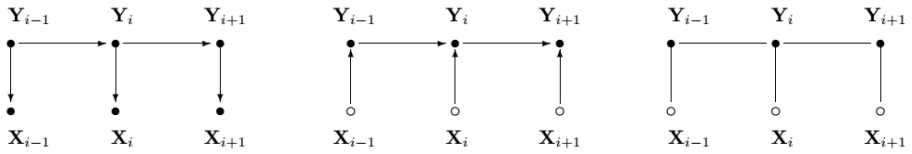

Figure 3.2: Graphical structures of simple HMMs (left), MEMMs (center), and the chain-structured case of CRFs (right) for sequences. Figure adopted from [20].

corresponding label (i.e. tag) sequences. Let Y be a finite label alphabet and all components Y_i of Y range over Y.

In a NLP sequence tagging problem, for example, X might range over language sentences and Y range over named-entity taggings of those sentences, with γ be the set of all possible named-entity tags.

Definition 1 *Let $G = (V, E)$ be a graph such that $Y = (Y_v)_{v \in V}$, so that Y is indexed by the vertices of G. Then (X, Y) is a conditional random field in case, when conditioned on X, the random variables Y_v obey the Markov property with respect to the graph: $p(Y_v|X, Y_w, w \neq v) = p(Y_v|X, Yw, w \sim v)$, where $w \sim v$ means that w and v are neighbours in G.*

CRFs have shown success in various sequence modeling tasks including noun phrase segmentation [46] and table extraction [47]. 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 sequences $= (s_1, s_2, ..., s_T)$ given an observation sequence $o = (o_1, o_2, ..., o_T)$ is calculated as:

$$P_v(s|o) = \frac{1}{Z_0} \exp \sum_{t=1}^{T} \sum_{k=1}^{K} \lambda_k * f_k(s_{t-1}, s_t, o, t)$$

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 $-\infty, ...+\infty$, but typically they are binary. To make all conditional probabilities sum up to 1, we must calculate the normalization factor,

$$Z_o = \sum_{s} \exp \sum_{t=1}^{T} \sum_{k=1}^{K} \lambda_k * f_k(s_{t-1}, s_t, o, t)$$

, 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:

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

where $(o^{(i)}, s^{(i)})$ 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 [27], a quasi-Newton method that is significantly more efficient, which results in only minor changes in accuracy due to changes in λ. When applying CRFs to the NERC 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 a open source C++ based CRF++[1] tool kit for building probabilistic models for sequence tagging tasks (i.e. NERC and PoS tagging). Training and test data both have to be in a particular format for CRF++ to work properly and correct interpretation. Training and test file consist of multiple tokens and a token consists of multiple but fixed number of columns. The definition of tokens is task-dependent. Here each token must be represented in one line, with the columns separated by spaces or tabular characters. Boundary between sentences is indicated by putting an empty line. In our training and testing data 1st column is 'word' itself, followed by a series of features related the word and the last column represents a true answer tag which is going to be trained by CRF.

CRF Feature Template: Each line in the template file denotes one template. In each template, special macro %x[row, col] will be used to specify a token in the input data. where row specifies the relative position from the current focusing token and col specifies the absolute position of the column. Sample template is shown in table 3.1.

[1]CRF++:Yet another CRF toolkit http://crfpp.googlecode.com/svn/trunk/doc/index.html

#Unigram	'U' letter for unigram template
U1:%x[-1, 2]	specifies second feature of previous word
U2:%x[0, 0]	current word itself
U4:%x[0, 1]	first feature of current word
U5:%x[0, 1]/%x[1, 1]/%x[2, 1]	Combination of features
# Bigram	'B' letter for bigram template
B	specifies combinations of previous output token and current word

Table 3.1: CRF++ sample feature templates

3.3 Unsupervised PoS tagging

Supervised methods available for PoS tagging task involve working from a pre-existing manually tagged corpus to learn tag probabilities, hence the reachability and application area of these methods are limited to resource-rich languages. Efficient PoS tagger for minority and resource poor domains can not be built by using supervised methods. As we know that assigning syntactic categories (i.e. PoS tags) to words is a crucial pre-processing step for many higher level NLP tasks. Machine translation, parsing, anaphora resolution, named entity recognition and information extraction (NERC), and many other NLP tasks involve use of PoS tags. This implies that if unavailability of efficient PoS tagger for a language affects the accuracy of various other NLP tasks for that language.

Unsupervised PoS tagging is the techniques that requires no pre-exiting manually tagged corpus to build a tagging model and hence highly suitable for resource poor languages. Idea of bootstrapping is employed for "unsupervised" tagging. Unsupervised tagging techniques use an untagged corpus for their training data and induce the tagset. These unsupervised techniques observes the patterns of words use in untagged corpus and extract different PoS categories from it. Word context play a very important role in inducing PoS categories. For example, words like p̈lay, r̈un, and ëatöccur in similar contexts, while words like ä, än, ẗheöccur in quite different ones. With sufficient iteration, similarity classes of words emerge that are remarkably similar to those human linguists would expect; and the differences themselves sometimes suggest valuable new insights.

Unsupervised PoS tagging results in slightly different categories as opposed to what is assumed by a linguistically motivated POS-tagger.

3.3.1 Unsupervised PoS system

There are a number of approaches to induce PoS tags (i.e. syntactic categories). Harris' distributional hypothesis states that words of similar parts of speech can be observed in the same syntactic contexts. Harris' distributional hypothesis is used by most of these approaches. Similarity of two words is measured by measuring to what level these two words appear in the same context. Function words form the syntactic skeleton of a language and almost exclusively contribute to the most frequent words in a corpus, and hence contexts in that sense are often restricted to the most frequent words. Feature words used to describe syntactic contexts. Target words are the words that are to be grouped into syntactic clusters. The general technique for inducing word syntactic categories information can be outlined as follows:

- Collect global context vectors of target words by counting how often feature words appear in neighbouring positions
- Apply a clustering algorithm on these vectors to obtain word classes.

Here we use Chinese Whispers (CW) algorithm for clustering purpose. It is a very basic and effective algorithm to partition the nodes of weighted, undirected graphs. Chinese Whispers clustering algorithm is inspired by the eponymous childrens game, in which children whisper words to each other. The goal of the game is to arrive at some funny derivative of the original message by passing it through several noisy channels, the CW algorithm aims at finding groups of nodes that broadcast the same message to their neighbors. The CW algorithm is sketched as follows:

Algorithm 2 Chinese Whispers Graph clustering CW (graphG (V, E))

```
for all v_i ∈ V do
    class(v_i) = i
end for
for it = 1 to noumber-of-iteration do
    for all v ∈ V, randomised order do
        class(v) = predominant class in neigh(v)
    end for
end for
return partition P induced by class labels
```

Input to this unsupervised PoS system is a substantial amount of unlabelled, tokenised monolingual text without any POS information. Process of building a Viterbi tagger for PoS induction is outlined below:

- Chinese Whispers is applied to distributional similarity data, which groups a subset of the most frequent 10,000 words of a corpus into several hundred clusters (tagset 1).
- Similarity scores on neighbouring co-occurrence profiles are used to obtain again several hundred clusters of medium- and low frequency words (tagset 2). The combination of both partitions yields sets of word forms belonging to the same induced syntactic category.
- To gain on text coverage, ambiguous high-frequency words that were discarded for tagset 1 are added to the lexicon.
- A Viterbi trigram tagger is trained with this lexicon and augmented with an affix classifier for unknown words.

