Domain Specific Named Entity Recognition Using Supervised Approach

Ashwini A. Shende *zashwini@rediffmail.com*
Department of Computer Science & Engineering, RCOEM,
Rashtrasant Tukdoji Maharaj, Nagpur University
Nagpur, 440013, India

Avinash J. Agrawal *avinashjagrawal@gmail.com*
Department of Computer Science & Engineering, RCOEM,
Rashtrasant Tukdoji Maharaj, Nagpur University
Nagpur, 440013, India

Dr. O. G. Kakde *ogkakde@vnit.ac.in*
Visvesvaraya National Institute of Technology
Nagpur, 440010, India

Abstract

This paper introduces Named Entity Recognition approach for text corpus. Supervised Statistical methods are used to develop our system. Our system can be used to categorize NEs belonging to a particular domain for which it is being trained. As Named Entities appears in text surrounded by contexts (words that are left or right of the NE*)*, we will be focusing on extracting NE contexts from text and then performing statistical computing on them. We are using n-gram model for extracting contexts from text. Our methodology first extracts left and right tri-grams surrounding NE instances in the training corpus and calculate their probabilities. Then all the extracted tri-grams along with their calculated probabilities are stored in a file. During testing, system detects unrecognized NEs from the testing corpus and categorizes them using the tri-gram probabilities calculated during training time. The proposed system is made up of two modules i.e. Knowledge acquisition and NE Recognition. Knowledge acquisition module extracts tri-grams surrounding NEs in the training corpus and NE Recognition module performs the categorization of unrecognized NEs in the testing corpus.

Keywords: Named Entity, Supervised Machine learning, N-gram, Context Extraction, NE Recognition

1. INTRODUCTION

The term "Named Entity" (NE) is frequently used in Information Extraction (IE) applications. It was coined at the sixth Message Understanding Conference (MUC-6) which influenced IE research in the 1990s. In defining IE tasks, people noticed that it is essential to recognize information units such as names including person, organization, and location names, and numeric expressions including time, date, money, and percentages. Identifying references to these entities in text was acknowledged as one of IE's important sub-tasks and was called "Named Entity Recognition (NER)." Named Entity Recognition is complex in various areas of automatic Natural Language Processing of (NLP), document indexing, document annotation, translation, etc. It is a fundamental step in various Information Extraction (IE) tasks.

1.1 Named Entity Recognition

The NER task consists of identifying the occurrences of some predefined phrase types in a text. In the expression "Named Entity," the word "Named" aims to restrict the task to only those entities for which one or many rigid designators, stands for the referent. Some tasks related to NER (David Nadeau et.al. [1]) can be listed as follows.

- **Personal Name Disambiguation :**

It is the task of identifying the correct referent of a given designator. In a given context, it may consist of identifying whether *Jim Clark* is the race driver, the film editor, or the Netscape founder. Corpus-wide disambiguation of personal names has applications in document clustering for information retrieval.

- **NE Descriptions Identification :**

It is the identification of textual passages that describe a given NE. For instance, Bill Clinton is described as "the President of the U.S.," "the democratic presidential candidate" or "an Arkansas native," depending on the document. Description identification can be use as a clue in personal name disambiguation.

- **Named Entity Translation :**

It is the task of translating NEs from one language to another. For instance, the French translation of "National Research Council Canada" is "Camseil national de recherché Canada." NE translation is acknowledged as a major issue in machine translation.

- **Analysis of Name Structu**

It is the identification of the parts in a person name. For example, the name "Doctor Paul R. Smith" is composed of a person title, a first name, a middle name, and a surname. It is presented as a preprocessing step for NER and for the resolution of co-references to help. Determine, for instance, that "John F. Kennedy" and "President Kennedy" is the same person, while "John F. Kennedy" and "Caroline Kennedy" are two distinct persons.

- **Entity Anaphora Resolution :**

It mainly consists of resolving pronominal co-reference when the antecedent is an NE. For example, in the sentence "Rabi finished reading the book and he replaced it in the library," the pronoun "he" refers to "Rabi." Anaphora resolution can be useful in solving the NER problem itself by enabling the use of extended co-reference networks. Meanwhile it has many applications of its own, such as in "question answering" (e.g., answering "Who put the book in the library?").

