TamilNLP: Low Resource Language Processing

By

Himanshu Singh

Under the supervision of

Dr. Rajiv Ratn Shah

Indraprastha Institute of Information Technology Delhi

July, 2022

TamilNLP: Low Resource Language Processing

By

Himanshu Singh

Submitted in partial fulfillment of the requirements for the degree of

Master of Technology

To

Indraprastha Institute of Information Technology Delhi

July, 2022

Certificate

This is to certify that the thesis titled *TamilNLP: Low Resource Language Processing* being submitted by *Himanshu Singh* to the Indraprastha Institute of Information Technology Delhi, for the award of the degree of Master of Technology, is an original research work carried out by him under my supervision. In my opinion, the thesis has reached the standards fulfilling the requirements of the regulations relating to the degree.

The results contained in this thesis have not been submitted in part or full to any other university or institute for the award of any degree/diploma.

July, 2022

Dr. Rajiv Ratn Shah

Indraprastha Institute of Information Technology Delhi

New Delhi 110020

Acknowledgements

I would like to express my special thanks of gratitude to my instructor Dr. Rajiv Ratn Shah who guided and supported me throughout the thesis . I am grateful to Dr. Sarveswaran who guided me with his valuable feedback throughout the research. I would also like to thank my partner, Tanya Sanjay Kumar who helped in various tasks in the research.

I would also like to thank Prof. Abirami, Prof. Suganya, Prof. Anita and the team of students of Thiagarajar College of Engineering, Madurai who played a major role in the development of annotation guidelines and the annotation process. Many thanks to William Tjhi and Weiqi Leong from AI Singapore team whose team helped us with their valuable feedback on various tasks in the project.

I would like to thank my parents and friends who helped me and kept me motivated in carrying out the research.

Abstract

In this paper, we worked on different aspects like dataset, annotation guidelines, annotation platform, and models to build a complete eco-system, aimed at making significant contributions towards NLP for the Tamil language. We focused on researching about morpho-syntactic relations in the Tamil text. A more diverse dataset was curated from 5 sources to form a treebank of 10,000 CoNLL-U format annotated sentences. Detailed annotation guidelines were developed for guiding the annotators and the users. We proposed hierarchical tag sets for POS and NER tasks, after testing various available tag sets for the Tamil language. To carry out the CoNLL-U format annotations efficiently, we introduce CoNLL-U GSheets. This annotation platform uses the highly accessible and easy-to-use Google sheets and equips it with all the necessary tools for annotations. The research also focused on developing the pipeline and the models for each task in the morpho-syntactic analysis. We have addresed the language-specific issues for each task in the morpho-syntactic analysis. We also took design decisions that promote flexibility in applications and assist in later NLP tasks.

Contents

List of Figures

List of Tables

Abbreviations

NLP	Natural Language Processing
UD	Universal Dependencies
IAA	Inter Annotator Score
POS	Part of Speech
NER	Named Entity Recognition
MWTT	Modern Written Tamil Treebank
Wiki	Wikipedia
HMM	Hidden Markov Model
CRF	Conditional Random Fields
SVM	Support Vector Machines
DL	Deep Learning
RNN	Recurrent Neural Network
LSTM	Long Short-Term Memory
min	minimum
max	maximum

Chapter 1

Introduction

Tamil is one of the longest-surviving classical languages in the world, possessing a vast knowledge of literature and grammar. It is part of the Dravidian Language Family. It is spoken in Tamil Nadu and Sri Lanka, in East-Asian countries like Burma, Malaysia, Singapore, Indonesia, Indio china, Fiji, in South-Africa and British Guinea and in islands like Mauritius and Madagascar etc. Tamil is an official language in Tamil Nadu and some of the foreign countries such as Sri Lanka and Singapore. The Indian government formally recognized it as a classical language in October 2004. There are countless literature present in Tamil ranging from music to medical, and so on. This language is often referred to civilization rather than just a language and has a lot of cultural aspects related to it.

There have been researches and developments to preserve and ease out the communication for this language. In this era of communication, Natural Language Processing (NLP), a field of AI, plays an important role in automation and solving a problem at a larger scale. NLP researchers aim to gather knowledge on how human beings understand and use language so that appropriate tools and techniques can be developed to make computer systems understand and manipulate natural languages to perform the desired tasks [1]. S.Sivakama Sundari also explains the importance of NLP in revival and development of Tamil language [2].

