Construction Grammar based Annotation Framework for parsing Tamil

by

Vigneshwaran Muralidharan, Dipti Misra Sharma

in

17th International Conference on Intelligent Text Processing and Computational Linguistics

Konya, Turkey

Report No: IIIT/TR/2016/-1

Centre for Language Technologies Research Centre
International Institute of Information Technology
Hyderabad - 500 032, INDIA
April 2016

Construction Grammar based Annotation Framework for parsing Tamil

Vigneshwaran Muralidaran, Dipti Misra Sharma

International Institute of Information Technology
Gachibowli, Hyderabad, Telengana 500032, India
vigneshwaran.m@research.iiit.ac.in, dipti@iiit.ac.in

Abstract. Syntactic parsing in NLP is the task of working out the grammatical structure of sentences. Some of the purely formal approaches to parsing such as phrase structure grammar, dependency grammar have been successfully employed for a variety of languages. While phrase structure based constituent analysis is possible for fixed order languages such as English, dependency analysis between the grammatical units have been suitable for many free word order languages. These approaches rely on identifying the linguistic units based on their formal syntactic properties and establishing the relationships between such units in the form of a tree. Instead, we characterize every morphosyntactic unit as a mapping between form and function on the lines of *Construction Grammar* and parsing as identification of dependency relations between such conceptual units. Our approach to parser annotation shows an average MALT LAS score of 82.21% on Tamil gold annotated corpus of 935 sentences in a five-fold validation experiment.

1 Introduction

A common theoretical assumption in dependency parser implementations is that grammar is a formal system of rules which arranges the morphosyntactic units that make up a sentence. In dependency grammar analysis, both the identified grammatical units and the dependency relations between them are structural or formal in nature. We observed that Dravidian languages, spoken in South India, have interesting morphological properties which can be better characterized from a functional approach to syntax such as *Construction Grammar*[1], [2] or *Cognitive Grammar*[3]. According to the theoretical assumptions of Construction Grammar, every grammatical formative in a language is analyzed as a meaningful mapping between its form and function. We identified that the set of morphological inflections that occur in Dravidian languages can be mapped to a set of meaningful 'construals' i.e. as a form function pairing. We will talk about what these 'construals' are in section 3. Wherever morphemes can be grouped together as one meaningful construal let us call that as one construction unit. Our idea is to identify which morphemes should be grouped together as one construction unit, identify the appropriate construction label for the grouped unit, chunk these construction units and finally identify the dependency relations between these chunks.

We describe the previous works on Parsing in Indian languages and other morphologically rich languages in section 2, the basics of construction based analysis of a text

in section 3 , the proposed annotation framework in section 4, experiments and results in section 5, error analysis in section 6 and conclusion and future work in section 7.

2 Previous Works

Parsing of natural language texts has been explored in various languages for more than two decades now. Here, we point out to various parsing approaches attempted in Indian languages and other morphologically rich languages so far. Indian languages are morphologically rich, free-word order languages and it is well understood that dependency framework suits better for the analysis of the various grammatical structures of such languages [4], [5], [6]. Akshar Bharati and Rajeev Sangal proposed a grammar formalism called as 'Paninian Grammar Framework' [7] that has been successfully used to create treebanks in Indian languages and extensively used for all free word order Indian languages. A simple parser for Indian languages has been proposed by Akshar Bharati et al. in a grammar-driven methodology [8]. The proposed methodology was based on Paninian grammatical approach in dependency framework. Prashanth Mannem proposed a bidirectional dependency parser for Hindi, Telugu and Bangla language [9].

Joakim Nivre presented the work of optimizing Malt Parser for three Indian languages namely Hindi, Telugu and Bangla in NLP Tool Contest at ICON 2009 [10]. Bharat Ram Ambati et al. explored MALT and MST parsers on three Indian languages Hindi, Telugu and Bangla [11]. A statistical syntactic parser for Kannada has been developed based on Penn Treebank structure [12]. Selvam et al. presented a statistical parser for Tamil language using phrase structure hybrid language model [13] . In 2011, Ramasamy Loganathan and Zdeněk Žabokrtský discussed their results on rule-based and corpus based approaches in Tamil dependency parsing. They report an accuracy of more than 74% for the unlabeled task and more than 65% for labeled tasks on a small corpus size of 204 sentences with 2961 words [14]. Straka et al report an LAS score of 69.7% and UAS score of 78.3% on the Universal Dependencies Treebank[15] for Tamil parsing using 600 sentences.

B. Venkata S.Kumari in 2012 presented a hybrid approach by combining the output of both the MALT and MST in an intuitive way and showed that this can perform better than both the parsers [16]. In 2011, a constraint based Hybrid dependency parser for Telugu was reported [17], with an LAS score of 68.06% with 1119 training data size. For other morphologically rich languages across the world, average LAS scores vary from 91.83%(German language with the largest training set) to around 83% (for Hebrew and Swedish with smallest training data) amongst the languages in SPMRL shared task 2013[18]. Some of the state-of-the-art parsing results reported for a few Indian languages in data-driven approach[1] to the best of our understanding are shown in Table 1.

[1] Hindi, Telugu and Bangla accuracies reported are experimented with Computational Paninian Grammar Framework[7]. The Tamil accuracy reported is based on the Universal Treebank results reported by Straka et al[15]

Table 1: Label accuracy scores(LAS) reported for various Indian languages

Language	LAS	Training size
Hindi	90.99%	12041
Telugu	68.06%	1119
Bangla	79.81%	1000
Tamil	69.7%	600

3 Our Approach

We take inspiration from the theoretical perspectives about syntax in *Cognitive Grammar* [3]. On the outset, we propose that there are two dimensions for understanding any syntactic unit functionally as a construction schema. (i) Composition - what concepts make up a construction schema (ii) Interaction - what kind of dependencies that it has with other such construction schemas. The four basic kind of interactions are shown below in figure 1. To make a fully functional characterization of syntax, the following

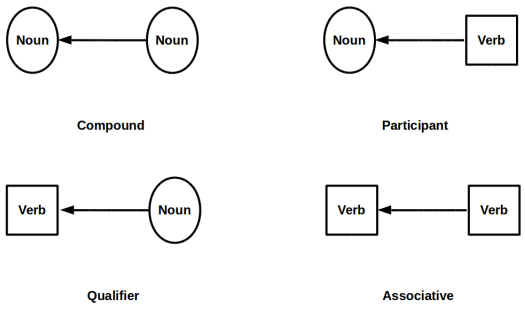

Fig. 1: Basic Types of Interactions in Discourse World

construction schemas are defined.