- **Acronym Identification :**

It is described as the identification of an acronym's definition (e.g., "IBM" stands for "International Business Machines") in a given document. The problem is related to NER because many organization names are acronyms (GE, NRC, etc.). Resolving acronyms is useful, again, to build co-reference networks aimed at solving NER On its own; it can improve the recall of information retrieval by expanding queries containing an acronym with the corresponding definition.

- **Record linkage :**

It is the task of matching named entities across databases. It involves the use of clustering and string matching techniques in order to map database entries having slight variations. It is used in database cleaning and in data mining on multiple databases.

- **Case Restoration :**

It consists of restoring expected word casing in a sentence. Given a lower case sentence, the goal is to restore the capital letters usually appearing on the first word of the sentence and on NEs. This task is useful in machine translation, where a sentence is usually translated without capitalization information.

Computational research aiming at automatically identifying NEs in texts forms a vast and heterogeneous pool of strategies, methods, and representations. In its canonical form, the input of an NER system is a text and the output is information on boundaries and types of NEs found in the text. The majority of NER systems fall in two categories: the Rule**-based** systems; and the **Statistical** systems. While early studies were mostly based on handcrafted rules, most of the recent systems preferred statistical methods. In both approaches, large collections of documents are analyzed by hand to obtain sufficient knowledge for designing rules or for feeding machine learning algorithms. Expert linguists must execute this important amount of work, which in turn limits the building and maintenance of large-scale NER systems.

The ability to recognize previously unknown entities is an essential part of NER systems. Such ability hinges upon recognition and classification rules triggered by distinctive modeling features

associated with positive and negative examples. When training examples are not available, handcrafted rules systems remain the preferred technique. The statistical methods collect statistical knowledge from corpus and determine NE categories based on the statistical knowledge. The statistical methods use supervised machine learning algorithms. The idea of supervised learning is to study the features of positive and negative examples of NE over a large collection of annotated documents and design rules that capture instances of a given type. The main shortcoming of Supervised Learning is the requirement of a large annotated corpus. The unavailability of such resources and the prohibitive cost of creating them lead to two alternative learning methods: semi-supervised learning **(SSL**); and unsupervised learning (**UL**).

The term "**semi-supervised**" or "**weakly supervised**" is relatively recent. The main technique for SSL is called "bootstrapping" and involves a small degree of supervision, such as a set of seeds, for starting the learning process. For example, a system aimed at "disease names" might ask the user to provide a small number of example names. Then, the system searches for sentences that contain these names and tries to identify some contextual clues common to the five examples. Then, the system tries to find other instances of disease names appearing in similar contexts. The learning process is then reapplied to the newly found examples, so as to discover new relevant contexts. By repeating this process, a large number of disease names and a large number of contexts will eventually be gathered.

The typical approach in **unsupervised** learning is clustering. For example, one can try to gather NEs from clustered groups based on context similarity. There are other unsupervised methods also. Basically, the techniques rely on lexical resources (e.g., WordNet), on lexical patterns, and on statistics computed on a large unannotated corpus.

This paper discusses the use of supervised machine learning approach for the problem of NE recognition. The aim of our study is to reveal contextual NE in a document corpus using n-gram modeling. A context considers words surrounding the NE in the sentence in which it appears, it is a sequence of words, that are left or right of the NE. In this work, we use supervised learning technologies, combined with statistical models to extract contexts from text document corpus, to identify the most pertinent contexts for the recognition of a NE.

1.2 n-gram Modeling

A useful part of the knowledge needed for Word Prediction can be captured using simple statistical techniques like the notion of the probability of a sequence (a phrase, a sentence). An ***n*-gram model** is a type of probabilistic model for predicting the next item in a sequence. n-gram probabilities can be used to estimate the likelihood

- Of a word occurring in a context (n-1)
- Of a sentence occurring at all

n-gram models are used in various areas of statistical natural language processing and genetic sequence analysis. n-gram language model uses the previous n-1 words in a sequence to predict the next word. These models are trained using very large corpora. n-gram probabilities come from a training corpus

- overly narrow corpus: -probabilities don't generalize
- overly general corpus:- probabilities don't reflect task or domain

A separate test corpus is used to evaluate the model, typically using standard metrics

- held out test set; development test set
- cross validation
- results tested for statistical significance

An ***n*-gram** is a subsequence of *n* items from a given sequence. The items can be phonemes, syllables, letters, words or base pairs according to the application. An *n*-gram of size 1 is referred to as a "**unigram**"; size 2 is a "**bigram**" (or, less commonly, a "digram"); size 3 is a "**trigram**"; size 4 is a "**four-gram**" and size 5 or more is simply called an "***n*-gram**"...