3.4 Distributional Thesaurus

Distribution thesaurus groups the words together according to the distribution similarity index. A DT contains, for every sufficiently frequent word, the most similar words as computed over the similarity of contexts these words appear in, which implements the distributional hypothesis [19]. This automatically induced lexical resource is used in [23] for lexical expansion of text by virtually expanding every content word in the text with the list of most similar words from the DT. In this way, it was possible to reach significant improvements in a word sense disambiguation system that assigns word senses in context by comparing their dictionary definitions with the context.

In [23], the authors utilize the concept of a Distributional Thesaurus to lexically expand a given text for the Word Sense Disambiguation (WSD) task. Especially techniques based on matching sense definitions from dictionaries with contexts of ambiguous terms to assign the correct sense suffer from a type of "lexical gap problem". The lexical gap problem arises out of having insufficient amount of overlapping vocabulary between the sense description and the context of the ambiguous term. To "close" this "lexical gap" the authors in [23] describe the technique of producing lexical expansions due to

a distributional thesaurus. This distributional thesaurus is constructed by harvesting statistical information from a unannotated corpora [23]. What the lexical expansion does is populates the text with additional wordforms that may convey the same meaning by deploying the distributional thesaurus, thereby enriching the context, so that the lexical gap is alleviated. Here, we are employing the technique inspired from the above discussed work to address sparsity issues for NERC.

3.5 Unsupervised PoS tags and DT similar words as features

In [6], author investigates the utility of an unsupervised part-of-speech (PoS) system in a task oriented way. PoS labels are treated as features for different supervised NLP tasks like Word Sense Disambiguation, Named entity recognition and classification (NERC) and Chunking for a variety of language. It is observed that supervised tagging accuracy improves significantly from unsupervised tagging. Further, results reveal that chunking benefits more from supervised PoS as compared to unsupervised PoS tag and unsupervised PoS tagging behaves similarly to supervised PoS in Word Sense Disambiguation. Overall results suggest that unsupervised PoS tagging is a veritable low-cost alternative for resource poor languages and if none or very little PoS training data is available for the target domain. Unsupervised PoS tagging provides an additional word-level feature, which can be computed for any language and domain, and has proven useful in domain adaptation and in situations of reduced training data. In our work, we are employing unsupervised PoS tags as one of the important language independent feature which can benefit NERC task for various Indian languages and German.

Additionally, another source of vital features for NERC task can be distributional thesaurus. In [4] author investigates the use of Distributional Similarity for Lexical Expansion in Knowledge-based Word Sense Disambiguation. We also investigates the use of features based on distributional similarity. Distributional thesaurus is used to fetch most similar words. We are incorporating three most similar words to a particular token as three features in training and test datasets. Figure 3.3 shows the three most similar words for tokens in a Hindi language sentence. Each token in training and test dataset is incorporated with three words which are most distributional

similar to itself.

Tokens from a Hindi sentence	Similar words from Distributional Thesaurus		
इनकी	उसकी	इसकी	इसकी
सहायता	मदद	राहत	सहयोग
से	को	उससे	के
ही	भी	वह	वे
कुशल	सफल	भारतीय	अच्छे
गुप्तचरों	अधिकारी	जासूसों	एजेंटों
को	उन्हें	उसे	द्वारा
नियुक्त	तैनात	पद	बनने
कर	करने	किया	करना
राम	कृष्ण	नारायण	प्रसाद
लंका	कोलकाता	तो	घर
में	मे	मंे	मेंे
सुरक्षित	खतरनाक	उचित	बेहतर
स्थान	मिनट	ओवर	नंबर
पर	से	को	तक
बन्दिनी	ND	ND	ND
सीता	संतोष	रमेश	मुकेश
को	उन्हें	उसे	द्वारा
खोज	तलाश	ढूंढ	निकाल
सके	सकते	सकती	सकता

Figure 3.3: Lexical expansion of tokens in Hindi language

3.6 Feature selection

In [12], the problem of feature selection for only one classifier, namely maximum entropy (ME) is formulated as a multiobjective optimization (MOO) problem. A MOO algorithm attempts to optimize more than one classification quality measures, conflicting in nature, simultaneously. For example, in case of NERC *recall* and *precision* are conflicting in nature. The increase of one's value may degrade the other's value.

In [14], some simulated annealing based classifier ensemble techniques were developed and those were applied for solving the Part-of-Speech tagging

problem for different Indian languages. Since, in PoS tagging, each token is assigned a label from a set of predefined PoS tags/classes, and in NE extraction the problem is to distinguish proper names from others and to classify them into some predefined set of NE classes, the methodologies are likely to be transferrable.

As it was shown in [6] that different NE classes, as well as different classes of proteins, diseases and genes are distinguished in general and medical, respectively. The choice of appropriate features seems to be highly appropriate.

3.7 Data collection and Preparation

As we have mentioned the research in NLP in Indian languages is still at the nascent stages because of its resource-constrained nature. Corpus, annotated corpus, name dictionaries, PoS tagger, NE tagger etc. are not readily available in the web. In this research we plan to work on two Indian languages, namely Bengali and Hindi. Hindi is the national language in India and Bengali ranks second in terms of native speakers. In [13], the authors develop a web crawler that searches and retrieves web pages from the archive of a popular Bengali newspaper available in the web. The web crawler converts the article data available online, into XML format for the corpus. Subsequent stages of this data collection then involve removal of noise and addition of HTML tags. The corpus is cleaned by removing the tables, images, advertisements etc. and made ready for text-processing applications. The authors also take into account the different encodings used by the newspaper and have developed the code for conversion of this encoding to Indian Standard Script Code for Information Interchange (ISCII) which was then subsequently converted to the de–facto standard for document text encoding *viz.* UTF–8. The size of the corpus is about 34 million wordforms with a collection of news data of 5 years, and can be increased dynamically.

Named entity (NE) annotated corpus [12] for developing supervised NERC system is annotated 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. We will also use the IJCNLP-08 NERC on South and South East Asian Lan-

guages (NERSSEAL)[2] Shared Task data of Bengali and Hindi. The approach used in the paper [13] can be also used to develop similar resources from the archives of other Indian language dailies available on the web such as Hindi, Tamil, Oriya etc.

3.7.1 Named Entity Features

In machine learning, a feature is an individual measurable property of a token under observation. Success of a pattern classification task is highly dependent on features used in algorithm. Features' values are often numeric, but in some machine learning problems, features such as strings and graphs are also used. The set of features of a token is often grouped in a feature array or vector so that it can be treated mathematically for classification purposes.

Following features are used for building the various classifiers based on conditional random field (CRF) classification model. Majority of the following features are not language dependent and can be extracted for almost all the languages with ease. By language independent features, we mean that these features don't require any language and domain specific resources or rules for their computation.

1 **Context words:** Context words of a token play a significant role in determining its true named entity tag. This is based on the observation that surrounding words carry effective information for the identification of NEs. Context words of current token are the words preceding and succeeding it.

2 **Word suffix and prefix:** Word suffixes and prefixes of fixed length are proved to very effective to detect NEs and work well for NERC task on highly inflective Indian languages. Word suffixes are the fixed number of letters stripped from rightmost position in word and prefixes are the letters stripped from leftmost of the word. If we are extracting suffixes and prefixes of size n and word length is less than n then we feature value is not defined and assigned ND. The main motive to include these features is the observation that named entities (NEs) has some common suffixes and/or prefixes values.