1. Any morphosyntactic unit that is conceived as outlining a *thing* irrespective of its internal complexity is a 'noun' schema. Anything which is conceived as outlining a *relation* is a 'verb' schema.
2. A relation can be conceived as outlined through time or immediately perceived as a configuration. We refer to the former interpretation as a 'process' and the latter as a 'status'.
3. If a relation attributes its relational properties to a noun, it is called as 'qualifier' schema
4. If a relation attributes its relational properties to another relation , it is called as 'associative' schema

5. If a thing is conceived to be an integral part of another thing, it is said to be in 'combinative' schema
6. If a thing is conceived to be an integral part of a relation, it said to be in 'participant' schema

The above mentioned conceptualizations suggest that every syntactic unit is reducible to two fundamental concepts namely 'thing' and 'relation'. (Refer to [3] to understand these in detail). The four possible ways they are conceived to interact in the speaker's conceptual world are the basis for all kinds of complex discourse interactions. These four kinds of interactions are shown in Figure 1. The direction of the arrow points to the dependent unit in the interaction. These conceptualizations dynamically made by the speaker are called 'construals' and the syntactic patterns that act as symbolic mappings to these construals are called construction schemas. It is these construction schemas that we intend to learn in our approach and it is between these construction schemas that the dependency relations are established. Let us understand these ideas with some examples.

1. I saw a **long** beach. *Status Qualifier schema*
2. The beach was **long**. *Status Pronominal schema*
3. The boy **who lives** near our home, is playing in the beach. *Process Qualifier schema*
4. I met **my** friend in the conference. *Status Qualifier schema*
5. **Three** people entered the classroom. *Status Qualifier schema*
6. **The fact that I spoke** to him on his birthday made my friend happy. *Process Pronominal schema*
7. The book is **mine** *Status Pronominal schema*

In the above examples, the entities which invoke some construction schemas are highlighted in boldface. For example various syntactic units such as *long (adjective), who lives (relative clause), my (personal pronoun), three (numeral quantifier)* in examples 1,3,4 and 5 exhibit a common pattern of describing or attributing some relational property on a noun. Such a discourse 'construal' of attributing some property on a *thing* is the construction pattern that we call as *Qualifier Schema*. Whereas the same adjective *long* in example 2, the predicate form of genitive case pronoun *mine* in 7, construction such as *the fact that I spoke* in example 6 exhibit the discourse 'construal' of creating referring expressions. For instance one can say 'Contrary to my expectation, the beach **was long**. That surprised me'. As you can see, the predicate of *being long* is an idea that you can anaphorically refer to as 'that'. Such a discourse construal of 'referring expressions' are what we call as *Pronominal schema*.

We call it 'pronominal' because a *qualifier* attributed on a generic *thing* creates a referring expression. The whole expression now will stand 'in place of' a generic *thing* in a discourse world and thus 'pronominal'. If the referring expression is from a *process* relation it is *Process pronominal* or if it is from *status* relation then it is *Status pronominal*. Thus we see that, it is not the formal structural properties that we want to capture in our grammatical analysis but the functional concepts. Neither are they entirely semantic from an information point of view. Rather these construals are dynamic viewpoints that

the speaker adopts while describing a given semantic information.

The observation we made was that these kinds of discourse construals are directly encoded through regular morphological inflections in Dravidian languages. For example in Tamil, whenever a *Qualifier Schema* is construed in discourse, morphologically it is expressed as a relative participial inflection(RP) of a verb. Tamil lacks lexical adjectives such as *broad, beautiful, long* and expresses these ideas as verbal participial inflections such as *agalam*-**An-a** *(breadth-become-RP), azhag*-**An-a** *(beauty-become-RP),* **nIND-a** *(elongate-RP)* with RP inflection in each case. But the same expressions, while functioning as predicates, take an additional pronominal suffix and become *agalam*-**An-a-du** *(breadth-become-RP-PronSuffix), azhag*-**An-a-du** *(beauty-become-RP-PronSuffix),* **nIND-a-du** *(elongate-RP-PronSuffix)* - thus creating referring expressions. These RP inflections and pronominal suffixes analogously occur in other parts of speech such as genitive markers, relative clauses, qualifiers, quantifiers and so on. We want to exploit these morphological regularities by mapping them to their construction schema patterns. The table 2 lists out the various formal grammatical units and their mapping to the construction patterns that we identified. As the mappings mentioned in table 2 are

Table 2: Mapping the formal grammatical units to construction schemas

Grammatical units	**Construction schemas**
Adjectives, genitive case markers, relative clauses, appositions, quantifiers as noun modifiers, subordinate clauses as noun modifiers	Qualifier Schema (Both process and status)
Predicate adjectives, genitives as predicates, rel.clause referring expressions, predicate quantifiers, subordinate clauses signaling entity-event relations	Pronoun Schema (Both process and status)
Case markers, prepositions, subordinate and coordinate conjunctions, complementizers, adverbs	Status Associative
Discourse event-event relations, grammatical aspects, modality, complex predicates	Process associative; Subtypes are: conjunctive, concursive, conditional, infinitive
Case assigned nouns	Participant
Nouns which are part of multiword expressions	Combinative

directly available as morphological regularities in Tamil, they can be easily identified for annotation. The following example demonstrates this idea.

(1) veLiy-uRavu **kuRittu** inRu vivAdam naDaipe-tR-adu.
External-affairs **regarding** today debate happen-PST-NonHum.SG
'Regarding (our) external relations, a debate happened (in parliament) today'

(2) veLiy-uRavu **kuRitt-a** vivAdam
External-affairs **regarding-RP** debate
'The debate (which is) about external affiars'

(3) vivAdam veLiy-uRavu **kuRitt-a-du**
Debate External-affairs **regarding-RP-PronSuffix**
'The debate is about external affiars'

As you can see in the above examples, even function words such as postpositions are expressed through verb morphology in Tamil. **kuRittu(about)** inflects as a relative participial form(RP) **kuRitta** when it attributes its property on a noun but takes a pronominal inflection **kuRittadu** when it acts as a predicate. These and other such morphological regularities that consistently map to construal functions form the basis of our annotation labels.