E.g. for the sequence "the big red ball"

unigram P (**ball)**
bigram P **(ball** / **red)**
trigram P **(ball** / **big red)**
four-gram P **(ball** / **the big red)**

In general
P (Word| Some fixed prefix)

As we increase the value of n, the accuracy of n-gram model increases, since choice of next word becomes increasingly constrained.

n-gram is a sequence of *n* words in a text document and one can get a set of n-grams by moving a floating window from the beginning to the end of the document. During the n-gram extraction from text document, duplicate n-grams must be removed and the frequency of the n-gram types should be calculated. Additionally, other values can be stored with n-gram type and frequency, e.g. n-gram unique number, but it is document and query model dependent.

FIGURE 1, shows a common architecture of an n-gram extraction framework. This framework usually includes:

1. **Document parsing** – it parses terms from input documents.
2. **Term pre-processing** – in this phase, various techniques like stemming and stop-list are applied for the reduction of terms.
3. **n-gram building and pre-processing** – it creates an n-gram as a sequence of *n* terms. Sometimes, n-grams are not shared by text units (sentences or paragraphs). It means, the last term of a sentence is the last term of an n-gram and the next n-gram begins by the first term of the next sentence.
4. **n-gram extraction** – the main goal of this phase is to remove duplicate n-grams. The result of this phase is a collection of n-gram types with the frequency enclosed to each type. This collection can be cleaned after this phase; for example, n-gram types with a low frequency are removed. However, it is not appropriate to apply this post-processing in any application. It can be used only when we do not need low frequency n-gram types. A common part of such a framework is n-gram indexing. A data structure is applied to speed up access to the tuple _ngram, id, frequency_, where ngram is a key; it means the ngram is an input of the query and id and frequency form the output. Although, it is necessary to create other data structures, for specific document and query models, one must always consider this global storage of the tuples.

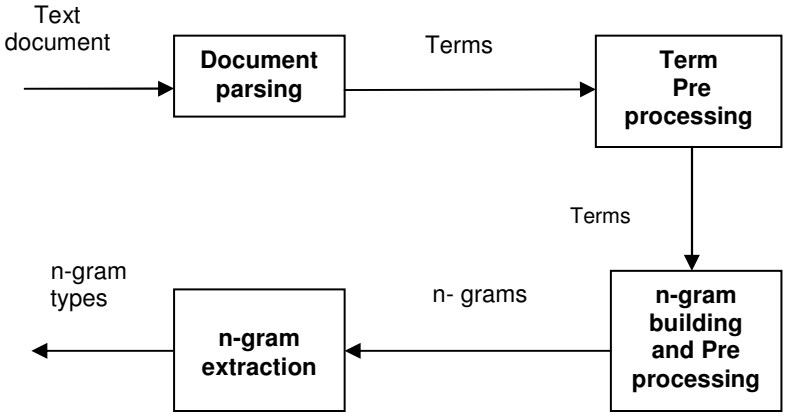

FIGURE 1: n-gram Extraction Framework

The remaining of the paper is organized as follows: Section 2 presents the review of the various methods used for Named Entity Recognition. Section 3 describes the Methodology and section 4 gives test results of our approach. Section 5 gives the Work's conclusion and Section 6 explains the future work recommended.

2. RELATED WORK

Named entity recognition can be used to perform numerous processing tasks in various areas: of Information Extraction systems, Text mining, Automatic Speech Recognition (ASR) etc. Several works are particularly interested in the recognition of named entities.

Mikheev et al. [2] have built a system for recognizing named entities, which combines a model based on grammar rules, and statistical models, without resorting to named entity lists.