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

3 **First word**: First word is a binary valued feature which takes value 1 when current word is the first token of the sentence and 0 for the other case. This feature is considered because of the observation that the first word of the sentence is highly likely to be a NE.

4 **Length of the word**: Length of a word is a binary valued feature which takes value 1 when number of characters in a token is greater than a predetermined threshold value (here, set to 5). This binary valued feature is considered with the observation that very short words are unlikely to be NEs.

5 **Infrequent word:** This feature is also a binary valued features which checks whether frequency of current word in training set crosses a threshold value. A list of frequent word is computed from training set by fixing a appropriate threshold value. Threshold value depend on the size of training set. For the present work, we fix the threshold value to 15 for Hindi and 10 for Bengali. This feature is included based on the observation that frequently occurring words are more probable to be NEs.

6 **Last word of sentence:** This binary valued feature checks whether the word is the last word of a sentence or not and turn on/off accordingly. Generally, in Indian languages, verbs appear in the rightmost position of a sentence. Majority of Indian languages follow subject-object-verb (SOV) structure. This feature is considered with the observation that it distinguishes NEs from the verbs.

7 **Capitalization:** This binary valued feature checks whether the word starts with a capital letter or not and take value accordingly. This feature is found to be an useful feature for German and English. Indian languages do not have any kind of capitalization cues, hence this feature is not used in such case.

8 **Part-of-speech (POS) information:** PoS tags provide information about the syntactic categories of tokens which is observed to be useful to identify the NEs.

9 **Chunk information:** Some of the possible values of chunk information feature are I-NP/B-NP (which implies that token is the ending/beginning of a noun phrase), I-VP/B-VP (which means token is the

Tag	Description
PER	Person Name
LOC	Location Name
ORG	Organization Name
MISC	Miscellaneous Name
O	Other the Named Entity (NE)

Table 3.2: Named entities tagset

ending/beginning of a verb phrase) etc. Chunk information is quite useful for NE identification task. Here, this feature is used only for German and English due to the non-availability of chunker for Indian languages.

10 **Digit features:** There are several digit features are defined based upon the presence and/or the number of digits and/or symbols in a token. These features are digitPercentage (token contains digit and percentage), digitComma (token contains digit and comma), 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).

11 **Dynamic NE information:** Dynamic NE information is the output tag (s) of the previous token (s). The value of dynamic NE feature is computed dynamically at run time.

3.8 Analysis and Evaluation of Data

As mentioned in previous sections, this research work will be aimed at improving upon the state–of–the–art systems in the NERC task for various Indian languages and biomedical texts. The feature selection and classifier ensembles will be evaluated on the datasets as mentioned above. We will use well-known algorithms such as Maximum Entropy (ME) [7], Conditional Random Field (CRF) [20] and Support Vector Machine (SVM)[28] as the supervised classifiers. The classifiers are trained mostly with the language independent features that can be generated automatically from the available training data. Initially we will use the features described in [17] for Indian

languages. Biomedical texts require more diverse set of features. Initially the set of features used in [16] will be used for the biomedical NERC. These setups will form the baselines. Thereafter, we shall investigate additional features from the lexical acquisition methods. All these will be performed in such a way that the proposed method can be transferred to other languages and domains easily. We also target to identify and implement some other domain-independent as well as domain specific features. Our main goal is to develop a system that can be easily transferred to any other resource-poor languages. Finally, MOO based feature selection technique will be used to determine the most relevant set of features for all the domains.

We develop methods based on the concept of multiobjective optimization (MOO). MOO, typically 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 [10]. 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 multiobjective evolutionary algorithms (MOEAs) have been suggested [9, 29]. The prime motivation for using evolutionary algorithms (EAs) to solve multiobjective problems is their population-based nature and ability to find multiple optima simultaneously. A simple EA can be easily extended to maintain a diverse set of solutions.

For biomedical texts, initially the system will be evaluated with each dataset. Thereafter, experiments will be carried out by combining different datasets. This will be an interesting experiment because the system that performs well for a domain often fails to to show reasonable performance in other domains. We want a system that will show good performance across different domains, i.e. the system is expected to perform well when datasets with different annotation schemes are combined. In each case the system will be compared with the state-of-the-art systems.

To ensure that we do not overfit to the datasets at hand, we will split all available annotated data into training, development and test sets. While optimization will be applied to the development set, we will test the final model on the held-out test data. Also we will produce learning curves with reduced training set to study the effects of new feature to the situations where training data is scarce.

The system will be evaluated in terms of the standard metrics, namely recall, precision and F-measure. The definitions of recall and precision are

given below:

$$\text{recall} = \frac{\text{Number of NEs correctly identified by the system}}{\text{Number of NEs in the gold standard test data}} \tag{3.2}$$

$$\text{precision} = \frac{\text{Number of NEs correctly identified by the system}}{\text{Number of NEs identified by the system}} \tag{3.3}$$

Precision is the ratio of the number of correctly found *NE chunks* (i. e., more than one token) to the number of found NE chunks, and recall is the ratio of the number of correctly found NE chunks to the number of true NE chunks. From the definitions, it is clear that while recall tries to increase the number of correctly tagged entries from the entire data set as much as possible, precision tries to increase the number of correctly tagged entries from the total number of identified entries. These two capture two different classification qualities.

The value of the metric F-measure, which is the weighted harmonic mean of recall and precision, is calculated as below:

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

.

3.9 Feature selection using MOO

Feature selection can be formulated as an optimization problem that involves choosing an relevant feature subset. Thus, in order to determine the best feature combination for NERC under the CRF framework, we simultaneously optimize both F-measure and feature count. Feature count should be lower and F-measure should be high. In order to achieve this, a MOO technique [10] is used.

As the performance of any classifier depends on the features of training and test data sets hence selecting relevant features for classification models is vital task. . Feature selection is the technique of selecting a subset of relevant features for building robust classifier. In a machine learning approach, feature selection is an optimization problem that involves choosing an appropriate feature subset. In CRF-based models, relevant feature selection is a crucial

problem and also a key issue to improve the classifier's performance. CRF does not provide a method for automatic feature selection, hence heuristics are used to find the appropriate set of features. In general, feature selection problems are solved using some single objective optimization techniques like GA [18], but as already described, these single objective optimization techniques can only optimize a single quality measures, for e. g., recall, precision, feature count or F-measure at a time. Feature count should be lower and F-measure should be high. Sometimes, a single measure cannot capture the quality of a good classifier reliably. A good classifier should have all its parameters optimized simultaneously in comparison with the high value of any parameter. Thus, here, we simultaneously optimize both F-measure and feature count in order to determine the best feature combination for NERC under the CRF framework.

A multiobjective optimization GA viz. NSGA-II, is used for solving the feature selection problem. This method selects relevant feature set for CRF models for German and two Indian languages, namely Bengali, Hindi.

