NAMED ENTITY RECOGNITION(NER) APPROACHES FOR HINDI TEXT DOCUMENTS

Dissertation submitted to the Jawaharlal Nehru University
in partial fulfillment of the requirements
for the award of the degree of

MASTER OF TECHNOLOGY
In
COMPUTER SCIENCE AND TECHNOLOGY

Submitted By

Aparna Kumari

Under the Supervision of

Dr. Aditi Sharan

SCHOOL OF COMPUTER AND SYSTEMS SCIENCES
JAWAHARLAL NEHRU UNIVERSITY
NEW DELHI – 110067
JULY-2011

जवाहरलाल नेहरू विश्वविद्यालय

School of Computer & Systems Sciences
JAWAHARLAL NEHRU UNIVERSITY
NEW DELHI- 110067, INDIA

DECLARATION

I hereby declare that this dissertation entitled "**NAMED ENTITY RECOGNITION(NER) APPROACHES FOR HINDI TEXT DOCUMENTS** " submitted by me to the School of Computer and Systems Sciences, Jawaharlal Nehru University, New Delhi, for the award of the degree of **"Master of Technology in Computer Science and Technology"**, is a record of bonafide work carried out by me under the supervision of Dr. Aditi Sharan. This dissertation comprises only my original work.

The matter embodied in the dissertation has not been submitted for the award of any other degree or diploma in any university or institute.

Aparna Kumari
Aparna Kumari
M.Tech(2009-2011)
School of Computer & Systems Sciences,
Jawaharlal Nehru University,
New Delhi-110067

जवाहरलाल नेहरू विश्वविद्यालय

School of Computer & Systems Sciences
JAWAHARLAL NEHRU UNIVERSITY
NEW DELHI- 110067, INDIA

CERTIFICATE

This is to certify that this dissertation entitled “**NAMED ENTITY RECOGNITION(NER) APPROACHES FOR HINDI TEXT DOCUMENTS**” submitted by **Miss Aparna Kumari** to the School of Computer and Systems Sciences, Jawaharlal Nehru University, New Delhi, for the award of the degree of “**Master of Technology in Computer Science and Technology**”, is a record of bonafide work carried out by her under the supervision of Dr. Aditi Sharan.

This work has not been submitted in part or full to any university or institution for the award of any degree or diploma.

Sharan

Supervisor 22-7-2011

Dr. Aditi Sharan
School of Computer & System Sciences
Jawaharlal Nehru University
New Delhi -110067

Dean

School of Computer & System Sciences
Jawaharlal Nehru University
New Delhi-110067

प्रोफेसर कर्मेषु/Professor Karmeshu
डीन / Dean
स्कूल ऑफ कम्प्यूटर और पद्धति विज्ञान संस्थान
School of Computer and Systems Sciences
जवाहरलाल नेहरू विश्वविद्यालय
Jawaharlal Nehru University
नई दिल्ली / New Delhi-110067

To My Loving Family & Friends…..

ACKNOWLEDGEMENTS

At the very outset I thank Almighty God for all the favors He Showered upon me throughout my life.

I am very glad to express my sincere gratitude and thanks to my supervisor ***Dr. Aditi Sharan*** *for her expert guidance in completing this work. Without her constructive suggestions, critical comments, encouragement, help and kind cooperation, this dissertation would not have taken a shape. I feel it is a great privilege to have had the opportunity to work under her prestigious supervision. Her constant encouragement and support helped me a lot. She led me into the exciting field of Information Retrieval and taught me how to approach the problem in rigorous way. The methodology and philosophy that I have learned in my research will definitely benefit my career for life.*

I also express whole hearted thanks to my friends, classmates and labmates for their care, valuable help, comments, suggestions for this dissertation and moral supports. The moments, I enjoyed with them during my M. Tech course will always remain as a happy memory throughout my life.

Where would I be without my family? My parents deserve special mention for their inseparable support and prayers. My Father, Mr. Ram Lakhan Banbari, in the first place is the person who put the fundament my learning character, showing me the joy of intellectual pursuit ever since I was a child. My Mother, Mrs. Usha Devi, is the one who sincerely raised me with her caring and gently love. Archana, Prasun, and Manjusha thanks for being supportive and caring siblings.

I am grateful to the entire faculty, staff of SC & SS and administration of JNU for providing a congenial environment for making our work a success. Finally, I would like to thank everybody who was important to the successful realization of thesis, as well as expressing my apology that I could not mention personally one by one.

Aparna Kumari

Aparna Kumari

ABSTRACT

Extracting named entities from web sources is an important operation in many applications, including data warehousing, web data integration and text mining. Named entities are perhaps the most important indexing element in text for most of the information extraction and mining tasks. In this dissertation, we consider the problem of Named Entity Recognition (NER) from web pages. The scale, unstructuredness, and diversity of the web pose challenges to named entities on the webpages. Traditionally, Rule based and Machine Learning techniques based systems have been used for this task. Rule based techniques require large amount of linguistic knowledge where as machine learning techniques require large amount of labeled data sets. This poses a serious limitation for resource poor language such as Hindi. Further, most of the approaches require a standard gazetteer list to start with. In Hindi even such gazetteer list are not available. Therefore NER becomes a very challenging problem.

In this work, we have tried to study and compare various approaches for named entity recognition in context of Hindi language. Further, we have tried to find out difficulties and challenges in implementing these approaches. To make the Hindi NER system, we have proposed a context pattern induction based domain specific approach. This approach can be used as a gazetteer preparation methodology. This approach uses a domain specific raw corpus and a few seed entities to learn context patterns and then the corresponding name lists are generated.

The proposed NER system addresses two major limitations. First, the system requires no human intervention such as manually labeling training data or creating gazetteers. Second, the system does not require any labeled dataset. We describe the system's architecture of the proposed system. The proposed context pattern induction methodology has been used for Hindi language but it can be easily applicable for other languages.

Table of Contents

I. List of Figures

II. List of Tables

List of Abbreviation

NE	**Named Entity**
NEs	**Named Entities**
NP	**Noun Phrase**
NER	**Named Entity Recognition**
IE	**Information Extraction**
NLP	**Natural Language Processing**
IR	**Information Retrieval**
ML	**Machine Learning**
HMM	**Hidden Markov Model**
ME	**Maximum Entropy**
MAXEnt	**Maximum Entropy**
MEMM	**Maximum Entropy Markov Model**
CRF	**Conditional Random Field**
SVM	**Support Vector Machine**
DT	**Decision Tree**
POS	**Part of Speech Tagger**

Chapter 1

Introduction

In today's world, overload of information, particularly through the World Wide Web, creates difficulties for the users to access the right information. This situation becomes more difficult due to the fact that a large amount of information is available on internet in different languages. Therefore, this is very important to apply an information process that will extract only the relevant facts that match the user's interests from all that volume of information and allow the user to access those facts. Information Extraction (IE) technology can meet these requirements, since unlike what happens with information retrieval and filtering technology, in IE the user's interests are on specific facts extracted from the documents and not just on the documents themselves. Some documents may contain the requested keywords but be irrelevant to the user's interests. Providing the specific facts instead of documents provides to users with information that is more relevant to their interests, gives all the reasons for smiles to the users [1].

"*Information extraction is the identification, and consequent or concurrent classification and structuring into semantic classes, of specific information found in unstructured data sources, such as natural language text, providing additional aids to access and interpret the unstructured data by information systems.*" [46]

The IE system can be use to extract, fixed information from documents in a fixed language. "*Web information extraction is the problem of extracting target information items from Web pages*" [47]. However, in order for the IE technology to be truly applicable in real life applications, IE systems need to be easily customizable to new domains and languages. In this work, our focus is towards named entity recognition (NER) which is a subtask of IE. The NER task involves typically two sub-tasks: the recognition of named entities (e.g. persons, organizations, locations, dates) involved in an event and the recognition of the relationships holding between named entities in that event (e.g. personnel joining and leaving companies in management succession events). The named entities are the smallest sort of information-hunting task [48]. A "**named entity**" (NE) is a phrase, which serves as a name for something or someone. According to this definition, the phrase in question must be a noun phrase (NP). Clearly, not all NPs are named entities.

1.1 Named Entity Recognition(NER)

NE Definition:

> "*Named entities are phrases that contain the names of persons, organizations, locations, times, and quantities.*" [11].

NER Definition:

- NER involves **identification** of proper names in texts and its **classification** into a set of predefined categories of interest.
- Three universally accepted categories: **Person, Location** and **Organization**
- Other common tasks: recognition of date/time expressions, measures (Percent, money, weight etc), email address, domain specific entities like name of drug etc [12].

As per the definition, NER task is concerned with marking occurrences of a specific object being mentioned. These mentions are then classified into standard categories include "person", "location", "geo-political organization", "facility", "organization", and "time" etc. For example, in the sentence "The George Washington Bridge spans the Hudson River," the phrase "George Washington Bridge" would be marked and classified as a facility, and "Hudson River" would be marked as a location. However, the sentence "The bridge spans the river" does not contain any named entity mentions, as the task is traditionally defined, because "bridge "and "river" do not refer to specific entities in this context; they are generic mentions [2]. Researchers have explored domain specific, multi-lingual, and standard Named Entity Recognition. Several conferences have been devoted to Mention Detection: MUC, ACE, and CoNLL-2002[11] and 2003 [3] [4] [5]. In these conferences, NER systems have been developed that achieve F-measure scores (a weighted average of precision and accuracy) of up to 88.7 on English newswire texts [5].

A large number of techniques have been developed to recognize named entities. Some of them are Rule based and others are Statistical techniques. The rule based approaches uses the morphological and contextual evidence [6] [7] of a natural language and consequently determine the named entities.

The statistical techniques use large annotated data to train model, like HMM [9] and subsequently examine it with the test data. Both the methods mentioned above required the efforts of language experts [10].

1.2 Standard Categories of Named Entity(NE)

The following universal categories have been developed in Message Understanding Conference in the 1990s. Top three categories are used for Name identification, Number Identification and time and Date identification respectively. Others category include the URL, email address and domain names identification [3] [50].

* ENAMEX
* NUMEX
* TIMEX
* Others

Most research on NER systems has been structured as taking an annotated block of text, such as this one:

" सचिन तेंदुलकर ने आइसीसी विश्व कप- २०११ में ५९ के औसत से ३७९ रन बनाए हैं, जो टूर्नामेंट में उनका तीसरा सर्वश्रेष्ठ स्कोर है।"

And producing an annotated block of text, such as this one:

***<ENAMEX TYPE="PERSON">* सचिन तेंदुलकर *</ENAMEX>* ने *<ENAMEX TYPE="ORGANIZATION">* आइसीसी *</ENAMEX>* विश्व कप *<TIMEX TYPE="DATE">* २०११ *</TIMEX>* में *<NUMEX TYPE="QUANTITY">* ५९ *</NUMEX>* के औसत से *<NUMEX TYPE="QUANTITY">* ३७९ *</NUMEX>* रन बनाए हैं, जो टूर्नामेंट में उनका *<NUMEX TYPE="QUANTITY">* तीसरा *</NUMEX>* सर्वश्रेष्ठ स्कोर है।**

From the annotated block of text, different text can be identified with sound meaning. The text"सचिन तेंदुलकर" is the name of the person i.e. a noun and it comes in category ENAMEX. " आइसीसी" is the organization name which comes under the ENAMEX category too. "२०११" is the year that represents time so that fall in TIMEX category. "५९", "३७९" is the numeric quantity so it falls under NUMEX category.

1.3 NE Basic System Architecture

A typical NER system involves the exploitation of a lexicon and a grammar, which need to be updated when the system is customized to a new domain. The lexicon is a set of gazetteer lists, containing names that are known beforehand and have been classified into named-entity types. The grammar is used to recognize named entities that are not in the gazetteer lists or they occur in more than one gazetteer list. Manual construction of these resources is a very time-consuming process and therefore, a number of worth examining methods is existing that could automate their construction for a particular application in a specific language. Automated knowledge acquisition, with the use of machine learning techniques, has recently been proposed as a promising solution to this and other similar problems in language engineering [8]. "Figure 1" shows the basic architecture of the NER system.

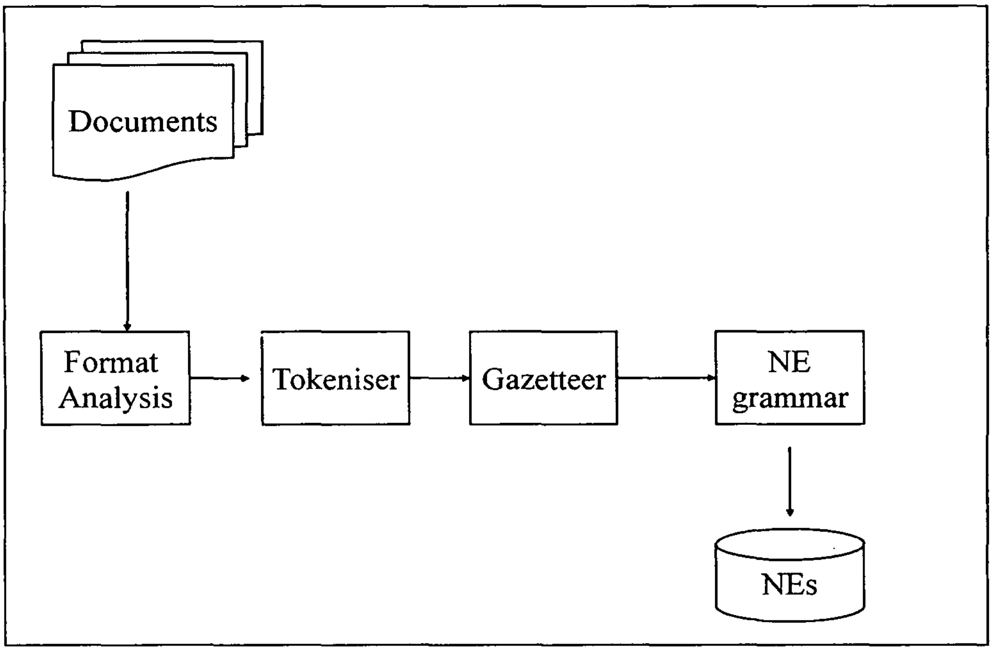


Figure 1: NE Basic System Architecture

Generally, the basic architecture of the NER system includes the following module:

- **Tokeniser**: It is a set of rules producing annotations. It segments text into tokens, e.g. words, numbers, punctuation.
- **Gazetteer lists:** It is finite set/lists of tokens/text use to classify words. Each list has attributes Major Type and Minor Type (and optionally language). Town, names, countries shows NEs and company designators, titles shows keyword.
  - NEs, e.g. towns, names, countries, ...
  - key words, e.g. company designators, titles, ...
- **Grammar**: For identification of NEs, there are different approaches to annotation. It uses annotations from format analysis, tokeniser and gazetteer modules. NE grammar uses contextual information and use the rule based on the priority, status and ordering.

