The Bilingual Named Entity Recognizer Framework in English and Hindi Language

Ruhi Mahajan[1] Ankita Koshti[2]
Department of Computer Engineering, MMCOE, University of Pune, India

Abstract—*Name Entity Recognition (NER) is one of the emerging field of Natural Language Processing (NLP) technology which can provide high value to various kinds of applications like Information Retrieval (IR), Question Answering (QA), Information Extraction (IE), text clustering etc. NER is basically used to identify proper names present in text and to classify them as different types of named entity such as people, locations, and organizations etc. In this research paper a framework is proposed for identifying the named entities like name, place, organization etc both for English and hindi language. The English language mixes case text so there is presence of some clues such as initial capitalized letters which clearly indicates the presence of name entities like name, place etc but Hindi language doesn't provide such clues, so it is very difficult to identify name entities in Hindi. In order to overcome such problem, we store corresponding Hindi text of English words in the database. To prove in the concept, the database MySQL5.5 is studied and used for this proposed framework*

Keywords: -Named Entity, Named Entity Recognition, Tag set.

I. Introduction

World Wide Web is a rapidly growing technology now a days and also a very important resource for fetching information from collections of huge amount of data with the help of imposing some queries and keywords. But there is problem of fetching information when the user wants the exact data because WWW contains more than one document related to a particular thing, person or incident etc. For instance, when we search for some data in the web with the help of some queries, we may get a lot of irrelevant as well as un-important data instead of getting the exact data or information what user actually wants. So, in order to fetch the exact information from large collection of documents what the user exactly wants there is great need of some methods or mechanisms. This gives rise to the Information Extraction Research. Information Extraction (IE) is a method which helps in extracting the required important or exact data. It is the process of fetching the required information from large collection of documents what the user actually wants. For this purpose, Information Extraction area includes sub tasks such as Named entity recognition, Relationship extraction. Named entity recognition is the process of extracting or identifying the named entities present in the given input text. This system finds named entities such as person, location, organization, number, date etc in the given text. Named entities are useful in many applications such as relation extraction, question-answering, etc. NER is basically used to recognize and classify information units in an unstructured text into pre-defined categories. Named entity refers to any object name in the real world but in the field of Information extraction, Named entities mainly refers to Person, Location, Organization names and Numerical entities like Time, Date and Number respectively. Named entity recognition (NER) is also known as entity identification and entity extraction system. It is the process of identifying and recognizing the named entities such as person names, organization names, location names and numeric data entities such as date, number etc. It is one of the important tasks in the research area of Information Extraction. Information Extraction (IE) is the process of extracting the relevant data from the available documents. NER is one of the important tasks in natural language processing area pertaining to Information Extraction, Question Answering, and Database Querying etc. Generally Named entity recognition tasks could find the seven types of entity names respectively. They are described as follows:

1. Person
2. Location
3. Organisation
4. Date
5. Time
6. Money
7. Percent

Among these entity names we are finding only date, number, person, location and organization names. Finding these entity names is very difficult when compared to the other entity names.

II. Proposed Work

In this research paper we are going to implement an English NER using some method , and a database. First we use Maximum Entropy Model; in which we tokenize every word from corpus, and then assign grammatical tag using POS tagger i.e we use a Stanford POS tagger API maxENT. Next we compare each noun in the above tagged data with the database consists of list of names. Then we assign each named entity identified to desired category of named entity with the tag of either person(<PERSON>),organization(<ORGANIZATION>)or location (<LOCATION>) respectively. In order to identify numbers or dates we used some already defined date patterns and pattern and matcher classes

A. Tokenization:

The first step of our NER system consists of breaking a stream of an input English text up into meaningful elements called tokens where each token is either a word or something else like a number or a punctuation mark. Therefore, we needfirst to keep the sentence boundaries where the sentence is something that ends with a full stop '.', a question mark'?' or an exclamation mark '!', since the system is only able to tag entities on a token-by-token basis.

