Hybrid, Three-stage Named Entity Recognizer for Tamil

S. Lakshmana Pandian
Anna University
Chennai. India
lpandian72@yahoo.com

Krishnan Aravind Pavithra
Anna University
Chennai. India

T.V. Geetha
Anna University
Chennai. India
rctamil@annauniv.edu

Abstract

The aim of this paper is to present the construction of a hybrid, three-stage named entity recognizer for Tamil. Named entity recognition performs an in-place tagging task for a given Tamil document in three phases namely shallow parsing, shallow semantic parsing and statistical processing. The E-M algorithm (HMM) is used in the statistical processing phase, with initial probabilities obtained from the shallow parsing phase, and a modification to the E-M algorithm deals with inputs from the shallow semantic parsing phase. This study is concentrated on entity names (personal names, location names and organization names), temporal expressions (dates and times) and number expressions. Both NER tags and POS tags are used as the hidden variables in the E-M algorithm. The average F-values obtained from the system 72.72% for the various entity types.

1. Introduction

The named entity recognition (NER) problem refers to the identification of names of people, locations and organizations from a document of data. Given a document of un-annotated data, the expressions to be annotated are "non-artifact unique identifiers" of entities (organizations, persons, locations) and times (dates, times) [3]. The tool constructed reads a Tamil text document, performs in-place tagging of named entities within the document, and stores all the entities found in a named entity table along with their attributes and probability vectors.

Four major challenges were identified in performing NER for a Tamil document: (1) Named entities in English are generally capitalized, but in Tamil there is no capitalization for the initial letter, which makes named entities in text tougher to spot. [3]. (2) the word order and sentence structure of Tamil and to an extent its grammatical structure are semantically oriented and are dependent on the meaning conveyed. Even the definitions of subjects and objects in Tamil depend on the function that these constructs perform. This means that a purely syntactic approach will be insufficient, and a certain level of semantic analysis is necessary. Tamil follows a partial free word order, thereby rendering a purely positional analysis insufficient for NER. (4) Tamil is heavily *conjugated*, like most Indian languages, and lemmatization is a difficult task. However, because of the conjugation, lemmatization must sometimes be performed to extract the attributes of the entities.

However, Tamil words contain several unique features like post-positions, case markers, and PNG markers which to an extent enable identification of entity types. It was found that using these contextual cues could help evaluate reasonably accurate seed probability estimates for words in the input data.

Taking into account the above challenges, it was found that a purely inductive or machine-learning oriented approach to NER in Tamil would not always suffice for Tamil documents. Tamil sentences have free-word ordering, and the positional order of the words in a sentence has no strong bearing upon whether a given word can be a named entity or not. A purely deductive or rule-based approach also proves insufficient, as it is not possible to encode the information required to tag named entities in a set of simple yet exhaustive and comprehensive rules.

We therefore adopt a hybrid approach to NER in Tamil with three distinct phases. In the first phase, shallow parsing, a dictionary of word clues and case markers are utilized as context cues to identify and tentatively classify named entities. The unique morphological components of Tamil words are hence utilized to obtain initial probabilities. In the second phase, shallow semantic parsing, a verb's syntactic information (PNG markers) and semantic information (meaning) are utilized to identify the named entity type. The semantic orientation of subjects and objects are exploited in this phase. In the final phase, statistical information from a training corpus is utilized. The E-M algorithm identifies the best tag sequence using the statistical information and inputs from the above two phases. The E-M algorithm is modified to resolve the free-word order problem, a modification termed *quantum entanglement.*

2. Related work

Recent work in named entity recognition has

INFOS2008, March 27-29, 2008 Cairo-Egypt

demons- trated a shift from hand-coded knowledge based methods towards supervised learning techniques. A survey was first made of papers adopting a machine learning approach to NER. The Co-NLL 2003 Shared Task on Language Independent NER [13] provided an insight on several statistical methods used for NER, notably Hidden Markov Models (HMM) and Support Vector Machines (SVM). Using SVM for NER was examined by [15] and it was found to be useful when the number of features was very large. However the best results were obtained by systems using a combination of HMM with the Maximum Entropy Approach. It was decided to adopt the E-M algorithm (HMM-based) as it was a fast and suitable algorithm for this type of task. Cucerzan and Yarowsky present the use of character based tries to store word-internal and external contextual evidence before smoothing[6]. The system SLINERC follows a multi-stage approach to NER, using orthographic context tries, but without performing smoothing across categories[10]. Megyesi presents a shallow parser that uses POS tags groups noun phrases together [14] An understanding of the E-M algorithm was obtained from [7] and the usage of the E-M algorithm in NLP tasks was presented in [9]. Vijayakrishna presents a domain focused NER for Tamil using CRF++ tool. It performs named entities tagging with a hierarchical tag set designed with focus to tourism domain. [12]. Lakshmana Pandian presents Named Entity Recognition in Tamil using context cues and EM algorithm. Using context cues and E-M method the word sequence probability has been enhanced. [11].