Collins et al. [3] suggests an algorithm for named entity classification, based on the meaning word disambiguation, and exploits the redundancy in the contextual characteristics. This system operates a large corpus to produce a generic list of proper nouns. The names are collected by searching for a syntax diagram with specific properties. For example, a proper name is a sequence of consecutive words in a nominal phrase, etc.

Petasis et al. [4] presented a method that helps to build a rules-based system for recognition and classification of named entities. They have used machine learning, to monitor system performance and avoid manual marking.

In his paper, Mann et al. [5] explores the idea of fine-grained proper noun ontology and its use in question answering. The ontology is built from unrestricted text using simple textual co-occurrence patterns. This ontology is therefore used on a question answering task to provide primary results on the utility of this information. However, this method has a low coverage.

The Nemesis system presented by Fourour et al.[6] is founded on some heuristics, allowing the identification of named entities, and their classification by detecting the boundaries of the entity called "context" to the left or right, and by studying syntactic, or morphological nature of these entities. (n-gram modeling) For example, acronyms are named entities consisting of a single lexical unit comprising several capital letters, etc.

Krstev et al. [7] suggested a basic structure of a relational model of a multilingual dictionary of proper names based on four-level ontology.

Etzioni et al. [8] planned the KNOWITALL system which aims at automating the process of extracting named entities from the Web in an unsupervised and scalable manner. This system is not intended for recognizing a named entity, but used to create long lists of named entities. However, it is not designed to resolve the ambiguity in some documents.

Friburger et al. [9] recommends a method based on rules for finding a large proportion of person names. However, this method has some limitations as errors, and missing responses.

Nadeau et al. [10] have suggested a system for recognizing named entities. Their work is based on those of Collins, and Etzioni. The system exploits human-generated HTML markup in Web pages to generate gazetteers, then it uses simple heuristics for the entity disambiguation in the context of a given document.

Kono Kim et. al. [11] proposed a NE (Named Entity) recognition system using a semi supervised statistical method. In training time, the NE recognition system builds error-prone training data only using a conventional POS (Part-Of-Speech) tagger and a NE dictionary that semi-automatically is constructed. Then, the NE recognition system generates a co-occurrence similarity matrix from

the error prone training corpus. In running time, the NE recognition system detects NE candidates and assigns categories to the NE candidates using Viterbi searching on the AWDs.

In view of works touching the recognition of named entities, we perceive that most of them are based on a set of rules in relation to predefined categories: morphological, grammatical, etc. or on predefined lists or dictionaries. The n-gram modeling domain is still in exploration. We adopted the idea of Nemesis based on the left and the right context of the named entity. However, our approach does not mark the context derived from syntactic or morphological rules, but identifies the context founded on learning phase. The objective is thus to carry out a system, able to induce the nature of a named entity, without requiring dictionaries or lists of named entities.

3. METHADOLOGY

This paper discusses the use of supervised machine learning approach for the problem of NE recognition. The aim of our study is to reveal contextual NE in a document corpus using n-gram modeling. A context consists of words surrounding the NE in the sentence in which it appears. It is a sequence of words, that are left or right of the NE. In this work, we use supervised learning technologies, combined with statistical methods to extract contexts from text document and to identify the most pertinent contexts for the recognition of a NE.

Our work mainly focuses on Context extraction i.e. extracting the left and the right context of the Named entity... Two or more words that tend to occur in similar linguistic context (i.e. to have similar Co-occurrence pattern), tend to be positioned closer together in semantic space and tend to resemble each other in meaning. Our objective is to carry out a system, able to induce the nature of a named entity, following the meeting of certain indicators.