3.9.1 Formulation of feature selection problem

Let us denote the N number of available features by by $f_1, f_2, \ldots, f_N$ and suppose that set of all features denoted by $F = f_i : i = 1toN$. Then the problem of feature selection can be stated as follows: Find a set of features G that will optimize a function $O(F)$ such that: $G \subseteq F$. Here, O is a measure of classification efficiency for the classifier trained using features set G. The particular type of problem like NERC has mainly three different kinds of classification quality measures, namely recall, precision, and F-measure and other possibility is number of features. Thus,

$$O \subseteq recall, precision, F - measure, no\_of\_features.$$

Thus, the feature selection problem under the single objective optimization framework looks as follows: Find a set of features G such that maximize

$$O(G); O \in recall, precision, F - measure, -(no\_of\_feaures) and G \subseteq F.$$

Here, we choose $O = F - measure$ as this is a combination of both recall and precision. The feature selection problem can be formulated under the MOO framework as below: Find a set of features G such that maximize $[O_1(G), O_2(G)]$, where

$$O_1, O_2 \in recall, precision, F - measure, -(no\_of\_features)$$

and $O_1 = O_2$. Here, $G \subseteq F$. We have chosen $O_1 = F - measure$ and $O_2 = -(no\_of\_features)$

3.9.2 Chromosome representation and population initialization

Let total number of features is N and size of the population is P and hence chromosome size will also be N. Fig. 3.4 represents the encoding of a particular chromosome with $N = 18$ which means maximum 18 different features are available. This chromosome represents the use of 9 features for constructing a classifier (first, second, fifth, eighth, tenth, twelfth, fourteenth, sixteenth and eighteenth features). The entries of each chromosome are randomly initialized to either 0 or 1. If the i^{th} position of a chromosome is 0, then it represents that i^{th} feature does not participate in feature template set for construction of CRF-based classifier. And if it is 1, then the i^{th} feature participates in feature template set for construction of CRF-based classifier. All the P number of chromosomes of this population are initialized in the way discussed above.

3.9.3 Fitness Computation

For the fitness computation, the following steps are executed.

- There are N number of features present in a particular chromosome (i.e., total N number of 1's in that chromosome).
- Build a CRF classifier with only these N features.
- Now Training data is divided in training set and development set of appropriate sizes. CRF classifier is trained on training data and tested on development set.
- F-measure and feature count is calculated
- Objective functions corresponding to a particular chromosome are F-measure and feature count. NSGA-II is used for optimization process using these two objective functions.

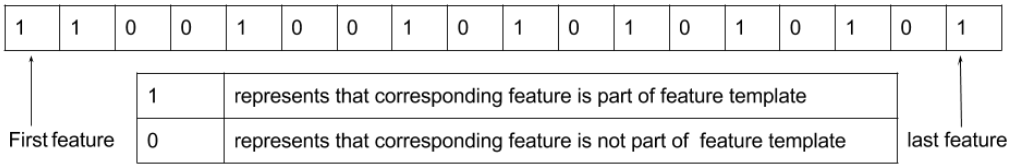

Figure 3.4: Chromosome representation of features

3.10 Evaluation Process

After incorporating unsupervised lexical acquisition features in training data MOO is used to select best feature set with single or multiple relevant objectives like recall, precision, F-score, cardinality of the selected feature set etc. NSGA–II algorithm is utilized for the optimization process. Non–Dominated Sorting Genetic Algorithm (NSGA II) is a elitist multi–objective evolutionary optimization technique that provides fast convergence rate to Pareto front. Due to presence of multiple objectives solution of Multi-objective optimization problem is set called Pareto optimal set. Hence we will get multiple number of features combinations which are incomparable to each other due to the presence of multiple objectives.

Classifier ensemble technique is employed to make the situation advantageous. In Classifier ensemble technique we do not build a single classifier but learn a set of classifiers. We combine the predictions of multiple classifiers. Benefits are reduced variance which is achieved as results are less dependent on peculiarities of a single training set and reduced bias also as a combination of multiple classifiers may learn a more expressive concept class than a single classifier. Our goal is to find which classifier should participate in ensemble process. Classification models like Maximum Entropy, Conditional Random field (CRF) and Support vector machine can be employed for building classifiers. As there are huge number of combination possible, we can utilize optimization techniques for taking this critical decision of choosing best.

In both cases, best features selection and best classifier ensemble selection we use NSGA–II algorithm as optimization technique. For best features selection, a fixed size binary valued chromosome is formulated in which each bit valued 1 represents presence of a particular feature and 0 represents absence

and in optimization process for best classifier ensemble, bits value represents participation of a different classifiers in determining of target class. Figure depicts the whole evaluation process.

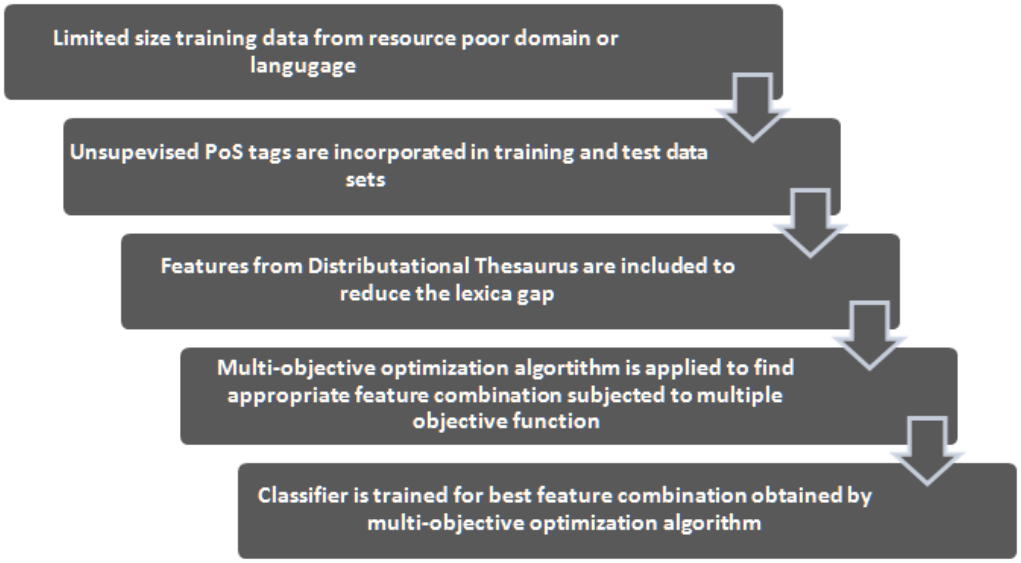

Figure 3.5: Evaluation process

Chapter 4

Experiments

In this chapter results of various Information extraction tasks on datasets of several languages (namely Hindi, Bengali, German) are discussed. We are using two objectives namely F1-Measure and feature count in every experiment.

4.1 NERC task for Hindi and Bengali language

Experiments on NERC tasks with various set of available features are performed on datasets of Hindi and Bengali language. In this section results of NERC task for Hindi and Bengali are discussed.

4.1.1 Datasets and Experimental Setup

Both Hindi and Bengali dataset contain a total of 27 features including the token itself. There are 22 syntactic features which are namely checkTwoDigit (1), checkFourDigit (2), checkDigitHyphen (3), checkDigitSlash (4), checkDigitDot (5), checkDigitPercentage (6), checkDigitComma (7), lastWord (8), wordFrequency (9), wordLength (10), wordPrefix (11-15), wordSufix (16-20), firstWord (21), PoS (22), followed by Unsupervised PoS (23) and then Distributional Thesaurus (DT) features (24-26). DT features are the top three most similar word.

Bengali Dataset: Training data for Bengali dataset contain 328,064 tokens of training examples and test data contain 34,200 tokens of Bengali

language NE tagged examples.

Hindi Dataset: Training data for Bengali dataset contain 462,120 tokens of training examples and test data contain 60,810 tokens of Bengali language NE tagged examples.

4.1.2 Results and Discussion

In all the experiments on NERC task for Hindi and Bengali language, we set the following initialization parameter values for NSGA-II algorithm: population size = 32, number of generations = 50, probability of crossover = 0.8 and probability of mutation = 0.0125. We use CRF as the base classifier.