B. Handling date and numbers:

This module is independent from database. In this module we handle all the patterns of date with the help of already defined Date Pattern Formats like dd/MM/yyyy, dd-MMM-yyyy, dd-MM-yyyy, dd.MM.yyyy which are defined in the java packages-

java.text.DateFormat,java.text.SimpleDateFormat.The java.text.DateFormat class, and its concrete subclass java.text.SimpleDateFormat, provide a convenient way to convert strings with date and/or time info to and from java.util.Date objects.DateFormat is an abstract class for date/time formatting subclasses formats and parses dates or time in a language-independent manner. Thedate/time formatting subclass, such as SimpleDateFormat, allows for formatting (i.e., date -> text), parsing (text -> date), and Normalization.

The numbers are handled with the help of java.util.regex.Pattern for pattern matching with regular expressions.A regular expression is a special sequence of characters that helps us match or find other strings or sets of strings, using a specialized syntax held in a pattern. They can be used to search, edit, or manipulate text and data.

The java.util.regex package primarily consists of the following three classes:

*1) Pattern Class:*A Pattern object is a compiled representation of a regular expression. The Pattern class provides no public constructors. To create a pattern, you must first invoke one of its public static compile methods, which will then return a Pattern object. These methods accept a regular expression as the first argument.

*2) Matcher Class:*A Matcher object is the engine that interprets the pattern and performs match operations against an input string. Like the Pattern class, Matcher defines no public constructors. You obtain a Matcher object by invoking the matcher method on a Pattern object. The matches () method in the Matcher class matches the regular expression against the whole text passed to the Pattern. matcher () method, when the Matcher was created.

*3) PatternSyntaxException:*PatternSyntaxException object is an unchecked exception that indicates a syntax error in a regular expression pattern.

C. HandlingPhrases:

In this module we handle all the phrases of name and organization i.e. if two or more than two consecutive words are initialized with capital letter and form a phrase then we combine all these words with " – " and then consider it as a single token.

E.g : Indian-Railways.

If an input text contains a word " Dr", " Miss" , "Mrs."., " Ms", " Prof", " Er", " Lect", " Mr" then the word follows it must be a person entity, so we combine one of the word that exists in an input text from all the words above with the next word with the help of "- "and tag it with a person entity and we also provide hindi transliteration of the hypenated word that is given by the transliterateclass.

E.g : Dr-Monika

If two more consecutive words are initialized with capital letter and form a phrase then we combine all these words with "-" and consider it as a single token and then we check if it ends with "Ltd","Limited","Corporation","Corp","University".If yes,we tag it with an organization entity and we also provide hindi transliteration of the hypenated word that is given by the transliterate class .

E.g : Abmtc-Limited

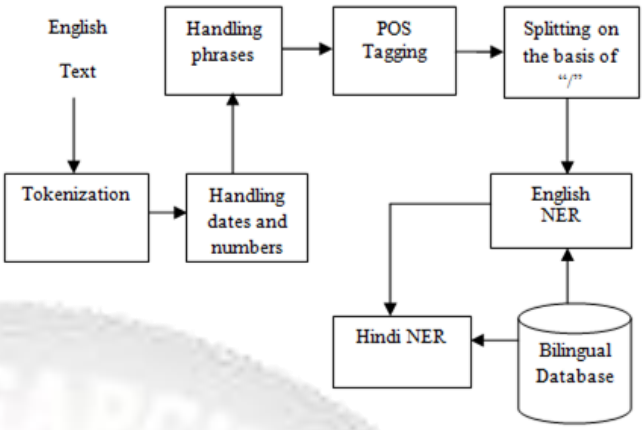

Fig. 1: Architecture of proposed NER system

D. POS Tagging:

Parts of speech tagger is used for identifying and recognizing the entity names. It determines each word in the document as noun, proper noun, verb etc. according to the grammar rules. For this we are using already existing tool Maximum entropy parts of speech tagger implemented by Stanford java NLP [1] . This POS tagger tags each word based on the PENN TREEBANK TAG SET.