3. System Overview

The NER system accepts a raw Tamil document as input and outputs a tagged document wherein named entities in text are tagged according to the entity type. The system also outputs a named entity table containing a list of named entities found in the document, the attributes with which they appeared, and the probability vector for the named entity types. Fig 3.1 gives the outline of the NER system. The various processes and components of the system include the following:

Preprocessing: An initial preprocessing of the input data is at first performed to render the input text suitable for machine parsing. The preprocessor scans the Tamil text document and indexes all the sentences into a data structure for easy access. Commas impart valuable information regarding chunking of entities, and hence commas are stored along with the word they follow here. Regular expressions for dates and times are also scanned here and suitable probability assignments are made. It was decided to use two levels of contextual clues to provide probabilities. Level one clues include word-internal evidence like case-markers and word-external evidence like titles for persons, and words like "park"," river", etc. For locations, level two clues or attributes refer to words indicating possession or emotion of a particular entity type. The contextual clues provide initial probability estimates within the shallow parsing phase.

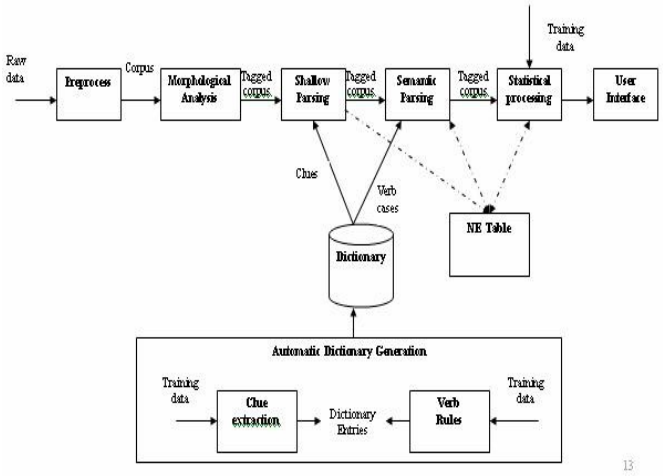

Fig. 3.1 Block diagram of E-M system

Morphological Analyzing and Clue Extraction: The input data is then passed on to an Extractor module. The Extractor module performs two major functions (1) morphological analysis of the input data (2) Extraction of the clues attributes verbs and prepositions from the dictionary.

Morphological analysis for Tamil is performed by a tool called Atcharam, developed by RCILTS, Chennai. The morphological analyser is entirely rule-based; it inputs a Tamil text document and outputs the Part Of Speech (POS) tag and the morphological components like case-markers, etc. for every word in the document. Words unresolved by the morphological analyzer are left untagged, and mostly consist of proper nouns (likely named entities). The morphological components of all words are stored for checking against context clues from the dictionary.

The other function performed by the Extractor is to read the contents of the dictionary and extract the rules within into the corresponding data structures.

Dictionary: The dictionary forms the heart of the deductive component of the NER system, and it is on permanent storage that contain the rules necessary to perform shallow and semantic parsing. Fig 3.2 illustrates the contents of the dictionary The context clues (used in the shallow parsing phase) in the dictionary are stored as the clue, and a set of heuristic values depicting the frequency of appearance of this clue for a particular entity type. Verbs (used in the semantic

parsing phase) in the dictionary are stored as a root word (of the verb), and the set of entity type arguments (subjects/objects) that the verb can take.