  This basic architecture assumes that all the NEs can be identified by accessing the standard gazetteer and NER grammar, provided efficient search techniques are available. In real life this approach is not feasible. Firstly because in most of the cases it is impossible to add all the possible NEs to the gazetteer and provide efficient search techniques to search in the gazetteer. Secondly, new NEs are generated so a static list is not sufficient. NEs are domain dependent and language dependent, therefore the basic architecture of the NER system needs modification for better result.

1.4 Application of NER

Name Entity Recognition (NER) can be independent to domain or domain specific. It applies in different field:-

- Information Extraction System
- Text Mining
- Relation Extraction
- Robust handling of proper names essential for many applications
- Pre-processing for different classification levels
- Information filtering
- Information linking
- Question-Answering system
- Automatic summarization
- Social Network Analysis

1.5 Motivation for NER system in Hindi

There has been done a lot of work in NER for English and other European languages with the claims for high accuracy. There are full-fledge NER systems, which are available in English and other European languages, but such systems are not available in Indian Languages (IL) especially in Hindi. This is because the research in natural language processing (NLP) in Hindi is still at preliminary state. Even the proper resources like lexicons, labeled data to develop an NER system is not available. Therefore, developing an NER in Hindi is a very challenging problem and hence it provides many research issues.

This interest has been largely motivated by the relative tractibility of the problem and the potential marketability of an accurate named entity system. This marketability is driven by the many obvious benefits of having an accurate named entity system, including:

- More accurate internet search engines.
- Automatic indexing of books. For many books, the majority of the items which would go in the index would be named entities.
- General document organization. A user can call up all documents on a company intranet which mention about a particular individual.
- Before reading an article a user can see a list of the people, places, and companies mentioned in the document.
- A named entity tagger can serve as a preprocessing step to simplify tasks of machine translation. It is an essential component of complex information extraction tasks.
- Magazine could use this to highlight the names of every person mentioned in bold. The Wall Street Journal could do the same with companies.

The motivation for our "context pattern based approach" to design a NER system comes from problems, issues, lack of resources and challenges faced for NEs extraction (discussed in coming section).

1.5.1 Difficulties to develop an NER system

In Hindi, the process of compiling symbolic rules for rule-based IE/NER systems has been described as a costly and very tedious task [14]. These types of difficulties, pose more general questions of robustness and portability, which this research has tried to address, such as:

1. What would be needed when NER system have to be reengineered whenever new texts in different domains and languages are to be analyzed?
2. What would be needed if new NEs categories are set to be identified?
3. Is there any automatic method which can be followed to acquire rules for a rule-based NER system?
4. Can we use rules developed for a NER system in other related applications without major changes?

5. How can a NER system properly classify unconventional person names?
6. How can a NER system properly tokenize organization names?

This dissertation reports on the research approach which we adopted to answer these questions and reports on the results of the study.

1.5.2 Parameters to develop an NER System

Following points/parameters should be consider while building NER system:

- Ease to change
- Portability (Domains and Language)
- Scalability
- Language Resources
- Cost-effective

The above mention most of the parameters have been already achieved by the NER system developed in English but, still researcher/developers of Hindi NER systems are struggling to achieve these parameters.

1.6 Organization of the Dissertation

The dissertation is organized as follows. We present our literature survey and related work is in Chapter 2. In Chapter 3, we present a formal description of the problem, and describe challenges, issues, assumptions and our approach. In Chapter 4, we describe implementation of our approach, Experimental results and analysis is also described in this Chapter. Chapter 5 highlights our conclusions. Finally, we conclude by presenting ideas about our future efforts.

Chapter 2

Approaches for Named Entity Recognition

Named entities (NEs) are words which belong to certain categories like persons, places, organizations, numerical quantities, expressions of times etc. A large number of techniques have been developed to recognize named entities for different languages. These techniques may involve use of gazetteer with proper lexicons and grammar. However, gazetteer based approach is not sufficient to extract all the NEs. There are some Rule based and others are Statistical techniques. The rule based approaches use the morphological and contextual evidence [6] [7] of a natural language and consequently determines the named entities.

The statistical techniques use large annotated data to train model, like HMM [9] and subsequently examine it with the test data. Both the methods mentioned above required the efforts of language experts. Consequently, the application of the statistical techniques for Indian Languages is not very feasible [10]. This eventually leads to formation of some language specific rules for identifying named entities. A typical NER system involves the exploitation of a lexicon and a grammar, which need to be updated when the system is customized to a new domain. The lexicon is a set of gazetteer lists, containing names that are known beforehand and have been classified into named-entity types. The grammar is used to recognize named entities that are not in the gazetteer lists or they occur in more than one gazetteer list. Manual construction of these resources is a very time-consuming process and it is therefore worth examining methods that could automate their construction for a particular application in a specific language. Automated knowledge acquisition, with the use of machine learning techniques, has recently been proposed as a promising solution to this and other similar problems in language engineering [8]. Statistical NER systems typically require a large amount of manually annotated training data. An appropriately large set of annotated data is yet to be made available for the Indian Languages.

There are a variety of techniques for NER. We have broadly classified these approaches as shown in the figure2 [15] [47].The main approaches are:

- Handcrafted Approach
- Machine learning based approach
- Hybrid Approach

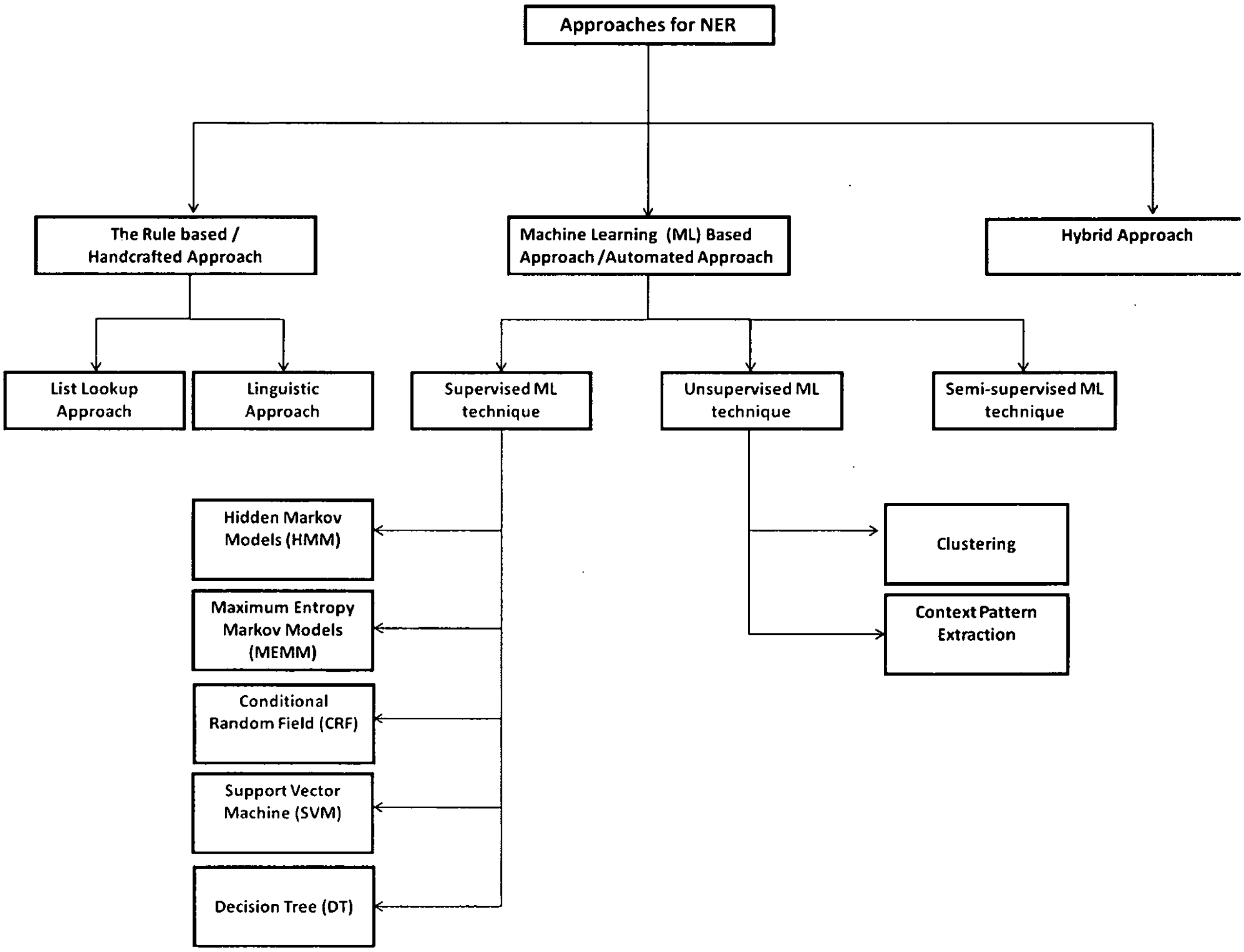

Approaches for NER
The Rule based / Handcrafted Approach
Machine Learning (ML) Based Approach /Automated Approach
Hybrid Approach
List Lookup Approach
Linguistic Approach
Supervised ML technique
Unsupervised ML technique
Semi-supervised ML technique
Hidden Markov Models (HMM)
Maximum Entropy Markov Models (MEMM)
Conditional Random Field (CRF)
Support Vector Machine (SVM)
Decision Tree (DT)
Clustering
Context Pattern Extraction

Figure 2: Approaches for NER

To increase the accuracy and coverage of the results comes from the above mentioned approach, few new approaches has also introduced. Like: **Multilingual Approach** [34].

2.1 Handcrafted Approach

- **2.1.1 Dictionary/Gazetteer Based Approach (List Lookup Approach):** NER system uses gazetteer to classify words. We just need to create a suitable list in gazetteer.

Advantages: It is simple, fast and language independent. It is also easy to retarget as we just have to create lists. It is useful for resource poor languages.

Disadvantages: Only works for lists in the gazetteer. We need to collect and maintain the gazetteer. It cannot resolve ambiguity and does not have learning capacity from the extraction of NEs. It also faces the searching issues if the dictionary is very large for large set of data. It is impossible to create generalized list for all the possible NEs. Therefore domain specific gazetteers are required for separate domains.

- **2.1.2 Linguistic Approach:** NER system uses some language based rules and other heuristic to classify words. The linguistic approach is the classical approach to NER. It typically uses rules manually written by linguists. Though it requires a lot of work by domain experts, a NER system based on manual rules may provide very high accuracy. There are several rule-based NER systems, containing mainly lexicalized grammar, gazetteer lists, and list of trigger words, which are capable of providing F-value of 88-92 for English [16], [17], [18]. It uses the transliteration techniques to identify the NEs. Here good knowledge of Hindi language is required.

Advantages: It needs rich and expressive rules and gives good results.

Disadvantages: The main disadvantages of these rule-based techniques are:

- They require huge experience and grammatical knowledge on the particular language or domain.
- It requires good rules and heuristic.
- The development is generally time-consuming and sometimes changes in the system may be hard to accommodate.
- Also, these systems are not transferable, which means that one rule-based NER system made for a particular language or domain cannot be used for other languages or domains.

2.2 Statistical and Machine Learning Approach (Automated Approach)

The Statistical and Machine Learning (ML) techniques acquire the high-level knowledge by the use of a large amount of annotated data. ML based techniques facilitate the development of recognizers in a very short time. Several ML techniques have been successfully used for the NER task. ML can be classified into three categories:

- **Supervised ML technique**: It generates a function that maps inputs to desired outputs (also called **labels**, because they are often provided by human experts labeling the training examples). For example, in a classification problem, the learner approximates a function mapping a vector into classes by looking at input-output examples of the function. The main approaches are Maximum Entropy (ME), Maximum Entropy Markov Model (MEMM), Conditional Random Field (CRF), and Support Vector Machine (SVM). Other than this, there are some other approaches also like Analytical learning, Decision tree learning, Nearest Neighbor Algorithm, Boosting, Back propagation and many more.

- **Unsupervised ML technique**: It refers to the problem of trying to find hidden structure in unlabeled data, like clustering and Relevant Document Selection etc.

- **Semi-supervised ML technique**: It combines both labeled and unlabeled examples to generate an appropriate function or classifier.

2.2.1 Supervised ML technique

These Supervised ML techniques are broadly used in the development of NER system. Here we mention a few ML techniques that can be used to develop NER. '**Identifinder**' is one of the first generation ML based NER systems which used Hidden Markov Model (HMM) [19]. Other ML approaches like Maximum Entropy (ME) [20], Maximum Entropy Markov Model (MEMM) [21], Conditional Random Field (CRF) [22], Support Vector Machine (SVM) [23] and Decision Tree (DT) [24] are the mostly used approaches. Let's have a look on statistical and machine learning approaches:

- **Hidden Markov Models (HMM):** It is a generative model. The model assigns a joint probability to paired observation and label sequence. Then the parameters are trained to maximize the joint likelihood of training sets.

$$P(X,Y)=\prod_i P(X_i,Y_i)\,P(Y_i,Y_{i-1})$$

It uses the forward-backward algorithm, Viterbi Algorithm and Estimation-Modification method for modeling. Example of a HMM based NER system is Nymble (1997) [30]. HMM is suitable for applications that require to process large amounts of text such as information retrieval or have real-time response. We can extend the original HMM to develop NER system. HMM is used back-off model to avoid sparse data problem in Entity recognition.

Advantages: Its basic theory is elegant and easy to understand. Hence it is easier to implement and analyze. It uses only positive data, so they can be easily scaled.

Disadvantages: In order to define joint probability over observation and label sequence HMM requires enumerate all possible observation sequence. Hence it makes various assumptions about data like Markovian assumption i.e. current label depends only on the previous label. Also, it is not practical to represent multiple overlapping features and long term dependencies in the development of NER system.

Solution: Maximum Entropy (ME) model, Conditional Random Field (CRF) or Support Vector Machine (SVM) are the solutions of the most of the problems which has been faced in the HMM approach. ME, CRF or SVM can make use of rich feature information for NER system.

* **Maximum Entropy Markov Models (MEMMs):** It is a conditional probabilistic sequence model; it can represent multiple features of a word and can handle long term dependency. It is based on the principle of maximum entropy (MaxEnt), which states that least biased model which considers all known facts is the one which maximizes entropy, where each source state has an exponential model that takes the observation features as input and output a distribution over possible next state. Output labels are associated with states.