Penn Treebank Tag Set: The PENN TREE BANK is a corpus consisting of over 4.5 million words of American English.A Treebank or parsed corpus is a text corpus in which each sentence has been parsed, i.e. annotated with syntactic structure. Syntactic structure is commonly represented as a tree structure; hence it is named as Treebank. The term Parsed Corpus is often used interchangeably with Treebank with the emphasis on the primacy of sentences rather than trees. Tree banks can be used in corpus linguistics for studying syntactic phenomena or in computational linguistics for training or testing parsers. PENN Tree Bank Tag Set consists of 36 POS tags and other punctuation tags respectively. Each tag has their significance related to a phrase according to the grammar rules. The description of each tag is

S.NO	TAG	DESCRIPTION
01	CC	Coordinating conjunction E.g. . . . and, but, or…
02	CD	Cardinal Number
03	DT	Determiner
04	EX	Existential there
05	FW	Foreign word
06	IN	Preposition or subordinating conjunction
07	JJ	Adjective
08	JJR	Adjective, comparative
09	JJS	Adjective, superlative
10	LS	List Item Marker
11	MD	Modale.g. Can, could, might, may...
12	NN	Noun, singular or mass
13	NNP	Proper Noun, singular
14	NNPS	Proper Noun, plural
15	NNS	Noun, plural
16	PDT	Predeterminer e.g. all, both ... when they precedes an article
17	POS	Possessive Ending e.g. Nouns ending in's
18	PRP	Personal pronoun e.g. I, me, you,

19 PRP$ Possessive Pronoun e.g. my, your,
20 RB Adverb. Most words that end in –ly as well as degree words
21 RBR Adverb, comparative. Adverbs with comparative ending –er,
22 RBS Adverb, superlative
23 RP Particle
24 SYM Symbol ,Should be used for mathematical or technical symbols
25 TO To
26 UH Interjection e.g. uh, well, yes, my…
27 VB Verb, base form
28 VBD Verb, past tense Includes the conditional form of the verb to be
29 VBG Verb, gerund or present participle
30 VBN Verb, past participle
31 VBP Verb, non-3rd person singular present
32 VBZ Verb, 3rd person singular present
33 WDT Wh-determiner which, and that when used as relative pronoun
34 WP Wh-pronoun e.g. what, who, whom
35 WP$ Possessive wh-pronoun
36 WRB Wh-adverb e.g. how, where why

S.No	TAG	PUNCTUATION
01	#	Dollar sign
02	$	Dollar sign
03	.	Sentence-final punctuation
04	,	Comma
05	:	Colon, semi-colon
06	(	Left bracket character
07	)	Right bracket character
08	“ ”	Left open double quote and Right close double quote
09	‘ ’	Left open single quote and Right close single quote

E. Splitting on the basis of “/”:

After getting the tagged output we split it on the basis of “/” with the help of already defined Split function in order to separate words from their respective tags so as to get each token separately.After getting each token separately we fetch the previous words of the tags only if its tag is NN or CD or JJ so as to work further on fetched words.

F. English Ner:

In this module we implement NER for English Language i.e. Identifies named entities like name, organization, location etc. After tagging and splitting we separate the nouns and with the help of database we compare each noun of an input string with the entity names present in the database and then finally fetched all the noun entities with their appropriate sub-entity i.e. name, place or organization. Here we created our own database. Database consists of collection of large number of entities of names,places, organization with columns words, entity (sub category) and its Hindi transliteration.

Steps:-

1. First of all we check if word contains tag “NN”.
2. If yes then we fetch its previous word and stored it into a database table “filled”.
3. Compare the entity stored in table filled with entities stored in another table“ambiguious” or “used”.
4. After that we check if corresponding to that word the previous word is “in or from or to ” then check the recent word in “ambiguious” table whether it is place or not.If yes then fetch it and if not check it in another table “used” to fetch its respective sub entity.
5. If the previous word is not “in or from or to” then check the recent word in “ambiguious” table whether it is name or not.If yes then fetch it as name entity else go to another table “used” to check whether it is name or place or organization.
6. If the tag of the word is “CD” its a number and hence we compare this word with the pre-defined pattern of the numbers .

