Dependency Parsing in a Morphological rich language, Tamil

Vijay Sundar Ram R and Sobha Lalitha Devi
AU-KBC Research Centre
MIT Campus of Anna University
Chennai
sobha@au-kbc.org

Abstract

Dependency parsing is the process of analysing the grammatical structure of a sentence based on the dependencies between the words in a sentence. The annotation of dependency parsing is done using different formalisms at word-level namely Universal Dependencies and chunk-level namely AnnaCorra. Though dependency parsing is deeply dealt in languages such as English, Czech etc the same cannot be adopted for the morphologically rich and agglutinative languages. In this paper, we discuss the development of a dependency parser for Tamil, a South Dravidian language. The different characteristics of the language make this task a challenging task. Tamil, a morphologically rich and agglutinative language, has copula drop, accusative and genitive case drop and pro-drop. Coordinative constructions are introduced by affixation of morpheme 'um'. Embedded clausal structures are common in relative participle and complementizer clauses. In this paper, we have discussed our approach to handle some of these challenges. We have used Malt parser, a supervised learning- approach based implementation. We have obtained an accuracy of 79.27% for Unlabelled Attachment Score, 73.64% for Labelled Attachment Score and 68.82% for Labelled Accuracy.

1 Introduction

Dependency parsing is the process of analysing the grammatical structure of a sentence based on the dependencies between the words in a sentence. It gives the necessary information for various sophisticated NLP tasks such as Information Extraction, Machine Translation, and detailed Sentiment Analysis etc. Extensive research in development of Dependency Treebanks and Dependency parsers are done in languages such as English, Czech, French, German, Arabic, Turkish etc. The annotation of dependency relations is done using different formalisms at word-level namely Universal Dependencies and inter and intra chunk-level namely AnnaCorra. In Indian languages, particularly, in Hindi, Telugu and Bengali, there are a good number of publications on Dependency parser development compared to other Indian languages. Lack of dependency relation annotated corpus in most of the Indian languages is the prime reason. In this paper, we have described dependency relation annotation task and the development of dependency parser for Tamil using a Data-driven approach.

The paper is structured as follows. In section 2, we have discussed the previous attempts in the development of dependency parsers in various Indian languages. A brief introduction on the characteristics of Tamil language is given in section 3. In section 4, we describe the annotation of dependency relations in Tamil sentences. Section 5 has the details of our approach for the development of Tamil dependency parser. This is followed by a section containing the details of the experiment, results and discussion. The paper concludes with the conclusion section.

2 Recent Works

Dependency Treebank work in Indian languages started with development of annotation schema for Indian languages developed by Bharati et. al. (2006). Few of the initial works in Indian

Proceedings of the ICON 2021 Workshop on Parsing and its Applications for Indian Languages (PAIL) , pages 20–26
December 16, 2021. Silchar, India.

languages in Dependency parser are as follows. Bharati et. al. (2008) has proposed a framework for dependency parsing for Indian languages using a grammar-driven methodology. They have presented how the rules made for one language can be effectively transferred to other similar languages. Bharati et. al. (2009) presented a two-stage constraint-based approach to dependency parsing. Here the different grammatical constructions were processed at appropriate stages. This algorithm was tested with Hindi Dependency Treebank data.

ICON Tool Contest on Dependency Parsing during 2009 and 2010 boosted the Dependency parser research in languages namely Hindi, Telugu and Bengali. Different Dependency parsers using Grammar-driven approaches, Malt parser and MST parser and hybrid approaches were developed. The detailed report on the tool contents is available in Husain S. et. al. (2009) and Husain S. et. al. (2010). There are substantial works in these languages using the data released in these tool contests. Few of the notable work in these languages are as follows.