The feature selection algorithm is run for two Indian languages Hindi and Bengali three times with different set of available features. So basically we design three experiments, one with only lexical features, second with lexical features plus unsupervised PoS tag and third experiment with three features from distributional thesaurus in addition of unsupervised tag and lexical features. Due to the presence of two objectives (i.e. feature count, $F_{\beta=1}$ score), we get a set of solution in final population. All of the them are best in their own and incomparable on the basis of aforementioned two objective collectively, but for building a CRF classifier we need only single set of features. We choose one solution with highest $F_{\beta=1}$ score value from the final population. Table 4.1 depicts difference performance metrics for NERC task on Hindi dataset without considering any lexical acquisition features (i.e. one unsupervised PoS tag, three DT features).

Overall results for this feature selection problem shows that there are 41 features in training feature set for classifier which produced best results among other feature set in final population. Precision, recall and $F_{\beta=1}$ score values yielded by this approach are 80.15%, 52.19%, and 63.22 respectively. These results can be considered as baseline for our further experiments on NERC tasks on Hindi.

Following are the 41 features used to train the CRF based classifier.

Unigram U1:%x[-1, 0] U2:%x[-1, 4] U3:%x[-1, 6] U4:%x[-1, 9] U5:%x[-1, 11] U6:%x[-1, 12] U7:%x[-1, 14] U8:%x[-1, 16] U9:%x[-1, 17] U10:%x[-1, 18] U11:%x[-1, 21] U12:%x[-1, 22]

U13:%x[0, 0] U14:%x[0, 1] U15:%x[0, 3] U16:%x[0, 4] U17:%x[0, 5] U18:%x[0, 6] U19:%x[0, 11] U20:%x[0, 12] U21:%x[0, 13] U22:%x[0, 15] U23:%x[0, 16]

Training Data	Precision	Recall	$F_{\beta=1}$	Feature Count
LOC	82.71%	47.97%	60.72	
MISC	83.37%	74.22%	78.53	
ORG	52.63%	29.85%	38.10	
PER	70.72%	29.15%	41.29	
Overall	80.15%	52.19%	63.22	41

Table 4.1: NERC performance metrics for Hindi data–set without lexical acquisition features , No. of Generations=50, Size of population=32

U24:%x[0, 17] U25:%x[0, 18] U26:%x[0, 19] U27:%x[0, 20] U28:%x[0, 22]

U29:%x[1, 0] U30:%x[1, 2] U31:%x[1, 3] U32:%x[1, 4] U33:%x[1, 5] U34:%x[1, 9] U35:%x[1, 13] U36:%x[1, 14] U37:%x[1, 16] U38:%x[1, 17] U39:%x[1, 19] U40:%x[1, 21] U41:%x[1, 22]

Bigram B

In the next experiment we incorporated unsupervised PoS tag in available set of feature for feature selection problem. When the feature selection algorithm is run on Hindi dataset for NERC we got the results shown in 4.2. Precision, recall and $F_{\beta=1}$ score values yielded by this approach are 79.22%, 54.45% and 64.54 respectively.

There is significant improvement in $F_{\beta=1}$ score and substantial reduction in feature count. Here we have 25 features in feature set selected from final population for building a CRF-based classifier. The improvement in result is because of incorporation of unsupervised PoS feature for training classifier. U16:%x[0, 23] represents the unsupervised PoS tag for the token itself. So the unsupervised part of speech tag for the word itself in combination with other lexical features reduced the total feature count and significant improvement in $F_{\beta=1}$ score is also observed.

Follwing is the feature set which yields highest $F_{\beta=1}$ score in final population of feature selection optimization problem after incorporating unsupervised PoS tag.

Unigram U1:%x[-1, 5] U2:%x[-1, 8] U3:%x[-1, 9] U4:%x[-1, 11] U5:%x[-1, 16] U6:%x[-1, 19]

U7:%x[0, 0] U8:%x[0, 1] U9:%x[0, 4] U10:%x[0, 13] U11:%x[0, 16] U12:%x[0,

	Precision	Recall	$F_{\beta=1}$	Feature Count
LOC	82.20%	49.24%	61.59	
MISC	83.00%	76.78%	79.77	
ORG	62.50%	29.85%	40.40	
PER	67.42%	32.14%	43.53	
Overall	79.22%	54.45%	64.54	25

Table 4.2: NERC performance metrics for Hindi data–set with Unsupervised PoS feature, No. of Generations=50, Size of population=32

17] U13:%x[0, 18] U14:%x[0, 20] U15:%x[0, 22] **U16:%x[0, 23]**

U17:%x[1, 4] U18:%x[1, 5] U19:%x[1, 7] U20:%x[1, 11] U21:%x[1, 12] U22:%x[1, 15] U23:%x[1, 18] U24:%x[1, 21]
Bigram B

Inspired from [23] where lexical expansion is utilized for closing the lexical gap for word-sense disambiguation problem, in our third experiment on NERC task we integrate some features from distribution thesaurus. We fetched three most similar word from distributional thesaurus for each token in training and test dataset. Now we have a total of four lexical acquisition features in our datasets which include one unsupervised PoS tag and three most similar word from distributional thesaurus. Algorithm for feature selection is run for Hindi dataset with additional distributional distributional features and results are reported in table 4.3. It is observed that there is substantial hike in overall $F_{\beta=1}$ score.

	Precision	Recall	$F_{\beta=1}$	Feature Count
LOC	72.88%	63.39%	67.81	
MISC	80.08%	82.76%	81.40	
ORG	55.13%	56.95%	56.03	
PER	63.87%	43.96%	52.08	
Overall	73.26%	66.44%	69.68	32

Table 4.3: NERC performance metrics for Hindi data–set with Unsupervised PoS feature and DT features, No. of Generations=50, Size of population=32

There are total 32 features in feature set selected for training the NE tag classifier. These features are given below in CRF++ feature template for-

mat. This feature combination include several lexical expansion features (or distributional thesaurus features) which are $U8 : \%x[-1, 25]$, $U9 : \%x[-1, 26]$ from previous context word of token and $U22 : \%x[0, 24]$, $U23 : \%x[-1, 25]$, $U24 : \%x[0, 26]$ from token itself. These distributional thesaurus in combination with unsupervised PoS tag and other lexical acquisition feature improved the performance of the classifier. Precision, recall and $F_{\beta=1}$ score yielded by this approach are 73.26%, 66.44% and 69.68 respectively. Fig. 4.2 depicts the Pareto optimal front of last generation in optimization process with unsupervised PoS Tag and DT features.

Unigram U1:%x[-1, 0] U2:%x[-1, 4] U3:%x[-1, 6] U4:%x[-1, 8] U5:%x[-1, 10] U6:%x[-1, 11] U7:%x[-1, 23] **U8:%x[-1, 25] U9:%x[-1, 26]**

U10:%x[0, 0] U11:%x[0, 2] U12:%x[0, 4] U13:%x[0, 7] U14:%x[0, 8] U15:%x[0, 11] U16:%x[0, 13] U17:%x[0, 17] U18:%x[0, 18] U19:%x[0, 19] U20:%x[0, 20] U21:%x[0, 23] **U22:%x[0, 24] U23:%x[0, 25] U24:%x[0, 26]**

U25:%x[1, 3] U26:%x[1, 8] U27:%x[1, 13] U28:%x[1, 15] U29:%x[1, 16] U30:%x[1, 19] U31:%x[1, 22] U32:%x[1, 23]

Bigram B

Figure 4.1 show the difference in output NE tags after incorporation of DT features and compares it with tags without DT features and actual NE tags.