G. Bilingual Database:

Here bilingual means we maintain our database in both languages i.e. For Hindi and English. In our database there are mainly three columns words, their Hindi transliteration and their entities i.e. name, places or organization.Our database contains mainly three tables filled, used, and ambiguous. Ambiguous table consists of those words that act as both name and place entities.Filled table consists of those words that are nouns present in our given input text .Used table consists of almost all entities that are names , place , organization and their respective Hindi transliteration

H. Hindi NER:

In this we directly fetch Hindi named entity corresponding to that of English from the database.

Steps:

1. Enter the english input text .
2. Tokenize the input text i.e break the input sentence into independent meaningful words.
3. If the input text contains date we identify it as date entity with the help of some pre-defined patterns i.e dd-mm-yyyy , dd.mm.yyyy , dd/mm/yyyy etc .
4. If text contains numbers we identify it as number entity with the help of the matcher function and some regular expressions .
5. If text contains phrases ie two consecutive words start with capital letter aand form a phrase , we combine it with a hypen.
   e.g Government-Of-India
6. If text contains phrases and ends with "Corp","Ltd","University", it must be an organization entity and then we tag it with organization entity.
   e.g Abmtc-Limited
7. If text contains phrases and starts with "Mr","Mrs","Er","Prof","Miss","Mr", it must be a name entity and then we tag it with person entity.
   e.g Mr-Ram
8. The hyphenated text if it contains phrases and the normal text is then passed to POS tagger which provide the words their respective tags .
9. Next we split the tagged data on the basis the “/” so as to fetch the previous word of the tag .
10. If tag is NN then we store that words in Filled table and then compare it with Used or Ambigious table. If it matches then we fetch its respective entity ie whether name , place or organization .

11. If tag is CD then we compare it with some already defined patterns with the help of matcher class and return its enity as Number.