4 Annotation Scheme

A gold annotation of dependency parsing in this Construction Grammar(CG) Framework involves two stages: (a) Processing the raw text into morphosyntactic patterns for which Construction labels have to be annotated (b) Grouping these construction units into chunks and then annotating the dependency relations between these chunks. The list of Construction schemas and the dependency relations that can exist between them are mentioned in tables 3 and 4. A simple example is given below to illustrate the annotation scheme:

(4)

timuka talaivar-um mun-nAL mudalamaiccar-um-**An-a** karuNAnidhi
DMK leader-also pre-day CM-also-**become-RP** Karunanidhi
nEtRu seidiyALargaL-ai sandit-tu pEs-in-Ar.
yesterday reporters-acc met-conjunctive speak-PST-honorific

'Mr. Karunanidhi, the DMK leader and the ex-Chief Minister, met the reporters yesterday and spoke (with them)'

In the above example, there is a proper noun *Karunanidhi* which is described by the apposition phrase *the DMK leader and the ex-Chief Minister* in English. Functionally this invokes a Qualifier schema because the noun is being described by the apposition phrase. In fact, in the above Tamil sentence this qualifier schema is encoded morphologically by the status verb 'Agu(become)' with RP inflection *Ana* that is highlighted above. The verb is not a process verb that describes an event, but only a status verb that describes a configuration. Therefore, in our annotation scheme the above raw text is annotated as follows:

(5) timuka talaivarum munnAL mudalamaiccarum Ana karuNAnidhi
NN NN NN_COMB NN ST_QUAL NNP
nEtRu seitiyALargaLai sandittu pEsinAr.
NST NN PR_CONJ PR_FIN

As a first stage, the given raw text is split into construction patterns like above (note that we have split the unit *Ana*, which is usually formally analysed as an adjectivializer, from the raw text since it invokes a construction pattern of Qualifier). Most of the tokens in Tamil are already directly mappable to construction schemas in the above example. The labels are shown below each token. In this way all the sentences are processed such

Table 3: Tagset for Construction Schemas in Tamil

Tag Name	Construction Schema
CC	Coordinating Conjunction
ECHO	Echo Word
NN	Noun
NN_COMB	Nouns in Combinative Schema
NST	Spatio Temporal Noun
OPER	Operator
PRON	Pronoun
PSP	Postposition
PSP_PRON	Postposition in Qualifier schema
PSP_QUAL	Postposition in Qualifier schema
QT_CONC	Quotative concursive
QT_COND	Quotative conditional
QT_CONJ	Quotative conjunctive
QT_FIN	Quotative complete
QT_PRON	Quotative Pronoun
QT_QUAL	Quotative Qualifier
RDP	Reduplication
ST_CONC	Status Concursive
ST_COND	Status Conditional
ST_CONJ	Status Conjunctive
ST_FIN	Status Complete
ST_PRON	Status Pronoun
ST_QUAL	Status Qualifier
SYM	Symbol
UNK	Unknown token
PR_CONC	Process Concursive
PR_COND	Process Conditional
PR_CONJ	Process Conjunctive
PR_FIN	Process Complete
PR_PRON	Process Pronoun
PR_QUAL	Process Qualifier

Table 4: Dependency relations between construction schemas

Tag Name	Construction Schema
ccof	Coordination conjunction
conc:man	Concursive- Manner
conc:seq	Concursive- Event concurrence
cond	Conditional
concess	Concession
conj:gram	Conjunctive- Grammatical
conj:man	Conjunctive- Manner
conj:seq	Conjunctive- Event continuance
inf:gram	Infinitive- Grammatical
inf:man	Infinitive- Manner
inf:seq	Infinitive- Purpose
k1	karta karaka
k1s	karta samanadhikarana
k2	karma karaka
k2s	karma samanadhikarana
k3	karana karaka
k4	sampradana karaka
k4a	anubhava karta
k5	apadana karaka
k7	vishayadhikarana
k7a	adhikarana extended
k7p	deshadhikarana
k7t	kaladhikarana
nmod:stat	Status Qualifier
pof	Complex Predicate
r6	Genitive case
ras	Associative relation
rh	Reason relation
main	Attachment to ROOT node
rt	Purpose relation
sent_adv	Process Conditional
status	Status functions in discourse
vmod:stat	Status Associative

that the tokens resulting after this preprocessing will be construction units that are ready to be labelled. The comprehensive list of construction labels and their meanings are shown in Table 3. After this labelling is done, the labeled construction units are chunked based on whether the consecutive construction units are *usage based syntactic freezes* or not. For example *multi-word expression(MWEs), numeric quantifiers modifying a noun, demonstrative adjectives modifying a noun* are the three instances where based on usage

in Tamil, the consecutive construction labels can be safely grouped in a single chunk. As an example the sample labelled sentence shown in 5 is chunked in the following manner:

(6) (timuka talaivarum) (munnAL mudalamaiccarum) (Ana)
(NN NN) (NN_COMB NN) (ST_QUAL)
(karuNAnidhi) (nEtRu)) (seitiyALargaLai) (sandittu) (pEsinAr).
(NNP) (NST) (NN) (PR_CONJ) (PR_FIN)

In the above sentence, two units *timuka* and *talaivarum* are grouped together as one chunk because it forms a noun-compound MWE which is a syntactic freeze as per its usage. i.e. You can chunk them together and only the head of the chunk is going to participate in the dependency relationship. After chunking, the final dependency analysis is shown in 2. It can be seen that the above dependency analysis uses the *karaka*

