WDS'11 Proceedings of Contributed Papers, Part I, 143–148, 2011. *ISBN 978-80-7378-184-2 © MATFYZPRESS*

TamilTB: An Effort Towards Building a Dependency Treebank for Tamil

L. Ramasamy

Charles University in Prague, Faculty of Mathematics and Physics, Institute of Formal and Applied Linguistics (ÚFAL).

Abstract. Annotated corpora such as treebanks are important for the development of parsers, language applications as well as understanding of the language itself. Only very few languages possess these scarce resources. In this paper, we describe our effort in syntactically annotating a small corpora (600 sentences) of Tamil language. Our annotation is similar to Prague Dependency Treebank (PDT 2.0) and consists of 2 levels or layers: (i) morphological layer (m-layer) and (ii) analytical layer (a-layer). For both the layers, we introduce annotation schemes i.e. positional tagging for m-layer and dependency relations (and how dependency structures should be drawn) for a-layers. Finally, we evaluate our corpora in the tagging and parsing task using well known taggers and parsers and discuss some general issues in annotation for Tamil language.

Introduction

The most important thing in Natural Language Processing (NLP) research is data, importantly the data annotated with linguistic descriptions. Much of the success in NLP in the present decade can be attributed to data driven approaches to linguistic challenges, which discover rules from data as opposed to traditional rule based paradigms. The availability of annotated data such as Penn Treebank [*Mitchell et al.*, 1993] and parallel corpora such as Europarl [*Koehn*, 2005] had spurred the application of statistical techniques [*Ratnaparkhi*, 1996], [*Collins*, 2003], [*Koehn et al.*, 2003] to various tasks such as Part Of Speech (POS) tagging, syntactic parsing and Machine Translation (MT) and so on. They produced state of the art results compared to their rule based counterparts. Unfortunately, only English and very few other languages have the privilege of having such rich annotated data due to various factors.

In this paper, we take up the case of building a dependency treebank for Tamil language for which no annotated data is available. The broad objectives for the design of the Tamil dependency treebank (TamilTB) include: (i) annotate data at morphological level and syntactic level (ii) in each level of annotation, trying for maximum level of linguistic representation and (iii) building large annotated corpora using automatic tools. We have chosen dependency annotation over constituency representation for one obvious reason: that dependency annotation works well for free word order languages and the annotation is quite intuitive and easy to represent. One other reason is that, since treebanking for other Indian languages such as Hindi and Telugu [*Begum et al.*, 2008] too focuses on dependency annotation scheme, it would be easier in the future to compare or adopt features from those efforts. The focus of the paper is primarily on the annotation process at morphological level and syntactic level and evaluation of the annotated resources using publicly available taggers and parsers.

There is an active research on dependency parsing ([*Bharati*, 2009], [*Nivre*, 2009] and [*Zeman*, 2009]) and developing annotated treebanks for other Indian languages such as Hindi and Telugu. One such effort is, developing a large scale dependency treebank [*Begum et al.*, 2008] (aimed at 1 million words) for Telugu, as of now the development for which stands [*Vempaty*, 2010] at around 1500 annotated sentences. For Tamil, previous works which utilised Tamil dependency treebanks are: [*Dhanalakshmi et al.*, 2010] which developed dependency treebank (around 25000 words) as part of the grammar teaching tools, [*Selvam et al.*, 2009] which developed small dependency corpora (5000 words) as part of the parser development. Other works such as [*Janarthanam et al.*, 2007] focused on parsing the Tamil sentences. Those works did not make use of treebank to the parser development, rather they were based on linguistic rules. A somewhat detailed description of an effort to develop a TamilTB appeared in [*Ramasamy et al.*, 2011]. To our knowledge, this is the first attempt to develop a dependency treebank for Tamil with respect to the objectives defined earlier. This will also be the continuation of the work mentioned in [*Ramasamy et al.*, 2011].

The Section 2 will describe the general linguistic aspects of the Tamil language in brief, Section 3 will describe the annotation process in general and explain the preprocessing step in the annotation

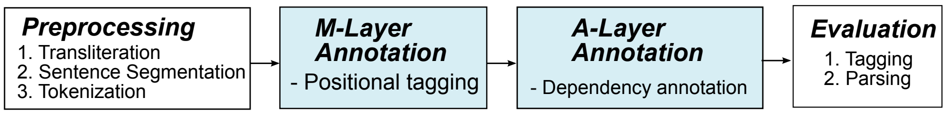


Figure 1. Annotation process.

process, Section 4 will introduce the morphological level annotation (m-layer annotation), Section 5 will introduce dependency annotation (a-layer annotation) and some of the issues involved and Section 6 will describe the evaluation on the developed resources.