    * **Maximum Entropy (MaxEnt):** MaxEnt is a flexible statistical model which assigns an output to each token based on its history and features. MaxEnt computes the probability $p\ (o|h)$ for any o from the space of all possible outputs O, and for every h from the space of all possible histories H. A history is all the conditioning data that enables output space from assign probabilities. In NER, history can be viewed as all information derivable from the training corpus relative to the current token w_i. The computation of $p(o|h)$ depends on a set of

features, which are helpful in making predictions regarding to the output. Given a set of features and a training corpus, the MaxEnt produces a model in which every feature f_i has a weight α_i. It can compute the conditional probability as [28].

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

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

The probability is given by multiplying the weights of active features. The weight α_i is estimated by a procedure called Generalized Iterative Scaling (GIS) [29]. This method iteratively improves the estimation of weights. The MaxEnt technique guarantees that, for every feature f_i , the expected value equals the empirical expectation in the training corpus. For the development of NER system, Java based MaxEnt toolkit [52] is freely available on internet. It gives the probability values of a word belonging to each class. That is, given a sequence of words, the probability of each class is obtained for each word. To find the most probable tag corresponding to each word of a sequence, we can choose the tag having the highest class conditional probability value. Sometimes this method results in inadmissible assignment for tags belonging to the sequences that never happen. To eliminate these inadmissible sequences, beam search algorithm can be used. This algorithm finds the most probable tag sequence from the class conditional probability values.

Advantages: A Maximum Entropy approach is a random process; it makes the distribution satisfy a given set of constraints, and making as few assumptions as possible. The constraints are specified as real-valued feature functions over the data points.

Disadvantages: The distribution of dataset should be as uniform as possible, which is difficult to achieve for free text dataset.

MEMM can be constructed using ME concepts. After the MaxEnt walkthrough, A MEMM consists of |o| conditional ME models po'(o|h) = p(o|h,o'), one for each o'. The model po'(o|h) estimates the probability of appearance of the label o immediately after the label o' in the context h. The probability of a whole label sequence o = o_1 o_2 o_3 ...... o_m, given the sentence h = h_1 h_2 h_3............. h_m, is the product

$$P(o \mid h) = P_0(o_1|h_1)\prod_{i=1}^{m-1} p_{\ i}(o_{i+1}|h_{i+1})$$

The best tagging can be found using Dynamic Programming similar to Vitterbi algorithm.

The model p_0 (o|h) used at the beginning of a sentence is separate.

Advantages: It solves the problem of multiple feature representation and long term dependency issue faced by HMM. Generally, it has increased recall and greater issue faced by HMM.

Disadvantages: It has Label Bias Problem. The probability transition leaving any given state must sum to one. So it is biased towards states with lower outgoing transitions. The state with single outgoing state transition will ignore all observations. To handle the Label Bias problem, the state-transition structure can be changed or it can start with fully connected model and let the training procedure decide a good structure.

* **Conditional Random Field (CRF):** It is a type of discriminative probabilistic model. It has all the advantages of MEMMs without the label bias problem. CRFs [22] are undirected graphical models (also known as random field) which is used to calculate the conditional probability of values on assigned output nodes given the values assigned to other assigned input nodes. A special case of CRF Lafferty et al.(2001) [22], correspond to conditionally-trained probabilistic finite state automata. Unlike English, for which the authors know many helpful lexical structures such as

capitalization patterns and word suffixes, the authors knew almost nothing about Hindi. However, given CRFs' tremendous freedom to include arbitrary features, and the ability of feature induction to automatically construct the most useful feature combinations, users of CRFs can simple provide a large menu of lexical feature tests consisting of anything they imagine might possibly be useful, and then let the training procedure automatically perform the feature engineering. The conditional probability of a state sequence s = ⟨s1, s2,... sT⟩ given an observation sequence o = ⟨o1, o2,... oT⟩ is calculated.

$$P_{\Lambda}(\mathrm{s}|\mathrm{o}) = \frac{1}{Z_{\mathrm{o}}} \exp\left(\sum_{t=1}^{T} \sum_{k} \lambda_k f_k(s_{t-1}, s_t, \mathrm{o}, t)\right),$$

where f_k (s_{t-1}, s_t, o, t) is a feature function whose weight λ_k is to be learned via training. Feature functions could ask arbitrary questions about the two consecutive states, any part of the observation sequence and the current position. Their values may range between −1 to +1, but typically they are binary. To make all conditional probabilities sum up to 1, calculation of normalization factor Z_o is must.

Advantages: It is a simple, customizable and use open source implementation. It has generally increased recall and greater precision then HMM. It has MEMs without the label bias problem. Unlike ME, CRF does not require careful feature selection in order to avoid overfitting. It has Freedom to include arbitrary features and also the ability of feature induction to automatically construct the most useful feature combinations. Conjunction of features (e.g., a conjunction feature might ask if the current word is in the person name list and then extword is an action verb 'कहा'(told)). It is very good to handle different types of data.

Disadvantages: It is Infeasible to incorporate all possible conjunction features due to overflow of memory. It is also a Language dependent methodology.

- **Support Vector Machine (SVM):** Support Vector Machines (SVMs), first introduced by Vapnik [23], is a famous supervised ML approach. SVMs are well-known for their good generalization performance and have been applied to many pattern recognition problems. In the field of NLP, SVMs are applied to text categorization, and are reported to have achieved high accuracy without falling into over-fitting even though with a large number of words taken as the features [31]. To solve a classification task by SVM, the task involves with training and testing data, which consists of some data instances. Each instance in the training set contains one "target value" (class labels, where class label 1 for positive and class label -1 for negative target value and several "attributes" (features). The goal of a supervised SVM classifier method is to produce a model which predicts target value of the attributes. For each SVM, there are two data set namely, training and testing, where the SVM used the training set to make a classifier model and classify testing data set based on this model with use of their features [15]. The TinySVM- 0.07 [54] classifier is the best, optimized toolkit among publicly available SVM toolkits.

Advantages: The SVM is good with the same kind of data. The SVMs have advantage over conventional statistical learning algorithms, from the following two aspects:

1) SVMs have high generalization performance independent of dimension of feature vectors. Conventional algorithms require careful feature selection, which is usually optimized heuristically, to avoid overfitting. So, it can more effectively handle the diverse, overlapping and morphologically complex Indian languages.
2) By introducing the *Kernel function*, SVMs can carry out their learning with all combinations of given non-linear features without increasing computational complexity.
3) It gives the best results where the data set is few, and with extended algorithms it can be used in multiclass problems.

Disadvantages: SVM Predict the classes depending upon the labeled word examples only. It cannot handle the NEs outside tokens. Predict the NEs based on feature information of words collected in a predefined window size only [23] [32].

- **Decision Tree (DT):** DT is a powerful and popular tool for classification and prediction [24]. Decision Tree is a classifier in the form of a tree structure where each node is either a leaf node-indicates the value of the target attributes(class) of expressions, or a decision node that specifies some text to be carried out on a single attribute value with one branch and sub-tree for each possible outcome of the text. It is an inductive approach to acquire knowledge on classification. The attractiveness of DT is due to the fact that in contrast to neural network, it presents rules. Rules can readily be expressed so that human can understand them or even directly use them in a database access language like SQL so that records failing into a particular category may be tree. The decision tree rebuilds the manual categorization of training documents by constructing well-defined true/false-queries in the form of a tree structure. In a decision tree structure, leaves represent the corresponding category of documents and branches represent conjunctions of features that lead to those categories. The well organized decision tree can easily classify a document by putting it in the root node of the tree and let it run through the query structure until it reaches a certain leaf, which represents the goal for the classification of the document (As shown in figure 3).

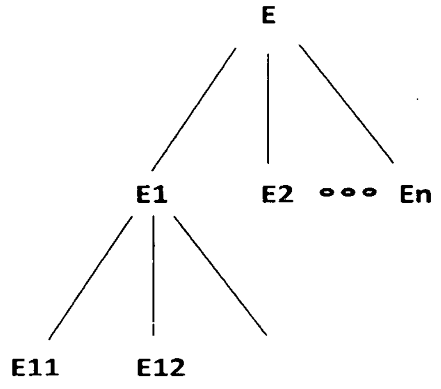


Figure 3: Decision Tree

Advantages: The decision tree classification method is outstanding with several advantages. The main advantage of decision tree is its simplicity in understanding and interpreting, even for non-expert users. Besides, the explanation of a given result can be easily replicated by using simple mathematics algorithms, and provide a consolidated view of the classification logic, which is a useful information for NE identifications.

Disadvantages: The major risk of implementing a decision tree is it over fits the training data with the occurrence of an alternative tree that categorizes the training data worse but would categorize the documents to be categorized better [46]. This is due to the classification algorithm of decision tree is made to categorize training data effectively, however neglect the performance of classifying other documents. Besides, huge and excessively complex structure of tree is built from a dataset with very large number of entries.

2.2.1 Unsupervised ML technique

Unsupervised learning is closely related to the problem of density estimation in statistics. However unsupervised learning also encompasses many other techniques that seek to summarize and explain key features of the data. Many methods employed in unsupervised learning are based on data mining methods used to preprocess data. On the basis of our survey, we can divide the unsupervised ML approaches in below categories:

- clustering
- Context Pattern extraction method

- **Clustering:** It organizes data instances into similarity groups, called clusters such that the data instances in the same cluster are similar to each other and data instances in different clusters are very different from each other [47]. There are a list of method available like k-means, mixture models, k-nearest neighbors, hierarchical clustering.

* **Context Pattern extraction method:** It is an unsupervised information extraction dispenses with hand-tagged training data. Because unsupervised extraction systems do not require human intervention, they can recursively discover new relations, attributes, and instances in a fully automated, scalable manner. This method is also a part of pattern recognition because it extracts the patterns from the raw corpus in some specific context. It is a very strong methodology for feature extraction and it doesn't need human intervention.

2.3 Hybrid Approach

Apart from handcrafted and machine learning approaches, NER also make use of the Hybrid model [33] which combines the strongest point from both the Rule based and statistical methods. This method is particularly used when complex Named Entities (NE) classes are used. Here, In Hybrid approach several Machine Learning and rule based system are combined together for more accuracy to identify the NEs in Hindi NER system. Some example of Hybrid systems are:

2.3.1 HMM approach and Rule Based approach

2.3.2 CRF approach and Rule Based approach

2.3.3 MEMM approach and Rule Based approach

2.2.4 SVM approach and Rule Based approach

There is some NER system in English which measures high value f- measure. These are:

* MAXEnt + Rule : Borthwick(1999) - 92% f-measure [20]

* MAXEnt + Rule : Endinburgh Univ. – 93.39% f-measure[15]

* MAXEnt + HMM + Rule : Srihari et al.(2000)- 93.5% f-measure [26]

2.4 Prior Work in Named Entity Recognition

This part includes the survey of existing work on Indian Language Hindi in NER. Named entity recognition is a classification problem which requires features of word at least in case of Hindi for proper identification of NEs. There are broadly two main approaches to NER are Linguistic approach and Machine Learning based approach.

There are several rule-based NER system, containing mainly lexicalized grammar, gazetteer lists and list of trigger words, which are capable of providing F-value of 88-92 for English.[4][5][6].

To overcome from the disadvantages of rule-based techniques Machine Learning based approach has been introduced. Machine Learning methods such as Hidden Markov Model (HMM) [19], Maximum Entropy (ME) [20], Maximum Entropy Markov Model (MEMM)[21], Conditional Random Field (CRF) [22], Support Vector Machine (SVM) [23] and Decision Tree (DT) [24] are the mostly used approaches. Apart from these two approaches, NER also make use of the Hybrid model[33] which combines the strongest point from both the Rule based and statistical methods. This method is particularly used when data is less and complex Named Entities (NE) classes are used. Sirhari et.al [26] introduce another Hybrid system by combination of HMM, ME and handcrafted grammatical rules to build an NER system and achieved F-value of 93.5.

Li and McCallum(2003) developed the Hindi NER using CRF with feature induction. They discover relevant features by providing a large array of lexical test and also used the feature induction to construct the features that increases the conditional likelihood. The Combination of Gaussian prior and early-stopping based on the results of 10-fold cross validation is used to reduce over fitting. The highest test set accuracy of their system, is the F-value of 71.50 [33].

Saha et.al(2008) describes the development of NER using ME approach for Hindi Language. The training data consists 234 k words, which is collected from the newspaper "Dainik Jagaran" and is tagged manually with 17 classes including one class for not name which consists of 16,482 NEs. The paper also shows the development of a module for semi-automatic learning of context pattern. The system was evaluated using a blind test corpus of 25K words having 4 classes and achieved the F-value of 81.52 [35].

Goyal (2008) build a NER for Hindi using CRF. This method was evaluated on test set1 and test set 2 and attains a maximum F1-value around 49.2% and nested F1-value around 50.1% for test set1 maximum F1-value around 44.97% and nested F1-value around 43.70% for test set2 and F-value of 58.85% on development set [36].

Saha et.al (2008) has identified relevant features for Hindi NER task that have been used to develop an ME based NER system for Hindi. Two-phase transliteration methodology was used. In this transliteration techniques the English lists was useful for the Hindi NER task. The system showed a considerable performance after using the transliteration based gazetteer lists. This transliteration approach is also applied to Bengali besides Hindi NER task and is found to be effective. The highest F-value achieved by ME based system is 75.89% which is then raised 81.2% by using the transliteration based gazetteer list [37].

Gupta and Arora (2009) describe the observation made from the experiment conducted on CRF model for developing Hindi NER. It shows some features which makes the development of NER system complex. It also describes the different approaches for NER. The data used for the training of the model was taken from Tourism domain and it is manually tagged in IOB format [38].

In the field of IR, patterns play an important role in identifying relevant pieces of information. Soderland et al.(1995)[39], Rillof and Jones(1999)[40], Lin et al.(2003)[25], Downey et al.(2004)[41], Etzioni et al.(2005)[42] described different approaches for context pattern extraction. Talukder et al. (2006) [43] combined grammatical and statistical techniques to create high accuracy patterns specific for NEs extraction. An approach to lexical pattern learning for Indian languages is described by Ekbal and Bandopadhyay (2007) [44]. They used seed data and annotated corpus to find patterns for NER. These context pattern based method is easy to use, can be use to generate a large gazetteer and work very efficiently for domain specific corpus.

2.5 Conclusion

In this Chapter, we briefly reviewed different approaches for Named Entity Recognition system and also have a look on the literature survey for the NER task, till so far in Hindi Language. The existing methods and models have tried to improve precision in recognition module and portability in recognition domain. As mentioned earlier, one of the most important problems and difficulties in NER is to change and switch data domain to new domain and that is called portability. In next chapter we will introduce our proposed methodology, where we have tried to reduce the portability issues with the help of suitable framework for domain specific corpus.