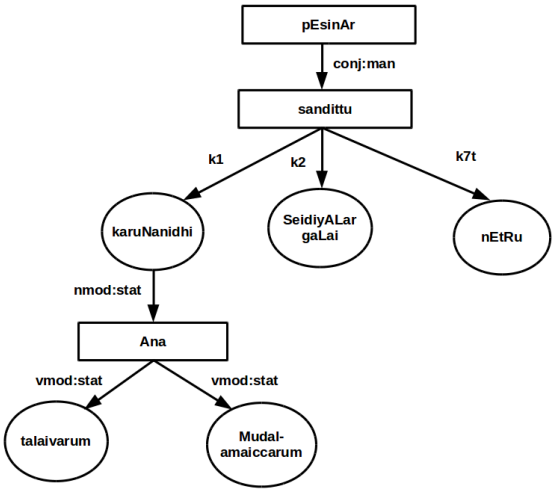

Fig. 2: Parsed Output

relations such as k1, k2, k7 as discussed in Computation Paninian Grammar(CPG) that has been successfully applied for Indian languages[7]. This is because the *karaka relations* such as *karta, karma, karana* etc. are already syntactico-semantic concepts that describe the meaningful role that a noun is conceived to play in an action denoted by a verb. These relations are well understood from traditional Sanskrit grammar and are extremely suitable for analysing relatively free-word ordered languages like Indian languages. Roughly we can say that the *karaka* roles are analogous to theta roles with the difference that these are not purely semantic from the information point of view but 'syntactico-semantic'. In that sense *karaka* roles are different from theta/ semantic roles. Refer to [7] and [6] to understand these more. In other words *karaka* theory

provides mapping between form (vibhakti markers) and construed meaning (karaka relations) which exactly is what Construction schemas are about. Hence we are not going to reinvent the wheel but rather use these traditional notions in the general Construction Grammar formulation. As a part of the larger picture, these karaka roles exemplify the 'participant' interaction that is shown in figure 1. In what way is the proposed Construction Grammar annotation different from the Computational Paninian Grammar annotation? The answer can be summarized as following:

- We use all the *karaka* relations as used in CPG as these are 'construals' of how a noun can interact meaningfully with a verb.
- Additionally, we treat the function words that can occur in a language as *status* verbs i.e. postpositions, adjectives, adverbs, complementizers etc. Instead of analyzing them as formal parts of speech, we treat them as relations which describe atemporal configurations and to that extent they are all verbs in *status* interpretation. (Refer to section 3 and [3] to understand this better). In fact, Tamil uses verbal morphology to derive these function words.
- We also handle various other meaningful discourse construals such as *Qualifier, Pronominal, Conjunctive, Concurrent, Condition and Infinitive* schemas which are necessary to understand the peculiarities in Dravidian syntax. Refer to [19–24] to understand some of the syntactic issues in Dravidian languages. How our discourse construals can explain these issues in a meaningful way is a theoretical enterprise that we are working on which is beyond the scope of the current work. Basically these discourse construals are included in our annotation framework as we hypothesize that they are inevitable to fully characterize Dravidian syntax meaningfully. Furthermore, we also hypothesize that these discourse construals are directly machine-learnable by way of morphological regularities in these languages.
- We also handle 'combinative' interaction in figure 1 by means of chunking multiword expressions, noun compounds as one chunk.
- In short, we are treating the *karaka* relations as one of the four types of discourse interactions that can happen between *things* and *relations* in the speaker's discourse world. Other three types of interactions are also captured though construction labels and chunking in our scheme.

In the third point above, we have mentioned a few discourse construals and pointed out that they are inevitable to understand certain syntactic peculiarities in Dravidian languages. Out of these, we have discussed about *Qualifier* and *Pronominal* schemas with illustrations in section 3. We will just briefly describe the other four schemas namely *Conjunctive, Concurrent, Condition, Infinitive*. These four schemas signify the *associative* interactions discussed in figure 1 i.e. how one relation interacts with another relation. For instance, if two events take place in discourse, they are perceived in the following ways.

1. Event 1 is perceived as having continuance with event 2 - *Conjunctive schema*
2. Event 1 is perceived as being concurrent/ parallel with event 2 - *Concurrent schema*
3. Event 1 is perceived as a condition/situation for describing event 2 - *Conditional schema*

4. Event 1 is perceived as yet to happen from the point of view of event 2 - *Infinitve schema*

All these four 'construals' are directly encoded in the verb morphology of Dravidian languages. Every Dravidian verb has four basic non-finite inflections that directly correspond to these construals that we mentioned above. It is these construals that are skilfully exploited by the languages to create subordinate clauses, grammatical aspects, modalities etc. by way of creating a series of non-finite verbs ending with one finite-verb. The bottom line is: formally different syntactic phenomena such as subordinate clause formation, grammatical aspects, modalities are constructed using the same above four construals. Again learning these construals instead of their formal properties is straightforward and suitable for these languages, because these discourse level construals are directly marked in morphology. For example in Tamil, *conjunctive* inflection of a verb occurs in clauses showing discourse sequence as well as in grammatical aspects. The *infinitive* inflection occurs in clauses showing goal/purpose as well as to describe moods and modalities. The *concurrent* inflection occurs in clauses conceiving concurrence as well as in describing manner adverbs. The *conditional* inflection occurs in conditional clauses, situated descriptions, in complement clauses which are based on some condition etc. The following examples serve to illustrate the point.

(7) rAman vITTuk-ku **pO-y** paDam pArt-t-An
Raman home-dat **go-conjunctive** *movie see-pst-agr*
'Ram went home and watched a movie'

(8) rAman munb-E paDatt-ai **pArt-tu** iruk-kiR-An
Raman before-only movie-acc **see-conjunctive** *exist-pres-agr*
'Ram has already seen the movie'

Note that the same conjunctive form of a verb is morphologically used by the language to indicate both clausal sequences and grammatical aspects. Usually, in formal analyses this is explained through grammaticalization theory. But in construction grammar, it is explained as the cognitive construal that underlies the conjuctive participial inflections. (Refer to [25] for cognitive operations underlying conjunctive inflection and how it relates to grammatical aspect). Discussing in detail why the same conjunctive, infinitive form are used for event - event relations as well as for aspects and modalities, with relevant grammatical evidences is a task we are not undertaking in this work. What is relevant for our parsing task at hand is, we treat every one of these non-finite verbal inflections as separate construction schemas and the dependency parser has to learn whether the relation that holds between them is event relation or merely grammatical and so on.