General Aspects of the Tamil Language

Tamil is a south Indian language that belongs to the Dravidian family of languages. Other major languages in the Dravidian family include Telugu, Malayalam and Kannada. The main features of the Tamil language include agglutination, relatively free word order, head final and the subject-verb agreement. Below we touch briefly on these features.

Morphology. Tamil is an agglutinative language [*Lehmann*, 1989] and has a rich set of morphological suffixes which can be added one after another to noun and verb stems (mainly) as suffixes. Tamil morphology is mainly concatenative and derivations are also possible by means of *adjectivalization, adverbialization* and *nominalization*. In general, Tamil morphology can be represented [*Lehmann*, 1989] as [*stem (+affix)*^{n}]. Though there are only eight basic POS categories, with no such restrictions placed on as to how many words can be glued together, Tamil morphology pose significant challenges to POS tagging and parsing.

Head Final and Relatively Free Word Order. Tamil is a head final language, meaning the head of the phrasal categories always occur at the end of a phrase or constituent. Modifiers and other co-constituents always precede the phrasal head. For example, *postposition* is the head of the postpositional phrase, and will be modified by noun phrases. There are very few exceptions (identifiable) such as the subject of a sentence occuring after the finite verb (head). In most cases, head final rule is preserved.

Tamil is a Subject Object Verb (SOV) language and the word order is relatively free. Within a clause, phrases can be moved to almost any position except to the postion of clause head which should always be a verb. Besides the above features, subjects in Tamil agrees with verb in person, number and gender. Certain verbs will not code agreement with them, for ex: *illai, muti* etc.

Annotation Process

Our annotation scheme is based on Prague Dependency Treebank (PDT) [*Hajic*, 1998] and [*Böhmová et al.*, 2001]. PDT annotates the data in 3 levels or layers: (i) morphological layer (m-layer) (ii) surface syntax annotation (a-layer) and (iii) tectogrammatical annotation (t-layer). As we have mentioned earlier, our annotation process includes only the first 2 layers i.e. m-layer and a-layer. The Figure 1 shows the annotation process and the Table 1 shows the general information about the data used for annotation. This section will introduce in brief how preprocessing is done prior to the actual annotation process.

Table 1. General statistics of the data.

Description	value
Source	`www.dinamani.com`
Format	UTF-8
Transliterated	yes
Number of sentences	600
Number of words	9581

Preprocessing

The preprocessing stage consists of 3 steps. Once the raw corpus is downloaded from the web (`www.dinamani.com` in our case), the corpus in UTF-8 is transliterated into Latin for the ease of representation inside the programming components. Then the sentence segmentation is performed on the transliterated data to split the raw corpus into sentences. We used simple heuristics such as *fullstop, name initials, attribution* etc. to split into sentences. Wrong sentence splitting is corrected later during

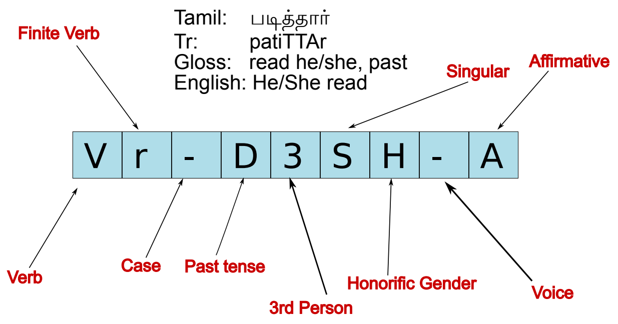


Figure 2. Positional tag.

the annotation. Tokenization is one of the important steps in preprocessing. The default delimiter in tokenization is *space*. However, Tamil is known to combine closed class words with general wordforms, which can be represented as separate words in languages such as English. For ex: Tamil, in certain situations combine *postpositions* with nouns, *clitics* with almost any wordforms and *auxiliary verbs* with verbs. We splitted those combination of words with the help of the list given in the Table 2. So given this list we will be able to split the agglutinative combinations such as *nouns + postpositions*, *verbs + auxiliaries* and etc. Initial splitting was done automatically using a few well known words from the list, and the remaining words or suffixes are found later when manually analyzing the data. This process will aid the m-layer annotation by reducing the tag complexity as well as data sparsity to some extent. We evaluated how much such combinations have been splitted from the original corpora. We found that 953 splits took place out of 9581 words. We can say that almost 10% of the additional corpus size is due to splitting some wordforms into separate tokens. The Table 3 shows an example sentence before and after applying the splitting.

