



An Analysis of Colloquial Computational Tamil Prose Features in NLP Using Untagged Patterns in Grammar Checker

Dr.K.Nirmala¹, M.K.Kalpana²

¹Associate Professor, ²Research Scholar

^{1,2}Department of Computer Science, Quaid-E-Millath Government College For Women (Autonomous),
Chennai-600 002, Tamil Nadu, INDIA.

Abstract: A Tamil is a south-Indian language, this research paper is customized its work in Modern Tamil Grammar checker. Grammar checker is one of the parts in Spell checker, Grammar Checker also differentiate from patterns. The patterns are known as Prose, Poetry and Novel etc., this paper, the research is explained the pattern and grammar features about the prose. All over the world language communication is available through spoken and written. For a formal documentation, people are using prose-based grammatical formats or patterns. In Modern or colloquial Tamil Prose-based features are Punctuation, Transliteration, Sandhi, Word-based Morphology and Words Understanding. These are the processes are available to create a Prose-based Grammar checker.

Keywords: Natural Language Processing, Modern Tamil, Rule based Machine Translation, Computational Prose Features, Untagged Patterns and Grammar Checker.

I. Introduction

Recent Tamil Computational Prose is put the punctuation to the capable places in given grammatical terms in a system. It is translated indirectly the input text into intermediate text for passing the grammatical rules, Sandhi is a set of grammar rules and consisting few process like insertion, deletion and alternation. Word-based morphology is the enhanced stage compared than the previous stage of the Sandhi. Finally, the word-based understanding also a part to create a Tamil Prose-based Grammar Checker in Tamil NLP.

II. Natural Language Processing

NLP stands for Natural Language Processing, this process is feeding a particular language pattern to produce an application, and it is differentiated upon the creativity or idea. NLP is applicable in mechatronics, computer science, robotics and electrical and electronics etc., Nowadays, machines are controlled by the computer programs, for an example humanoid robot. Apart from English, Indic script is having responsibilities to create a number of application like English language.

III. Modern Tamil

Modern Tamil is a colloquial (spoken) language in south India. Few years before, Tamil people were used Classical Tamil and Middle Tamil. Each category of Tamil is having different dimensions and functionalities. This research is took Modern Tamil data for the uses of creating formal text-based application. Compared than other category of the Tamil, the Modern Tamil application is used to the people for their formal needs. It is allowing north-indian letters also. This is lightly following English style to preparing the documentation, sentence making etc.,.

IV. Rule-Based Machine Translation

Tanveer Siddiqui, U. S. Tiwary says, "Rule-based Machine Translation systems parse the source text and produce an intermediate representation, which may be a parse tree or some abstract representation. The target language text is generated from the intermediate representation. These systems rely on specification of rules of morphology, syntax, lexical selection and transfer, semantic analysis and generation, and are hence called rule-based systems.

Interlingua machine translation

In interlingua-based machine translation approach, the source language text is converted into a language independent meaning representation called 'interlingua'."

V. Methods Of Computational Prose Features In Tamil

A. Tamil Transliteration:

A word from one language is written in other way in another language, why because, each and every language syllables procedure and word structure was different. One word of a language is matched with another language spells and letters is named as transliteration.

Transliteration Demands

When a new word formation and translation procedures are difficult. On that time we use transliteration for the temporary relief. When the new words are comes on colloquial practices. The transliteration is automatically hidden slowly from the utilities.

Analytical Results: Tamil having 300 letters including North-Indian languages like Hindi, Sanskrit etc., Each and Every letter is having intermediate source language text; it is named as Transliteration or Bi-lingual Transliteration or Indirect Machine Translation or Interlingua-Machine Translation. This Method is used to passing the rules. Under the following the table can explained about it briefly.