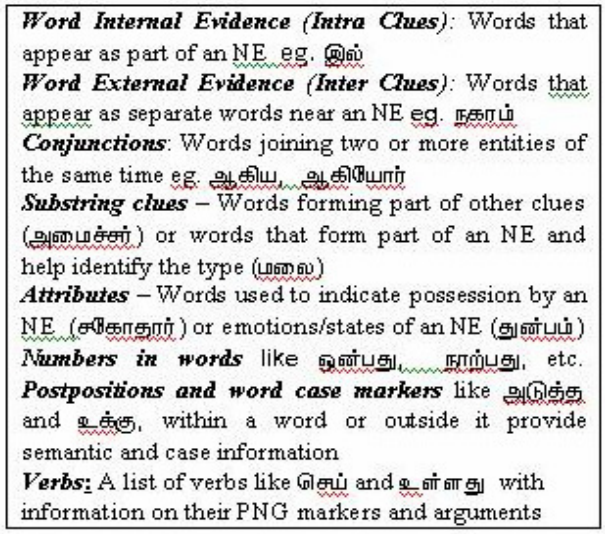

Figure 3.2. Dictionary Components

Shallow Parser: The POS tagged input sentences enter the shallow parser module, which checks for context clues and assigns initial probabilities to an entity chunk. The module outputs a named entity table with the entity chunk and an initial probability vector. Two levels of contextual clues are utilized: (1) Clues, which represent word-internal (morphological components) or word-external evidence ('naming' words) with heuristic values from the dictionary and (2) Attributes, which represent possession or emotion related to a type of entity.

A blind chunking of all untagged items together is performed at first in the hope that all words in the chunk would refer to the same entity. An entity chunk is created for further processing and re-chunking is performed later. Two handlers examine word-internal clues or intra clues and word-external or inter clues found in the entity chunk and compares them with the equivalent clues found in the dictionary. If a match is found, the clues and their probability influence values are stored with the chunk. A Turing machine uses the clues obtained from all the above modules to perform a probability update. Attributes (words related in sense to an entity) are also examined in this phase and used to update probabilities. The intermediate output at this phase is an entity chunk with its clues, attributes and probability vector assigned by the Turing machine. A re-chunking is performed at this stage based on conjunctions to split wrongly chunked entities. Before indexing the correct entity chunk into the NE Table, a lookup is made to find if the entity has occurred previously. If so, the chunk is indexed as an alternate occurrence of the previous entry. If not, a new table entry is created for a particular entity chunk. The output of the shallow parser module is the NE Table containing entities, their probability vectors, and the clues and attributes.

Semantic parsing: The next phase of the NER system is the shallow semantic parsing phase, wherein verbs and postpositions are used to identify the type of subjects/ objects in a sentence. This phase uses the verbs in the dictionary to help identify and accurately classify the entities already present in the NE Table. The semantic parsing phase comprises of two main modules: a Tiling module and an Entanglement module.

The input sentences are analyzed and split into logical groups of words called tiles. Termed *tiling*, creates logical chunks of words or tiles containing verbs or postpositions, which are then used to identify subjects and objects in a sentence. This phase connects a verb to various entities subjects and objects, and outputs a list of these connections. A set of procedures were devised for building tiles from input text such that each tile contained one of (1) a verb (2) a postposition or an entity along with its case-marker (3) entities or noun phrases without any case markers. The case-markers represent the equivalent of prepositions in English and are good object identifiers. Fig. 3.3 illustrates an input sentence split into tiles.

The tiles and the input sentence are then passed on to an Entangler module, which tries to "entangle" the verb in the group of tiles with its subject and objects. A simple rule of thumb was adopted to identify the subjects and objects: entities accompanied by case-markers or postpositions were objects and entities with no such markers were the subject of the verb, an observation that holds true for almost 90% of Tamil sentences.

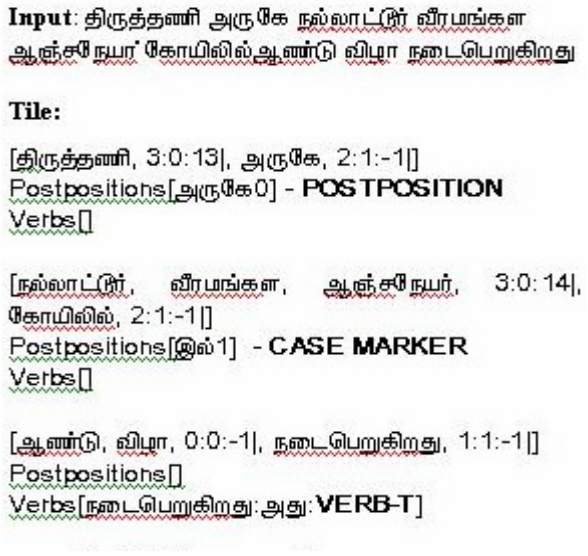

Figure 3.3. Tiles created from a sentence

Knowing the subject and objects of the verb, and the meaning and PNG markers of the verb itself, it is possible o specify the type of the subject and objects to the accuracy required for the six categories of entities. The entangler module does this by creating an *entanglement packet*, which contains the entity found in the sentence, its index in the entity table and the type of entity it represents. These packets are used as inputs in the statistical processing phase.