Morphological analysis and syntactic analysis are the fundamental and starting steps

of NLP. In order to build efficient and accurate parsers of the language, we need annotated corpus containing such morphosyntactic information. There are unsupervised methods for training a parser, but they often suffer on accuracy [3].The accuracy of these parsers have a direct impact on the efficiency of tasks like machine translation, question answering and information retrieval. Hence we rely on supervised learning, which in turn needs a sufficient amount of annotated data.

Working in the direction of contributing annotated dataset, we followed the universal dependency guidelines as the groundwork of our proposed guidelines. Universal Dependencies is an initiative to develop and maintain cross linguistically consistent treebank annotation for many languages [3]. Cross lingual learning, multilingual parsers, and research based on parsing from language typology perspective will be facilitated due to this initiative. We aimed to include elements of universal dependency guidelines to make our dataset useful for generalization and uniformity. We also included language specific guidelines that will help in understanding and explaining the syntactic structure in Tamil more precisely. Currently there are two treebanks for Tamil in the Universal Dependencies repository. Both of them contain roughly around 600 annotated sentences each. They also lack the diversity in the source of data which does not give a proper representation of Tamil in the real world. The comparative study of these two treebanks can be found in the Appendix C. Hence we aim to provide a larger treebank with sentences from diverse sources to depict a better representation of the Tamil language.

Grammatical annotations are time consuming and require a desired amount of knowledge to produce correct annotations. The annotators are expected to read and follow the annotation guidelines throughout the annotation process. Hence the quality of annotation depends on the clarity and in depth explanations given in the guidelines. We developed annotation guidelines for annotating data in CoNLL-U format covering tasks like tokenization, POS tagging, morphology, NER and dependency parsing. One of the main reasons to form a separate annotation guidelines was to cover Tamil language specific morphological and syntactic features. Though UD provides generalization, we need to highlight language specific details to develop accurate tools in NLP for that language.

We will be looking into the proposed annotation guidelines later in the report.

Decision of an appropriate annotation platform or tool affects the pace and effort of the annotation process. This decision can be based on the number of annotators, level of collaborativeness, number of fields to be annotated, relation between annotated fields, ease of use and availability. There are many tools available that help in POS tagging, NER tagging, dependency parsing and morphological annotations. There are quite a few tools listed on Universal Dependencies to annotate data in CoNLL-U format. We tested and analyzed some of these tools to see fit to our needs. We identified some shortcomings of the existing tools and developed a new platform "CoNLL-U GSheets", integrated with google sheets that proved to be an easier and effective choice. The motivation and the features are explained later in the report.

The contributions of this research can be broadly divided into -

1. Extracting and Collating Tamil text from multiple sources
2. Building CoNLL-U GSheets, a platform for CoNLL-U format annotations
3. Developing Annotation Guidelines for Tamil language
4. Building and analyzing machine and deep learning models

Chapter 2

Dataset

2.1 Related Work

There are many sources of raw Tamil text available online. Kaggle holds many datasets of varying size and Indic NLP recently contributed a huge corpora consisting of 2 crore sentences [4]. But there exists only a few publicly available dataset consisting of grammatical annotations. In this category, most of the annotations are done only for either POS or NER. FIRE conferences present annotated NER corpuses for Tamil and other languages [5]. These single feature annotations also differ in terms of tagsets and annotation guidelines. Further focusing on publicly available dataset containing all the morphosyntactic fields or CoNNL-U annotations, we end up with universal dependencies treebanks. Currently there are only two treebanks available for Tamil in the universal dependencies repository.

These two treebanks are called TTB and MWTT. TTB is the older dataset containing 600 sentences. It contained errors in tokenization and annotations like inconsistencies in dependency relations nmod and obl. MWTT was formed with the motivation to provide accurate annotations and incorporate the Enhanced Universal Dependencies scheme [6]. MWTT consists of 534 grammar sentences. One of the areas of improvement in this treebank is that it contains short sentences from a single source which does not truly represent

the real world Tamil instances. These treebanks do not contain NER annotations. An in depth comparison of the two treebanks can be found in the Appendix C. We identified the need for a bigger and diverse treebank. We plan to annotate 10,000 sentences in CoNLL-U format along with NER annotations. Additional information on the CoNLL-U format can be found in the Appendix A.