Example Table of Tamil Transliteration

Sl.No	Diacritical letters	Tamil Letters
1	Kā	கா
2	ṅā	ஙா
3	Cā	சா
4	ñā	ஞா
5	ṭā	டா
6	ṇā	ணா
7	ṭā	தா
8	ṇā	நா
9	pā	பா
10	mā	மா

Evaluation and Results: The Evaluation of Transliteration system can be done manually or automatically by the use of metrics like Transliteration Accuracy. For evaluating the performance of this transliteration system, transliteration accuracy of the transliterated system is checked by determining the total number of Tamil input data generated divided by total number of English generated transliterations.

$$\text{Transliteration Accuracy} = \frac{\text{Total No. of Language input data}}{\text{Total No. of converted English transliterated data.}} \times 100$$

$$= 300/300 \times 100 = 100\%$$

B. Modern Tamil Sandhi Rules Generator:

Analytical Results: Hybrid Model:[3]

In computer-based information systems, errors of sandhi rules constitute a very common source of variation between strings. These errors have been widely investigated. All investigations agree that single character insertion, deletion and alternation are the most common sandhi rules mistakes. In an early investigation, Damearu(1964) reported that over 80% of the typing errors were single-error misspellings:

- (1)substitution of a single letter,
- (2)deletion of a single letter
- (3)insertion of a single letter and
- (4)transposition of two adjacent letters.

Shafer and Hardwick (1968) found that the most common type of single character error was substitution, followed by omission of a letter, and then insertion of a letter, Single character omission occurs when a single is missed (deleted). Insertion error refers to the presence of an extra character in a word; Substitution error occurs when a wrong letter is typed in place of the right one and reversal/transposition of two adjacent letters refers to a situation in which the sequence of characters is reversed.

Context dependent error detection and correction methods, utilize the context of a word to detect and correct errors. This requires grammatical analysis and is thus more complex and language dependent. Even in context dependent methods, the list of candidate words must first be obtained using an isolated-word method before making a selection depending on the context.

The spelling correction algorithm has been broadly categorized by kukich(1992) as follows.

Minimum edit distance The minimum edit distance between two strings is the minimum number of operations (insertions, deletions, or substitutions) required to transform one string into another. Spelling correction algorithms based on minimum edit distance are the most studied algorithms.

Rule based techniques in spelling corrections In a rule-based technique, a set of rules (heuristics) derived from knowledge of a common spelling error pattern is used to transform misspelled words into valid words. For example, if it is known that many errors occur from the letters *g* is typed as *t*, then we may write the rule that represents this.

Minimum Edit Distance The minimum edit distance is the number of insertions, deletions and substitutions required to change one string into another (Wagner and Fischer 1974). When we talk about distance between two strings, we are talking of the minimum edit distance between 'anga calai' and 'antacalai' is 2: We substitute 't' for 'g' and insert 'c' before 'c'. No smaller edit sequence can be found for this conversion. Therefore, the minimum edit distance is 2. Edit distance between two strings can be represented as a binary function; it maps two strings to their edit distance.

Edit distance can be viewed as a string alignment problem. By aligning two strings, we can measure the degree to which they match. There may be more than one possible alignment between two strings. The best possible alignment corresponds to the minimum edit distance between the strings. The alignment shown here, between anga calai and antacalai, has a distance of 2.

a	n	g	a	—	c	a	l	a	i
a	n	t	a	c	c	a	l	a	i

A dash in the upper string indicates insertion. A substitution occurs when the two alignment symbols do not match (shown in bold). We can associate a weight or cost with each operation. The *levenshtein* distance between two sequences is obtained by assigning a unit cost to each operation. Another possible alignment for this sequence is:

a	n	—	a	—	c	a	l	a	i
a	n	t	a	c	c	a	l	a	i

which has a cost of 3. We already have a better alignment than this one.

The problem of finding minimum edit distance seems quite but in fact is not so. A choice that seems good initially might lead to problems later. Dynamic programming algorithms can be quite useful for finding minimum edit distance between two sequences. Dynamic programming refers to a class of algorithms that apply a table-driven approach to solve the problems by combining solutions to sub-problems. The dynamic programming algorithm for minimum edit distance is implemented by creating an edit distance matrix. This matrix has one row for each symbol in the source string and one column for each matrix in the target string. The (i, j)th cell in this matrix represents the distance between the first i character of the source and the first j character of the target string. Each cell can be computed as a simple function of its surrounding cells. Thus, by starting at the beginning of the matrix, it is possible to fill each entry iteratively. The value in each cell is computed in terms of three possible paths.