Statistical processing : The final phase is the statistical processing phase, which utilizes inputs from the above two phases and a tagged training corpus to identify a final probability vector for every entity. Fig 3.4 gives an outline of the statistical processing phase. The E-M algorithm, modified as defined above, is used in this phase

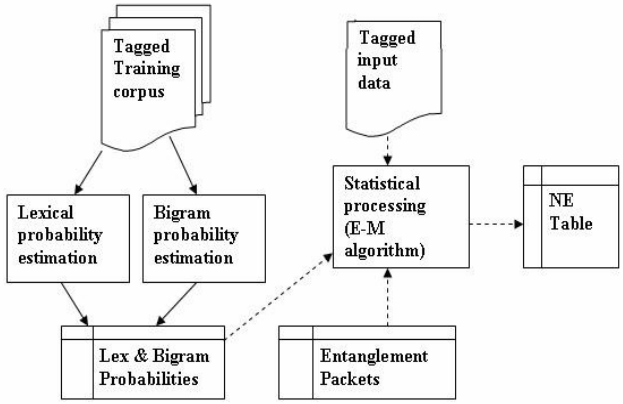

Fig. 3.4 Statistical processing phase

அர்ஜுன் மூர்த்தியை சந்தித்தார்

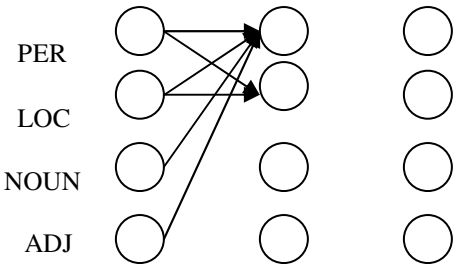

Fig. 3.5 A portion of Trellis for Named entity sequence

The E-M algorithm [7] belongs to a class of clustering algorithms used when incomplete data is provided and the presence of "hidden" or latent variables can be used to augment the incorrect values and yield accurate or correct values after an iteration. The E-M algorithm's initial probabilities could be obtained from the shallow parsing phase, and the output would be the best category

அர்ஜுன் மூர்த்தியை சந்தித்தார்

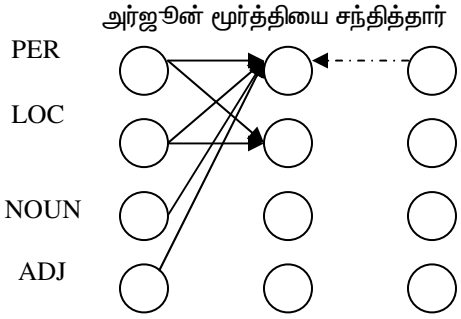

Fig 3.6 Modified EM with Quantum entanglement

for each word in the input sentence, including entity types as categories. However, two probems were encountered with the traditional E-M algorithm (shown in Fig 3.1) (1) It performs only positional analysis , and a modification was required for free word order languages like Tamil (2) it is **syntactically oriented,** and modification was required to include semantic information. The modification proposed, **Quantum entanglement**, solves both the above problems. The positional analysis is supplemented by allowing words out of sequence to also update probabilities for any word. Semantic information from the semantic parsing phase is used to find which two words should be thus entangled. As shown in Fig 3.6 word no. 3 influences the probability that word no. 2 is a LOC category word.

To use EM for Named Entity Recognition, we phrase the NER problem in terms of hidden variables (categories) and bserved sequences (a word sequence). The most general procedure of the E-M algorithm follows:

Define a function Q as

$$Q(\theta \mid \theta^k) = E((X, Z \mid \theta) \mid X, \theta^k) \qquad 3.1$$

where Z represents the hidden variables (categories)
X represents the observed data (words)
θ represents the parameters of the model (probability estimates).
E is the expectation function
k refers to the kth word in the word sequence

The two steps are as follows:

E-step: Compute $Q(\theta \mid \theta^k)$. In our model, this is equivalent to computing the probabilities for the word in the k+1th position based on the probabilities of the word in the kth position. The Viterbi algorithm (lexical and bigram probabilities is used in this step)

M- step: Choose θ^{k+1} to be a value of θ that maximizes $Q(\theta \mid \theta^k)$ In our model, we identify the category that the word in a sequence has maximum probability of belonging to, and store a back pointer from the next word to that category.