Kesidi R. S. et. al. (2011) has presented a two-stage constraint-based approach to Telugu dependency parser, where they perform a selective identification and resolution of the dependency relation at different stages. The ranking strategy using S-constraints is used to get the best parse. Praveen Gatla (2019) has presented a work on development of Telugu annotated corpus and their experiment in developing Telugu dependency parser using Malt parser and MST parser. The annotated treebank had 2424 sentences. Nallani.S et. al. (2020) has presented a simple and effective dependency parser for Telugu using BERT model built using Telugu Wikipedia corpus. They have attempted to use contextual vector representations instead of hand-crafted features using linguistic information such as part-of-speech and morphology.

Naman Jain (2016) has done a considerable study on Hindi Dependency parser development. He has explained two different ensembling approaches namely re-parsing algorithm and word-by-word voting algorithm in improving the Malt parser. Dhar A. et. al. (2012) came up with a two-stage dependency parser for Bengali, where in the second stage, Bangla specific constraints using Bangla verb frames were used. Morphological features, Part-of-speech, Chunk and Named Entity information were used in both stages.

In Tamil, there are very few published works in Dependency parser. Ramaswamy L. and Zabokrtsky Z. (2011) attempted to build Tamil dependency parser using rule-based technique and also using Malt parser and MST parser. Their annotation schema was partially based on Prague Dependency Treebank (PDT). They annotated a corpus of 3000 words. They observed that the both the rule-based and corpus-based approaches performed poorly in identifying the co-ordinate constructions. Sarveswaran K. and Dias G. (2020) has presented a neural-based dependency parser for Tamil namely TamizhiUDp, developed using UUparser engine developed using Styme S. et. al. (2018) for training with heterogeneous treebanks. As they had 600 Tamil Dependency relation annotated sentences, they experimented multilingual training. They jointly trained Tamil data with Hindi HTTB v2.6 sentences. It has 16647 sentences. They also tried training with other languages such as Telugu, Turkish, but they got better result with Hindi data as the data size was bigger. They got 62.39% as Label Attachment Score (LAS) accuracy. Since Telugu MTG Udv2.6 had only 1328 sentences and without morphological information, it did not suit for combined training in their experiment. In the following section, we give a brief introduction on characteristics of Tamil language.

3 General Characteristics of Tamil Language

Tamil belongs to the South Dravidian family of languages. It is a verb final language and allows scrambling. It has postpositions, the genitive precedes the head noun in the genitive phrase and the complementizer follows the embedded clause. Adjective, participial adjectives and free relatives precede the head noun. It is a nominative-accusative language like the other Dravidian languages. The subject of a Tamil sentence is mostly nominative, although there are constructions with certain verbs that require dative subjects. Tamil has Person, Number and Gender (PNG) agreement.

Tamil is a relatively free word order language, but when it comes to noun phrases and clausal constructions it behaves as a fixed word order language. As in other languages, Tamil also has optional and obligatory parts in the noun phrase.

Head noun is obligatory and all other constituents that precede the head noun are optional. Clausal constructions are introduced by non-finite verbs. Complementizer clause occurs with complementizer markers 'endru' and 'ena'. Subject drop occurs in Tamil. The other characteristics of Tamil are copula drop, accusative drop, and genitive drop. Co-ordinate constructions are also introduced with 'um' suffix. In the next section, we will discuss about the annotation of Dependency relation in Tamil sentences.

4 Tamil Dependency Treebank Annotation

As there is no publicly available Tamil Dependency Treebank, we prepared it inhouse. In the present work of annotating Tamil sentences with dependency relations, we have followed the annotation guidelines given in AnnaCorra (Dipti et. al.. 2012). It is based on modifier-modified relationship. The dependency relations are hierarchically defined as inter-chunk and intra-chunk relations. The grammatical relations that are considered in the guidelines are of two types; (1) Karaka, and (2) Relations other than karakas.

'Karakas' are the roles of various participants in an action. An action in a sentence is denoted through a verb. For a noun to hold a karaka relation with a verb, it is important that they (noun and verb) have a direct relation.

4.1 Relations and Tag labels

The scheme contains about 40 tags which are arrived at considering various types of sentence constructions. These labels represent the following relations (a) karaka and non-karaka dependency relations (b) some underspecified tags of the type vmod, nmod etc and (c) some tags which indicate relations which are not exactly dependency relations but are required to represent the sentence structures. The labels are presented in fig 1.