Chapter 3

Proposed Work: NER using Context Pattern

NER is an important component for many Natural Language Processing (NLP) applications like Information Extraction, Question Answering, and Machine Translation, Automatic Text summarization, indexing for information retrieval, Document classification and Natural Language Understanding. The scope of this work is to study in detail, the problems and challenges in developing NER in Hindi, study and compare approaches for extracting named entity and implement a context based NER in Hindi.

3.1 Challenges in Hindi NER

While significant amount of work has been done in English NER, with a good level of accuracy, work in Indian languages like Hindi has started to appear only very recently. Since the research in Hindi NER is still in preliminary level, it is very difficult to design and implement a full fledge NER in Hindi. In this work we suggest and develop an approach for identifying named entities using the context pattern. During this whole research we faced two major challenges: Challenges faced due to the properties/features of the Hindi language and Challenges due to the scarcity of proper resources.

3.1.1 Challenges faced due to the properties/features of the Hindi language

Following challenges occur due to the properties/features of the Hindi Language:

Capital letter indicator is missing: Capitalization is a major clue for most European Language. There is no concept of capitalization in Hindi language unlike English and other European languages where leading characters of names plays an important role in identifying NE's , either directly or indirectly [36].

* **Brahmi Script:** It has high phonetic characteristic which is difficult to utilize by NER system.
* Number of frequently use words (common nouns) which can also be used as proper names are very large. Also, common noun which used frequently against proper name is more or less unpredictable.
* **Morpheme Boundary** is difficult to detect: Hindi language is morphologically rich in nature and highly inflectional, which create difficulties to identify the NE's.
* Place names are frequently **homographic** with common words or with person names, presence of a number of exonyms (foreign language equivalences), endonyms (local variants) and historical variants for many place names etc.
* Lack of **standardization** and **spelling**.
* **Free word order** language : सचिन तेंदुलकर ने धौनी को or सचिन तेंदुलकर को धौनी ने |

Following examples clarify above issues more or less. Also, these examples show the difficulties and challenges due to the ambiguity in Hindi Language:

1. Common noun Vs proper noun- Sometimes, Common noun occurs as a person name such as "आकाश" which means sky, which is creating ambiguities between common noun and proper noun.
2. Variation of NEs: महेंद्र सिंह धौनी, कप्तान धौनी, म. सिंह धौनी, म. सिं. धौनी, माही ....

   "महेंद्र सिंह धौनी "is the person name but it can be found in various forms in text documents.
3. Person name Vs Organization – "महात्मा गांधी" used as a person name as well as an organization that creates ambiguity between proper noun and group indicative noun.

   महात्मा गांधी भारत के पिता हैं।(Person name)

   सचिन ने महात्मा गांधी ट्रस्ट में १००००/- सहयोग राशी दी । (Organization)

3. Location name Vs Organization – “इंडिया” (India) which act both as a location and organization.

विश्व कप–२०११ के सारे मैच इंडिया मे होंगे ।(Location)

इंडिया क्रिकेट-२०११ का विश्व चैंपियन बना ।(Organization)

3. Person name Vs Location name- The word “गंगा” is usually used as a person name and also as the name of a river i.e. as a location name.

गंगा बहुत सुशील है। (person name)

गंगा के किनारे बहुत बडा डेल्टा है।(Location name i.e. River name)

6. Person vs. Adverb: “संभव” occurs as a name of a person and sometimes it is used in sentence as adverb, which means “possible”. In the case of adverb it is well known that “संभव” cannot be a NE.

संभव गुप्ता आज दिल्ली आये| (Person name)

यह काम मुझसे संभव नही| (Adverb)

7. Organization vs. Noun: Here, “कॉलेज” word shows its uses in the form of organization as well as noun.

डीग्री कॉलेज का एक छात्र …..(Organization)

तबीयत खराब होने के कारण मै कॉलेज नही गया (Noun)

8. Location vs. Verb: “गया” is a verb, which means “gone” and also it is a place name.

गया में महात्मा बुद्ध को ग्यान प्राप्त हुआ (Location) वह जम्सेदपुर गया था (Verb)

3.1.2 Challenges due to the scarcity of proper resources

Following problems were faced due to the unavailability of proper resources:

* Limited availability of Unicode data (specially in Hindi)
* Scarcity of lexical **resources**: Unavailability of resources such as large gazetteer, Parts of speech (POS) tagger, good morphological analyzer etc. There is lacking in standardization, spelling of data also. Name lists are available in web which is in English but there is no list of names in Hindi language is available on web.
* Lack of **labeled data**
* Scarcity of **Data mining** and **Machine Learning tools.**

Apart from these challenges and difficulties, Hindi Language has some features also, which can help to identify NEs. We will discuss it in next section.

3.2 Features of Hindi NER Task

A text document and words in the documents are associated with some specific features. These features can be independent of the language or language dependent. In order to identify NEs, proper selection of features is very important. Some of the features that can be used for NER are listed below:

* **Surrounding Words:** Surrounding words are very important to recognize the NEs. As a feature, previous and next words of a particular word are used. The previous *m* words $[w_{i-m} \ldots w_{i-1}]$ of a particular word (can say it a placeholder) and next *n* words $[w_{i+1} \ldots w_{i+n}]$ of the same particular word (placeholder) can be treated depending on the training data size, total number of candidate features etc. During experiment different combinations of previous three words to next three words are used as features. These features are multivalued. For a particular word w_i , its previous word w_{i-1} can be any word in the vocabulary, which makes the feature space very high. Such high-dimensional features do not work well if amount of training data is not sufficient.

- **Binary Word Feature:** The binary feature is used to modify the multi-valued feature to reduce the feature space. Class specific lists are compiled taking the frequent words present in a particular position. For example, for the previous word of the "person name" class, frequent words are collected in "Previous Person" list. Such lists are compiled for each class and each position (previous *m* to next *n*). Now "C" *binary* features replace the word feature for a particular position, where "C" is the number of classes. The word in a particular position is checked that whether it is in the corresponding position list for specific class or not. List of words can be prepared, which has high frequency in a particular position corresponding to a class.

- **Context Lists:** Context words are defined as the frequent words present in a word window for a particular class. The idea of binary word feature is used to define the class context features. In our experiment we have listed all the frequent words present anywhere in context window [w_{i-3} w_{i-2} w_{i-1} **w** w_{i+1} $w_{i+2}w_{i+3}$] for a particular class. "**w**" is the placeholder word. Then this list is manually edited to prepare the context word list for a class. For example, location context list contains रोड (road), शहर(shahar), गढ़(garh), राजधानी (capital), स्थित (located in), जाकर (going to), विहार(vihar), बाद(bad), प्रदेश(pradesh), देश(desh), पुर(pur), पुरम(puram), नगर(nagar) etc. The feature is defined as, for a word w_i, if any of its surrounding words [w_{i-3} w_{i-2} w_{i-1} **w** w_{i+1} $w_{i+2}w_{i+3}$] (we set the window size as 7) is in a class context list then the corresponding class context feature is 1.

- **Named Entity Tags of the Previous Words:** Named entity (NE) tags of the previous words [$w_{i-m}...w_{i-1}$] are used as feature. This feature is dynamic and the value of the feature for w_i is available after obtaining the NE tag of w_{i-1}.

- **First Word:** If the word is the first word of a particular sentence, then this feature is set to 1. Otherwise, it is set to 0.

- * **Containing Digit:** If a word contains digit(s) then the binary feature is set to 1.

- * **Numerical Word:** If a word is a numerical word, i.e. it is a word denoting a number (e.g. सात (seven), आठ (eight), नौ (nine) etc.) then the binary feature is set to 1.

- * **Made up of 4 Digits :** If, in a word all the characters are digits and having only 4 digits in it, then the feature "fourDigit" is set to 1. This feature is helpful for identifying year. A little modification of the feature might give better result. As in our development, we are working in cricket domain; the years are limited to 1800-2100 in most cases (first cricket match held in 1844). Then we have modified the feature as if it is a four-digit word and its value is between 1800 and 2100 then the feature value is 1.

- * **Word Suffix:** Suffix information is very useful to identify the named entities. This feature can be used in two ways. The first one is that a fixed length word suffix of current and surrounding words and another is variable length word suffix. During evaluation, it was observed that this feature is useful and able to increase the accuracy by a considerable amount. Still, better approach is to use suffix based binary feature. Variable length suffixes of a word can be matched with predefined lists of useful suffixes for different classes of NEs. For location name, Suffix list is very useful since most of the location names in India end with a specific list of suffixes. Suffix list of locations contains 116 suffixes like रोड (road), गढ़(garh), विहार(vihar), बाद(bad), प्रदेश(pradesh), देश(desh), पुर(pur), पुरम(puram), नगर(nagar), गंज(ganj) etc.

- * **Word Prefix:** Prefix information of a word is also very useful. A fixed length word prefix of current and surrounding words can be treated as feature.

- * **Parts-of-Speech (POS) Information:** The POS of the current word and the surrounding words are important to recognize named entities like सचिन ने is a pattern where "ने"

generally comes with the entities name. There is very few number of POS tagger is available for Hindi Language and it is also not available easily. There is a coarse-grained POS tagger, which have only three tags - nominal, postpositional (PSP) and others. These coarse grained POS values of current and surrounding tokens are helpful for name recognition. Some binary features can be defined using the POS information.

* **Root Information of Word:** Indian languages are morphologically rich. Words are inflected in various forms depending on its number, person, case, tense etc. Identification of NEs becomes difficult due to these inflections. The task becomes easier if instead of checking the inflected words, corresponding root words could be checked whether these are NE or not [13].

3.3 Context Pattern and its Role in NE extraction

Our work is focused towards extracting named entity by identifying good context patterns. Our context pattern based methodology has been tested for domain specific corpus. This method is simple to use, does not require labeled example, for certain domains it may give good result and it can be used as a starting point for collecting NE for gazetteer preparation.

3.3.1 Context Pattern

A pattern is an arrangement or sequence regularly found in comparable objects or events. A context pattern is a pattern enclosing certain context. In our case, the context is determined by the seed[1] entity.

> " *Context Pattern is a non local form of patterns, which allows the matching of sub terms without fixed distance from the root of the whole term* " [45].

[1] Root word for specific context. Like "सचिन" is the person name in context of named entity.

Typical application of context patterns are functions which search a data structure and possibly transform it, especially functions which operate on programs as data objects. This concept can easily be adopted for any languages using pattern matching [45]. For Example, “SEED ने कहा की” pattern is an example of good pattern, where SEED indicate the root word. Here root word denotes a person name in context of NE. This pattern can extract names of the persons; those are coming before set of words “ने कहा की” in any set of documents.

3.3.2 Problem Formulation

This subsection formally introduces the problem of extracting named entities using context patterns. In our approach, after data collection and preprocessing of data, we start with identifying the NEs set to be extracted such as person name, place name, organization, date and time. For each NE set, some seed[2] entities are identified.

Seed Entities : Seeds are actually named entities, which manually identified at the beginning. These seed entity are expected to provide good pattern, which can be helpful in extracting new named entities of that type. With the help of seed entity all possible context string are generated.

Context String: Context string is a non local form of patterns, which encloses some specific root term (root terms are seed entities). In the place of root term, a placeholder string is used.

$$w_{i-m} \ldots w_{i-2}\, w_{i-1}\, \mathbf{p}\, w_{i+1}\, w_{i+2} \ldots w_{i+n}$$

In above context string example, “**p**” represent the placeholder.

Once context strings are identified, these strings can be used for extracting new NEs. This string allows us to extract new entities by searching context string in the corpus, where placeholders are the candidates for new entities.

[2] Named Entity, which is manually identified at the beginning of the experiments.

Context Pattern/ Good Context String: By analyzing the results returned from the context string, good context patterns is identified by using the evaluation metrics of IE i.e. precision and recall. Finally, only good patterns are retained. It is expected that these patterns can extract NE with high accuracy.

Evaluation Measures: In the field of information extraction, there are several types of accuracy measurements for evaluating the performance of the system. The most widely measurements are Precision, Recall, and F-measures; each one holds a specific characteristics. These measurements for information extraction define a correspondence between the extracted items and facts within the documents.

Precision:

It answers the question that, for every item in the extracted outcomes, if there is a corresponding fact in the documents.

$$\boxed{\textbf{Precision/Accuracy}=A/\ (A+B)}$$

Where A= number of extracted items matching the facts

B= number of wrong extracted items

Recall:

Recall corresponds to question that, for every fact in the documents, if there is a corresponding item shown in extracted outcomes.

$$\boxed{\textbf{Recall/Coverage} = A/\ (A+C)}$$

Where A is the same as above

C=number of facts failing to be extracted

F-measure:

Low recall can be fixed by increasing the redundancy of the corpus, and low precision can be improved by adding more constraints in the system processing loop. If someone is interested in taking care of both recall and precision, F-measure can be used:

$$\boxed{\textbf{F-measure}= ((1+\beta^2)\ (\textbf{Precision} * \textbf{Recall}))/\ (\beta^2 * \textbf{Precision} + \textbf{Recall})}$$

$$\boxed{\textbf{F-measure}= 2*P*R/\ (P+R)}$$

Where P=precision, R=recall-rate

β= 1 (Here value of β is taken as 1)

On the basis of recall, precision, F-measure algorithm 4 is used to find the best/good patterns from the list of context strings.

Overall approach of the problem is based on extracting NEs using context patterns. The problem requires a text corpus and entity sets to be identified. For each entity set some seed entities are identified. These seed entities are used to extract context strings, which are refined to provide good context patterns for extracting NEs. Finally, these patterns are used for identifying named entities. Overall approach of the proposed work is discussed in next section.

3.4 Proposed Work

The whole work process of our proposed Hindi-NER system is discussed as follows. In this work process data (text data corpus) is collected and then preprocessed the documents. After the preprocessing an index file needs to be generate. Entity set is identified for each category of NE to be extracted. For each entity set seed entities are selected. The Inverted index file is constructed for seed entities. This index file helps to reduce the search space of seed entity in the text documents. For each seed, context string is extracted from the set of text documents, with the help of algorithm2. Using algorithm3 good context strings/context patterns are extracted.

Algorithm for NER System using Context Pattern Extraction method

1. Collect and preprocess the text documents.
2. Generate the Index file
3. Identify entity set to be extracted from text documents.
4. Identify seed entities for each set.
5. For each entity set, perform steps 6 to 8
6. Use the context String Extraction Algorithm 2 to extract the pattern.
7. Extract NEs with the help of NEs extraction Algorithm 3.
8. Find good patterns with the help of "Good Pattern Selection Module" (Algorithm4).

Algorithm 1: Proposed Hindi-NER system's steps using context pattern

The implementation of this algorithm is explained in chapter 4.