2.2 Dataset Extraction

We wanted to represent the real life occurrences of Tamil in our dataset. We focussed on multiple domains like ebooks, news, movie reviews and wikipedia. These sources consist of mostly formal and structured sentences. We also included the sentences from the simple grammar books, following the motivation used in MWTT to cover every possible dependency relation. We did not focus on the social media sources due to its dynamic and noisy nature [7]. The extraction from the mentioned sources were done as follows -

- Ebooks - They were downloaded from freeTamilebooks site [1] using the request library and then parsed using a custom parser written with the help of epub and BeautifulSoup python libraries. We focused on the ebooks containing stories or novels. These selected ebooks consist of publication dates ranging from 1900 to 2021.
- News - News articles were scrapped from Theekkathir [2]. Theekkathir is a Tamil newspaper run by Toiling Masses Welfare Trust Tamil Nadu. The articles were scraped using a custom parser written with Selenium and BeautifulSoup python libraries. The data consisted of formal news articles. The scraped news articles mostly consist of political matters. The news articles were scrapped in the years of 2021-22. .
- Movie Reviews - These were taken from a Kaggle repository [3]. The movie reviews present in this kaggle dataset were scraped from a puthiyathalaimurai website [4].

[1] https://freetamilebooks.com/
[2] https://theekkathir.in/
[3] https://www.kaggle.com/sudalairajkumar/tamil-nlp
[4] puthiyathalaimurai.com

- Wikipedia - These wiki articles were also taken from a Kaggle repository [5] and we also scrapped some articles. We decided to do targeted scrapping based on the requirement of better representation of some particular named entities or tags during the annotation process.

- Grammar Sentences - Simple grammar sentences were included from government school books. These were introduced to under-represented forms of relations of tokens in Tamil.

2.3 Data Curation

We decided to have paragraphs [8] instead of single sentences because of its benefits to higher level tasks like coreference resolution, next sentence prediction and text summarization. We decided to have short paragraphs of maximum 4 sentences. This decision was backed by a practical need to save data from some sources, which was being lost in the selection process. The data was taken as sentences, only from the grammar books source,. We were able to collect 2 lac paragraphs from all the sources. 30,000 sentences were initially selected through a series of steps - Data was curated in the form of paragraphs along with the fields like minimum sentence length, maximum sentence length, number of words, number of sentences, number of numerals, number of modern punctuations like (’“‘”()-[];:‘”<>/@$*_ %’) .

Paragraphs beginning with invalid characters like ’’,’!’,’,’, or having a number of numerals/English words more than half the words or had modern punctuations were dropped. Next we put a size constraint on the sentences to prevent extra large or extra small sentences. We try to incorporate sufficiently large sentences by finding min max threshold (3 token length and 30 token length) on sentence lengths in the paragraph that led to at most 70 % data retention. Now we analyzed the availability of data obtained from different categories. More about the original sample distribution of the scraped data can be found in the Appendix B. After several rounds of discussion involving scarcity of data in

[5] https://www.kaggle.com/disisbig/tamil-wikipedia-articles

few categories, aiming for equal representation of different sources, and so on, we finalized percentages of different types:

- Grammar books (5%)
- Ebooks (25%)
- Wiki (30%)
- News (20%)
- Movies (20%)

Now to select 30,000 sentences (around 7903 paragraphs), we applied stratified random sampling on the data. More priority was given to the fewer paragraphs containing few english words and numerals were given. Rest were selected randomly for each source.

This gave a corpus of 30,000 sentences, 7900 paragraphs consisting of sufficiently large sentences, numerals and English words.

2.4 Data Exploration

We had around 30,000 sentences after the data curation process. We planned to annotate 10,000 sentences out of this selected pool of data. Since our dataset consisted of paragraphs of a maximum of 4 sentences, we present the general statistics related to them in 2.1. Our dataset also contains 500 simple grammar sentences, mostly comprising of small to medium (7-10 tokens) length sentences. The exact distribution of data from each source can be found in Fig. 2.1. We performed an analysis of the data present in the form of paragraphs. The majority of the instances are formal in nature with some informal text coming from movie articles.