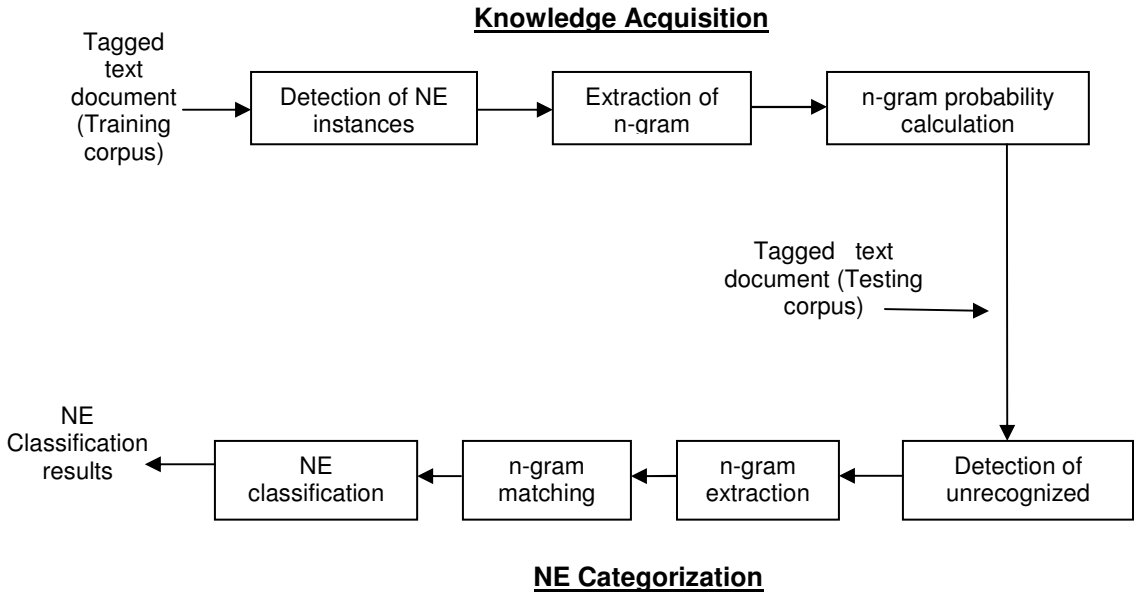

FIGURE 2: Block diagram of Proposed System

FIGURE. 2 shows block diagram of the proposed system. The proposed system consists of two modules. First module is a Knowledge acquisition module which detects NE instances from the training corpus. Then, it extracts left and right tri-grams surrounding those NE instances and calculates its probability occurrence in the training corpus. After calculating all probabilities, extracted tri-grams along with their probabilities are stored in a text file for reference. When testing corpus is given for testing, NE recognition module finds all unrecognized NE instances from it by using the same method used in knowledge acquisition module. Then, it classifies each

unrecognized NE instance in the testing corpus into one of the domain specific categories using the tri-gram probabilities already stored in a file.

3.1. Knowledge Acquisition

The main functioning of this module is to extract the tri-grams surrounding NEs from given domain specific text document. The document acts as the training corpus for learning. Our system input is a tagged text document. For our corpus, all NEs should have numerical tagging. Some of the sentences from our corpus are given below.

- when [q] vidarbha [1] express [n] reaches [v] wardha [2]
- what [q] is [x] the [d] status [n] of [p] mumbai [1] mail [n]
- what [q] is [x] departure [n] time [n] of [p] vidarbha [1] express [n]
- when [q] mumbai [1] mail [n] reaches [v] mumbai [2]
- what [q] is [x] the [d] position [n] of [p] the [d] gitanjali [1] express [n]

ALGORITHM:-

- Locate all NEs from training corpus.
- Extract left & right trigrams surrounding NEs.
    - If trigram does not exist then extract bigrams.
        - If bigram does not exist then extract unigrams.
- Remove duplicate trigrams / bigrams / unigrams and calculate the probability of each in the corpus.
- Store the unique trigrams / bigrams / unigrams along with probability in a file.

The first step of our algorithm is to locate Named Entities in each sentence by reading the text corpus. NE's are the words which are followed by numerical tagging. E.g. "*vidarbha* ", "*Mumbai*","*wardha*" etc are NE instances in the above examples. After locating the NEs, surrounding trigrams are extracted from the text corpus. Trigrams are the 3 consecutive words to the left or right of NE. For efficiency purpose we will extract both left and right trigrams for each NE. following structure is used to store the trigram.

```
public class TriGramElement
{

    public String[] LeftElements = new String[3];
    public String[] RightElements = new String[3];
    public String CentreElement;
    public String[] LeftValue = new String[3];
    public String[] RightValue = new String[3];
    public String CentreValue;

};
```