NEs Extraction using Context string: Context strings are helpful for identifying NEs. As manual identification of context strings takes much manual labor and linguistic knowledge, we have developed a module for semi-automatically learning of context string. The summary of the context string learning module is given follows:

```
1. E -> Entity set, S-> SEED Entities for each class (obtained from algorithm1)
2. For each seed s € S, do following
        2.1 From the corpus find context strings (C -> collection of context strings) comprised of a placeholder e (s is used as e here) for the class instance, m tokens before e and n tokens after e [We have used m=3, n = 3]. This set of tokens form initial pattern.
                $w_{i-m} \ldots w_{i-2}\, w_{i-1}\, \mathbf{e}\, w_{i+1}\, w_{i+2} \ldots w_{i+n}$
            a) When m=0, then Suffix Pattern is extracted.
                $\mathbf{e}\, w_{i+1}\, w_{i+2} \ldots w_{i+n}$          (where n>=1)
            b) When n=0, then Prefix Pattern is extracted.
                $w_{i-m} \ldots w_{i-2}\, w_{i-1}\, \mathbf{e}$          (where m>=1)

3. Return C, collection of context String along with seed entities.
```

Algorithm 2: Context String Extraction Algorithm

"NE extraction algorithm 3" takes the context strings as an input that comes from the algorithm 2, with the help of these patterns/strings, words/token is extracted. The "NE extraction algorithm 3" is used to extract the NEs from a domain specific corpus. Good patterns are retained and can be used for further processing.

```
1. For each context string (found in algorithm 2), search the pattern in whole corpus.
   and extract the words/tokens, which are represented by placeholders.
2. From the list of words/tokens found in step1, identify the NEs using predefined
   list.
```

Algorithm 3: NE Extraction Algorithm

Good pattern extraction using Context string and NEs : The "NE extraction algorithm 3" is used to extract the NEs from a domain specific corpus. These NEs are added to the gazetteer and can be used for further processing. The context patterns or good context strings are extracted from using precision, recall and f-measures as discuss earlier in previous section 3.4.The algorithm4 calculates the precision, recall and f-measures and on the basis of good precision ,recall and f-measure value it returns the list of the good context strings. These context strings can be used for further processing and extracting of NEs from the same type of domain.

1. Find the coverage and precision/accuracy of context strings. The performance of the system has been evaluated in terms of the standard recall, precision and f-score parameters.

2. Discard the patterns/strings containing low precision.

3.Generalize the patterns by dropping one or more tokens to increase coverage.

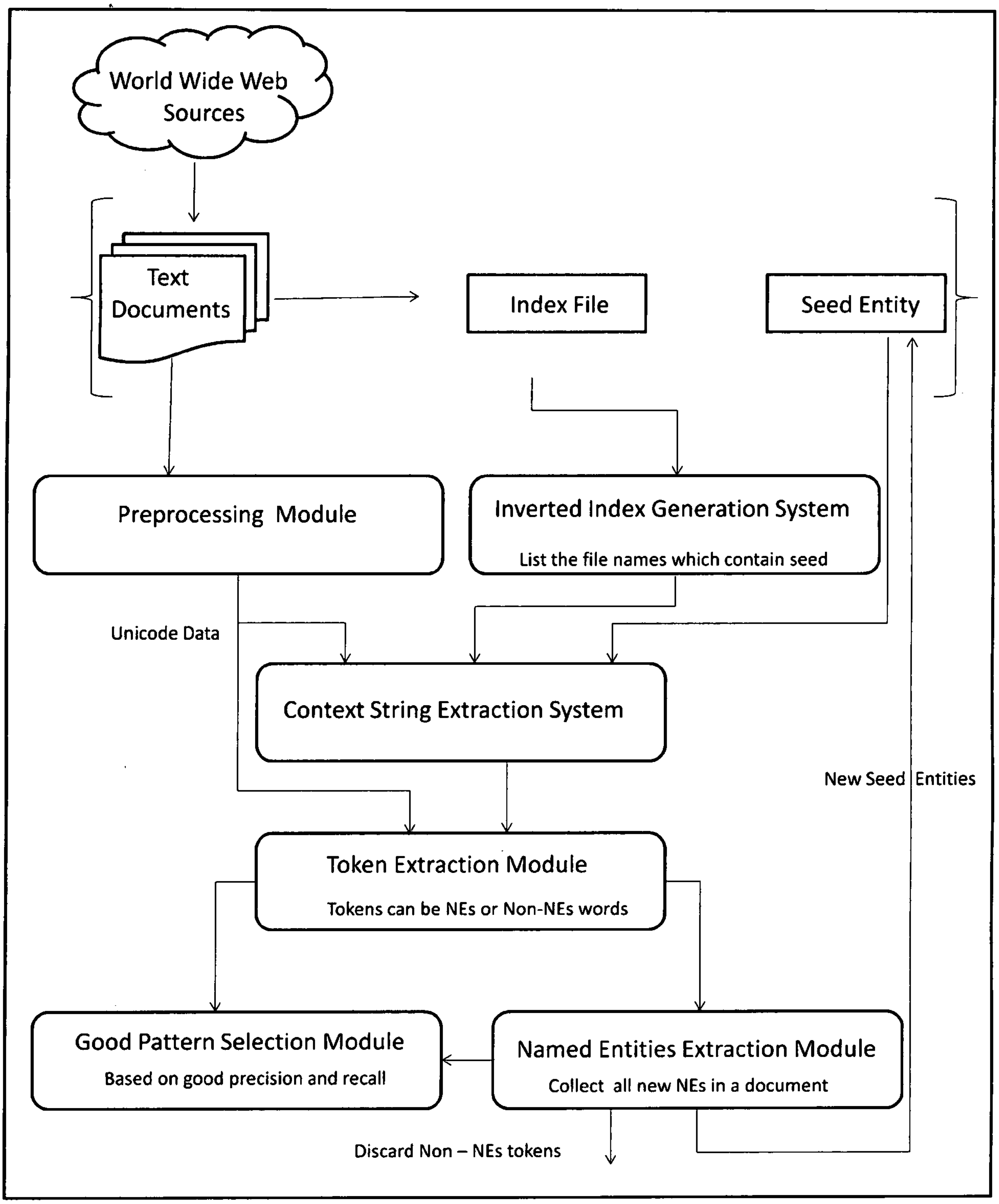
If a subset of a pattern is giving better result, then pattern is dropped and subset- pattern is used as a good pattern i.e. Selection of those patterns has to be done for which no subset of those is a good pattern. This generalization can also done on the basis of good prefixes and suffixes.

4. Find best patterns having good accuracy/precision and coverage.

Algorithm 4: Good Pattern Extraction Algorithm

3.5Architecture of The Proposed System

This section presents the architecture of "context pattern extraction methodology" for extracting the named entities (NEs). This method seeks the high precision context strings by using some seed entities and finds the patterns to the raw corpus and collects the NEs [40]. As manual identification of context patterns takes much manual labor and linguistic knowledge, automatic or semi-automatic modules need to be developed for learning of context pattern. The modules of the proposed architecture are shown in figure 4.


World Wide Web Sources
Text Documents
Index File
Seed Entity
Preprocessing Module
Inverted Index Generation System
List the file names which contain seed
Unicode Data
Context String Extraction System
New Seed Entities
Token Extraction Module
Tokens can be NEs or Non-NEs words
Good Pattern Selection Module
Based on good precision and recall
Named Entities Extraction Module
Collect all new NEs in a document
Discard Non – NEs tokens

The "World Wide Web" is the big source of information for dataset. The "n" numbers of text documents are collected from the "World Wide Web Sources". These "Text Documents" contain the Unicode text files. For each entity set, SEED entity is identified and "Index File" will be generated. The "Text Documents" is preprocessed through the "Preprocessed Module". Then, "Inverted Index Generation System" takes processed "Text Documents", "Index File" file and "SEED" as an input and generates the list of filenames which contain SEED entities. "Inverted Index Generation System" is used to reduce the search space in the whole raw corpus. "Context string Extraction System" takes the processed "Text Documents" (Unicode data), Inverted Index file (generated from the "Inverted Index Generation System") and "SEED" as an input and generates the Pattern list. These pattern lists contain good patterns, best patterns and few bad patterns also. "Token Extraction Module" takes the processed "Text Documents" (Unicode data), context string lists (collected from the "Context string Extraction System") as an input and extract the words/tokens that can be NEs. Then, "Named Entities Extraction Module" will select the named entities. "Good Pattern Selection Module" is proposed to extract the good pattern on the basis of recall and precision. These measures are discussed in detail in section 3.5. To generalize the system few new identified NEs can be used as seed and the process can be repeated iteratively. The implementation detail of this system is given in the next chapter "chapter 4".

3.6 Conclusion

This chapter presented the proposed work, which is based on extracting named entities by identifying good context patterns. We do not claim that it is a full proof methodology or very efficient. However, this approach is simple to use, does not require labeled example, for certain domains it may give good result and can be used as a starting point for collecting NE for gazetteer preparation. The following chapter 4 will show the implementation details of this methodology.

Chapter 4

Experiments & Result

Objectives: ***To extract the named entities with the help of semi-automatic NER system in Hindi using context pattern extraction method.***

The Experiments is divided into two parts:

1. **Context String Extraction (First experiments)**
2. **NEs Extraction (Second experiments)**

Input

1. The experiment has been performed on text documents from a specific domain. Domain Selected: Cricket Domain.
2. Text files for specified domain, one collected "Dainik Jagarn". Total number of text files is 609.
3. Select named entity set E corresponding to which entities are to be extracted. Entity sets has defined for each category i.e. **ENAMEX, NUMEX, TIMEX** category. Refer to Appendix A.

Table 1: List of the NEs set

Category	**Entity Set**
ENAMEX	**Person name, Location name, Organization name, Team name**
NUMEX	**Number of runs, Number of wickets**
TIMEX	**Year, Time**

4. **Selection of Seed Entity:** For each NE sets of ENAMEX, NUMEX and TIMEX category, seed values are identified manually (listed in Appendix A). For example, EAMEX contain Person name, Location name, Organization name, Team name as the NE set. Person name NE set takes 3 seed entities "सचिन तेंदुलकर, रिकी पोंटिंग, मुथैया मुरलीधरन". Some of the NEs set do not required the seed entity like "Year". These NEs set uses the regular expression to extract the NEs. The below table shows the list of seed which has been taken as input for the experiment.

Table 2: List of the seed entities with their NEs set

Entity Set	Seed NEs
Person name	सचिन तेंदुलकर, रिकी पोंटिंग, मुथैया मुरलीधरन
Location name	ईडन, नागपुर, मोहाली
Organization name	पीसीबी, बीसीसीआइ, आइसीसी
Team name	इंडिया, आस्ट्रेलिया, बांग्लादेश
Number of runs	No Seed
Number of wickets	No Seed
Year	No Seed
Time	No Seed

Output

As we discussed in the algorithm 1 that this whole experiment can divided into the two parts. First experiment for context string extraction and second is for NEs Extraction. The corresponding input and output is shown in table3 for each algorithm discussed in chapter 3.

Table 3: Output of the corresponding algorithm

Input	Algorithm number	Output
Text Documents, Seeds	Algorithm 2	List of Context String
List of Context String	Algorithm 3	List of NEs
List of Context String, List of NEs	Algorithm 4	Good context Patterns

Steps

Steps for the implementation of the NER using context pattern extraction based methodology, has been discussed already in algorithm1 in chapter3. The software's and packages use to implement these steps are listed in Appendix D & Appendix G.

4.1 Design and Implementation of Experiments

We divided our experiments on the basis of Architecture and **Algorithm 1** discussed in chapter3. In this chapter all the implementation and experiments details will be focused for the development of semi-automatic NER system in Hindi. The details of materials and methods used during the implementation of this NER system are listed in Appendix G. Here, we are going to discuss the implementation of context pattern extraction methodology to extract the good pattern and named entities for Hindi language. Steps are as follows:

I. **Data Collection and Preprocessing**
II. **NEs set selection and identify seed entities for each set**
III. **Context String Extraction**
IV. **NEs Extraction**
V. **Good Patterns selection**

I. Data Collection and Preprocessing: Our first requirement for analysis was a domain based corpus in Hindi. Cricket was chosen to be the domain for our purpose and relevant sports news were downloaded from popular Hindi newspaper "Dainik Jagaran" websites covering information about various matches played during year 2011(till April). The downloaded news were saved in text format in different files which formed our corpus[1]. For testing of our approach we have collected 192 text files, called testing data[2]. Our cricket domain corpus contains total 609 text file in unicode format, which contains about 1.5 lakhs words. Java API URLDataReader.java has been used to extract the data

[1] Collection of text documents with unicode data.

[2] Dataset which is used for testing of the applied methodology.

from the "Dainik Jagran" website. Then Preprocessing.java API is used to preprocess the raw corpus data by removing all the stop words listed in Appendix B and also, remove the duplicate words from the text file. These APIs are listed in listed in Appendix D. We have also worked on preparing the lists of cricket players name, list of team names which participated in the match played the year 2011 and organization name. This exhaustive task were use to identify NEs as a benchmark to identify the accuracy of the extracted NEs.

II. **NEs set selection and identify seed entities for each set:** The steps for the implementation of the NER system in Hindi has been discussed in details in **algorithm1**. Firstly, we identify the four entity sets to identify from the corpus we have collected. These NE sets are for ENAMEX, NUMEX and TIMEX category is listed in Appendix A. For example, ENAMEX contain Person name, Location name, Organization name, Team name as the NE set. Person name NE set contain 3 seed entities "सचिन तेंदुलकर, रिकी पोंटिंग, मुथैया मुरलीधरन". Appendix A is referring for reset of the NE sets with their seed entities.

III. **Context String Extraction (First experiments)**

To extract the patterns for a particular category, we select a part of the corpus where the target seed is available with high frequency. Inverted Index file, seed entities and raw corpus is required. Inverted Index file is generated using the Indri **Indexer UI 1.0 & Indri Retrieval UI 1.0.** Steps/Snap shots are shown in Appendix C.

For example to get the patterns for the names of the cricketers. The list of the most frequent words related to cricket has been prepared like, रन (run), बल्लेबाज (batsman), गेंदबाज (bowler) etc. In our development the cricket contains 1.5 lakh words. For a particular seed, we find the occurrences of the seed entity in the corresponding raw corpus. By the use of the context string extraction **Algorithm**

2, extract the context string. Few of the samples of patterns are listed in Appendix E. Table 4 shows the number of context strings extracted for each seed entities. Here, Java API PrintPattern.java uses to extract context string from the corpus.