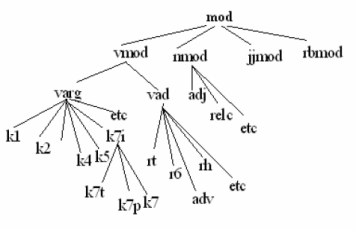

Figure 1: Dependency Relations

S.No	Relation	Meaning
1	k1	Agent / Subject / Doer
2	k2	Theme / Patient / Goal
3	k3	Instrument
4	k4	Recipient / Experiencer
5	k5	Source
6	k7	Spatio-temporal
7	rt	Purpose
8	rh	Cause
9	ras	Associative
10	k*u	Comparative
11	k*s	(Predicative) Noun / Adjective Complements
12	r6	Genitives
13	relc	Modification by Relative Clause
14	rs	Noun Complements (Appositive)
15	adv	Verb modifier
16	adj	Noun modifier

Table 1: Dependency relations and their meaning

To handle the co-ordinate sentence formed by 'um' suffix, and sentences with copula drop, pro-drop, we manually introduce a NULL in that required slot in the sentence and mark the dependency relations. Few of the example sentences are given below.

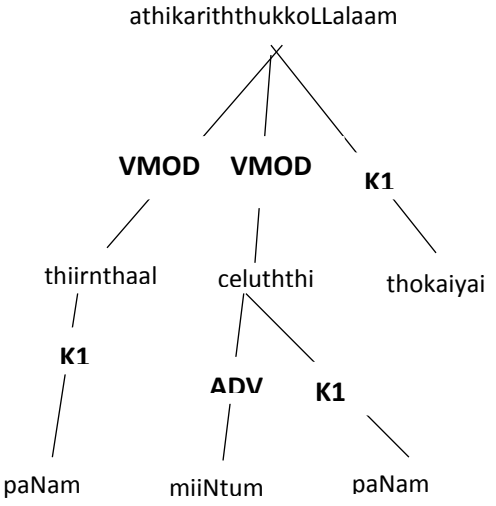

Figure 2: Dependency Tree for example 1

Ex 1:
paNam thiirnthaal miiNtum
Money(N) complete(V+COND) again (Adverb)
paNam celuththi thokaiyai
money(N) pay(V+VBP) amount(N)
athikariththukkoLLalaam .
increase(V+Finite)

(If the money gets emptied, again paying the money, we can increase the amount.)

The sentence in example 1, has a conditional clause ‘paNam thiirnthaal’ (if money gets empty) and a non-finite clause ‘miiNtum paNam celuththi’ (by paying money again). These two clausal verbs are attached to the finite verb has vmod relation. And it is shown in fig 2.

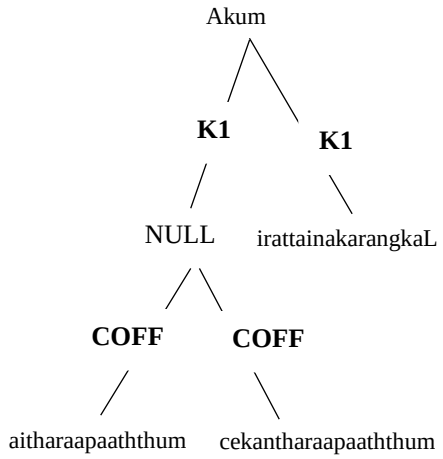

Figure 3: Dependency Tree for example 2

Ex 2:
aitharaapaaththum cekantharaapaaththum
Hyderabad(N+INC) Secunderabad(N+INC)
irattainakarangkaL aakum.
twin-cities be(copula)
(Hyderabad and Secunderabad are twin cities.)

In example 2, the sentence has co-ordination between two nouns aitharaapaaththum (Hyderabad) cekantharaapaaththum (Secunderabad) introduced by ‘um’ suffix. While annotating this sentence, a NULL is introduced between these two nouns and the nouns are related to the NULL with ‘ccof’ relation. The dependency tree for this sentence is given in fig 3.