Every NE occurrence cannot guarantee presence of trigram surrounding it, especially if NE occurs as the first or last word of the sentence in a corpus. In such cases our system is flexible to consider either Bigram or unigram. E.g. for NE "*vidarbha*" left context is a unigram and right context consists of a trigram. For "*Mumbai*", left context is trigram and right content is unigram and for "*hawrah*" both left and right context consists of trigrams. Some sample extracted trigrams from the corpus is mentioned below.

when **vidarbha** express reaches wardha

the status of **mumbai** mail

by what time **hawrah** mail will come

The next step of the algorithm is to remove duplicate n-grams. Removal of duplicate trigrams is necessary to apply statistical methods on it. For probability calculation we need to get the occurrence count of each trigram. Our system generates a list of unique trigrams and stores them in a text file along with their probabilities. The sample trigrams stored in a text file is shown below.

```
1    ,,,when          :   wardha,reaches,express    -->  0.02

1    ,the,status,of   :   is,what,mail              ->  0.02

1  ,position,of,the   :   is,what,express           -->  0.02

3  ,the,fare,from     :   what,gondia,to            -->  0.02
```

FIGURE 3: List of sample trigrams stored in a file

3.2 NE Categorization

After detecting unrecognized NEs, the NE recognition module assigns categories to them using the trigram probabilities calculated by Knowledge acquisition module.

ALGORITHM:

- Detect unrecognized NE instance from testing corpus.
- Extract left and right trigrams for it.
    - If trigram does not exist then extract bigram.
        - If bigram does not exist then extract unigram.
- For every unrecognized NE instance in testing corpus, search for left trigram / bigram / unigram in the list stored in a file. (Generated from training corpus) using linear search.
- If match not found search for right trigram / bigram / unigram in the list.
- If match not found for left as well as right trigram / bigram / unigram then marked the corresponding NE as unrecognized.
- Find out the category of maximum probability trigram / bigram / unigram match.
- Assign maximum probability category to the unrecognized NE.
- Repeat above steps for all unrecognized NE instances in the testing corpus.
- Store NE categorization results in a file.

NE categorization module will first extract all NE instances, from the testing corpus by applying the same method used in knowledge acquisition module. We are assuming that testing corpus is a tagged corpus in which all unrecognized NE's are marked with tag [0]. After detecting NE's, NE categorization module will create a list of unrecognized NE instances. For each NE stored in list, left and right content words are extracted from the testing corpus in the form of trigrams.

To categorize NE, our system will compare its left context words with the tri-gram entries (generated from training corpus) stored in a file using linear search algorithm. Our system prefers left context words over right context words as left context is more relevant in comparison to right context for recognition. If the match is found then its probability count value will be extracted. After checking all the entries, NE categorization module will compare probability count of all matched entries and will find the maximum probability count out of it. Then unrecognized NE will be classified to the matched category for which probability count is maximum. In absence of left trigrams, right trigrams will be considered for matching. Our system is flexible to use bigrams as well as unigrams in absence of trigrams for categorizing NEs. After categorizing all NE's categorization results will be stored in a text file.

Consider the following sentence from the testing corpus

how [q] many [u] trains [n] are [x] of [p] type [n] **doronto** [0]

In the above sentence word with tag [0] is detected as unrecognized NE. i.e. "*doronto*". Next step is to find out the context words to the left and right of NE. The extracted tri-gram for the "*doronta*" is

are of type **doronto** --- ---- -----

In above case left trigram consists of 3 words whereas right tri-gram is null as "*doronto*" is the last word of the sentence. Our algorithm gives precedence to left context words .So it will search the Tri-gram entries stored in a file to get a match for tri-gram "*are of type* ". The match is found with probability count 0.02 and the category type is train name. So "*doronto*" will be categorized to Train name and result will be stored in a text file.

4. EXPERIMENTAL RESULTS

4.1 Test Collections

To evaluate the performance of the proposed system, we used a test collection of a Railway Reservation domain. The testing corpus is a collection of routine railway enquiries consisting of domain specific NE categories like train names, source and destination train names, reservation classes etc. Categories are labeled with numerical tagging in the testing corpus. We think that the preliminary experiments have some meaning as our goal is to recognize NE categories with supervised statistical methods.