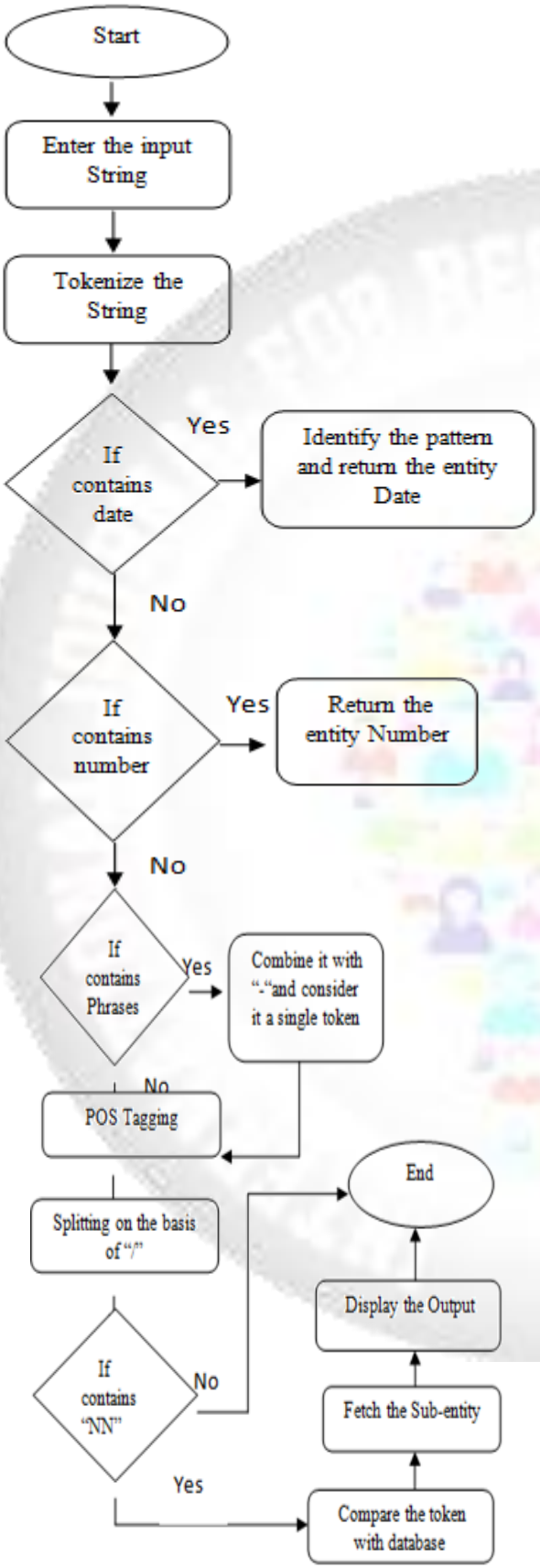

Fig. 2: Flow Chart

III. NER FOR HINDI LANGUAGE

NLP research around the world has taken major turn in the last decade with the advent of effective machine learning algorithms and the creation of large annotated corpora for various languages. But not much work has been done in NER for Hindi language because annotated corpora and other lexical resources have started appearing very recently in India. As common feature function like capitalization are not available in Hindi language and due to lack of standardization and spelling variation, so English NER cannot be directly used for Indian languages. So there is a great need to develop an accurate NER system for Hindi language.

A. Characteristics and Problems faced by Hindi language:

- No capitalization
- Non-availability of large gazetteer
- Lack of standardization and spelling
- Lack of labeled data
- Scarcity of resources and tools
- Free word order language

B. Some points to consider while building NER System:

- Ease to change
- Portability
- Scalability
- Language Resources
- Cost-effective

C. Performance Evaluation Metrics are:

1. Precision: Precision is the ratio of the number of items of a certain named entity type correctly identified to all items that were assigned that particular type by the system.
   Precision (P) = correct answers/answers produced
2. Recall: Recall measures the number of items of a certain named entity type correctly identified, divided by the total number of items of this type.
   Recall (R) = correct answers/total possible correct answers
3. F-measure: The F_1 score (also F-score or F-measure) is a measure of a test's accuracy. It considers both the precision P and the recall R of the test to compute the score. The F_1 score can be interpreted as a weighted average of the precision and recall, where an F_1 score reaches its best value at 1 and worst score at 0. It combines Recall (R) and Precision (P) using the formula .The traditional F-measure or balanced F-score (F_1 score) is the harmonic mean of precision and recall:

$$F_1 = 2 \cdot \frac{\mathrm{precision} \cdot \mathrm{recall}}{\mathrm{precision} + \mathrm{recall}}$$

D. Resources used:

The various resources used to build the bilingual NER are:

1. Parts of Speech (POS) tagger**:** A Part-Of-Speech Tagger (POS Tagger) is a piece of software that reads text in some language and assigns parts of speech to each word (and other token), such as noun, verb, adjective, etc., Maximum entropy parts of speech tagger implemented by Stanford java NLP is used [1].This

POS tagger tags each word based on the PENN TREEBANK TAG SET.

2. Gazetteer Lists: Due to the scarcity of resources in electronic format for Hindi language, so the gazetteer lists are prepared manually from websites and newspaper available online. Different types of lists are prepared such as:

1. First-Name
2. Last-Name
3. Location-Name
4. Organization-Name

With the help of these gazetteer lists we have created a database with tables that contains all location names, organization names, person names as well as their hindi transliteration and their sub entity types.

IV. Results And Analysis

We have tested our system with a corpus which consisted of both English language data. The testing on the corpus of around 100 sentences revealed the following results:

Categories	Precision	Recall	F-Measure
Name	0.35	0.5	0.41
Places	0.42	0.6	0.49
Organization	0.60	0.75	0.66
Dates	0.82	0.86	0.83
Numbers	0.70	0.77	0.73

V. Conclusion

In this thesis the bilingual NER has been developed. The NER requires a database and POS tagger as its important elements to generate correct output. In this we worked on different and almost all date patterns as a result of which our system generates correct output for date entities and hence provides us high value of F -measure.In addition to date patterns we also worked on Number entites with the help of some regular expressions and matches method as a result of which our system correctly recognizes number entities present in the input text.The recogniziton of name, place and organization entities is dependent on database , so we conclude here that our system gives us correct result in case of above three entites only if all the names , places and organizations are available in database . This means that the above three entities are 90% dependent on database . In it we also worked on ambiguous words like the words having same name for person and placeso that the efficiency of the system will be increased. Our proposed NER system will be able to find efficiently all the named entities in different domains depending on enitites present in database used . Hence, we conclude that our proposed NER system have good performance in finding named entities and achievinghighF-measurescorewithlimitedsizecorpora
The quality of NER system depends upon size of database and quality of database .