Table 4: Number of context String extracted from

Seed NEs	No of Context Strings
सचिन तेंदुलकर	149
रिकी पोंटिंग	89
मुथैया मुरलीधरन	92
ईडन	84
नागपुर	95
मोहाली	23
पीसीबी	12
बीसीसीआइ	11
आइसीसी	10
2003	43
1994	34
2011	42
04:00	15
09:50	17
105	6
59	5
0	3
6	7
3	11
1	9

Then we extracted tokens enclosed within context string. A placeholder (SEED for cricketers) replaces the seed. The placeholder and the surrounding tokens p−3 p−2 p−1 placeholder p+1 p+2 p+3) forms the initial set of patterns. For the seed सचिन तेंदुलकर (Sachin Tendulkar) we extract 149 initial patterns. Context string list has been mentioned in the Appendix E. Some of which are:

- मास्टर ब्लास्टर SEED को
- बल्लेबाज SEED ने
- मेरे विचार से SEED को तक्कर देने

IV. NEs Extraction (Second Experiments)

From the list of context strings (discussed in previous section), tokens are extracted. TokenSelection.java API is used. It takes all kind of generalized patterns, suffix and prefix pattern as an input to extract token. **Algorithm 3** is for NEs extraction. For example, बल्लेबाज SEED ने is one of the context string extracted from "सचिन तेंदुलकर". The tokens extracted from this pattern are like: वकार यूनुस, गैरी किर्सटेन, गौतम गंभीर, शाह्दि आफरीदी etc. List of the some tokens extracted are listed in Appendix F.

Table5 shows the few of the context patterns and number of entities (for person name NEs set only, listing of all patterns here is not possible) they extracted with their precision and recall value. NefromExtractedTokenFile.java is used to extract the NEs from the list of tokens extracted earlier. The entities can be taken as seeds to identify more entities.

Here is the details of variable used in Table5.

M: Total number of word values found
N: Total number of named entity found
R:Total number of named entity in gazetteer

Table 5: Result of Gazetteer based Hindi NER

Patterns	M	N	R	% Accuracy (N/M*100)	% Coverage (N/R)*100	f-Score value
NE ???? ??? ????? ???	3	2	254	66.67	0.79	1.56
NE ?? ? ? ?	6	5	254	83.33	1.97	3.85
NE ??	278	242	254	87.05	95.28	90.98
? ????? NE	252	214	254	84.92	84.25	84.58
? ??? ? ??????? NE	150	123	254	82.00	48.43	60.89
?????? ? ??????? NE	2	2	254	100.00	0.79	1.56
?????? ???????? NE	10	1	254	10.00	0.39	0.76
NE ????? ? ? ??	198	167	254	84.34	65.75	73.89
? ??????? NE ??	267	249	254	93.26	98.03	95.59
???? ??? NE ??	204	199	254	97.55	78.35	86.90
NE ?? ????? ?????	251	223	254	88.84	87.80	88.32
NE ?? ????????? ???	143	134	254	93.71	52.76	67.51
NE ?? ????? ?????	1	1	254	100.00	0.39	0.78
NE ?? ?????	15	10	254	66.67	3.94	7.43
?????? ?? ?????? NE	16	1	254	6.25	0.39	0.74
????? ? ? ??????? NE	10	9	254	90.00	3.54	6.82
??????? ?? ? ? ??? NE	7	1	254	14.29	0.39	0.77
???? ?? NE ?? ???	4	4	254	100.00	1.57	3.10

V. Good Pattern Selection:

We measured the quality of a pattern depending on its precision and coverage. Precision is the ratio of correct identification and the total identification. Coverage/ recall is the ration of NEs extracted from the total NEs present in the dataset. In our case precision and coverage both have their own importance. There may be some patterns with 100% precision but very less (almost zero percent) coverage, such patterns are specific to a particular seed entity. Therefore these patterns are not expected to increase the coverage and thus are not expected to identify new entities. Hence theses patterns are not good context patterns even though they have very high precision. By this way we have prepared a list of good patterns for a particular gazetteer type. Result of selection of some good pattern is shown in Table5 for NE set "Person name". This result set contain long list of patterns, here it is contain only top most good patterns. Some mixed examples of good patterns are:

1. SEED

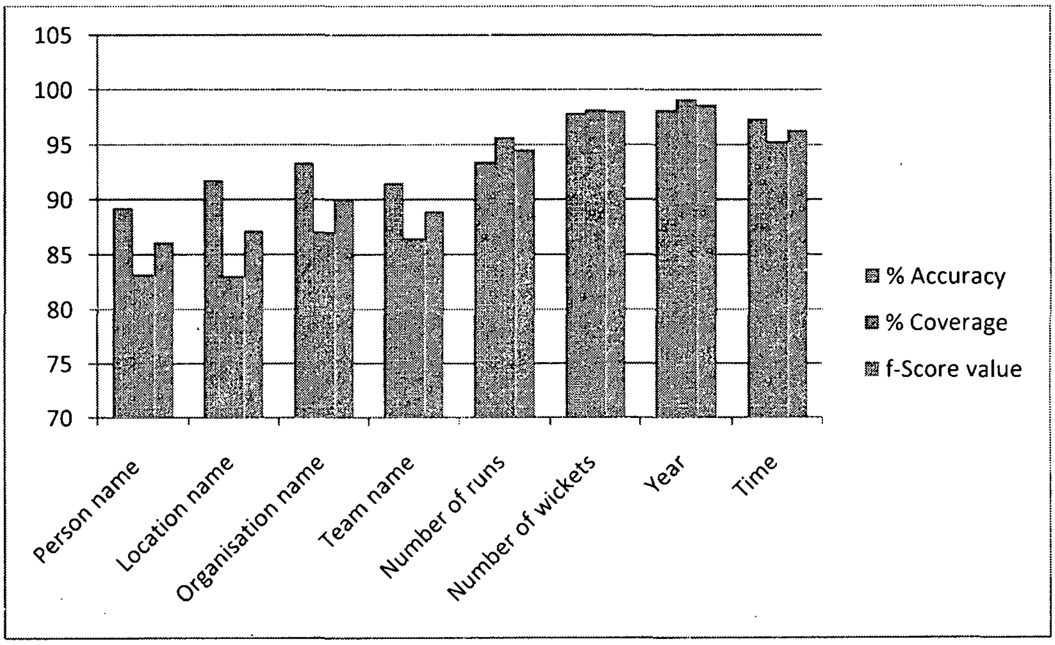
2. SEED

3. LOC-SEED

At lasts the combined result of each NEs set is shown in Table6 with its graph 1.

Table 6: Accuracy & coverage for each NEs set

NEs Set	**% Accuracy**	**% Coverage**	**f-Score value**
Person name	89.23	83.18	86.09885041
Location name	91.76	83.03	87.17698724
Organization name	93.34	87.07	90.09604567
Team name	91.5	86.5	88.92977528
Number of runs	93.43	95.67	94.53673295
Number of wickets	97.87	98.2	98.03472229
Year	98.12	99.12	98.61746502
Time	97.34	95.3	96.3091985

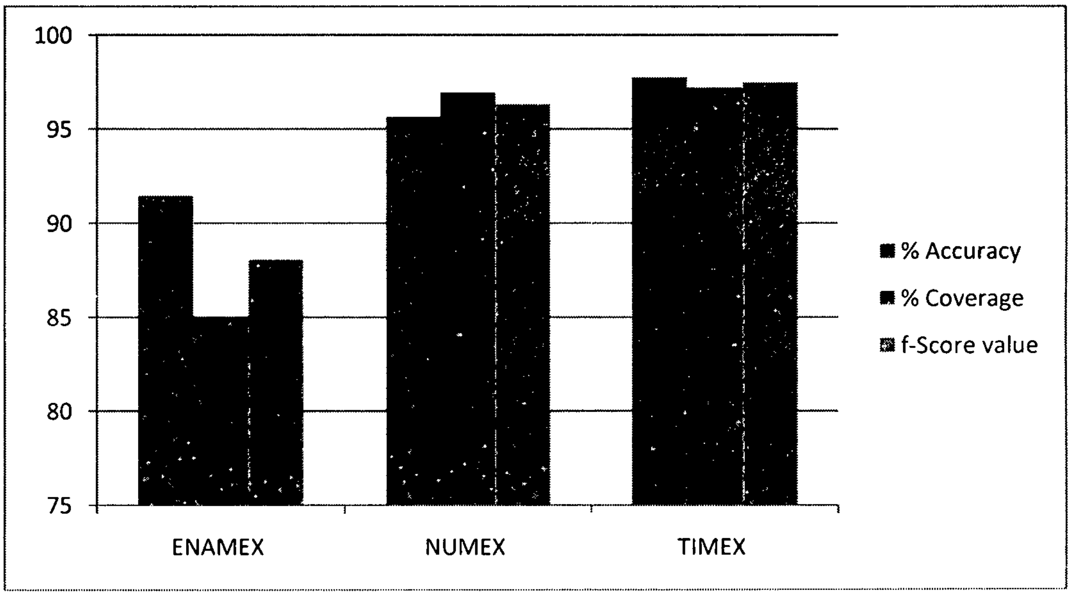


Graph 1: Accuracy & coverage for each NEs set

Final results for the each category of NER, has been also determined. It is shown in table7 and corresponding graph is in graph2.

Table 7: Accuracy & coverage for each NE category

NEs Category	% Accuracy	% Coverage	f-Score value
ENAMEX	91.4575	84.945	88.08103442
NUMEX	95.65	96.935	96.288213
TIMEX	97.73	97.21	97.46930645



Graph 2: Accuracy & coverage for each NE category

4.2 Analysis of Results

At the time of 'good' pattern selection we have made some interesting observations.

- There are some patterns which satisfy the 100% precision criteria but the coverage is very poor in terms of new entity extraction. For example, "   SEED   " is a pattern with 100% precision. The pattern has 24 instances in the 'part' corpus, but all the extracted entities are 'sachina tedulkara'. We have also examined the pattern in the total raw corpus. It is capable of extracting 'sachina tedulkara' (सचिन तेंदुलकर) only. So in spite of having high precision, this is not a 'good' pattern.

* Another interesting observation is, that there are some patterns which are 'good' patterns in the context of the 'part' corpus, but when used in the total raw corpus, it extracts non-relevant entities. For example "मेरे विचार से SEED को तक्कर देने" is a 'good' pattern for cricketer name. This is a good pattern, however if this is used in raw corpus it may not return the cricketer name only.
* There are patterns with very high coverage but precision is just below 100%. We have analyzed these patterns and manually identified the wrongly extracted entities. If the wrong entities can be grouped together and are having some specific properties then we have added these properties in a 'pattern exception list'. Then the pattern is used as a 'good' pattern and the exception list is used to detect the wrong identifications. In the following we have given some example of 'good' patterns
    * बल्लेबाज SEED को टीम के
    * बल्लेबाज SEED ने
    * SEED का अर्धसतक

* The extracted 'good' patterns are capable of identifying NEs from a raw corpus. The seeds form the initial gazetteer list for a particular gazetteer type. The 'good' patterns are used to extract entities from the total raw corpus. The entities identified by the patterns are added to the corresponding gazetteer list. In that way we can add more entities in our first phase gazetteer list. These new entities are taken as seeds for the next phase. Then the same procedure is followed repeatedly to develop a large gazetteer. These gazetteers are prepared just to prove the efficiency of our approach. By using only 3 seed entities we become able to prepare a gazetteer which contains 80 names of the cricketers. Even using this approach only one seed 'सचिन तेंदुलकर' extracts 42 names after the second iteration.

Chapter 5

Conclusion

Named Entity Recognition (NER) can be useful for any collection of text data sets, from which we want to extract specific entities. Explosion of information on web and need to extract information from web, is the prime motivation to develop an efficient NER system. The objective of the work was to propose a method for extracting NEs by extracting context patterns. Domain specific corpus was used for performing the experiments. The data set from cricket domain was used for the experiments.

Most of the NER approaches require the gazetteer and labeled data for NEs extraction and verification. In Hindi no gazetteers and labeled data are available as such. Therefore it is very difficult to design and implement a supervise learning technique for extracting NEs from Hindi text documents. Hence, unsupervised learning approaches may provide a starting base for extracting NEs from Hindi texts. In this context, context based approach may helpful in extracting NEs without any background requirement. Context pattern extraction method has a strong theoretical foundation. In this dissertation, we have tried to provide a general architecture for extracting named-entities from unstructured web pages (free text) using Context Pattern Method. The context pattern extraction method can be used for the preparation of standard gazetteer.

Finally we can conclude that this work is an attempt to explore a new emerging area of applying text mining and machine learning techniques to extract information from web pages containing information in Hindi. In this dissertation, where we focused towards the procedure of good context patterns extractions for NER. we do not claim that it is a full proof methodology or very efficient. However, this approach is simple to use, does not require labeled example , for certain domains it may give good result and can be used as a starting point for collecting NE for gazetteer preparation. The proposed methods and models supports the portability i.e. it can easily be ported to new domain.

At the time of development of this Hindi NER system we faced the problem of unavailability of resources and tools. There should be availability of standard corpus in different domain and also some tools are required to developed for Unicode encoding so that they can handle any language. Since statistical analysis fundamentally doesn't require the knowledge of grammatical rules, usage patterns, script used etc. so availability of such package/tools, which can handle all languages would be very beneficial for language analysis and research purposes.

Future work includes to anticipate Hindi NER using other techniques like DT, Genetic algorithm, Artificial and Neural Network, Fuzzy system etc that already showed an excellent performance in the other languages like English, Germany etc.