4.2 Corpus Details

We have collected sentences with different sentence structures from various Tamil web Newspapers. We have annotated 2500 sentences, where 70% of the sentences are simple sentences, 20% of the sentences are two clause sentences, 5% of compound sentences and remaining 5% of the sentences had multi clauses (more than two clauses).

5 Our Approach

We have used Malt parser for developing Tamil dependency parser. Malt parser is a language independent system, which can be used to train for any given language. Maltparser is an implementation of inductive dependency parsing. The syntactic analysis of a sentence amounts to the derivation of dependency structure. It is possible to improve the performance of the system by optimizing the parameter of the transition system and optimizing the features used for the classifier system. Maltparser implementation has nine deterministic parsing algorithms, viz, Nivre arc-eager, Nivre arc-standard, Convington non-projective, Convington projective, Stack projective, Stack swap-eager, Stack swap-lazy, planner and 2-planner. It supports two machine learning packages, LIBSVM and LIBLINEAR. In the present work, we have used arc-eager transition and morphologically rich features. Nivre. J (2009) has presented the importance of morphological information in developing Dependency parser for morphologically rich language namely Telugu. We have used LIBSVM classifier with the following features POS, chunk information along with the root word, case suffix, other suffixes and TAM (tense-aspect-model) and PNG (Person, Number and Gender) information from verbs. This information is obtained from the morphological analysis of the words.

Dependency relation annotated sentences were enriched with morphological information, Part-of-Speech, Chunk and chunk-head information using a robust Morphological analyser build using a paradigm approach, CRFs based part-of-speech tagger and chunker and rule-based head chunk identification module. The head chunk information is required for marking the inter-chunk relation.

As Tamil sentences have copula drop and co-ordinate sentences with ‘um’ suffix, we need to re-structure these sentences by adding NULL in the required positions.

5.1 Copula Drop Handler

We check for the finite verb in the sentence, if it does not exist, NULL is introduced as the last word of the sentence before the punctuation marker. Consider the following example sentence.
Ex 3:
naan doctor.
I am a doctor .

After addition of NULL, the sentence is as follows,

"naan doctor NULL ."

5.2 Co-ordinate Construction Handler

'um' suffix is also used as an emphatic marker. Using linguistic rules, we disambiguate the sentences with emphatic 'um' and co-ordinate suffix 'um'. Co-ordinate suffix 'um' suffix can occur in the series of nouns and verbs. The algorithm to handle this is explained below.

Step1: If sentence has multiple words with 'um' suffixes, then step 2

Step2: If sentence has a series of nouns or verbs with 'um' suffix
then
introduce NULL between the last pair of noun/verb with 'um' suffix.

Consider the sentence in example 4.

Ex 4:
aitharaapaaththum cekantharaapaaththum
Hyderabad(N)+INC Secendrabad(N)+INC
irattainakarangkaL aakum.
twin-cities be(copula)
(Hyderabad and Seceundrabad are twin cities.)
The sentence in example 4, has two nouns with 'um' suffix.

After Correction:
"aitharaapaaththum NULL cekantharaapaaththum irattainakarangkaL aakum. "

Dependency tree for the sentence in example 4 is shown in figure 3.

6 Experiment and Results

The dependency relation annotated sentences were randomly divided into 80% and 20% for training and testing purpose. Both the training data and the testing data are processed with morphological analyser, POS tagger, Chunker and chunk head identification module. Further sentences with Copula drop and sentences with 'um' co-ordinate suffix are corrected by introducing a NULL. This processed data is presented to Malt parser in CoNLL column format for training and testing.

We have evaluated the performance of the model with standard measures namely Label Attachment Score (LAS), Unlabeled Attachment Score (UAS), Labeled Accuracy (LA) metrics. The training data had 2000 sentences and the testing data had 500 sentences. We performed a detailed analysis on the contribution of each of the features in improving the accuracy. The performance scores obtained when introducing different features are given table 2.