Thus with this understanding of the concepts that we have discussed, we formulated the construction schema labels and the dependency labels. The complete set of Construction labels in our annotation scheme is listed in Table 3. The complete set of dependency relations that can exist between these construction schemas are listed in Table 4. To understand the *karaka* related concepts such as *karta samanadhikarana, deshadhikarana* etc. please refer to the dependency annotation guidelines[26]. Other

concepts such as conjunctive, concurrent, conditional, infinitive and their subfunctions namely grammatical/ manner/ event relations in discourse are already discussed. Extended adhikarana (k7a) is a *karaka* construal that is specific to Tamil. Volitionality conceived upon the giver or receiver in a transaction event is given the label k7a.

5 Experiments and Results

To verify if the interactions between the proposed conceptual schemas are true and suitable for learning dependencies, we collected a set of 935 sentences for our annotation experiments. Out of these, 354 sentences are taken from newspaper data that we crawled from various online newspapers in Tamil. The crawled data had a total of approximately 5 lakh sentences(5,17,421 sentences). This crawled corpus is raw, uncleaned and unprocessed. The remaining 581 sentences are taken from a portion of the gold standard annotated training data for Tamil full parser, which is available as part of ILMT consortium[2]. The gold annotation[3] of the 581 sentences was available on CPG framework. These sentences are from various domains such as tourism, health, religion etc. While the former 354 sentences are general newspaper sentences that could be on any topic, the latter 581 sentences are collected from domain specific data and therefore the type of sentences, length of sentences could differ.

We annotated all the 935 sentences with the proposed tags and dependencies. We used MALT parser[4] for the purpose of our experiments. The parser settings are of Ambati et al.[27]. The features given to the MALT parser are the stem and morphological inflections. Experiments were conducted for the following test scenarios to verify the performance of the parser.

1. Varying the sources of data (AUKBC or Newspaper data)
2. Varying the size of the training data for the given source
3. Varying the gold annotation scheme (existing CPG annotation vs our proposed annotation)
4. Combining both the data and verifying the performance with the proposed annotation

Since we took only 581 sentences from AUKBC data(CPG annotation) and annotated those sentences in our proposed CG framework, the parser evaluation that can be compared between the two annotation schemes for only this amount of data. Therefore the fourth testing scenario on 935 sentences reports only the parsing accuracy on our proposed annotation scheme. In every one of these test cases, a five fold validation was done. The results are reported in Table 5. The annotation type *CG* refers to the proposed Construction Grammar framework and *CPG* refers to the Computation Paninian Grammar framework. The average number of chunks in a sentence mentioned in the table is the length of a sentence in terms of number of chunks in it. As it can be seen, the number of chunks are not the same between the two annotation schemes. In fact since

[2] Indian Language Machine Translation Project funded by DIT, Government of India

[3] The gold annotation was carried out by AU-KBC Research Centre, Chennai

[4] http://www.maltparser.org/download.html

Table 5: Parser accuracy test cases

Ann. type	Data size	Data type	Avg. No. of chunks in a sentence	Avg. LAS
CPG	176	AUKBC	5.1	56.02%
CG	176	AUKBC	5.8	72.98%
CPG	581	AUKBC	5.2	60.57%
CG	581	AUKBC	5.9	82.24%
CG	354	Crawled data	10.6	72.93%
CG	354+176	(Crawled+AUKBC) data	9.0	74.96%
CG	354+581	(Crawled+AUKBC) data	7.7	82.21%

we split a token wherever status interpretations of verbs could be made, our annotation scheme almost always has more number of chunks in a sentence than the CPG annotation. For instance, note that for the data size of 176 sentences, the average number of chunks in CPG is 5.1 whereas in the proposed CG scheme it is 5.8. The average length of sentences is more in the crawled newspaper data than the AUKBC data. The last three rows show that as the average number of chunks in a sentence in the corpus decreases, the accuracy increases which is only expected because there will be lesser long distance dependencies. Every row that is shown in the result is the average of five fold validation performed on the data. For every given data size we chose 80% for training and 20% for testing. The average LAS accuracy of the parser on a total of 935 sentences is 82.21%.

5.1 Partial evaluation results of Dependency labels and attachment

Technically the LAS scores are themselves not directly comparable because the annotation labels and linguistic scheme are different. Therefore, we will show the partial evaluation results of dependency labels which reveal as to how within a given linguistic framework various labels are learnt consistently.

Table 6: Precision and Recall of DepRel and Attachment; 176 sentences; CG annotation

Deprel	Gold	Correct	System	Recall %	Precision %	Deprel	Gold	Correct	System	Recall %	Precision %
adv	8	6	7	75.00	85.71	k7	13	10	14	76.92	71.43
ccof	3	0	0	0.00	NaN	k7a	1	0	0	0.00	NaN
conj:gram	8	7	7	87.50	100.00	k7p	2	0	0	0.00	NaN
conj:man	1	0	0	0.00	NaN	k7t	1	0	3	0.00	0.00
inf:gram	9	8	8	88.89	100.00	main	36	36	36	100.00	100.00
k1	33	28	51	84.85	54.90	nmod:stat	5	5	5	100.00	100.00
k1s	2	2	2	100.00	100.00	pof	5	1	1	20.00	100.00
k2	12	5	6	41.67	83.33	r6	13	10	11	76.92	90.91
k2s	1	1	1	100.00	100.00	rsp	1	0	0	0.00	NaN
k3	2	0	0	0.00	NaN	sent adv	3	1	1	33.33	100.00
k4	6	1	1	16.67	100.00	status	2	2	2	100.00	100.00
k5	2	0	0	0.00	NaN	vmod:stat	21	20	36	95.24	55.56

Table 7: Precision and Recall of DepRel and Attachment; 176 sentences; CPG annotation