APPENDIX A

I. List of SEED entities for the different NEs set

For ENAMEX Category

1. **Person name** { सचिन तेंदुलकर, रिकी पोंटिंग, मुथैया मुरलीधरन }
2. **Location name** { ईडन, नागपुर, मोहाली }
3. **Organization name** { पीसीबी, बीसीसीआइ, आइसीसी}
4. **Team name** { इंडिया, आस्ट्रेलिया, बांग्लादेश }

For NUMEX Category

1.Number of runs { No seed}
2.Number of wickets { No Seed}

For TIMEX Category

1.Year { No seed}
2.Time { No seed}

II. List of frequent words

बल्लेबाज, विश्व कप, टीम, ऑलराउंडर, स्पिन, स्पिनर, कप्तान, रन, विकेट, आक्रामक, पिच, गेंदबाज, स्टेडियम, गेंद, बैट,

III. List of synonyms of some NEs

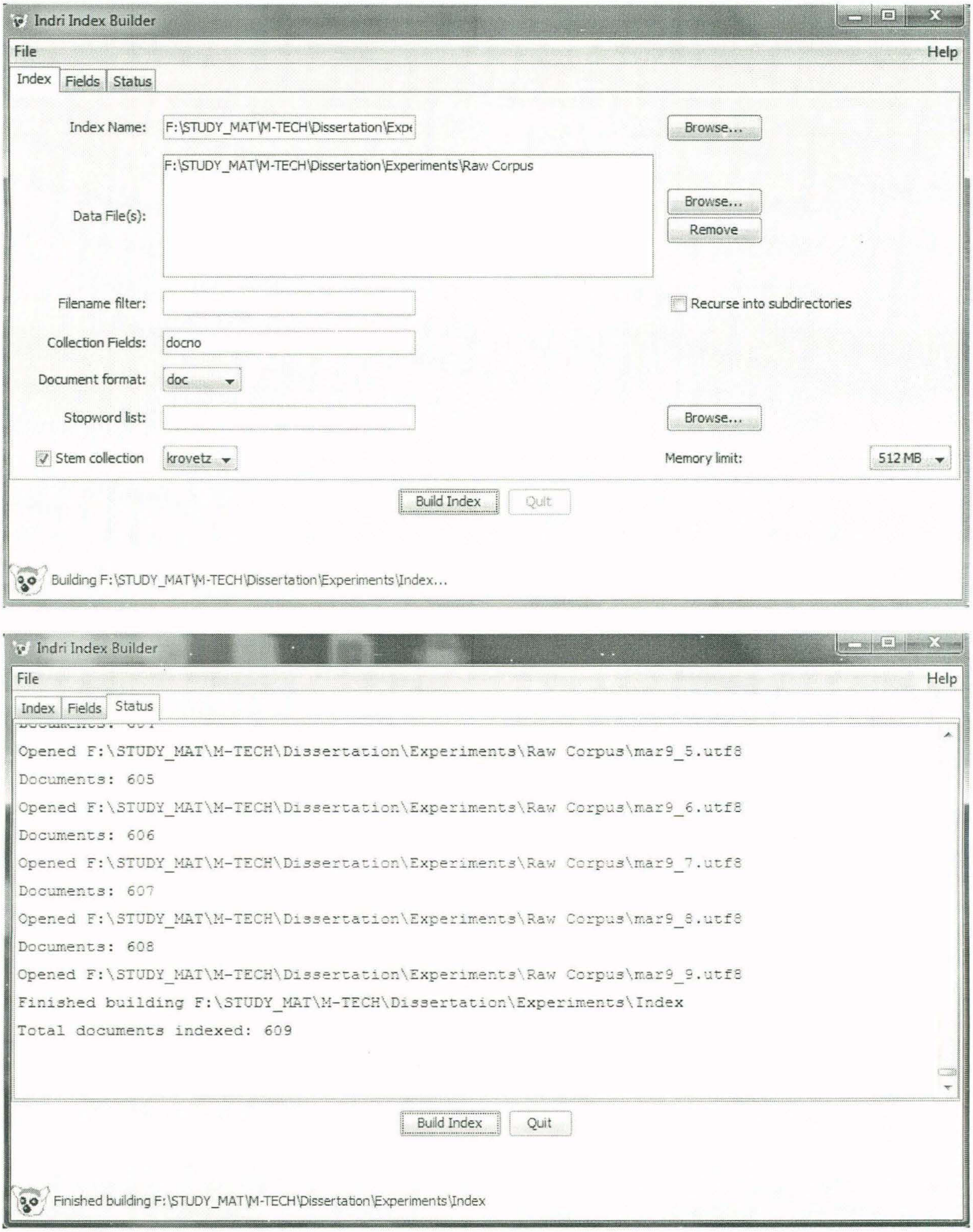
NEs	Synonyms
न्यूजीलैंड	कीवी
सेहवाग	वीरु
सचिन तेंदुलकर	मास्टर-ब्लास्टर, तेंदुलकर, सचिन
हरभजन सिंह	भज्जी
धौनी	महेंद्र सिंह धौनी, म. सिंह धौनी, म. सिं. धौनी,
अफ़्रीका	प्रोटियास
इंडिया	भारत

APPENDIX B

List of Stop-Words used during the experiment

1. (
2.)
3. {
4. }
5. :
6. @
7. #
8. *
9. ?
10. <
11. >
12. ,
13. .
14. |

APPENDIX C



Index Name: F:\STUDY_MAT\M-TECH\Dissertation\Expe

Data File(s): F:\STUDY_MAT\M-TECH\Dissertation\Experiments\Raw Corpus

Filename filter:

☐ Recurse into subdirectories

Collection Fields: docno

Document format: doc

Stopword list:

☑ Stem collection: krovetz

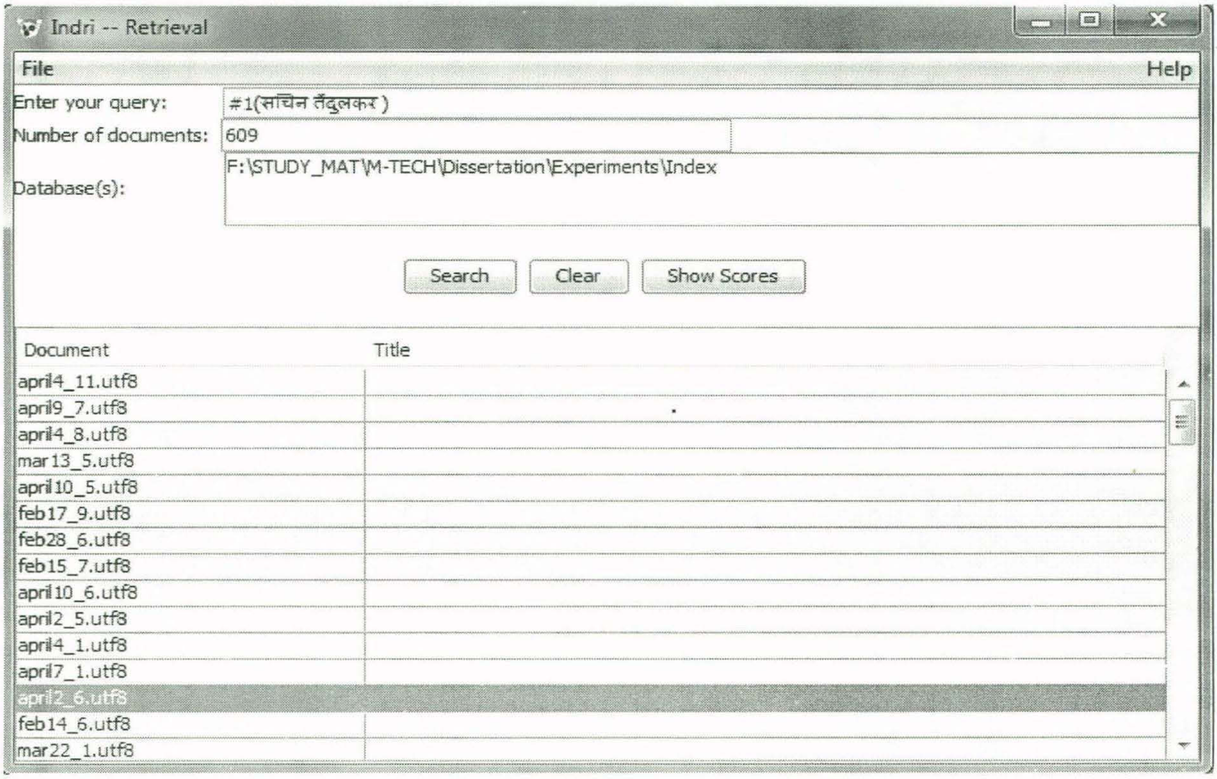
Memory limit: 512 MB

Building F:\STUDY_MAT\M-TECH\Dissertation\Experiments\Index...

```
Opened F:\STUDY_MAT\M-TECH\Dissertation\Experiments\Raw Corpus\mar9_5.utf8
Documents: 605
Opened F:\STUDY_MAT\M-TECH\Dissertation\Experiments\Raw Corpus\mar9_6.utf8
Documents: 606
Opened F:\STUDY_MAT\M-TECH\Dissertation\Experiments\Raw Corpus\mar9_7.utf8
Documents: 607
Opened F:\STUDY_MAT\M-TECH\Dissertation\Experiments\Raw Corpus\mar9_8.utf8
Documents: 608
Opened F:\STUDY_MAT\M-TECH\Dissertation\Experiments\Raw Corpus\mar9_9.utf8
Finished building F:\STUDY_MAT\M-TECH\Dissertation\Experiments\Index
Total documents indexed: 609
```

Finished building F:\STUDY_MAT\M-TECH\Dissertation\Experiments\Index

Figure 4.1 :Screen shot of the Index generation using Indri Indexer UI 1.0



Indri -- Retrieval

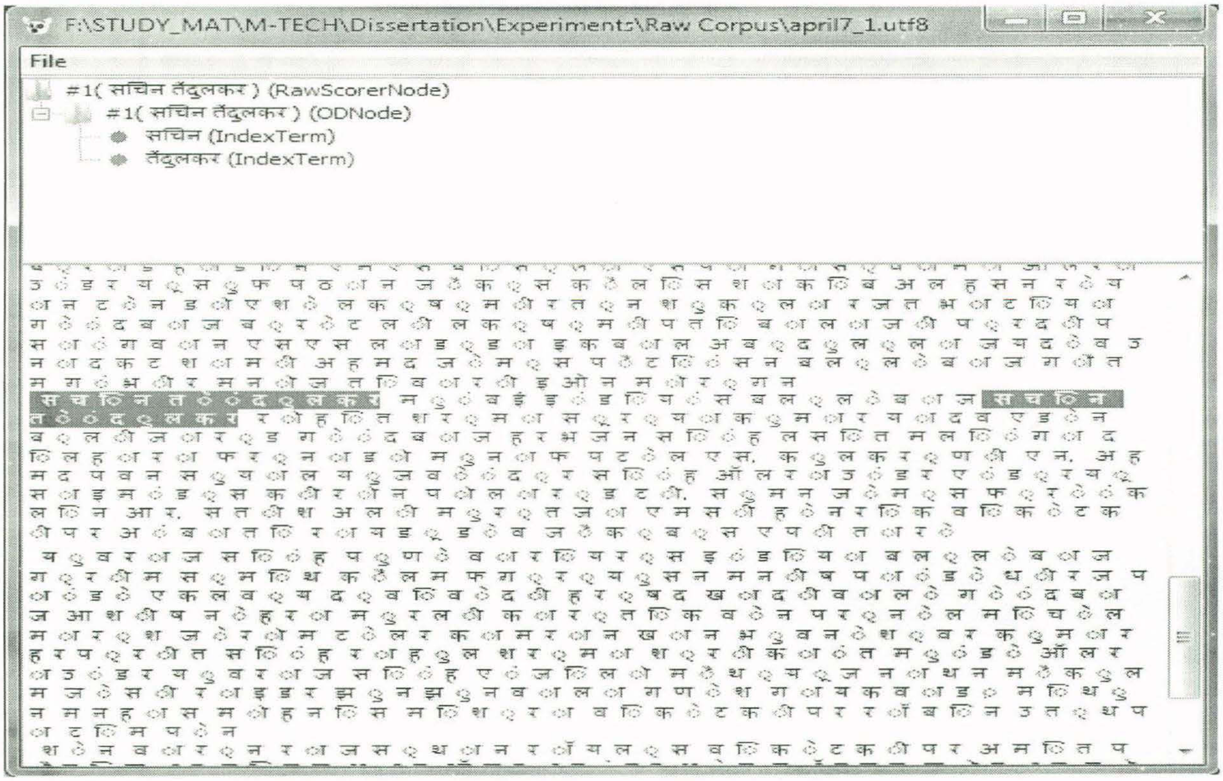
File Help

Enter your query: #1(सचिन तेंदुलकर)

Number of documents: 609

Database(s): F:\STUDY_MAT\M-TECH\Dissertation\Experiments\Index

Document	Title
april4_11.utf8	
april9_7.utf8	
april4_8.utf8	
mar13_5.utf8	
april10_5.utf8	
feb17_9.utf8	
feb28_6.utf8	
feb15_7.utf8	
april10_6.utf8	
april2_5.utf8	
april4_1.utf8	
april7_1.utf8	
april2_6.utf8	
feb14_6.utf8	
mar22_1.utf8	



F:\STUDY_MAT\M-TECH\Dissertation\Experiments\Raw Corpus\april7_1.utf8

File

```
#1( सचिन तेंदुलकर ) (RawScorerNode)
  #1( सचिन तेंदुलकर ) (ODNode)
    सचिन (IndexTerm)
    तेंदुलकर (IndexTerm)
```

[illegible]

Figure 4.2 : Inverted Index generation using Indri Retrival UI 1.0

APPENDIX D

I. List of Java APIs developed for Hindi NER system

S. No.	Java API Name	Application
1	URLDataReader.java	UTF8 Data Extraction from the Web and save it to UTF8 format file.
2	PreProcessing.java	Preprocess the raw corpus data by removing all the stop words listed in Appendix B and also, Replace a line or word in a file which is redundant to the text file.
3	PrintPattern.java	Extract context patterns from the corpus with the help of input of n-gram seed, context pattern length, inverted index file and seed value.
4	TokenSelection.java	Extract the tokens with the help of context patterns extracted from javaAPI PrintPattern.java. Also, take the suffix and prefix pattern as an input to extract token. Notation Used: -1: suffix (seed/new tokens position is prefix in pattern window) 1: Prefix 0:ContextPattern via 1 word, 2 word or 3 word
5	NefromExtractedTokenFile.java	Extract the NEs from the list of tokens extracted from the java API TokenSelection.java

APPENDIX E

There is total 149 patterns only for seed "सचिन तेंदुलकर" for NE set person name, Listing of all patterns is not possible here, so the below patterns is only for context window size 7.