S.No	System	Features	UAS	LAS	LA
1	Baseline	Word, root word	61.45	57.29	54.84
2	Baseline +POS	Word, Root word, POS	66.14	61.72	56.34
3	Baseline +POS +Case	Word, root word, POS, Case marker	75.46	71.84	66.23
4	Baseline +POS +Case +Other suffix	Word, root word, POS, Case marker, PNG information, other suffix	79.27	73.64	68.82

Table 2: Performance Scores of different systems

The performance scores in table 2, show the contribution of different features in improving the accuracy of the parsing. Tamil being a morphologically rich language with highly productive suffixation, the information from the processing modules contribute a lot of information. Case markers affixed to the nouns help in determining the semantic role of that noun phrase in that sentence. Both finite and non-finite verbs have clear suffix markers affixed to the verbs. These suffixes are vital features for the classifier. This is evident from the accuracy obtained for 3rd and 4th system, where, case markers and other suffixes are included as features. Though 'vmod' relations are identified properly, the system has poorly identified the Karakas relations in the sub-ordinate clause. Dropping of genitive case and accusative case

markers are common characteristics in Tamil. This affects the proper identification of K2 and r6 relations. It also introduces wrong K1. The sentences with embedded clause were not correctly handled. This requires re-ordering of sentence into linear form to correctly identify the relations.

The Overall performance measures obtained are presented in the following table 3.

UAS	LAS	LA
79.27%	73.64%	68.82%

Table 3: Overall Performance Measures

The performance measures for the frequently occurring tags are given in table 4.

Dep-Rel	Recall	Precision	F-measure
k1	81.64	78.23	78.84
k1s	62.49	64.45	63.45
k2	76.62	72.41	74.46
k2p	74.98	73.12	74.04
k2s	23.92	31.48	27.18
k4	71.53	64.54	67.86
k4a	19.34	56.62	28.83
k7	43.69	49.27	46.32
k7p	63.37	62.74	63.05
k7t	67.84	65.23	66.51
nmod	26.54	68.57	38.27
vmod	82.74	75.61	79.02
adv	77.83	74.81	76.29
Ccof	77.34	73.38	75.31
r6	74.85	67.45	70.96
rh	77.63	71.54	74.46

Table 4: Performance Measures of Major Tags

On analysing the output, we had the following observations.

k2 karaka relation is marked as k1 is a common error. It has happened in the sentences where the accusative case marker is dropped in the object noun phrase. There are instances where k1 is marked as k2. r6, which marks the possessive relation between the nouns were poorly identified. As genitive case drop is common in Tamil sentences, the relation between the nouns were not captured. This led to wrong tagging of dependency relations.

The parser has failed to handle embedded clause sentences, where karaka relation of the clausal verb and the main verb were wrongly attached. In these sentences, the 'vmod' relation between the clausal verb and the main verb were marked correctly.

K4a- karaka relation which refers to the experiencer relation, were wrongly tagged as k4 or k1. And these relations were less frequently occurred in the corpus. The parser has not handled the multiclause sentences correctly. Here the 'vmod' relations were correctly tagged but the karaka relations with clausal verb and main verb were not correctly tagged.

Thus, to improve the parser efficiency, we need to increase the training data. Further the rules should be included to handle genitive case drop, accusative case drop and subject drop. We need to identify the sentence structures using the clausal verb and handle them separately, it will reduce the errors due to wrongly attached NPs with other verbs. We should also use the post-processing rules to improve the parser efficiency.

Conclusion

It is also advised to supplement non-English characters and terms with appropriate transliterations and/or translations since not all readers understand all such characters and terms. We presented a work on developing Dependency parser for Tamil, which is a less attempted task. Tamil, a south Dravidian language, is a morphologically rich and highly agglutinative language. We used Malt parser to train the language model with morphologically rich features. We enrich the sentences with morphological information, PoS tags, Chunks and chunk head information using shallow processing tools. The copula-drop and sentences with co-ordinate 'um' suffixes are rewritten by introducing NULL in the required position in the sentences, before processing it with a parser. Accusative case drop, genitive case drop and subject drop are common in Tamil sentences and these affect the efficiency of the parser. Embedded clause occurs in relative participle clause and complementizer clause sentences and these sentence constructions are poorly handled by the parser. We plan to build a confusion matrix for deeper understanding of the errors.