We wanted our dataset to represent the real-life Tamil textual instances. Hence we allowed some English words and numerals in our datasets. We were not aiming for a complete code-mixed dataset. This decision will also help in giving more examples to

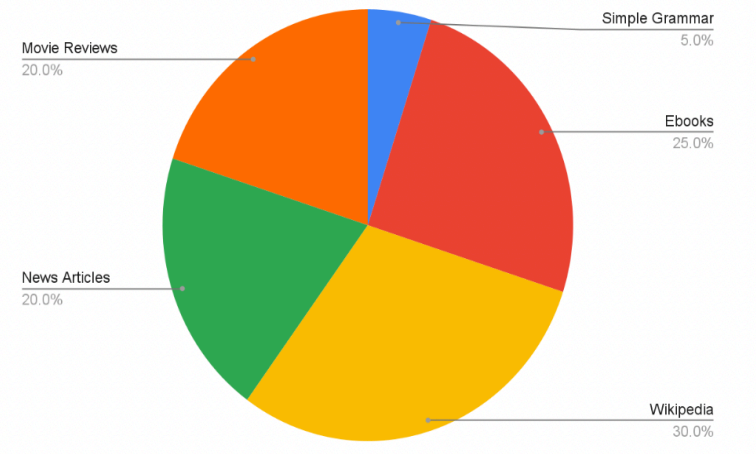


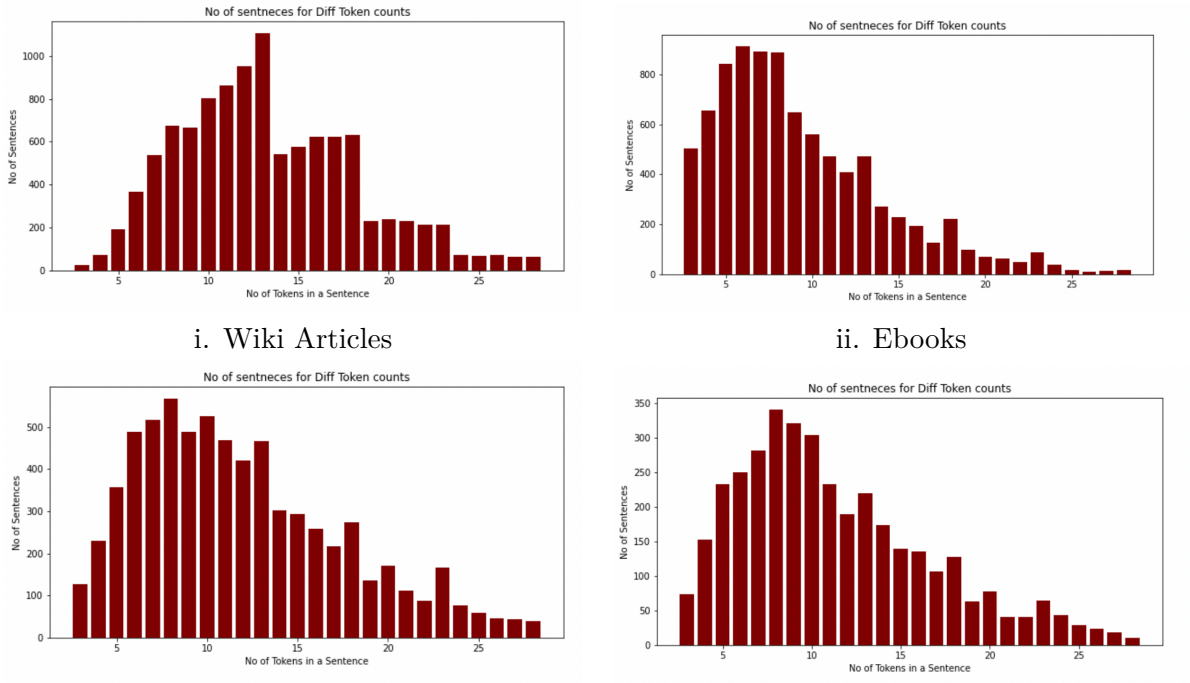
Figure 2.1: Contribution of Different Sources in Curated Dataset

Source	No of Paragraphs	No of Sentences	No of Paragraphs having English words / numerals
Ebooks	2347	8771	199
Movie Reviews	939	3714	87
News Articles	1857	6960	759
Wiki Articles	2760	10750	1395
TOTAL	7903	30195	2440