The two probability tables input to the algorithm are (1) Symbol emission probabilities or lexical probabilities, i.e the probability that a word at a given position in the sequence is realized by a category, and (2) State transition probabilities or bigram probabilities, i.e the contextual probability that one category follows another within the training corpus. These are calculated by the process of Maximum Likelihood Estimation from the counts of words and categories in the training corpus and represent the hidden variables in the model.

$$\sum_j F(W_{i-1} \cap C_j) * P(C_i \mid C_j) * P(W_i \mid C_i) \qquad 3.5$$

The data is then passed to the E-M algorithm, which performs positional probability smoothing using the trellis as shown in Fig. 3.5. Forward and backward probabilities for each word and each category are recursively calculated using the product of the previous/next word's forward/backward probability, the emission probability of the word for a category, and the transition probability from one category to the other, as depicted in the formula.

It can be seen from the figure 3.5 that named entity types also form categories within the trellis. The output of the algorithm, which is the best unambiguous category sequence for a word sequence, also helps unambiguously identify named entities.

The initial probabilities required for the E-M algorithm are provided for entities from the shallow parsing phase, and for other categories from the morphological analyzer. In the maximization step of the E-M algorithm,

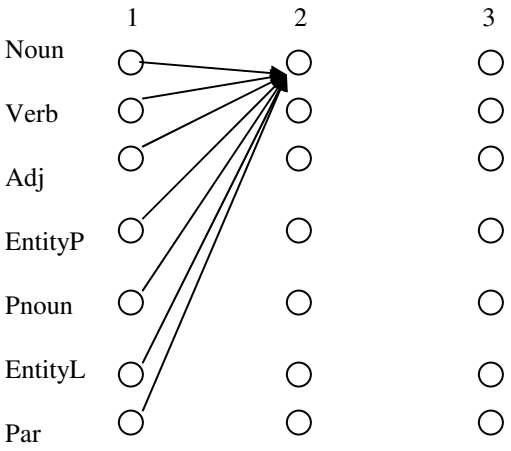

Fig 3.7 Calculation of probabilities for a node in the trellis from the previous word's probability vector. The rows in the trellis represent categories(N-Noun, V-Verb, etc) the column show word w for 1 from 1 to 4

entanglement information from the semantic phase is taken into account, and the probabilities for entities are affected based on the contents of the entanglement packet. The final probabilities for entities are therefore influenced by the processing in all three phases, and the processing in the final phase does not only depend upon the positional order of the words but also semantic information from any word in the sentence, according to the *entanglement.* Multiple iterations can be run to improve performance.

Automated Dictionary generation: This component acts independent of the above phases. Apart from using the clues already defined in the dictionary, a learning module scans the training corpus and performs automatic dictionary generation based on the proximity and frequency of a word near an entity in the training corpus. The process of Maximum Likelihood estimation (MLE) is used for performing this process.

The dictionary generation module inputs a format example, and details such as how many words near an entity it is supposed to scan. If clues appear frequently near the entity, then they are added on to the dictionary. The learning is semi-supervised, as the output is first scanned before it is inserted into the dictionary.

4. Experiments & Results

Two standard performance measures for NER are adopted – **Precision and Recall**, defined below (according to their definition at the MUC-7 conference [5])

Precision = No. of entities correct / No. of entities actually identified 4.1

Recall = No. of entities correct / No. of actual entities as identified by a human 4.2

For each named entity type, the precision and recall values were computed at the end of the deductive phase (**shallow parsing – depicted as S.P.)** and at the end of three stages (**hybrid approach – depicted as FINAL**). These values are then charted and the graphs are depicted below:

4.1. Shallow parsing

Three different test corpus with 1000, 2000 and 2500 words size are used for testing and tabulated results as follows

a) Precision

Words / NE	1000	2000	2500
PER	70.37	71.87	75.93
LOC	62.5	69.23	66.24
ORG	46.31	41.42	47.53
DATE	100	72.72	65.38

TIME	75	80	71.42
CURR	66.66	63.63	73.68
OVERALL	70.14	66.48	66.7

b) Recall

Words / NE	**1000**	**2000**	**2500**
PER	63.33	61.6	66.88
LOC	59.38	66.25	62.26
ORG	43.8	36.07	41.34
DATE	100	61.53	56.66
TIME	50	57.14	50
CURR	66.66	63.63	73.68
OVERALL	63.86	57.7	58.47

Table 4.1 Precision and recall values of NER with shallow Parsing.