To improve the efficiency of the parser we plan to train the model with more annotated data and to

add linguistic rules to handle accusative case drop, genitive case-drop and subject drop. Further we plan to improve the algorithm to selectively handle the different sentence structures. Increasing the annotated data substantially, we also plan to train a neural-based dependency parser, where we can use resources such as BERT which provides a contextual vector representation, to build a robust parser.

References

Bharati, A., Sharma, D. M., Bai, L, Sangal, R.: AnnCorra: Annotating Corpora Guidelines for POS and Chunk Annotation for Indian Languages. LTRC-TR31. (2006)

Bharati, A., Husain, S, Sharma, D.M., Sangal, R.: A Two-Stage Constraint Based Dependency Parser for Free Word Order Languages. In: the COLIPS IALP. (2008)

Bharati, A., Husain, S., Vijay, M., Deepak, K., Misra, D., Sangal, R.: Constraint Based Hybrid Approach to Parsing Indian Languages. In: The 23rd Pacific Asia Conference on Language, Information and Computation. Hong Kong (2009)

Dhar. A., Chatterji. S., Sarkar. S., Basu. A. 2012, In: Proceedings of the 10th Workshop on Asian Language Resources, COLING 2012, Mumbai, December 2012, pages 55–64

Husain, S., 2009. Dependency Parsers for Indian Languages. In: Proceedings of the ICON09 NLP Tools Contest: Indian Language Dependency Parsing. India.

Husain, S., Mannem, P., Ambati, B., Gadde, P.2010. The ICON-2010 tools contest on Indian language dependency parsing. In: ICON-2010 tools contest on Indian language dependency parsing. Kharagpur, India

Jain, N., 2016. Advancements to Hindi Dependency Parsing: Semantic Information, Ensembling and PAC (Doctoral dissertation, PhD thesis, International Institute of Information Technology Hyderabad).

Kesidi S.R., Kosaraju. P., Vijay. M. and Husain. S. 2011. A Constraint Based Hybrid Dependency Parser for Telugu. In Proceedings of the 12th CICLing, Tokyo, Japan.

Amba kulkarni, 2021, Sanskrit Parsing Following Indian Theories of Verbal Cognition, ACM Transactions on Asian and Low-Resource Language Information Processing, Vol 20, Number 2, pp 1-38,

Kulkarni A 2019, Sanskrit Parsing based on the theories of Shabdabodha, D. K. PrintWorld and Indian Institute of Advanced Study, August 2019

Nallani, S., M. Shrivastava & D. Sharma. 2020.A Fully Expanded Dependency Treebank for Telugu. Proceedings of the WILDRE5– 5th Workshop on Indian Language Data: Resources and Evaluation. Marseille: Language Resources

Nivre, J., Hall, J., and Nilsson, J. 2006. Maltparser: A data-driven parser-generator for dependency parsing. In Proceedings of LREC, volume 6, pages 2216–2219.

Nivre, J., 2009. Parsing Indian Languages with MaltParser. In: Proceedings of the ICON09 NLP Tools Contest: Indian Language Dependency Parsing.

Praveen Gatla 2019. Dependency Parsing for Telugu Using Data-driven Parsers. Language in India. Vol. 19:1

Ramaswamy L. and Zabokrtsky Z. 2012. Prague dependency style treebank for Tamil. In Proceedings of Eighth International Conference on Language Resources and Evaluation (LREC 2012), pages 1888–1894, Istanbul, Turkey

Sarveswaran K. and Dias. G. 2020. ThamizhiUDp: A Dependency Parser for Tamil. CoRR, abs/2012.13436. https://arxiv.org/abs/2012.13436}