VI. Future Scope

There can be following further directions for Bilingual NER:

1. The work can be extended to solve the ambiguity problem which can be solved by training the system with a large training corpus of various kinds of news, so that it contains a variety of combinations of names.
2. The work can also be extended to solve unknown words problem which can be solved by using some lists that contain names, especially foreign names.
3. The work can also be extended to make NER more general .
4. The work can also be extended in making more efficient transliteration rules .
5. Since the Hindi POS tagger is not currently available to us, we can't do so much work in Hindi NER . The

work can also be further extended in case of Hindi NER.

References

[1] Http://Nlp.Stanford.Edu/Javanlp/Postagger
[2] Animesh Nayan, B. Ravi Kiran Rao, Pawandeep Singh, Sudip Sanyal And Ratna Sanyal-Iiit Allahabad-Name Entity Recognizer For Indian Languages.
[3] Ijcsi International Journal Of Computer Science Issues, Vol. 7, November 2010-A Survey Of Named Entity Recognition In English And Other Indian Languages
[4] Darvinder Kaur, Vishal Gupta, November 2010, A Survey Of Named Entity Recognition In English And Other Indian Languages, Department Of Computer Science & Engineering, Panjab University ,Chandigarh.
[5] Mitchell P. Marcus, Mary Ann Marcinkiewicz, Building A Large Annotated Corpus Of English: The Penn Treebank, University Of Pennsylvania
[6] Andrei Mikheev, Marc Moens And Claire Grover, Named Entity Recognition,Database , Hcrc Language Technology Group,University Of Edinburgh
[7] Xiaoyi Ma, Toward A Name Entity Aligned Bilingual Corpus, Linguistic Data Consortium, 3600 Market St. Suite 810, Philadelphia
[8] Awaghad Ashish Krishnarao, Himanshu Gahlot, Amit Srinet, D. S. Kushwaha, A Comparative Study Of Named Entity Recognition For Hindi Using Sequential Learning Algorithms, Motilal Nehru National Institute Of Technology, Allahabad.
[9] Bruno Pouliquen, Ralf Steinberger, Camelia Ignat, Irina Temnikova, Anna Widiger, Wajdi Zaghouani, Jan Žižka,Multilingual Person Name Recognition And Transliteration**,** European Commission Joint Research Centre, Bulgarian Academy Of Sciences, University Of Brno.
[10] Thierry Poibeau And The Inalco Named Entity Group**,** The Multilingual Named EntityRecognition Framework, Inalco/Criivi 2 Rue De Lille 75007 Paris.
[11] Penny Treebank Tagset Is Available Http://Www.Ldc.Upenn.Edu/Catalog/Docs/Ldc95t7/C193.Html.
[12] Hai Leong Chieu And Hwee Tou Ng, "Named Entity Recognition: A Maximum Entropy Approach Using Global Information," In 19th International Conference On Computational Linguistics (Coling 2002), Taipei, Taiwan, 2002.
[13] Deepti Chopra, Nusrat Jahan, Sudha Morwal , Hindi Named Entity Recognition ByAggregating Rule Based Heuristics AndHidden Markov Model in International

Journal Of Information Sciences And Techniques (Ijist) Vol.2, No.6, November 2012

[14] Nusrat Jahan , Sudha Morwal And Deepti Chopra,Named Entity Recognition In IndianLanguages Using Gazetteer Method AndHidden Markov Model: A HybridApproach , Nusrat Jahan Et Al./ International Journal Of Computer Science & Engineering Technology (Ijcset).