$$\text{dist}[i,j] = \begin{cases} \text{dist}[i-1,j] + \text{insert_cost}, \\ \text{dist}[i-1,j-1] + \text{subst_cost}[source\ i, target\ j] \\ \text{dist}[i,j-1] + \text{delete_cost} \end{cases}$$

The substitution will be 0 if the character in the source matches with character in the target. The minimum edit distance algorithm is shown in Fig 1. How the algorithm computes the minimum edit distance between anga calai and antacalai is shown in Fig 2.

```

Input: Two strings, X and Y
Output: The minimum edit distance between X and Y
m<---length(X)
n<---length(Y)
for i=0 to m do
  dist[i,0]<---i
for j=0 to n do
  dist[0,j]<---j
for i=0 to m do
  for j=0 to n do
    dist[i-1,j]+insert_cost,
    dist[i,j]=dist[i-1,j-1]+subst_cost[Xi,Yj]
    dist[i,j-1]+delete_cost}
    
```

Figure 1. Minimum edit distance algorithm

	#	a	N	t	a	c	c	A	l	a	i
#	0	1	2	3	4	5	6	7	8	9	10
a	1	0	1	2	3	4	5	6	7	8	9
n	2	1	0	1	2	3	4	5	6	7	8
g	3	2	1	0	1	2	3	4	5	6	7
a	4	3	2	1	0	1	2	3	4	5	6
	5	4	3	2	1	1	2	3	4	5	6
c	6	5	4	3	2	1	2	3	4	5	6
a	7	6	5	4	3	2	1	2	3	4	5
l	8	7	6	5	4	3	2	1	2	3	4
a	9	8	7	6	5	4	3	2	1	2	3
i	10	9	8	7	6	5	4	3	2	1	2

Figure 2. Computing minimum edit distance

Minimum edit distance algorithms are also useful for determining accuracy in Modern Tamil Sandhi Rules Generator systems. Kemal Oflazer (1996) proposed an efficient algorithm based on spelling algorithm based on spelling correction with finite state automata.

Table of Prose-based Sandhi Rules Generator

Parametric Study of Sandhi Rules Generator for Modern Tamil					
Insertion and Alternation Sandhi Input	Insertion Sandhi letters	Insertion Sandhi Output	Alternation/Replacement	Deletion Input	Alternation/Replacement If any unwanted letters came between two words and single words, replaced with space/without space.
anta cālai	c	antaccālai	FALSE	nalla katai	TRUE
inta tērtal	t	intattērtal	FALSE	iṇṇa peyar	TRUE
enta pakkam	p	entappakkam	FALSE	iṅraiya tamil	TRUE
pōka kaṇṭēṅ	k	pōkakkāṇṭēṅ	FALSE	paṭitta puttakam	TRUE
vara col	c	varaccol	FALSE	eḷutāta katai	TRUE
paṭikka paṭikka	p	paṭikkappaṭikka	FALSE	kaṇṭa iṭattil	TRUE
viṭa kūrmai	k	viṭakkūrmai	FALSE	pōka eḷuntēṅ	TRUE
carivara kēḷ	k	carivarakkēḷ	FALSE	palā ilaiyil	TRUE
kūṭa koṭu	k	kūṭakkoṭu	FALSE	mātiri pēcu	TRUE
mika cari	c	mikaccari	FALSE	ceṭi koṭi	TRUE
pōla peritu	p	pōlapperitu	FALSE	kuṭi taṇṇir	TRUE
āka colli	c	ākaccolli	FALSE	veṭi poruḷ	TRUE
avaṇāka pēca	p	avaṇākap pēca	FALSE	ceṭi kaḷ	TRUE
ciṇṇa kuṭai	k	ciṇṇak kuṭai	FALSE	moḷi iyalpu	TRUE

Sandhi rules is a basic grammar rule In Tamil language. It is used to joining the two words or single word using space and without space, insertion, deletion and alternation. This method is explained in about table. In this Modern Tamil Sandhi Rules Generator is categorized Sandhi into two, one is ‘aga sandhi’, it is otherwise called internal joining of single word using insertion, deletion and alternation, and the another is named as ‘pura sandhi’, it is otherwise called external joining of the two words using insertion, deletion and alternation.