Deprel	Gold	Correct	System	Recall %	Precision %
adv	12	7	14	58.33	50.00
ccof	13	9	12	69.23	75.00
jjmod	1	0	0	0.00	NaN
k1	29	21	47	72.41	44.68
k1s	2	0	1	0.00	0.00
k2	22	10	16	45.45	62.50
k3	1	0	0	0.00	NaN
k4	5	2	2	40.00	100.00
k7	5	0	3	0.00	0.00
k7a	1	0	0	0.00	NaN
k7p	5	2	9	40.00	22.22
k7t	5	0	2	0.00	0.00
lwg__psp	2	2	3	100.00	66.67
main	35	34	35	97.14	97.14
nmod	5	0	0	0.00	NaN
r6	11	7	10	63.64	70.00
rh	1	0	0	0.00	NaN
rsp	1	0	0	0.00	NaN
sent- adv	1	0	0	0.00	NaN
undef	3	0	0	0.00	NaN
vmod	3	2	9	66.67	22.22

Table 8: Precision and Recall of DepRel and Attachment; 581 sentences; CG annotation

Deprel	Gold	Correct	System	Recall %	Precision %
adv	23	18	19	78.26	94.74
ccof	13	12	13	92.31	92.31
conj:gram	28	26	29	92.86	89.66
conj:man	7	5	9	71.43	55.56
conj:seq	3	0	0	0.00	NaN
inf:gram	37	36	37	97.30	97.30
inf:man	1	0	1	0.00	0.00
inf:seq	3	0	0	0.00	NaN
k1	110	98	140	89.09	70.00
k1s	16	14	15	87.50	93.33
k2	32	20	24	62.50	83.33
k2s	1	0	0	0.00	NaN
k3	9	9	9	100.00	100.00
k4	17	13	14	76.47	92.86
k4a	0	0	2	NaN	0.00
k5	2	2	3	100.00	66.67
k7	49	39	47	79.59	82.98
k7a	2	0	0	0.00	NaN
k7p	9	4	5	44.44	80.00
k7t	9	2	7	22.22	28.57
main	115	113	116	98.26	97.41
nmod:stat	16	14	14	87.50	100.00
pof	18	13	18	72.22	72.22
r6	29	25	25	86.21	100.00
ras	0	0	4	NaN	0.00
rsp	1	0	0	0.00	NaN
rt	1	0	0	0.00	NaN
sent adv	11	4	4	36.36	100.00
status	11	10	11	90.91	90.91
vmod:stat	90	83	99	92.22	83.84

Table 9: Precision and Recall of DepRel and Attachment; 581 sentences; CPG annotation

Deprel	Gold	Correct	System	Recall %	Precision %
adv	40	33	46	82.50	71.74
ccof	43	32	48	74.42	66.67
jjmod	2	0	0	0.00	NaN
k1	104	77	154	74.04	50.00
k1s	15	2	3	13.33	66.67
k2	63	22	59	34.92	37.29
k2g	4	2	2	50.00	100.00
k2p	1	0	0	0.00	NaN
k2s	1	0	0	0.00	NaN
k3	6	2	2	33.33	100.00
k4	18	12	24	66.67	50.00
k5	1	1	1	100.00	100.00
k7	20	6	8	30.00	75.00
k7a	1	0	0	0.00	NaN
k7p	27	22	39	81.48	56.41
k7t	17	5	9	29.41	55.56
lwg__psp	10	4	4	40.00	100.00
main	116	114	116	98.28	98.28
nmod	13	1	5	7.69	20.00
nmod_relc	3	0	3	0.00	0.00
r6	32	24	28	75.00	85.71
ras- k1	2	0	0	0.00	NaN
ras- k2	2	0	0	0.00	NaN
ras- neg	2	0	0	0.00	NaN
rd	2	0	0	0.00	NaN
rh	4	1	2	25.00	50.00
rsp	4	0	0	0.00	NaN
sent- adv	4	2	5	50.00	40.00
undef	8	0	0	0.00	NaN
vmod	18	15	25	83.33	60.00

The precision and recall of dependency labels and attachment taken together for a training size of 176 sentences annotated using the proposed method is reported in Table 6. Table 7 shows the precision and recall of dependency relation + attachment for the training data size of 176 using the CPG annotation. Similarly, 8 and 9 report the precision and recall of dependency labels + attachment for the training size of 581 sentences, annotated using CG framework and CPG framework respectively. The dependency label counts are not readily comparable even for the same label because the chunks and the relationships that exist between them are different in the two frameworks. For instance, in CG output shown in Table 8 the gold count of vmod:stat is 90 when the training size was 581 sentences. But in CPG output shown in Table 9, the gold count of vmod is just 18. This discrepancy can be understood if we notice that the ccof count in the CPG annotation is 43 but in CG it is just 13. i.e. what are analyzed as conjunction of two entities in CPG are analyzed as list of entities associating with a status verb in CG. Conjunctions are actually done by particles such as 'um' whose syntactic behaviour is different from a proper conjunction like 'and' in English. It semantics is similar to MO particle of Japanese well discussed in literature[28]. That explains why there is more vmod:status in CG parser output. Overall, it can be seen that the dependency label + attachment precision and recall are comparatively better in our proposed CG annotation consistently.

5.2 Reasons for better learning

There are two reasons for better learning of dprel relations and overall LAS accuracy.

- Our dependency relations are directly learnable from morphological inflections
- The form- function pairing analysis is able to generalize the peculiar constructions that occur in Tamil syntax better.

The dependency labels that we are using in our annotation scheme are not coarser but fine grained. For instance, what is just a 'vmod' in CPG annotation is annotated with fine-grained labels such as 'conj:gram', 'conj:man', 'conj:seq', 'conc:man', 'conc:seq' etc. Yet these labels are learnt better because these labels are directly inferred from the morphological inflections given as a feature to MALT parser. The fact that construction analysis performs better across 5 fold experiments shows that most of the functional properties of the language are directly encoded in morphology.