4.2 Performance Evaluation

Since any NER system or method must produce a single, unambiguous output for any Named Entity in the text, the evaluation is not based on a system architecture in which Named Entity Recognition would be completely handled as a preprocess to sentence and discourse analysis. The task requires that the system recognize what a NE represents, not just its superficial appearance and the answer may have to be obtained using techniques that draw information from a larger context or from reference lists.

A scoring model developed for the MUC and Named Entity Task evaluations measures, both precision (*P*) and recall (*R*) terms borrowed from the information-retrieval community. These two measures of performance combine to form one measure of performance, the *F*-measure, which is computed by the uniformly weighted harmonic mean of precision and recall.

To evaluate performance of the proposed system, we used the performance measures like precision, recall and the F-score. Precision (p) is the proportion of correct responses out of

returned NE categories, and Recall (r) is the proportion of returned NE categories out of classification targets. Following graph shows the performance measure results of our system.

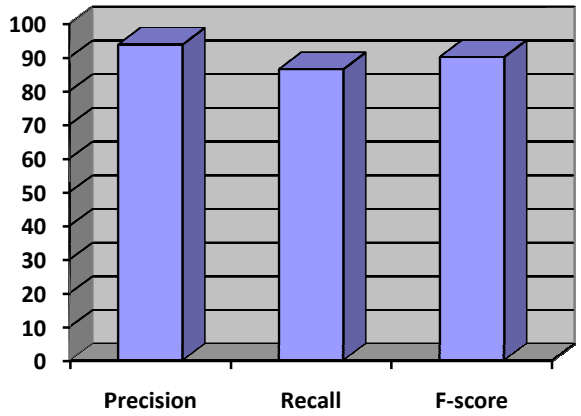

FIGURE 4: Performance Measure results

5. CONCLUSION

We proposed a NE recognition system using Supervised Statistical methods. Our goal is to uncover Named Entity in a document corpus. NE occurs frequently accompanied by contexts: i.e. sequence of words, that are left or right of the NE. In training time, the proposed system extracts all NE instances from a given domain specific text document. Then, the proposed system generates a list of unique tri-grams surrounding NEs in the training corpus and calculate probability occurrence for each. This information is stored in a file as a reference for testing. During testing, this information is referred to identify most pertinent contexts for the categorization of unrecognized NEs from the testing corpus. This enables to derive a model for NE recognition. In the preliminary experiments on Railway Reservation domain the proposed system showed 90.04% average F-score measure**.**

Recall and precision are usually admitted parameters for measuring system performance in the NER field.

$$\textbf{Precision} = \frac{\textbf{(No. Of correct responses)}}{\textbf{(No. of responses)}}$$

$$\textbf{Recall} = \frac{\textbf{(No. Of correct responses)}}{\textbf{(No. correct in key)}}$$

$$\textbf{F- measure} = \frac{\textbf{Precision} \times \textbf{Recall}}{\textbf{½ (Precision + Recall)}}$$

For NER task it is observed that though Hand-made rule based approach can get high rate results in specific domain but it has problem with broad and new domain. As Hand-made rule based method are dependent to domain, Machine learning–based methods is the best independent solution for NER. Machine Learning methods can get good result in precision and recall with high portability and it can be best independent and portable solution for text mining and specially NER. But high performance of this kind of methods depends on the data training value. This type of approach can get high precision in recognition when amount of data training is huge and the result is strictly reduce when data training value is few or malfunction of algorithm. The Hybrid methods gave good results but portability of this type of approach is reduced when they improve precision in recognition by using huge value of fixed rules. Though traditionally Rule based systems were more popular now a days machine learning approach is preferred for developing NER systems. TABLE 1 shows the comparison of the results obtained from proposed systems with the existing systems.