Example of Aga sandhi:

Input: āṭu

Result: āṭṭut

Example of Pura sandhi:

Input: anta cālai

Result: antaccālai

C. Word-Based Morphology Generator:

Morphology is classified into three:

1. Word-based Morphology
2. Lexeme-based Morphology and
3. Morpheme-based Morphology.

In these above three categories, this paper explained about Word-based Morphology, it is also otherwise called untagged data or word-based paradigm. In Modern Tamil, the Word-based Morphology concentrates under two methods. The first methods are *Words in Grammatical Terms* and the second methods are *Procedure for writing individual words*.

Table of Word-based Morphology Generator

Method 1:

Words in Modern Tamil Grammatical Terms	
Noun+Noun peyar+peyar	kaṇṇacaivu(kaṇ+acaivu)
	maṇṇacaivu(maṇ+acaivu)
	meyyaṇpu(mey+ṇpu)
	kallaṭi(kal+aṭi)
	uḷḷarai(uḷ+aṇai)
	ticcūṭar
	mākkōlam
	pūccetī

Method 2:

Procedure for writing individual words	
a	akkālattil
	accamayam
	attakuti
	avvūril
aṭai	nāṇ viyappaṭaintēṇ
	nāṇkaḷ cōrvaṭaintuviṭṭōm
	nāṇ makiḷcci aṭaintēṇ.
	nāṇkaḷ perumai aṭaikirōm.
	nāṇ kuḷappam aṭaintirukkiṭēṇ.
arukil, arukē	katavarukil
	kiṇṇararukē
alla	eṇ kaṇṇalla, atait tantuviṭu!
	atu oruporuṭṭalla
	itu vākkuṭi alla
	periya paṇkaḷā alla
	makaḷ alla, marumakaḷ.

4. Modern Tamil Punctuation Rules In Prose-Based Grammar Checker: When people are spoken on times we gave small and large time intervals, between the words. In these ways, we should avoid the confusions in news communication. These are the same procedures, we following in the time of writing the sentences; we place the punctuation marks to the equal of spoken communication in written. This written communication with punctuation marks used in poetry and prose. But, in Tamil poetry, any references did not mention about the punctuation marks is mostly used for the formal document preparation for the colloquial Tamil/Prose. In Tamil, a grammatical rule is set the Punctuation Marks in a particular position like a English. Because, the official document preparation this modern Tamil was formatted by the government bodies. Every punctuation is having a vast dimension in Tamil Grammar.

5. Modern Tamil Word Sense Disambiguation:

Table of Untagged Word-Sense Disambiguation

Modern Tamil Word Sense Disambiguation in the Terms of Meaning							Modern Tamil Word Sense Disambiguation in the Terms of Grammar			
A Modern Tamil word with a multiple meaning	One meaning given many words	Modern Tamil words with Ambiguity	Happenings on words joining	Noun word become Verb action	Ambiguity in hereditary sentences	sentence is not in the position	Ambiguity in prepositions and adjoining words		Grammar matching demands	
kāṣmīr, ūṭṭi enṇu kuḷṟvāṇa ṟattirku alaittucentṟāl nallatu.	kavitā ... koṇca vāratukku mugṇāla oru vārap pattirikaṟiyā cīṟukatai eḷutiyiruntā	nam periyavarkaḷ, pakavāṇavita āccāriyarkuḷ uttāṇ, atāvatu pakavāṇaḷ kāṭṭikkotukku m kuruvukuttā ṇ atika matippu koṭṭukkiṟkaḷ	inta nilayil, 1990ām āṇṭukkuḷ āṇaivarukku m kuṭinir vaḷaṅkum ilāṅkai eṭṭuvatu kāṭṭikkotukku kat tikalṟatu.	māvattat talainaka rkaḷil māvatta āṭciyar makaluṭaṇ aluvalak am mugpāk a ... ārpāṭṭa mceyvat u eṇa muṭivāṇ iyatu.	in hereditary sentences makaluṭaṇ kūṭavē paṭappitipp ukku vantu tayāṇipāḷar kalṇ pirāṇaṇaṇ piyikkum naṭattirat tāymārkaḷ. perṇaṇ.	napikaḷṇ uraiyaḷ kēṭṭu arīyāmaiyl mūḷkiyirunta utpā meyyaṇivu perṇaṇ.	vakkil maṇaiviy ait tī vattuk kolla muyarci		eḷuvāy, payaṇilai uṇavu vāṅkuvataṟkāka eppōtumē raṣiyāvil kāṇappaṭum kiyū ippōtu mēlum niḷamākiṇiṭṭaṇa.	