Table 2. Closed class words for tokenization.

Category	**Word or suffix list**
Clitics	um, E, EyE, AvaTu
Postpositions	kUta, utan, pati, kuRiTTu, iliruwTu, anRu, uL, ARu, Tavira, pOTu, pOla, pinnar, pin, arukE, aRRa, inRi, illATa, mITu, kIz, mEl, munpE, otti, paRRi, paRRiya, pOnRa, mUlam, vaziyAka etc.
Auxiliary Verbs	patta, pattu, uLLa, pata, mAttATu, patuvArkaL, uLLAr, uLLanar, illai, iruwTAr, iruwTaTu, pattaTu, pattana, mutiyum, kUtATu, vENtum, kUtum, iruppin, uLLana, mutiyATu, patATu, koNtu, ceyTu etc.
Particles	Aka, Ana and their spelling variants such as Akak, Akac, AkaT
Demonstrative pronouns	ap, ac, ic, iw, aw

Table 3. An example splitting of word combinations.

Before splitting	puTiya cattaTTin**pati** , pATukAkkap**patta** winaivuc cinnaTT**iliruwTu** 1000 ati varai ewTa kattumAnam**um** katta anumaTi illai .
After splitting	puTiya cattaTTin **pati** , pATukAkkap **patta** winaivuc cinnaTT **iliruwTu** 1000 ati varai ewTa kattumAnam **um** katta anumaTi illai .

M-Layer Annotation

The m-layer annotation simply corresponds to POS tagging of the data. We decided to use *positional tagging* scheme to annotate our corpus. The main advantage of the positional tagging is that it can accommodate morphological features. We can easily train the POS taggers for *coarse grained* or *fine grained* tagsets. The main difference compared to ordinary POS tagging is that the *positional tag* for each word has a *fixed length* characters. Each character in the tag signifies a particular feature of that word. For our purpose, we have defined the length of the positional tag to be *9*.

The Figure 2 shows an example Tamil word and its *tag*. As you can see, the first letter of the tag is **V** which indicates that the word is a verb. The second position (**r**)[1] indicates that the verb is a finite verb and so on. The Figure 3 (a) & (b) shows the positional tagging system with a possible values for

[1]SUBPOS values are not given in this paper. We will release the data and the full annotation scheme soon.

Position	Feature	#Possible Values
1	POS	14
2	Sub POS	42
3	Case	10
4	Tense	05
5	Person	04
6	Number	03
7	Gender	06
8	Voice	02
9	Negation	02

(a) Each position & num of possible values

Value	Description
A	Adverbs
C	Conjunctions
D	Determiners
I	Interjections
J	Adjectives
N	Nouns
P	Postpositions
Q	Quantifiers
R	Pronouns
T	Particles
U	Numerals
V	Verbs
X	Unknown
Z	Punctuations

(b) POS values

Description	Value
Corpus size	9581 words
Vocabulary size	3583 words
# of tags for this corpus	217
# words received unique tags	3464
# words received 2 tags	109
# words received 3 tags	9
# words received 4 tags	1

(c) m-layer annotation statistics

Figure 3. Positional tag system.

No	Afun	Afun	Examples
1	AAdjn	Adverbial Adjunct	Optional adverbs, optional PP phrases attaching to verb
2	AComp	Adverbial Complement	Obligatory adverbs, obligatory PP phrases attaching to verb
3	AdjAtr	Adjectival Attribute	Adjectivalized verbs, or relative clauses
4	Apos	Apposition	Heads of the apposition clauses - clauses attaching to 'enRa'
5	Atr	Attribute	Noun modifiers
6	AuxA	Determiners	Demonstrative pronouns (iwTa-'this', awTa-'that')
7	AuxC	Subordinating Conjunctions	Subordinating Conjunctions (enRu, ena, Aka)
8	AuxG	Punctuations	-, ", ', $, rU., (,), [,]
9	AuxK	Terminal Punctuation	:, . , ?
10	AuxP	Postpositional head	mITu-'on', paRRi-'about', kIz-'under'
11	AuxS	Technical Root	Technical Root
12	AuxV	Auxiliary Verb	uL, koNtu, iru
13	AuxX	Comma (not coordination)	,
14	AuxZ	Emphatic particles (clitics)	TAn(emphasis), um-'also, even', E-'even'
15	CC	Part of a word	kiLarwTu ezuwTu - 'rise' as in *rising against*, written as 2 words
16	Comp	Complement (not adverbial)	Obligatory attachments to non verbs, "belongs to the batch **of 1977**"
17	Coord	Coordination node	maRRum - 'and', um
18	Obj	Object	Object
19	Pnom	Nominal Predicate	Nominal Predicate , nouns as predicates
20	Pred	Main Predicate	Main Predicate
21	Sb	Subject	Subject

Figure 4. Dependency relations (Analytical functions).