	System	Precision	Recall	F -Score
1	**Proposed System**	**93.68**	**86.40**	**90.04**
2	NYU System (Rule based)	90	86	88.19
3	IsoQuest,Inc (Rule based)	93	90	91.60
4	MENE (Machine learning based)	96	89	92.20
5	Association Rule Mining (Machine learning based)	83.43	66.34	70.16
6	IdentiFinder (Machine learning based)	92	89	90.44
7	LTG (hybrid)	95	92	93.39
8	NYU Hybrid (hybrid)	93	85	88.80

TABLE 1: Comparison of proposed system results with the existing systems

All the proposed methods and models developed for NER task have tried to improve precision in recognition module and portability in recognition domain as one of the major problem and difficulty in NER systems is to change and switch over to a new domain called portability. Most distinguishing feature of the proposed system is that it is easily portable to the new domain as it is based on supervised machine learning approach. Proposed system is using n-gram model for extracting NE context which is also contributing to the portability of the proposed system across multiple domains. It is not required to maintain large gazetteer lists as NE recognition for our system is context based. Context is extracted from the corpus itself (training as well as testing) and not dependent on gazetteer lists. Based on the experimental results it can be said that the proposed system is a good solution to address NER problem as it is capable of recognizing NEs from the given domain corpora dynamically without maintaining huge large NE dictionaries or gazetteer lists.

6. FUTURE WORK

Though primarily we have applied the proposed approach to address NER problem, it is not restricted to that problem only. The proposed approach can be applied to solve many problems in Natural Language Processing domain. It can be used in various research areas like machine translation, Question answering systems etc.

As we have stated that proposed system is portable in nature we need to use our systems across diverse domains and get its performance analysis across those diverse domains.

Our future work recommendations are as follows.

- To test the system on different domain corpora,
- To discern and to measure similarity between contexts. We can use this measurement to cluster similar contexts.
- Though we have primarily applied our approach to NER problem, we can also attempt some additional concepts

7. REFERENCES

[1] David Nadeau "Semi-Supervised Named Entity Recognition: Learning to Recognize 100 Entity Types with Little Supervision "

[2] Mikheev, M. Moens, and C. Grover, "Named Entity Recognition without Gazetteers", *in Proceedings of Conference of European, Chapter of the Association for Computational Linguistics, EACL '99,* pp. 1-8, University of Bergen, Bergen, Norway June 1999.

[3] M. Collins and Y. Singer, "Unsupervised models for named entity classification", *in Proceedings of the Joint SIGDAT Conference on Empirical Methods in Natural Language Processing and Very Large Corpora, 1999, pp. 189–196*

[4] G Petasis, F Vichot, F Wolinski, G Paliouras, V. Karkaletsis, and C. D. Spyropoulos, "Using machine learning to maintain rule-based named-entity recognition and classification", *in Proceedings of the 39th Annual Meeting on Association for Computational Linguistic*s, pp. 426 – 43, Toulouse, France, 2001

[5] G.S. Mann, "Fine-grained proper noun anthologies for question answering", *International Conference on Computational Linguistics*, *COLING-02 on SEMANET: building and using semantic networks, 2002, Vol. 11*,

[6] N. Fourour, and E.Morin, "Apport du Web dans la reconnaissance des entités nommées". *Revue québécoise de linguistique, 2003, vol. 32, n° 1, pp. 41-60.*

[7] Krstev, D. Vitas, D. Maurel, M. Tran, "Multilingual ontology of proper name", *in Proceedings of the Language and Technology Conference, pp. 116–119, Poznan, Poland, 2005*

[8] O. Etzioni, M. Cafarella, D. Downey, S. Kok, A. Popescu, T. Shaked, S. Soderland, D. Weld, and A.Yates, "Unsupervised named-entity extraction from the web: An experimental study", *Artificial Intelligence*, *2005, vol. 65,pp. 91–134*

[9] N. Friburger, "Linguistique et reconnaissance automatique des noms propres", *Meta : journal des traducteurs,2006, vol. 51, n° 4, pp. 637-650*

[10] David Nadeau, Peter D. Turney and Stan Matwin "Unsupervised Named-Entity Recognition: Generating Gazetteers and Resolving Ambiguity", In Proceedings of the *19th Canadian Conference on Artificial Intelligence*, *2006*

[11] Kono Kim, Yeohoon Yoon , Harksoo Kim, and Jungyun Seo ",Named Entity Recognition Using Acyclic Weighted Digraphs: A Semi-supervised Statistical Method", *PAKDD 2007, LNAI 4426, pp. 571–578, 2007. © Springer-Verlag Berlin Heidelberg 2007*