Table 2.1: General Statistics of the Curated Dataset

Source	Count of English Words	Count of Numerals	No of Sentences having both English words and numerals
Ebooks	102	276	6
Movie Reviews	36	81	1
News Articles	81	1138	1
Wiki Articles	851	2146	26
TOTAL	1070	3641	34

Table 2.2: Occurrence of the English words and numerals in Curated Dataset



i. Wiki Articles ii. Ebooks

iii. News Articles iv. Movies Reviews

Figure 2.2: Sentence Length Distribution of Different Sources

features, tags, and relations like Foreign, NumType, and so on in UD which are not present or are underrepresented in the current two Tamil treebanks. Table 2.2 shows the frequency of English words and numerals. There was mostly only one English word or numeral seen in a sentence in the dataset. Also, there were very few sentences containing both of them. News and wiki datasets contain the majority of numerals in the form of factual counts for an instance or dates. Wiki dataset contains the majority of English words in the form of terminologies or entities.

We plotted the sentence length distribution for different sources as seen in Fig. 2.2. They all show a similar trend in the distribution due to stratified sampling. A choppy or step-type trend is seen in the case of Wikipedia articles because of the high availability of data for each length value. Ebooks have the majority of the sentence within the 10 token length value. The news and movie datasets are very similar to each other. There is a significant amount of longer sentences in these datasets due to the conversational nature of the text present in them. These plots also show the importance of incorporating longer sentences in the dataset if we want models to perform well on real-life datasets.

i. Wiki Articles

ii. Ebooks

iii. News Articles

iv. Movies Reviews

Figure 2.3: Wordclouds of Data Curated from Different Sources

We formed word clouds from the datasets curated from different sources which can be seen in Fig. 2.3. Stopwords were removed from the text, though the list of stopwords was not exhaustive. Top 10 frequent words were also noted from each source. "Indian", "year", and "yes" were some of the most frequent words in Wikipedia source data. This also supports the analysis of the majority of numerals coming from this source. The majority of the articles are in the Indian context too. "Aravindan", "but" and "girl" were some of the most frequently used words in the ebooks dataset. Since most of the ebooks are related to fantasies or stories, not much can be commented on after observing the most frequent words. The news articles are scraped during the time period when farmer protests were going on in India in the year 2021-22. Hence the words like "Government", "farmers" and "Indian" are frequently seen in the dataset. "Love", "story" and "Vijay" are some of the frequently used words in this dataset. This prompts us toward the aspects generally covered in a movie review like the storyline, the actor or actress, genre, and the views. Also, most of the reviews are related to South Indian movies.

The next step consisted of choosing and annotating 10,000 sentences from this selection.

Chapter 3

Annotation Platform

3.1 Related Work

There are a number of existing annotation platforms or tools available for both offline and online use to annotate data in CoNLL-U format. There are quite a few listed on the Universal Dependencies site too [1]. Like any system, every platform has its pros and cons. We will go through some of these platforms. Brat is a web based tool that supports many NLP labeling tasks including morphosyntactic annotations [9]. It is a configurable and collaborative tool . This tool needs to be installed locally or on a web server if available. Focusing on the ConLL-U format, manually selecting the same token for different field tags can include errors. Also, working with longer text spans proves to be inconvenient in this. WebAnno is a web-based annotation tool for a wide range of linguistic annotations [10]. It is flexible, configurable and offers crowdsourcing options to speed up the process. Brat and WebAnno are powerful tools but they are not available online and are difficult to install [11]. UD Annotator is another open-source tool for annotating Universal Dependencies [12]. It is mainly focused on a single file annotated by a single user. It provides a closer look to CoNLL-U data type annotation but lacks other small features like searching and validation tests of Universal Dependencies. Conlu Editor is a tool which facilitates the editing of syntactic relations and morphological features of

[1] https://universaldependencies.org/tools.html

files in CoNLL-U format [13]. Unlike the previous platforms, it is solely focused on CoNLL-U annotations and also provides tree level view of dependency graphs along with flat ones. Datasaur [2] is an online collaborative tool which can be used for text annotation for POS, NER, morphology and dependency parsing. But it is not solely focused on CoNLL-U based annotations and hence lacks features like lemma and misc. We analyzed several other tools to find the one fitting best to our purpose. We felt the need of a collaborative and accessible platform solely focused on CoNLL-U annotations.