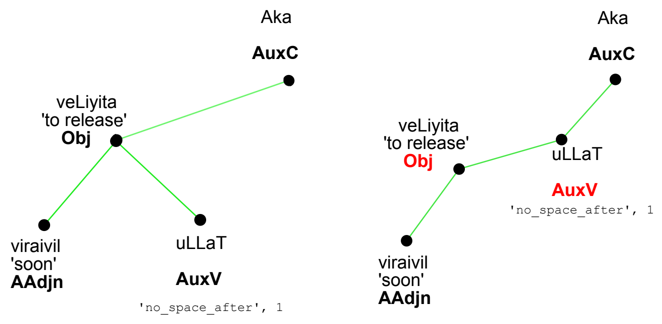
the first position i.e. main POS tag. Figure 3 (c) shows the basic statistics of the m-layer annotation. From the Figure, we observe that the entire corpus was tagged by 217 tags. The Figure also shows how many tags each word in the vocabulary can take. Over 96% of the wordforms are *unambiguous*. Only little over 3% of wordforms are ambiguous by having 2 tags. 3 tags and 4 tags are negligible. Lemmas for each wordform will also be stored as an attribute (`lemma` attribute) in m-layer annotation. At present, lemmas are identified partially through automation. Remaining are edited or added manually.

A-Layer Annotation

The a-layer annotation corresponds to dependency annotation. This step consists of two stages: (i) identifying the structure by attaching the *dependent* word as child to the *governing* word and (ii) labeling the relation with which the dependent and governing nodes (words) are related. Thus each sentence corresponds to a tree structure rooted at the predicate of the sentence or at the technical root. Each edge has a label and it signifies the relation between the parent and child nodes.

So far we have defined 21 dependency relations or analytical functions (afun) for labeling the edges. The Figure 4 shows the afun with some examples. After the *m-layer* annotation is performed, the structure and afun labels for the edges have been produced automatically by the rule based parser and edited manually. The *m-layer* and *a-layer* annotation have been performed for the dataset mentioned in Table 1.

In *a-layer* annotation, issues such as, handling of auxiliary verbs whether the auxiliaries should be hanged under the lexical verbs or the lexical verbs should be hanged under the auxiliary verbs, still remain. One reason for this dilemma is, that in Tamil, lexical verbs always precede auxiliary verbs but it is the auxiliary verb which codes the agreement and establishes morphological clues when there is an embedding of a clause into another clause. On the one hand, it is the lexical verb which is the head of a clause, so we can make the lexical verb as the head. On the other hand, it is the auxiliary verb which



(a) Auxiliary under lexical verb (b) Lexical verb under auxiliary verb

Figure 5. Auxiliary attachment dilemma.

Description	Value
# Sentences for training	479 (7715 words)
# Sentences for testing	121 (1866 words)
Morce Tagger	98.6 % (accuracy)
TnT Tagger	87.0 %

(a) Tagging performance

Description	Value
# Sentences for training	490 (7866 words)
# Sentences for testing	110 (1715 words)
MST (unlabeled)	77.8% (accuracy)
MST (labeled)	67.7%
Malt (unlabeled)	74.8%
Malt (labeled)	65.1%

(b) Parsing performance

Figure 6. Performance evaluation.

makes a connection between the embedded clause and the clause being embedded into, so the auxiliary is the head and the lexical verb will be the child. We chose to go by the first solution, i.e. auxiliary verb under the lexical verb. The Figure 5 shows an example for the auxiliary attachment problem. The `'no_space_after',1` indicates that the suffix "Aka" is part of the auxiliary verb "uLLaT".

As is common in the dependency approach, non-projective constructions can appear at the *a-layer*. They are observed at most in three situations: (i) when adverbs try to modify clauses by jumping the next immediate clause (ii) when arguments are shared between two clauses and when trying to attach some arguments to the first clause and some other to the second clause and (iii) when structures not belonging to Tamil occur.

Evaluation

This is actually not the evaluation of the TamilTB, rather, since the direct application of treebank is the parser development, we decided to evaluate how well the developed resource performs for the tagging and parsing tasks. We evaluated both *m-layer* and *a-layer* annotation independently with different training and test data. For *m-layer* annotation, we evaluated with Morce and TnT tagger. For *a-layer* annotation, we evaluated with Malt (projective) and MST parsers. The MST parser is trained with first order and projective algorithm settings. The data for training and testing are chosen randomly (around 80% for training and remaining for testing). The Figure 6 shows the evaluation results.