Table 4.4 depicts the F1-score and feature count for Bengali dataset. It shows a significant improvement in F1-score after including unsupervised PoS tag and DT features. Although there is not much difference between F1-score of first two experiments but there is substantial reduction in feature count.

Training Data	F1–Score	Feature Count
Without unsupervised PoS Tag and DT features	72.44	30
With unsupervised PoS Tag only	72.72	14
With unsupervised PoS Tag and DT features	73.50	21

Table 4.4: NER performance metrics for Bengali data–set, No. of Generations=50, Size of population=52

Tokens from a Hindi sentence	Similar words from Distributional Thesaurus			Actual NER Tag	NER Tag without DT Features	NER tag with DT features
इनकी	उसकी	इसकी	इसकी	O	O	O
सहायता	मदद	राहत	सहयोग	O	O	O
से	को	उससे	के	O	O	O
ही	भी	वह	वे	O	O	O
कुशल	सफल	भारतीय	अच्छे	O	O	O
गुप्तचरों	अधिकारी	जासूसों	एजेंटों	O	O	O
को	उन्हें	उसे	द्वारा	O	O	O
नियुक्त	तैनात	पद	बनने	O	O	O
कर	करने	किया	करना	O	O	O
राम	कृष्ण	नारायण	प्रसाद	I-PER	O	I-PER
लंका	कोलकाता	तो	घर	I-LOC	O	I-LOC
मैं	मे	मंे	मेेंं	O	O	O
सुरक्षित	खतरनाक	उचित	बेहतर	O	O	O
स्थान	मिनट	ओवर	नंबर	O	O	O
पर	से	को	तक	O	O	O
बन्दिनी	ND	ND	ND	O	I-PER	I-PER
सीता	संतोष	रमेश	मुकेश	I-PER	I-PER	I-PER
को	उन्हें	उसे	द्वारा	O	O	O
खोज	तलाश	ढूंढ	निकाल	O	O	O
सके	सकते	सकती	सकता	O	O	O

Figure 4.1: Difference in prediction tag after incorporating DT features

4.2 NERC task on German

4.2.1 Dataset Description and Experimental setup

In named entity recognition dataset for German language, we extract a total of twelve features including the token itself. In this feature set we have different standard lexical features, unsupervised PoS tag and three most similar words from distributional thesaurus (DT) in the following order. Token (1), FirstLetterCapital (2), IsDigit (3), TwoLetterPrefix (4), TwoLetterSuffix (5), BaseForm (6), ChunkTag (7), PoS (8), UnsupervisedPoS (9), DTFirstSimilarWord (10), DTSecondSimilarWord (11), DTSecondSimilarWord (12) and thirteenth column contain output class i.e. named entity tag of corresponding token. Table 4.5 shows a few lines from German NERC dataset.

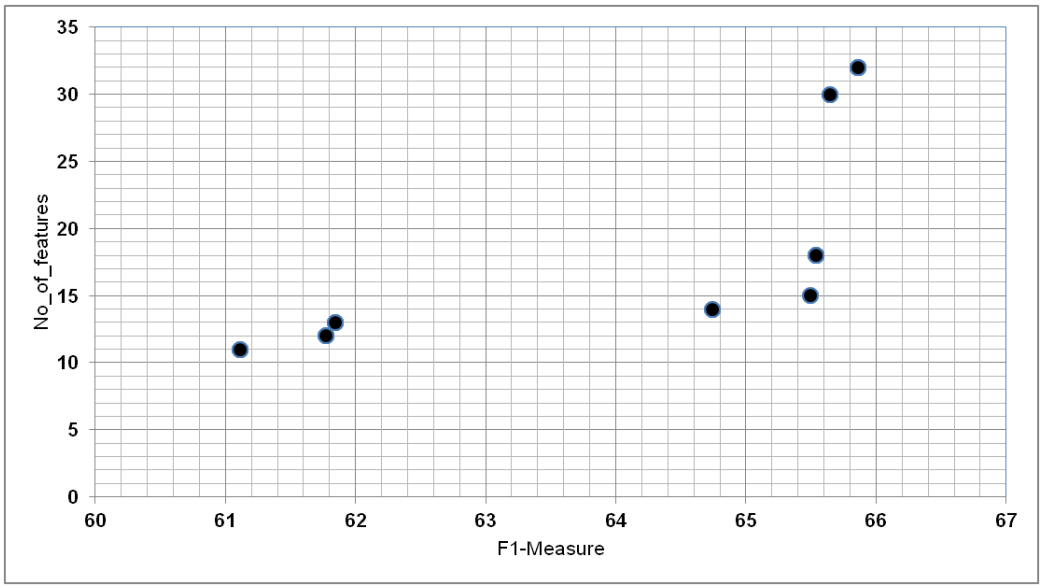

Figure 4.2: Pareto Optimal front for last generation in a experiment on Hindi Data with syntactic features, unsupervised PoS tag and distributional thesaurus features

Token	FC	DG	PRE	SUF	BSF	CH	POS	UNS_POS	DT1	DT2	DT3	NE
Großer	1	0	Gr	er	Große	I-NC	NN	13	Diesen	Dieser	Großen	O
Foto-Wettbeweb	1	0	Fo	eb	<unknown>	I-NC	NN	0	ND	ND	ND	O
”	0	0	ND	ND	”	O	$ (	280	<<	’	>>	O
NORDEND	1	0	NO	ND	<unknown>	I-NC	NN	8	Cedric	415	Ava	I-LOC
.	0	0	ND	ND	.	O	$.	305		!	. . .	O

Table 4.5: German NERC Dataset format

Training dataset for German NERC task contain 220,187 tokens of training examples and test data contain 54,711 tokens of German language NE tagged examples. In all the experiments on NERC task for German language, we set the following initialization parameter values for NSGA-II algorithm: population size = 32, number of generations = 50, probability of crossover = 0. 8 and probability of mutation = 0. 125. We use CRF as the base classifier.

4.2.2 Results and discussion

Table 4.6 depicts various performance metrics for NERC task on German dataset without considering any unsupervised lexical acquisition features (i.e. one unsupervised PoS tag, three DT features).

Overall results for this feature selection problem shows that there are 20 features in training feature set for classifier which produced best results among other feature sets in final population. Precision, recall and $F_{\beta=1}$ score values yielded by this approach are 80.43%, 64.11% and 71.35 respectively. These results can be considered as baseline for our further experiments on NERC tasks on German.

Unigram U1:%x[-1, 0] U2:%x[-1, 1] U3:%x[-1, 3] U4:%x[-1, 4] U5:%x[-1, 5] U6:%x[-1, 6] U7:%x[-1, 7]
U8:%x[0, 0] U9:%x[0, 1] U10:%x[0, 2] U11:%x[0, 3] U12:%x[0, 4] U13:%x[0, 5] U14:%x[0, 7]
U15:%x[1, 0] U16:%x[1, 1] U17:%x[1, 3] U18:%x[1, 4] U19:%x[1, 6] U20:%x[1, 7]
Bigram
B

	Precision	Recall	$F_{\beta=1}$	Feature count
LOC	77.36%	67.94%	72.34	
MISC	80.52%	30.10%	43.82	
ORG	73.47%	59.76%	65.91	
PER	86.83%	68.68%	76.70	
Overall	80.43%	64.11%	71.35	20

Table 4.6: NERC performance metrics for German data–set with syntactic features only, No. of Generations=50, Size of population=52

In the next experiment on German NERC dataset we incorporated DT features in available set of features for feature selection problem. Table 4.7 depicts various performance metrics for NERC task on German dataset with DT features considered. Precision, recall and $F_{\beta=1}$ score values yielded by this approach are 82.89%, 65.72% and 73.31 respectively and feature count is 19.