4.2. Shallow, Semantic, and E-M

The same test corpus with 1000, 2000 and 2500 words size are used for testing and tabulated results as follows

a) **Precision**

Words / NE	**1000**	**2000**	**2500**
PER	82.35	83.63	86.56
LOC	89.06	91	89.25
ORG	52.17	59.18	60
DATE	100	88.88	86.36
TIME	75	62.5	66.66
CURR	100	100	100
OVERALL	81.81	82.29	83.01

b) **Recall**

Words / NE	**1000**	**2000**	**2500**
PER	68.85	69.69	72.05
LOC	64.77	65.46	61.71
ORG	52.17	47.54	46.98
DATE	100	66.66	73.07
TIME	50	50	54.54
CURR	100	100	100
OVERALL	66.31	64.75	64.7

Table 4.3 showing precision and recall values. Almost 10 to 12% improvement in precision and recall values using a hybrid approach.

The precision and recall values over all the entity types taken as an average are charted below. As the chart depicts, there was a 24% improvement in Precision and 10% improvement in Recall

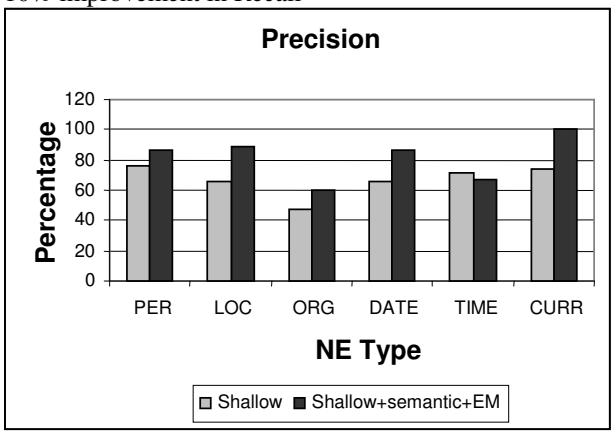

Fig 4.1. NE type wise Precision comparison for NER with Shallow parsing and NER with Shallow Semantic and Modified EM

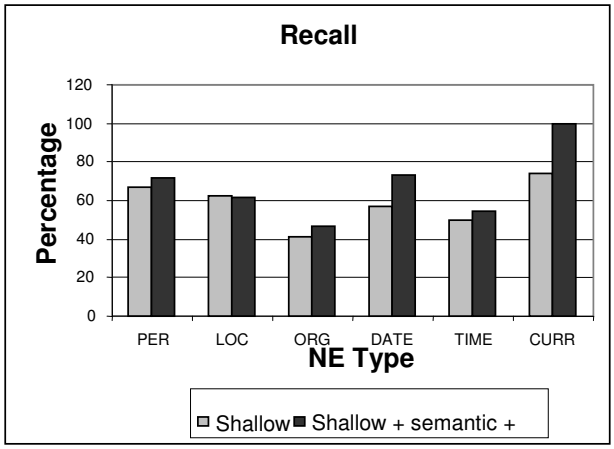

Fig 4.2. NE type wise Recall comparison for NER with shallow parsing and NER with Shallow, Semantic and Modified EM

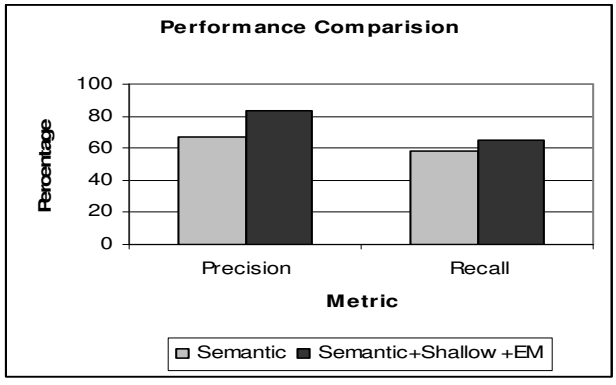

Fig 4.3. The overall Precision and recall comparison for NER with shallow parsing and NER with Shallow, Semantic and Modified EM