3.2 Requirements

We wanted a setup that is easily accessible by all the annotators with a minimum setup requirement. We also wanted the setup to be similar to some existing software so that the sense of familiarity can reduce the learning curve for the platform. We had a group of annotators working together to annotate different fields of a sample, so collaborativeness became a mandatory requirement for the platform. We found that information from separate fields helped in each other's annotation. So we wanted all the annotations to be present at one place for a sample, more like the CoNLL-U format itself. Basic requirements like easy searching, option of reverting back to some version of our dataset, log of activity and conditional formatting were also desired from the setup.

We decided to go with the google sheets [3] setup for carrying out the annotation process. The google sheets is free, online, accessible, collaborative, easy to use platform fulfilling most of our basic requirements too. They have version control and also track the activity of users. It can be customized to make the data look more understandable. Features like comment and reviews make it ideal for asynchronous workflow. But it is general purpose software for managing spreadsheets. There are projects where the developers have used google spreadsheets along with their developed tools for annotation purposes in fields like ontology and others [14]. To make it ideal for the CoNLL-U annotation process, we developed a web platform equipped with all the necessary features. The entire system is

[2] https://datasaur.ai/
[3] https://www.google.com/sheets/about/

called CoNLL-U GSheets. CoNLL-U GSheets

The system consists of Google Sheets, Google Drive and a set of developed tools. The tools are presented on a web platform. This platform is developed using Flask, HTML, CSS and Javascript. The logic is implemented with the help of python libraries like pandas [4], numpy [5] and sklearn [6]. The google sheets are connected to the backend by using the python library gspread [7]. Now let us look at the workflow identified for the annotation process which works best with this system.

3.3 Workflow

There are three google sheets used in the setup. Two of them are for the two teams of annotators and the third one is for the reviewers. Sentences are uploaded to the two sheets for annotators containing only a fraction of the same sentences. This set of same sentences are also uploaded to the review sheet, to be annotated by the reviewers which then will be referred to as the gold dataset. The purpose of the same sentences is to perform qualitative analysis like the Inter Annotator Agreement (IAA). The sentences are uploaded in the CoNLL-U format to the sheet by the program containing both metadata and initial tokens produced. The program allows users to add metadata according to their requirements. The initial tokenization is performed based on white spaces and punctuation marks. The annotators then start annotating. The first task of annotation is multiword decomposition. After the validation of this multiword decomposition by the reviewers using our tools, annotations of the rest of the fields can start parallely. After the annotations are done, the reviewers can perform format and validation checks using our tools to make corrections and provide feedback to the annotators. Other functionalities like label statistics and inter annotator score can be used to analyze the quality of annotation. Once this set of annotated data is given an approval by the reviewers, it is freezed and the whole process can be repeated on a new bunch of uploaded sentences. This provides the flexibility in

[4] https://pandas.pydata.org/
[5] https://numpy.org/
[6] https://scikit-learn.org/stable/
[7] https://docs.gspread.org/en/latest/