S. No.	CONTEXT PATTERN BY 3 WORD
1	कि टीम सचिन तेंदुलकर को विश्व कप
2	पास पहुंच जाएगी तेंदुलकर के आउट होने
3	अभ्यास नहीं किया तेंदुलकर हालांकि दो बार
4	चैंपियन बल्लेबाज सचिन तेंदुलकर ने बुधवार को
5	है अजहरुद्दीन और तेंदुलकर ने भारत के
6	नहीं दे सकता तेंदुलकर अपने आउट होने
7	का सामना किया तेंदुलकर ने पहली गेंद
8	एजेंसी सचिन तेंदुलकर की क्षमता और
9	ही पड़ेगा सचिन तेंदुलकर रविवार को ऑस्ट्रेलिया
10	मास्टर ब्लास्टर सचिन तेंदुलकर एक स्थान फिसलकर
11	मिला जहां सचिन तेंदुलकर वीरेंद्र सहवाग और
12	भेजें प्रिंट संस्करण तेंदुलकर सबसे पारंगत विश्व
13	भारत के सचिन तेंदुलकर और वीवीएस लक्ष्मण
14	थ्रो फेंका जिससे तेंदुलकर 28 रन पर
15	मास्टर ब्लास्टर सचिन तेंदुलकर वीरेंद्र सहवाग और
16	शतक जड़ चुके तेंदुलकर ने डेढ़ घंटे
17	गए मैच में तेंदुलकर ने अपनी इस
18	इसे पिछले साल तेंदुलकर ने ही वनडे
19	उन्होंने कहा कि तेंदुलकर टूर्नामेंट के स्टार
20	पेश करते हुए तेंदुलकर ने चौथे ओवर
21	हुए सहवाग और तेंदुलकर ने भारत को
22	ज्यादा रन जुटाए तेंदुलकर इसलिए सफल नहीं
23	चेपक स्टेडियम सचिन तेंदुलकर के महाशतक के
24	ट्रेनिंग ही की तेंदुलकर और नेहरा दोनों
25	करनी है कि तेंदुलकर मंगलवार को दोपहर

APPENDIX F

I. List of the some Tokens extracted from the seed “सचिन तेंदुलकर” :

रत्नाकर शेट्टी
सकें शेट्टी
एंडी फ्लावर
द सन
मुताबिक फ्लावर
पर बेदी
अजित वाडेकर
मधुर भंडारकर
संजय मांजरेकर
चाहृ वाले
वकार यूनुस
एजाज बट
अनुभव धौनी
एकादश माही
अहृम धौनी
भारतीय कप्तान
लेकिन धौनी
लालचंद राजपूत
और इमरान
शाहिद आफरीदी
क्रिकेट टीम
कि हरभजन
तथा खिलाड़ियों
और गंभीर
सुरेश रैना
गैरी किर्सटेन
मुंबई इंडियंस
रिकी पोंटिंग
मेजबान श्रीलंका

APPENDIX G

Materials & Methods: This section contains overview of the software's which are used to perform our experiments and the basic data requirements needed for empirical evidences to check the validity of result of our context pattern extraction based approach for Hindi NER system.

1 Notepad++: Notepad++ is a text editor and source code editor for Windows. It can also run on Linux and Mac OS X, using software such as Wine. We have used notepad++ v5.8.7, which is a unicode supporter. It can be downloaded from notepad website [56].

2 Indri: Indri[57] is an academic information retrieval system written in C++. It Supports UTF-8 encoded text, A java user interface contain two modules; first is **Indri Indexer UI** and second is **Indri Retrieval UI**.

2.1 Indri Indexer UI 1.0 :Here, we just need to give the index filename and the dataset path(corpus) to build the index. Appendix C "Figure 4.1" is showing the process of indexing.

2.2 Indri Retrieval UI 1.0 : Here, we need to select the Index file which has been created by the Indri Index UI 1.0. Now we can retrieve the seed entity query to get the list of the files which could be used as inverted Index to reduce the searching space. "Figure 4.2" in Appendix C has the detail view.

3 Java APIs : There is a list of java APIs for different task. These APIs has been developed by me and is listed in the Appendix D. It can be downloaded from the website of java [58].

4 Quality Measure : The performance of the system has been evaluated in terms of the standard recall, precision and f-score parameters as defined below:

$$\text{Recall} = \frac{\text{NEs retrieved by the system}}{\text{NEs present in the test set}} \times 100$$

$$\text{Precision} = \frac{\text{NEs correctly retrieved by the system}}{\text{NEs retrieveed by the system}} \times 100$$

$$\text{F} - \text{score} = \frac{2 \times \text{Recall} \times \text{Precision}}{\text{Recall} + \text{Precision}} \times 100$$

We have given more importance to precision and we have marked a pattern as effective if the precision is more than 86%. This quality measures has been already discussed in details in previous chapter 3.

References

[1] R. Gaizauskas and Y. Wilks. “Information Extraction: Beyond Document Retrieval”, *Computational Linguistics and Chinese Language Processing* vol. 3, no. 2, pp. 17-60, August 1998.

[2] Lisa Ferro, Nancy Chinchor, Erica Brown and Patty Robinson, ”1999 Named Entity Recognition Task Definition”, 1999.

[3] Nancy Chinchor, “MUC-7 Named Entity Task Definition”, In *Proceedings of MUC-7*, 1997, 1997.

[4] G. Doddington, A. Mitchell, M. Przybocki, L. Ramshaw, S. Strassel, and R. Weischedel, ”The Automatic Content Extraction (ACE) Program–Tasks, Data, and Evaluation”, In *Proceedings of LREC 2004*, pp. 837–840, 2004.

[5] Erik F. Tjong Kim Sang and Fien De Meulder, “Introduction to the CoNLL-2003 Shared Task:Language-Independent Named Entity Recognition”, In *Walter Daelemans and Miles Osborne, editors, Proceedings of CoNLL-2003*, Edmonton, Canada, pp. 142–147, 2003.

[6] J. Kim & P. C. Woodland (2000a),” Rule Based Named Entity Recognition”, *Technical Report CUED/F-INFENG/TR.385*, Cambridge University Engineering Department, 2000.

[7] J. Kim & P. C. Woodland (2000b),”A Rule-based Named Entity Recognition System for Speech Input”, In *Proceedings of the International Conference on Spoken Language Processing*, pp. 521–524, 2000.

[8] V. Karkaletsis, G.Paliouras, G. Petasis, N. Manousopoulou and C. D. Spyropoulos,” Named-Entity Recognition from Greek and English Texts”, *Intelligent and Robotic Systems, 26*, pp. 123–135, 1999.

[9] Robert Malouf, “Markov models for language-independent named entity recognition”, In *Proceedings of CoNLL-2002 Taipei*, Taiwan, pp. 591-599, 2002.

[10] G.V.S.Raju, B.Srinivasu, S. Viswanadha Raju & Allam Balaram, ”Named Entity Recognition for Telugu Using Conditional Random Field”, In *proceeding International Journal of Computational Linguistics (IJCL), Volume (1): Issue (3), CSC Journals*, Kuala Lumpur, Malaysia, pp. 36-44, December 2010.

[11] Erik F. Tjong Kim Sang,” Introduction to the CoNLL-2002 Shared Task: Language-Independent Named Entity Recognition”, In *Proceedings of CoNLL-2002,* Taipei, Taiwan, 2002.

[12] Alireza Mansouri, Lilly Suriani Affendey, Ali Mamat," Named Entity Recognition Using a New Fuzzy Support Vector Machine",In *proceeding of IJCSNS International Journal of Computer Science and Network Security, VOL.8 No.2*, February 2008.

[13] Saha, S., S. Sarkar, and P. Mitra. 2008, "A Hybrid Approach for Named Entity Recognition in Indian Languages", In *Proceedings of the 3rd International Joint Conference in Natural Language Processing (IJCNLP 2008)*, pp. 17-24, 2008.

[14] Yangarber, R. and Grishman, "NYU: Description of the Proteus /PET system as used for MUC-7 ST" In *Proceedings of the Seventh Message Understanding Conference (MUC-7)*, Columbia, MD, 1998.

[15] B. Sasidhar, P. M. Yohan, Dr. A. Vinaya Babu, Dr. A. Govardhan, "A Survey on Named Entity Recognition in Indian Languages with particular reference to Telugu", In *Proceeding of IJCSI International Journal of Computer Science Issues, Vol. 8, Issue 2*, March 2011.

[16] Grishman R.," Where's the syntax? The New York University MUC-6 System", In *Proceedings of the Sixth Message Understanding Conference*, 1995.

[17] McDonald D.," Internal and external evidence in the identification and semantic categorization of proper names", In *B.Boguraev and J. Pustejovsky (eds), Corpus Processing for Lexical Acquisition*, pp. 21-39, 1996.

[18] Wakao T., Gaizauskas R. and Wilks Y.," Evaluation of an algorithm for the recognition and classification of proper names ", In *Proceedings of COLING-96*, 1996.

[19] B. D. M, M. Scott, S. Richard, and W. Ralph, "A High PerformanceLearning Name-finder," In *Proceedings of the fifth Conference on Applied Natural language Processing*, pp. 194–201, 1997.

[20] A. Borthwick,"A Maximum Entropy Approach to Named Entity Recognition", *Ph.D. thesis*, Computer Science Department, New York University, 1999.

[21] R. Sirhari, C. Nui, and W. Li, "A Hybrid Approach for Named Entity and Sub-Type Tagging," In *Proceedings of the sixth conference on Applied natural language processing, Acm Pp*, 2000.

[22] J. Lafferty, A. McCallum, and F. Pereira, "Probabilistic Models for Segmenting and Labelling Sequence Data," In *Proceedings of the Eighteenth International Conference on Machine Learning(ICML-2001)*, 2001.

[23] C. Cortes and V. N. Vapnik, "Support Vector Network , Machine Learning" , pp. 273–297, 1995.

[24] F. Bechet, A. Nasr, and F. Genet, “Tagging Unknown Proper Names using Decision Trees,” In *Proceedings of the 38th Annual Meeting of the Association for Computational Linguistic*, 2000.

[25] L. Winston, Y. Roman and G. Ralph, “Bootstrapped learning of semantic classes from positive and negative examples”, In *Proceedings of ICML-2003 Workshop on The Continuum from Labeled to Unlabeled Data*, 2003.

[26] R. Srihari,”A Hybrid Approach for Named Entity and Sub-type Tagging”, 2000.

[27] Kumar, N. and P. Bhattacharyya., ”Named Entity Recognition in Hindi using MEMM”, *Technical report, IIT Bombay*, India, 2006.

[28] S. D. Pietra, V. D. Pietra and J. Lafferty, ”Inducing features of random fields”, *IEEE Transactions on Pattern Analysis and Machine Intelligence,* pp. 19(4): 380-393, 1997.

[29] J. N. Darroch and D. Ratcliff,” Generalized iterative scaling for loglinear models”, *Annals of Mathematical Statistics,* pp. 43(5):1470-1480, 1972.

[30] D. M. Bikel, S. Miller, R Schwartz and R. Weischedel,” Nymble: A high performance learning name-finder”, In *Proceedings of the Fifth Conference on Applied Natural Language Processing,* pp. 194-201, 1997.

[31] H. Taira and M. Haruno, “Feature Selection in SVM Text Categorization”, *In Proceedings of AAAI-99,* 1999.

[32] A. Ekbal and S. Bandyopadhyay, “Named Entity Recognition using Support Vector Machine: A Language Independent Approach,” *International Journal of Computer, Systems Sciences and Engg(IJCSSE), vol. 4*, pp. 155–170, 2008.

[33] W. Li and A. McCallum, “Rapid Development of Hindi Named Entity Recognition using Conditional Random Fields and Feature Induction (Short Paper),” *ACM Transactions on Computational Logic*, pp. 290–294, Sept 2003.

[34] S. K. Saha, S. Narayan, S. Sarkar, P. Mitra, “A composite kernel for named entity recognition”, *Pattern Recognition Lett. (2010)*, doi:10.1016/ j.patrec.2010.05.004, 2010.

[35] S. K. Saha, S. Sarkar, and P. Mitra, “A Hybrid Feature Set based Maximum Entropy Hindi Named Entity Recognition,” in *Proceedings of the 3rd International Joint Conference on NLP,* Hyderabad, India, pp. 343–349, January 2008.

[36] A. Goyal, “Named Entity Recognition for South Asian Languages,” in *Proceedings of the IJCNLP-08 Workshop on NER for South and South- East Asian Languages*, Hyderabad, India, pp. 89–96, Jan 2008.

[37] S. K. Saha, P. S. Ghosh, S. Sarkar, and P. Mitra, "Named Entity Recognition in Hindi using Maximum Entropy and Transliteration," *Research journal on Computer Science and Computer Engineering with Applications*, pp. 33–41, 2008.

[38] P. K. Gupta and S. Arora, "An Approach for Named Entity Recognition System for Hindi: An Experimental Study," in *Proceedings of ASCNT- 2009, CDAC, Noida, India*, pp. 103–108, 2009.

[39] S. Stephen, F. David, A. Jonathan, L. Wendy, "CRYSTAL: Inducing a Conceptual Dictionary", In *Proceedings of the Fourteenth International Joint Conference on Artificial Intelligence*, 1995.

[40] E. Riloff, "Automatically Generating Extraction Patterns from Untagged Text", In *Proceedings of the Thirteenth National Conference,* 1996.

[41] D. Downey, O. Etzioni, S. Soderland, D.S. Weld," Learning text patterns for Web information extraction and assessment", In *AAAI-04 Workshop on Adaptive Text Extraction and Mining*, pp. 50–55, 2004.

[42] O. Etzioni, M. Cafarella, D. Downey, A. M. Popescu, T. Shaked, S. Soderland, D. S. Weld and A. Yates," Unsupervised named-entity extraction from the Web: An experimental study", In *Artificial Intelligence,* pp. 91-134, 2005.

[43] P. Pratim Talukdar, T. Brants, M. Liberman and F. Pereira.," A context pattern induction method for named entity extraction", In *Proceedings of the Tenth Conference on Computational Natural Language Learning (CoNLL-X)*, 2006.

[44] A. Ekbal and S. Bandyopadhyay," Lexical Pattern Learning from Corpus Data for Named Entity Recognition", In *Proceedings of International Conference on Natural Language Processing (ICON)*, 2007.

[45] M. Mohnen, "Context Pattern in Haskell", In W. Kluge et. Al.. editor, Lehrstuhl fur Informatik II, RWTH Aachen, Germany, Selected papers of the 8th *International Workshop on Implementation of Functional Language (IFL),* number 1268 in Lecture Notes in Computer
Science, Springer-Verlag, pp. 41-58,1997.

Books

[46] M. F. Moens, *Information extraction: algorithms and prospects in a retrieval context,* Springer, pp. 225-234, 2006.

[47] B. Liu, *Web Data Mining: Exploring Hyperlinks, Contents and Usage Data*, Springer, 2007.

[48] S. Chakrabarti, *Mining the Web: Discovering knowledge from Hypertext Data*, Springer, 2005

[49] M. Konchandy, *Text Mining Application Programming*, Charles River Media, Boston, Massachusetts, 2007.

World wide web sources

[50] Message Understanding Conference, http://www .nlpir.nist.gov/related_projects/muc

[51] http://www.webknox.com/blog/2010/09/named-entity-definition/

[52] http://www.maxent.sourceforge.net/

[53] http://chasen.org/~taku/software/yamcha/

[54] http://cl.aist-nara.ac.jp/ taku-ku/software/TinySVM

[55] http://www.cs.ualberta.ca/ ~aixplore/ learning/ DecisionTrees

[56] http://notepad-plus-plus.org/

[57] http://www.lemurproject.org/indri/

[58] http://www.java.com/en/download/