By mapping the morphological features to meaningful construals, we are able to explain the peculiar morpho-syntactic constructions in Tamil, which are otherwise difficult or impossible to characterize. Take for example a construction like the one shown below:

(9)

nIND-a-d-um	kaLaippu	mikk-a-d-um-An-a
elongate-RP-PronSuffix-also	*tiredness*	*exceed-RP-PronSuffix-also-became-RP*
payaNam toDar-nd-adu		
journey continue-pst-agr		

'The long and arduous journey continued'

As it can be seen there are no pure adjectives in Tamil and therefore an expression like *long and arduous* is rendered by a complex expression *nINDadum kaLaippu mikkadumAna* whose literal morphological glosses are given above. Conventionally, the parts of speech and chunking in CPG framework is as follows:

1. ((nINDadum NN)) - Noun chunk
2. ((kaLaippu NN)) - Noun chunk
3. ((mikkadumAna JJ) (payaNam NN)) - Noun chunk
4. ((thodarndhadhu VB)) - Verb chunk

Note that the word 'nINDadum' is treated as a noun because morphologically it exhibits the property of a noun with pronominal suffix. But in terms of dependency analysis, 'nINDadum' and 'kaLaippu mikkadumAna' should be analyzed as adjectives that together modify the noun 'payaNam'. However, since chunk is structurally defined in CPG as minimal non-recursive unit of analysis[29] the second adjective 'mikkadumAna' and the noun 'payaNam' together are by definition grouped together as one Noun chunk. Now once it is chunked like this, there is no way to learn that there are two adjectives in the expression that are modifying a noun. The most likely dependency tree that the parser will end up learning with this configuration would be as shown in figure 3. This is a type of error that cannot be handled no matter how much training data

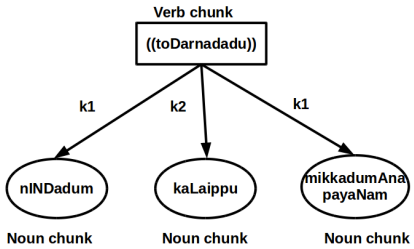

Fig. 3: Wrong dependency that is likely to be learnt

is provided because the problem here has a theoretical basis where there is a mismatch between what the morphology of the language says and what formal syntactic category the word stands for.

In construction grammar approach, this problem is easily handled because we map the word 'nINDadum' as *Status Pronominal Schema*, 'kaLaippu' as *Noun Schema*, 'mikkadum' as *Status Pronominal Schema*, 'Ana' as *Status Qualifier Schema*. Thus the three words in the raw text are treated as four construction units that map to their meaningful 'construals' and now the dependency relations are established between them as shown in figure 4. It might look that after all the final dependency relations are the same as in CPG, so what is the CG contribution here? Recognizing 'Ana' not as a formal adjectivializer suffix but a *Status Qualifier* schema that can take 'configurations' as its arguments allows us to chunk the tokens differently and build the dependency tree shown in figure 4.

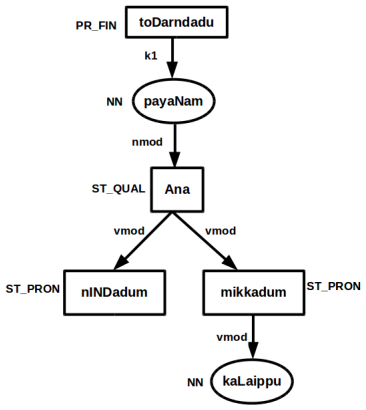

Fig. 4: Dependency analysis according to CG framework

This and many other expressions where the language uses its morphological properties to create 'verbiness' are very well captured by following a construction based approach. In the next section, we will discuss about what kinds of errors commonly occur in parser output, analyse those error scenarios and what improvements can be introduced to handle those errors.

6 Error Analysis

We performed a five fold validation of parser output for varying training sizes ranging from 354 to 935 sentences and checked what are the most frequent errors that are encountered by the parser trained with the proposed CG framework. The table 10 shows the top five frequent errors across different folds of iteration when the training data size was 935 sentences.

Fold 1			**Fold 2**			**Fold 3**			**Fold 4**			**Fold 5**		
Gold	System	Frequency	Gold	System	Frequency	Gold	System	Frequency	Gold	System	Fequency	Gold	System	Frequency
k1	k2	11	k1	k2	16	k1	k2	11	k1	k2	20	k1	k7t	13
k1	pof	11	k1	pof	11	pof	k1	8	k7	k7p	12	k7	k7p	9
k7	k7p	6	pof	k2	6	k7	k7t	6	k1	nmod	11	k1	pof	6
conj:seq	conj:man	6	k4	rt	5	k1	pof	6	pof	k1	10	conj:man	conj:seq	6
k7	k7t	5	k7	k7p	4	vmod:stat	k4	5	k1	vmod	9	k7	k7t	5

Table 10: Five frequent drel errors in different folds of iteration for 935 sentences Training Data

The most frequent errors that are observed in the parser output are as follows:

1. k1 is misidentified as k2 - karta(*agent* roughly speaking) is misidentified as karma(*patient* roughly speaking)
2. pof (Complex predicate) is misidentified as k1 (doer)
3. k7p (spatial location) is misidentified as k7(general locative)

4. k4 (goal) instead of rt (purpose)
5. conj:gram (grammatical) instead of conj:man (manner)

The most frequent error is: the gold label is k1 while the system identifies it as k2 and vice versa. This usually happens because it is quite common in Tamil that the accusative case is not morphologically marked on a noun and the language is relatively free word ordered. The following sentence is taken from one of the test file in which the parser misidentified k2 as k1. It illustrates the point.

(10) vinAyaga cadurttik-ku **paDam** veLiyiDuv-a-du kamal-in tiTTam
Vinayaka chaturthi-dat **movie** *release-RP-pron kamal's plan*
'It is kamal's plan to release **the movie** during Vinayaka chaturthi(a religious festival)'

In the above example, the highlighted word 'paDam' does not morphologically show the accusative marker. Because the parser is trained on morphological features and 'k1' and 'k2' are potential candidates in the similar context, these two tags are the most misidentified. The confusion between 'pof' and 'k1' is only expected because the noun which is a part of a complex predicate can be easily confused as a participant of the verb. The other two frequent errors are due to the granularity of the dependency relation that should be identified. *k7p* is more granular than *k7* and therefore difficult to learn. Same is true for *conj:gram* and *conj:man*. The other interesting candidate is 'rt(purpose)' being misidentified with 'k4(goal)'. Because conceptually purpose and goal are metaphorically directed towards some object, the same fourth case marker is used in Tamil to denote these two functions. Hence it is difficult to tell apart one from another. Since these error scenarios typically involve making distinctions between granular relations, with more training data these can be learnt better.