Finding the number of missed entities: the number of entities present in the input document is 2500 words and missed by the NER were tracked in both phases. There was a 28.5 % reduction in the number of missed entities, as depicted in fig 4.4

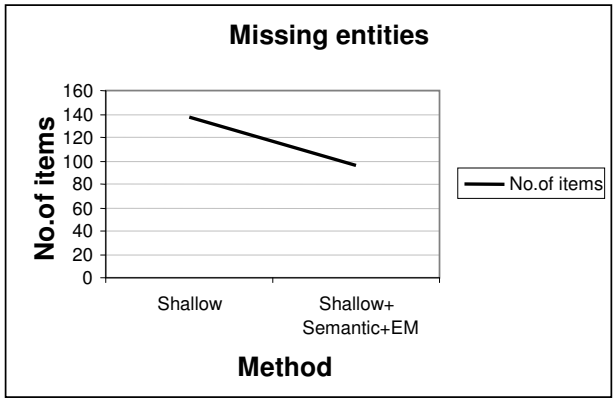

Fig 4.4 The comparison of No. of Entities missed to recognize for both NER

5. Conclusions

It was found that using the contextual cues provided by Tamil words to assign initial probabilities led to higher values of precision and recall than if the seed probabilities were initialized at random. Therefore the task of NER in Tamil is greatly simplified by the use of a mix of linguistic and statistic methods.

The accuracy of the NER is highly dependent on the training corpus and best results are obtained when the test corpus is similar to the training corpus. Further work includes adapting the algorithm to tri-gram and n-gram models and refining the technique for smoothing using conditional transitions through the trellis.

References

[1] James Allen, *Natural Language Understanding, Second Edition,* Benjamin Cummings Pulishing Company, 1995.

[2] David Berger, James Della Pietra, and Stephen Della Pietra A Maximum Entropy Approach to Natural Language Processing.

[3] Colin Cherry; Shane Bergsma An Expectation Maximization Approach to Pronoun Resolution. In: *Proceedings of CoNLL-2005*,

[4] Chinchor Named entity recognition task guidelines- In*: Proceedings of the MUC-7 conference 1997*

[5] Chinchor Named entity recognition evaluation scheme - *Proceedings of the MUC-7 conference1997*

[6] Silviu Cucerzan and David Yarowsky, Language Independent NER using a Unified Model of Internal and Contextual Evidence. In: *Proceedings of CoNLL- 2002, Taipei, Taiwan, 2002*, pp. 171-174.

[7] P. Dempster, N. M. Laird, and D. B. Rubin. Maximum likelihood from incomplete data via the em algorithm. *Journal of the Royal Statistical Society*:

[8] Dan Klein, Joseph Smarr, Huy Nguyen and Christopher D. Manning Named Entity Recognition with Character-Level Models In: *Proceedings of CoNLL . 2003*

[9] Christopher D. Manning and Hinrich Schutze , .Foundations of Statistical Natural Language Processing*., MIT Press, 1999.*

[10] Jon Patrick, Casey Whitelaw and Robert Munro, SLINERC: The Sydney Language-Independent Named Entity Recogniser and Classifier. In: *Proceedings of CoNLL-2002, Taipei, Taiwan*, 2002, pp. 199-202.

[11] Lakshmana Pandian, T.V.Geetha, Krishnan, .Named Entity Recognition in Tamil using context cues and EM algorithm. ,Indian International conference on Artificial Intelligence (IICAI 2007).

[12]Vijayakrishna R , Sobha L , .Domain Focused Named Entity Recognizer for Tamil Using Conditional Random Fields. Proceedings of the IJCNLP-08 Workshop on NER for South and South East Asian Languages, pages 59.66, Hydrabad 2008

[13]Erik F. Tjong Kim Sang,Fien De Meulder Introduction to the CoNLL-2003 Shared Task: Language-Independent Named Entity Recognition In Proceedings of CoNLL-2003 Edmonton, Canada, 2003

[14] B. Megyesi. Shallow parsing with pos taggers and linguistic knowledge. Journal of Machine Learning Research, 2002.