There is improvement in $F_{\beta=1}$ score and reduction in feature count by one. Here we have 19 features in feature set selected from final population

	Precision	Recall	$F_{\beta=1}$	Feature Count
LOC	81.40%	69.93%	75.23	
MISC	79.22%	29.61%	43.11	
ORG	74.50%	57.02%	64.60	
PER	88.31%	72.40%	79.56	
Overall	82.89%	65.72%	73.31	19

Table 4.7: NERC performance metrics for German data–set with DT features, No. of Generations=50, Size of population=32

for building a CRF-based classifier. The improvement in result is because of incorporation of DT features. Below is feature template for German dataset with DT features with DT features in bold letters. Features with index [*, 8], [*, 9], [*, 10] are DT features.
Unigram U1:U2:%x[-1, 6] U3:%x[-1, 8] **U4:%x[-1, 9]**
U5:%x[0, 1] U6:%x[0, 2] U7:%x[0, 4] U8:%x[0, 5] U9:%x[0, 6] U10:%x[0, 7] **U11:%x[0, 8] U12:%x[0, 9] U13:%x[0, 10]**
U14:%x[1, 0] U15:%x[1, 2] U16:%x[1, 6] U17:%x[1, 7] **U18:%x[1, 8] U19:%x[1, 10]**
Bigram
B

In the next experiment on German NERC dataset we incorporated DT features as well as unsupervised PoS tag in available set of features for feature selection problem. Table 4.8 depicts various performance metrics for NERC task on German dataset with all unsupervised lexical acquisition features considered. Precision, recall and $F_{\beta=1}$ score values yielded by this approach are 86.21%, 71.52% and 78.18 respectively and feature count is 21.

There is improvement in $F_{\beta=1}$ score. Here we have 21 features in feature set selected from final population for building a CRF-based classifier. The improvement in result is because of incorporation of unsupervised PoS tag and DT features. Below is feature template produced by feature selection module for German dataset with unsupervised PoS tag and DT features marked in bold letters. Feature with index [*, 8] is unsupervised tag and [*, 9], [*, 10], [*, 11] are DT features.
Unigram U1:U2:%x[-1, 6] U3:%x[-1, 7] **U4:%x[-1, 8] U5:%x[-1, 9] U6:%x[-1, 10] U7:%x[-1, 11]**
U8:%x[0, 0] U9:%x[0, 1] U10:%x[0, 4] U11:%x[0, 5] U12:%x[0, 7] **U13:%x[0,**

	Precision	Recall	$F_{\beta=1}$	Feature count
LOC	84.87%	72.60%	78.26	
MISC	79.75%	30.58%	44.21	
ORG	74.64%	61.99%	67.73	
PER	93.07%	82.15%	87.27	
Overall	86.21%	71.52%	78.18	21

Table 4.8: NERC performance metrics for German data–set with Unsupervised PoS feature and DT features, No. of Generations=50, Size of population=52

8] U14:%x[0, 9] U15:%x[0, 11]
U16:%x[1, 4] U17:%x[1, 6] U18:%x[1, 7] **U19:%x[1, 8] U20:%x[1, 9] U21:%x[1, 11]**
Bigram
B

4.3 Chunking Experimental Results

Promises of the scheme has been demonstrated by performing chunking experiment on English language datasets. The goal is to come up with classifier that can predict chunk tag with higher accuracy. Whole experiment is performed twice, once using training data without lexical acquisition features and again with lexical acquisition features. Experiments are performed for two training datasets of different size.

Both the training data and the test data have to be in a particular format for CRF++ [1] to work properly. Training and test file consist of multiple tokens and a token consists of multiple but fixed number of columns. The definition of tokens is task-dependent. Here each token must be represented in one line, with the columns separated by spaces or tabular characters. Boundary between sentences is indicated by putting an empty line. In our training data and testing 1st column is 'word' itself, second column is 'POS tag', third column is 'Unsupervised lexical acquisition feature' and the last column represents a true answer tag which is going to be trained by CRF.

[1]CRF++:Yet another CRF toolkit http://crfpp. googlecode. com/svn/trunk/doc/index. html

Dataset-I: Classifiers based on CRF model are trained with dataset-I for chunk tag prediction. This dataset contains 1973 lines and each line holds 4 columns where 1st column is 'word' itself, second column is 'POS tag', third column is 'Unsupervised lexical acquisition feature' and the last column represents a true answer chunk tag. Chunks are represented in the IOB2 format (B for BEGIN, I for INSIDE, and O for OUTSIDE). NSGA-II is configured to run for 20 generations with population size of 60. After running the whole experiment for training data with and without unsupervised PoS feature we got the results displayed in table 4.10. Result shows a little improvement after including unsupervised PoS feature. CRF++ templates of the Best feature set is shown in table 4.9. Each line in the template file denotes one template. In each template, special macro %x[row, col] is used to specify a token in the input data. row specifies the relative position from the current focusing token and col specifies the absolute position of the column. A total of 14 features [-3, 0] to [3, 1] represented by a binary valued chromosome of size 14 bits. Only one optimization function is used i.e. maximize F1-score.

Without unsupervised PoS feature	With unsupervised PoS feature
#Unigram	#Unigram
U2:%x[-1, 1]	U2:%x[-1, 1]
U3:%x[0, 0]	U3:%x[0, 1]
U4:%x[0, 1]	**U4:%x[0, 2]**
#Biagram	#Biagram
B	B

Table 4.9: Best feature templates for smaller dataset I

Training Data	Recall	Precision	F–Score
Without unsupervised PoS Tag	84.67	85.92	85.29
With unsupervised PoS Tag	86.16	84.85	85.50

Table 4.10: Chunking performance metrics for smaller data–set, No. of Generations=20, Size of population=60

Dataset-II: Dataset–II is comparatively larger in size than dataset–II and contains 46491 lines where each line holds 4 columns and 1st column is

'word' itself, second column is 'POS tag', third column is 'Unsupervised lexical acquisition feature' and the last column represents a true answer chunk tag. The whole experiment is performed for both training data with and without unsupervised Lexical acquisition feature. Results are compared in table 4.11. NSGA-II software is configured for single objective only i.e. maximize F1-score. In further experiments multiple objective will also be explored. Population size is taken 60 and optimization is carried out for 20 generations. Best feature set is shown in form of CRF template in table 4.12.

Training Data	Recall	Precision	F–Score
Without unsupervised PoS Tag	91.39	91.64	91.52
With unsupervised PoS Tag	91.82	91.66	91.74

Table 4.11: Chunking performance metrics for Larger data–set, No. of Generations=20, Size of population=60

Without unsupervised PoS feature	With unsupervised PoS feature
# Unigram	# Unigram
U1:%x[-3, 0]	U1:%x[-2, 1]
U2:%x[-2, 1]	U2:%x[-1, 0]
U3:%x[-1, 0]	U3:%x[-1, 1]
U4:%x[-1, 1]	**U4:%x[-1, 2]**
U5:%x[0, 0]	U5:%x[0, 0]
U6:%x[0, 1]	U6:%x[0, 1]
U7:%x[1, 0]	**U7:%x[0, 2]**
U8:%x[1, 1]	U8:%x[1, 0]
U9:%x[2, 1]	U9:%x[1, 1]
B	B

Table 4.12: Best feature templates for dataset II

Chapter 5

Conclusion and Outlook

In this present work, we proposed a unsupervised lexical acquisition and MOO-based technique for building NERC system. Our underlying assumption is that using features from unsupervised lexical acquisition resources and rather than heuristically selecting a feature-set to train a classifier, using MOO based feature selection can be alternative research direction to deal with scarcity of training data for resource poor languages.