Table of Untagged Word-Sense Disambiguation is describing that the table is classified the ambiguity into two major parts:

1. Modern Tamil ambiguity in the terms of meaning
2. Modern Tamil ambiguity in the terms of grammar.

1. Modern Tamil WSD in the terms of meaning:

This part consists of ambiguities in these below mentioned categories:

- i. A Modern Tamil word with a multiple meaning,
- ii. One meaning given many words,
- iii. Modern Tamil words with Ambiguity,
- iv. Happenings on words joining,
- v. Noun word become Verb action,
- vi. Ambiguity in hereditary sentences and
- vii. Sentence is not in the position.

2. Modern Tamil WSD in the terms of grammar:

This part consists of ambiguities in these below mentioned categories:

- i. Ambiguity in prepositions and adjoining words,
- ii. Grammar matching demands.

VI. Untagged Pattern

Tagged patterns are named an entities and processing the functionalities of the application using querying, it is querying the data through labeling. Labelling having a common name of the particular set. For an example, <noun>Kalpana</noun> like this tag several entities available in a customized corpus. Like a same way, other kinds of sets also available. Using this, common name of a single set patterns are recognized. But, in a tagged method accuracy is less than untagged corpus compare than tagged corpus or information. Each entities are concerntrates in a untagged patterns, in this way end user can get highest accuracy from an usage of text-based applications.

Example:

PUNCTUATION:

Comma(.) or kālpuḷḷi:

Input Sentence: nīṅkaḷ veliṇiṭṭirunta nakaiccuvaṭṭi tuṇukkukalaṇ paṭittēṇ racittēṇ ciritēṇ(without comma).

Pattern: \w+(?<=ttēṇ)(?=(\w\w\w)+\${,})

Output Sentence: nīnkaḷ veḷiyiṭṭirunta nakaiccuvaṭ tuṇukkukaḷaip paṭittēṇ, racittēṇ, ciritēṇ(inserted comma using pattern).

VII. Conclusion

This untagged based colloquial or Modern Tamil Prose-based Grammar checker is querying each and every grammatical term of the sentences using patterns or expressions. This customized Grammar checker is processing with pattern in WSD and Punctuation. But, in sandhi and transliteration it is functioning in a way of word-based paradigm in a rule-based machine translation. These are the technical analysis bringing next to the stage of implementation.

References

- [1] Machine Learning of Sandhi Rules for Tamil, K.Rajan, Dr.V.Ramalingam, Dr.M.Ganesan, 2012.
- [2] Dhanalakshmi, Velliangiri, et al. "Grammar Teaching Tools for Tamil Language." Technology for Education (T4E), 2010 International Conference on. IEEE, 2010.
- [3] "Natural Language Processing and Information Retrieval", Tanveer Siddiqui, U. S. Tiwary, 2008.
- [4] "Morphological Analyzer for Classical Tamil Texts - A Rule based approach: special Reference to Plural Markers", R.Akilan, Prof E.R.Naganathan, Dr. G.Palanirajan,V7, 2011.
- [5] V.Dhanalakshmi and Anand kumar M, Rajendran S & Somu KP, Natural Language Processing Tools for Tamil Grammar Learning and Teaching, 2010.Vol 8.
- [6] Entity Disambiguation with Freebase, Zhicheng Zheng, Xiance Si, Fangtao Li, Edward Y. Chang, Xiaoyan Zhu, The 2012 IEEE/WIC/ACM International Conference on Web Intelligence (WI'2012) (to appear)..
- [7] <https://en.wikipedia.org/wiki/Punctuation>.
- [8] YJoshi Shripad S., "Sandhi Splitting of Marathi Compound Words", 2013, Vol2.
- [9] Burston, Jack. "Exploiting the potential of a computer-based grammar checker in conjunction with self-monitoring strategies with advanced level students of French." Calico Journal 18.3 (2013): 499-515.