7 Conclusions and Future Work

In this work, we have discussed that by exploiting the morphological regularities in Tamil and by mapping these morphological forms to meaning on the theoretical basis of Construction Grammar, we are better able to learn the morpho-syntactic peculiarities in Tamil data. Since these 'construals' are actually learned through morphological features consistently across 5 folds for varying training data size, it shows that there is a merit in applying form-function pairing as a means of syntactic analysis. Though we have hinted at these morphological properties with a few illustrations, we are working on a full theory in which the syntactic problems that are discussed in linguistic literature [19–24] can be explained based on the functional 'construals' underlying these peculiar morphological inflections. In future, we are looking forward to apply this idea of form-function pairing for other morphologically rich languages as well, since most meaningful information about the syntax comes from morphology in these languages.

References

1. Goldberg, A.E.: Construction grammar. Wiley Online Library (2002)

2. Fried, M., Östman, J.O.: Construction grammar. Construction Grammar in a cross-language perspective (2011)
3. Langacker, R.W.: Cognitive grammar: A basic introduction. Oxford University Press (2008)
4. Shieber, S.M.: Evidence against the context-freeness of natural language. Springer (1987)
5. Melčuk, I.A.: Dependency syntax: theory and practice. SUNY Press (1988)
6. Bharati, A., Chaitanya, V., Sangal, R., Ramakrishnamacharyulu, K.: Natural language processing: a Paninian perspective. Prentice-Hall of India New Delhi (1995)
7. Bharati, A., Sangal, R.: Parsing free word order languages in the paninian framework. In: Proceedings of the 31st annual meeting on Association for Computational Linguistics, Association for Computational Linguistics (1993) 105–111
8. Bharati, A., Gupta, M., Yadav, V., Gali, K., Sharma, D.M.: Simple parser for indian languages in a dependency framework. In: Proceedings of the Third Linguistic Annotation Workshop, Association for Computational Linguistics (2009) 162–165
9. Mannem, P.: Bidirectional dependency parser for hindi, telugu and bangla. Proceedings of ICON09 NLP Tools Contest: Indian Language Dependency Parsing, India (2009)
10. Nivre, J.: Parsing indian languages with maltparser. Proceedings of the ICON09 NLP Tools Contest: Indian Language Dependency Parsing (2009) 12–18
11. Ambati, B.R., Gadde, P., Jindal, K.: Experiments in indian language dependency parsing. Proceedings of the ICON09 NLP Tools Contest: Indian Language Dependency Parsing (2009) 32–37
12. Antony, P., Warrier, N.J., Soman, K.: Penn treebank-based syntactic parsers for south dravidian languages using a machine learning approach. International Journal of Computer Applications **7** (2010) 14–21
13. Selvam, M., Natarajan, A., Thangarajan, R.: Structural parsing of natural language text in tamil using phrase structure hybrid language model. International Journal of Computer, Information and Systems Science, and Engineering **2008** (2008) 2–4
14. Ramasamy, L., Žabokrtskỳ, Z.: Tamil dependency parsing: results using rule based and corpus based approaches. In: Computational Linguistics and Intelligent Text Processing. Springer (2011) 82–95
15. Straka, M., Hajic, J., Straková, J., Hajic jr, J.: Parsing universal dependency treebanks using neural networks and search-based oracle. In: International Workshop on Treebanks and Linguistic Theories (TLT14). (2014) 208
16. Kumari, B.V.S., Rao, R.R.: Hindi dependency parsing using a combined model of malt and mst. In: 24th International Conference on Computational Linguistics, Citeseer (2012) 171
17. Kesidi, S.R., Kosaraju, P., Vijay, M., Husain, S.: A constraint based hybrid dependency parser for telugu. International Journal of Computational Linguistics and Applications **2** (2011) 53
18. Seddah, D., Tsarfaty, R., Kübler, S., Candito, M., Choi, J., Farkas, R., Foster, J., Goenaga, I., Gojenola, K., Goldberg, Y., et al.: Overview of the spmrl 2013 shared task: cross-framework evaluation of parsing morphologically rich languages, Association for Computational Linguistics (2013)
19. Amritavalli, R., Jayaseelan, K.: Finiteness and negation in dravidian. The Oxford Handbook of Comparative Syntax (2005) 178–220
20. Amritavalli, R.: Separating tense and finiteness: anchoring in dravidian. Natural Language & Linguistic Theory **32** (2014) 283–306
21. McFadden, T., Sundaresan, S.: Finiteness in south asian languages: an introduction. Natural Language & Linguistic Theory **32** (2014) 1–27
22. Jayaseelan, K.: The serial verb construction in malayalam. In: Clause structure in South Asian languages. Springer (2004) 67–91
23. Jayaseelan, K.: Coordination, relativization and finiteness in dravidian. Natural Language & Linguistic Theory **32** (2014) 191–211

24. Herring, S.C.: Aspect as a discourse category in tamil. In: Annual Meeting of the Berkeley Linguistics Society. Volume 14. (2011)
25. Karmakar, S., Kasturirangan, R.: Cognitive processes underlying the meaning of complex predicates and serial verbs from the perspective of individuating and ordering situations in bānlā. In: Proceedings of the First International Conference on Intelligent Interactive Technologies and Multimedia, ACM (2010) 81–87
26. Bharati, A., Husain, D.S.S., Bai, L., Begam, R., Sangal, R.: Anncorra: Treebanks for indian languages, guidelines for annotating hindi treebank (version–2.0) (2009)
27. Ambati, B.R., Husain, S., Nivre, J., Sangal, R.: On the role of morphosyntactic features in hindi dependency parsing. In: Proceedings of the NAACL HLT 2010 First Workshop on Statistical Parsing of Morphologically-Rich Languages, Association for Computational Linguistics (2010) 94–102
28. Szabolcsi, A.: What do quantifier particles do? Linguistics and Philosophy **38** (2015) 159–204
29. Bharati, A., Sangal, R., Sharma, D.M., Bai, L.: Anncorra: Annotating corpora guidelines for pos and chunk annotation for indian languages. LTRC-TR31 (2006)