[15] James Mayfield,Paul Mc Namee and Christine Piako, Named Entity Recognition using Hundreds of Thousands of Features , Proc of CoNLL-2003, Edmonton,Canada, pp. 184-187,2003.

Appendix – I - Tamil International phonetic Alphabet

Consonant	ISO 15919	IPA
க்	k	[k], [g], [x], [ɣ], [h]
ங்	ṅ	[ŋ]
ச்	c	[ʧ], [ʤ], [ʃ], [s], [ʒ]
ஞ்	ñ	[ɲ]
ட்	ṭ	[ʈ], [ɖ], [ɽ]
ண்	ṇ	[ɳ]
த்	t	[t], [d], [ð]
ந்	n	[n]
ப்	p	[p], [b], [β]
ம்	m	[m]
ய்	y	[j]
ர்	r	[ɾ]

ல்	l	[l]
வ்	v	[v]
ழ்	ẓ , l̲ , ṛ	[ɻ]
ள்	l̤	[l̤]
ற்	r̲ , R	[r], [t], [d]
ன்	n̲ , N	[n]

Vowel	ISO 15919	IPA
அ	a	[ʌ]
ஆ	ā	[aː]
இ	i	[i]
ஈ	ī	[i:]
உ	u	[u], [ɯ]
ஊ	ū	[u:]
எ	e	[e]
ஏ	ē	[e:]
ஐ	ai	[ʌj]
ஒ	o	[o]
ஓ	ō	[o:]
ஔ	au	[ʌʋ]

Consonant	ISO 15919	IPA
ஜ	j	[ʤ]
ஷ	ṣ	[ʂ]
ஸ	s	[s]
ஹ	h	[h]
க்ஷ	kṣ	[kʂ]

Compound form	ISO 15919	IPA
க	ka	[kʌ]
கா	kā	[kaː]
கி	ki	[ki]
கீ	kī	[ki:]
கு	ku	[ku], [kɯ]
கூ	kū	[ku:]

Tamil Characters

ஃ	அ	ஆ	இ	ஈ	உ	ஊ	எ	ஏ	ஐ	ஒ	ஓ	ஔ
க்	க	கா	கி	கீ	கு	கூ	கெ	கே	கை	கொ	கோ	கௌ
ங்	ங	ஙா	ஙி	ஙீ	ஙு	ஙூ	ஙெ	ஙே	ஙை	ஙொ	ஙோ	ஙௌ
ச்	ச	சா	சி	சீ	சு	சூ	செ	சே	சை	சொ	சோ	சௌ
ஞ்	ஞ	ஞா	ஞி	ஞீ	ஞு	ஞூ	ஞெ	ஞே	ஞை	ஞொ	ஞோ	ஞௌ
ட்	ட	டா	டி	டீ	டு	டூ	டெ	டே	டை	டொ	டோ	டௌ
ண்	ண	ணா	ணி	ணீ	ணு	ணூ	ணெ	ணே	ணை	ணொ	ணோ	ணௌ
த்	த	தா	தி	தீ	து	தூ	தெ	தே	தை	தொ	தோ	தௌ
ந்	ந	நா	நி	நீ	நு	நூ	நெ	நே	நை	நொ	நோ	நௌ
ப்	ப	பா	பி	பீ	பு	பூ	பெ	பே	பை	பொ	போ	பௌ
ம்	ம	மா	மி	மீ	மு	மூ	மெ	மே	மை	மொ	மோ	மௌ
ய்	ய	யா	யி	யீ	யு	யூ	யெ	யே	யை	யொ	யோ	யௌ
ர்	ர	ரா	ரி	ரீ	ரு	ரூ	ரெ	ரே	ரை	ரொ	ரோ	ரௌ
ல்	ல	லா	லி	லீ	லு	லூ	லெ	லே	லை	லொ	லோ	லௌ
வ்	வ	வா	வி	வீ	வு	வூ	வெ	வே	வை	வொ	வோ	வௌ
ழ்	ழ	ழா	ழி	ழீ	ழு	ழூ	ழெ	ழே	ழை	ழொ	ழோ	ழௌ
ள்	ள	ளா	ளி	ளீ	ளு	ளூ	ளெ	ளே	ளை	ளொ	ளோ	ளௌ
ற்	ற	றா	றி	றீ	று	றூ	றெ	றே	றை	றொ	றோ	றௌ
ன்	ன	னா	னி	னீ	னு	னூ	னெ	னே	னை	னொ	னோ	னௌ