As it is not a priori clear that features from which of unsupervised lexical acquisition techniques are more useful for a particular task or language, feature selection using MOO proved to be handy.

We came up with sufficient experiments to support our hypothesis that using features from unsupervised lexical acquisition and applying MOO for feature selection could be a more fruitful approach. The proposed approach encodes the features in its chromosome. The two objective functions for feature selection using MOO are feature count and $F_{\beta=1}$ for a classifier trained using features encoded in a particular chromosome. We used conditional random field (CRF) classifier as the base classifier. Multiple classification models are developed by varying available feature-set size. Other interesting and important characteristic of purposed system is that it makes use of mostly language-independent features that can be easily derived for almost all the languages without any knowledge of them a priori. Thus, a variety of languages are benefited by our proposed algorithm. Proposed unsupervised lexical acquisition and MOO-based NERC system is evaluated for German and two resource poor Indian languages, namely Hindi and Bengali. Some chunking experiments are also performed for English language. It has been consistently observed that incorporation of unsupervised lexical acquisition

features and using MOO-based feature selection results in significant improvement in performance for a variety of information extraction tasks and languages.

In future we would like to include more language independent features in our available feature set from various existing resources and tools. Rather than selecting single best-fitting feature set from best population produced by MOO algorithm, ensembling several classification systems where each one is developed using different feature set and/or different classification technique can be explored. It has been observed that there is slight improvement in F-Score when using unsupervised lexical acquisition features.

Bibliography

[1] Constitution of India, 1950, Art. 344(1) and 351. page 330, 1950.

[2] Bogdan Babych and A. Hartley. Improving Machine Translation Quality with Automatic Named Entity Recognition. In *Proceedings of EAMT/EACL 2003 Workshop on MT and other Language Technology Tools*, pages 1–8, Hungary, 2003.

[3] Daniel Bär, Chris Biemann, Iryna Gurevych, and Torsten Zesch. Ukp: Computing semantic textual similarity by combining multiple content similarity measures. 2(3):435–440, 2012.

[4] C. Biemann, G. Heyer, U. Quasthoff, and M. (Richter. The leipzig corpora collection:. In *Proceedings of the Corpus Linguistics Conference*. 2007.

[5] C. Biemann, C.and Giuliano and A. Gliozzo. Unsupervised part of speech tagging supporting supervised methods. In *Proceedings of RANLP-07*, 2007.

[6] Chris Biemann. Unsupervised part-of-speech tagging in the large. *Research on Language and Computation*, ISSN 1572-8706.

[7] A. Borthwick. *Maximum Entropy Approach to Named Entity Recognition*. PhD thesis, New York University, 1999.

[8] A. Clark. Combining distributional and morphological information for part of speech induction. In *In proceedings of European chapter of the Association for Computational Linguistics (EACL-03)*, pages 59–66, 2003.

[9] Carlos A Coello Coello. A comprehensive survey of evolutionary-based multiobjective optimization techniques. *Knowledge and Information Systems*, 1(3):269–308, August 1999.

[10] Kalyanmoy Deb. *Multi-objective Optimization Using Evolutionary Algorithms*. John Wiley and Sons, Ltd, England, 2001.

[11] Kalyanmoy Deb, Amrit Pratap, Sameer Agarwal, and T. Meyarivan. A fast and elitist multiobjective genetic algorithm: NSGA-II. *IEEE Transactions on Evolutionary Computation*, 6(2):181–197, 2002.

[12] A. Ekbal and S. Saha. Multiobjective Optimization for Classifier Ensemble and Feature Selection: An Application to Named Entity Recognition. *International Journal on Document Analysis and Recognition (IJDAR)*, 8, 2011.

[13] Asif Ekbal and Sivaji Bandyopadhyay. A web-based Bengali news corpus for named entity recognition. *Language Resources and Evaluation*, 42(2):173–182, 2008.

[14] Asif Ekbal and Sriparna Saha. Simulated annealing based classifier ensemble techniques: Application to part of speech tagging. *Information Fusion (in press)*.

[15] Asif Ekbal and Sriparna Saha. Weighted vote-based classifier ensemble for named entity recognition: A genetic algorithm-based approach. *ACM Transactions on Asian Language Information Processing (TALIP)*, 10(2):9, 2011.

[16] Asif Ekbal and Sriparna Saha. Feature selection and stacked-based ensemble for biomedical entity extraction. *Knowledge based Information Systems (accepted)*, 2013.

[17] Asif Ekbal and Sriparna Saha. Multiobjective optimization for classifier ensemble and feature selection: an application to named entity recognition. *International Journal on Document Analysis and Recognition (IJDAR)*, pages 1–24, March 25, 2011.

[18] D. E. Goldberg. *Genetic Algorithms in Search, Optimization and Machine Learning*. Addison-Wesley, New York, 1989.

[19] Zellig S. Harris. Methods in structural linguistics.

[20] John D. Lafferty, Andrew McCallum, and Fernando C. N. Pereira. Conditional Random Fields: Probabilistic Models for Segmenting and Labeling Sequence Data. In *ICML*, pages 282–289, 2001.

[21] D. Lin. Automatic retrieval and clustering of similar words. pages 768–774, 1998.

[22] Thomas Mandl and Christa Womser-Hacker. The Effect of Named Entities on Effectiveness in Cross-language Information Retrieval Evaluation. In *Proceedings of the 2005 ACM Symposium on Applied Computing (SAC'2005)*, pages 1059–1064, 2005.

[23] Tristan Miller, Chris Biemann, Torsten Zesch, and Iryna Gurevych. Using distributional similarity for lexical expansion in knowledge-based word sense disambiguation. *Proceedings of COLING–12, Mumbai, India*, 2012.

[24] David Nadeau and Satoshi Sekine. A survey of named entity recognition and classification. *Lingvisticae Investigationes*, 30(1):3–26, 2007.

[25] C. Nobata, S. Sekine, H. Isahara, and R. Grishman. Summarization System Integrated with Named Entity Tagging and IE Pattern Discovery. In *Proceedings of Third International Conference on Language Resources and Evaluation (LREC 2002)*, pages 1742–1745, 2002.

[26] Luiz Augusto Pizzato, Diego Molla, and Cecile Paris. Pseudo Relevanc Feedback Using Named Entities for Question Answering. In *Proceedings of the 2006 Australian Language Technology Workshop (ALTW-2006)*, pages 89–90, 2006.

[27] Fei Sha and Fernando Pereira. Shallow Parsing with Conditional Random Fields. In *Proceedings of NAACL '03*, pages 134–141, Canada, 2003.

[28] Vladimir N. Vapnik. *The nature of statistical learning theory*. Springer-Verlag New York, Inc., New York, NY, USA, 1995.

[29] D. Van Veldhuizen and G. Lamont. Multiobjective evolutionary algorithms: Analyzing the state-of-the-art. *Evolutionary Computations*, 2:125–147, 2000.