Conclusion and Future Work

In this paper, we described our ongoing efforts to develop a dependency treebank for Tamil language. As part of this development, we introduced our annotation scheme at word level and syntactic level. We also used our treebank resource to evaluate the performance in tagging and parsing tasks. The developed resource is still a small amount of data, and we are still trying to improve the annotation scheme and removing inconsistencies in the treebank data. As a future work, we will standardize the annotation scheme, optimize tools for low amount of data, and last but not the least, we will add more annotated data.

Acknowledgment. The research leading to these results has received funding from the European Commission's 7th Framework Program (FP7) under grant agreement n° 238405 (CLARA). We also would like to thank reviewers for their useful comments.

References

Begum, R., Husain, S., Dhwaj, A., Sharma, D., Bai, L., Sangal, R., Dependency Annotation Scheme for Indian Languages, In: Proceedings of the Third International Joint Conference on Natural Language Processing (IJCNLP), Hyderabad, India, 2008.

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 (ACL-IJCNLP 2009), pp. 162–165, *Association for Computational Linguistics*, 2009.

Böhmová, A., Hajič, J., Hajičová, E., Hladká, B., The Prague Dependency Treebank: Three-Level Annotation Scenario, In: Anne Abeillé (Ed): Treebanks: Building and Using Syntactically Annotated Corpora, *Kluwer Academic Publishers*, 2001.

Collins, M., Head-Driven Statistical Models for Natural Language Parsing, *Comput. Linguist.* 29, 589–637, 2003.

Dhanalakshmi, V., Anand Kumar, M., Rekha, R.U., Soman, K.P., Rajendran, S., Grammar Teaching Tools for Tamil Language, In: Technology for Education Conference (T4E 2010), pp. 85–88, India, 2010.

Hajic, J., Building a Syntacticly Annotated Corpus: The Prague Dependency Treebank, In: Issues of Valency and Meaning, pp. 106-132, Karolinum, Prague, 1998.

Janarthanam, S., Nallasamy, U., Ramasamy, L., Santhoshkumar, C., Robust Dependency Parser for Natural Language Dialog Systems in Tamil, In *Proceedings of 5th Workshop on Knowledge and Reasoning in Practical Dialogue Systems(IJCAI KRPDS-2007)*, pp. 1–6, Hyderabad, India, 2007.

Koehn, P., Europarl: A Parallel Corpus for Statistical Machine Translation, In: MT Summit, 2005.

Koehn, P., Och, F.J., Marcu, D.: Statistical Phrase-Based Translation, In: Proceedings of the 2003 Conference of the North American Chapter of the Association for Computational Linguistics on Human Language Technology, pp. 48–54, *Association for Computational Linguistics*, 2003.

Lehmann, T., A Grammar of Modern Tamil. Pondicherry Institute of Linguistics and Culture (PILC), Pondicherry, India, 1989.

Mitchell P.M., Mary Ann, M., Beatrice, S., Building a Large Annotated Corpus of English: the Penn Treebank. *Comput. Linguist.* 9, 313–330, 1993.

Nivre, J., Parsing Indian Languages with MaltParser, In: Proceedings of the ICON09 NLP Tools Contest: Indian Language Dependency Parsing, pp. 12–18, 2009.

Ramasamy, L., Žabokrtský, Z., Tamil dependency parsing: results using rule based and corpus based approaches, In: Proceedings of the 12th international conference on Computational linguistics and intelligent text processing - Volume Part I, CICLing'11, pp. 82–95, Tokyo, Japan, 2011.

Ratnaparkhi, A., A Maximum Entropy Model for Part-Of-Speech Tagging. In: Proceedings of the Empirical Methods in Natural Language Processing, pp. 133-142, 1996.

Selvam, M., Natarajan, A.M., Thangarajan, R., Structural Parsing of Natural Language Text in Tamil Language Using Dependency Model, *Int. J. Comput. Proc. Oriental Lang.*, Volume 22, 2009.

Vempaty, C., Naidu, V., Husain, S., Kiran, R., Bai, L., Sharma, D., Sangal, R., Issues in Analyzing Telugu Sentences towards Building a Telugu Treebank, In: Alexander F., G. (Ed): CICLing 2010, LNCS 6008, pp. 50–59, 2010.

Zeman, D., Maximum Spanning Malt: Hiring World's Leading Dependency Parsers to Plant Indian Trees, Proceedings of ICON09 NLP Tools Contest: Indian Language Dependency Parsing, *NLP Association of India*, Hyderabad, India, 2009.