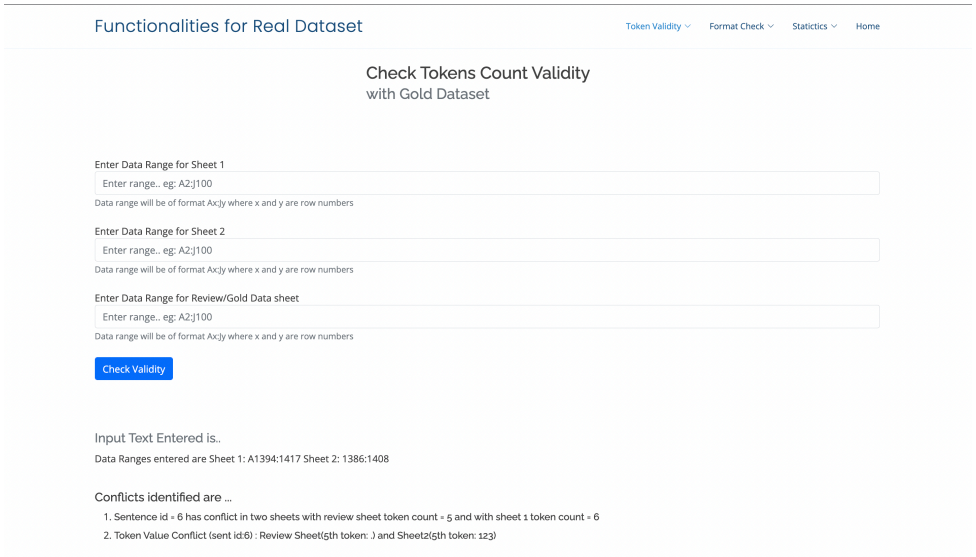


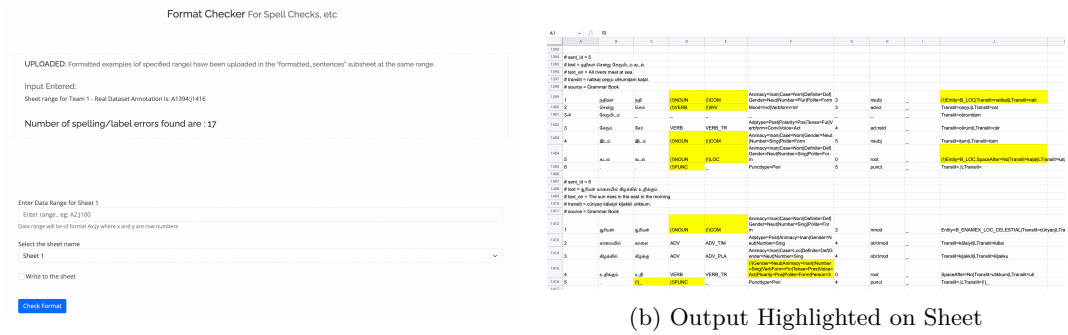
Figure 3.1: CoNLL-U GSheets Functionality: Tokenization Validation of Similar Sentences

the annotation process, adopting a kind of agile approach. The following section will give an overview on the features provided by the developed web platform.

3.4 Features of CoNLL-U GSheets

- Tokenization Validity - This tool allows to check whether there are any discrepancies in the tokens of the similar sentences across the google sheets. It prints the conflicts, if any, highlighting the dissimilar token counts or dissimilar tokens across the sheets for the same sentence as seen in Fig. 3.1. This tool is necessary for keeping the tokenization of the similar sentences consistent so that we can calculate Inter Annotator Scores across the sheets. Multiwords are also highlighted so that reviewers can analyze them at a glance.

- Spell and Format Validation - This tool is useful after the annotators have done the first round of annotation. It performs the format checks on the tags written with the tags present in the tag set. Spelling checks are performed for POS, Morphology, NER and Dependency fields. Other checks involve highlighting the empty fields for a non multi word token. These errors are written to the google sheet having these inconsistencies highlighted by a color as seen in Fig 3.2.b.



(a) Format Check Frontend

(b) Output Highlighted on Sheet

Figure 3.2: CoNLL-U GSheets Functionality: Format and Spell Check of Entered Range

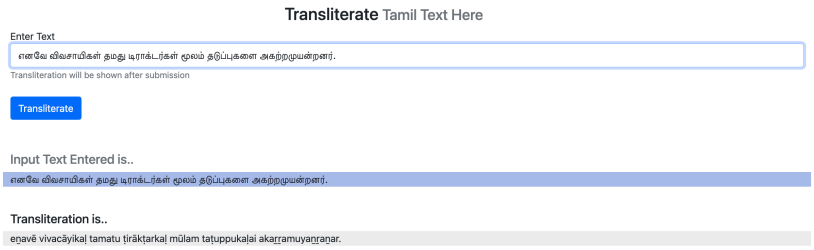


Figure 3.3: CoNLL-U GSheets Functionality: Transliteration of Input Text

- Transliteration - It helps in transliterating any Tamil text as visible in Fig. 3.3. This tool is also used internally by the format validation tool to automatically generate the transliterations for sentence, words and lemmas.
- Universal Dependencies Validation - The platform allows to run the UD validation tests on the annotated data so that those errors can be corrected at the start only and the data is release ready. These tests contains 6 levels of checks ranging from basic CoNLL-U format check to language specific guidelines[8].
- Label Statistics - This helps in visualizing and recording the count of tags present in different fields like UPOS, FEATS, DEPREL and MISC in the annotated data. It can help in understanding the tag distribution in different sources or finding the underrepresented tags.
- Inter Annotator Score (IAA) - This tool calculates and records the IAA between

[8] https://universaldependencies.org/validation-rules.html

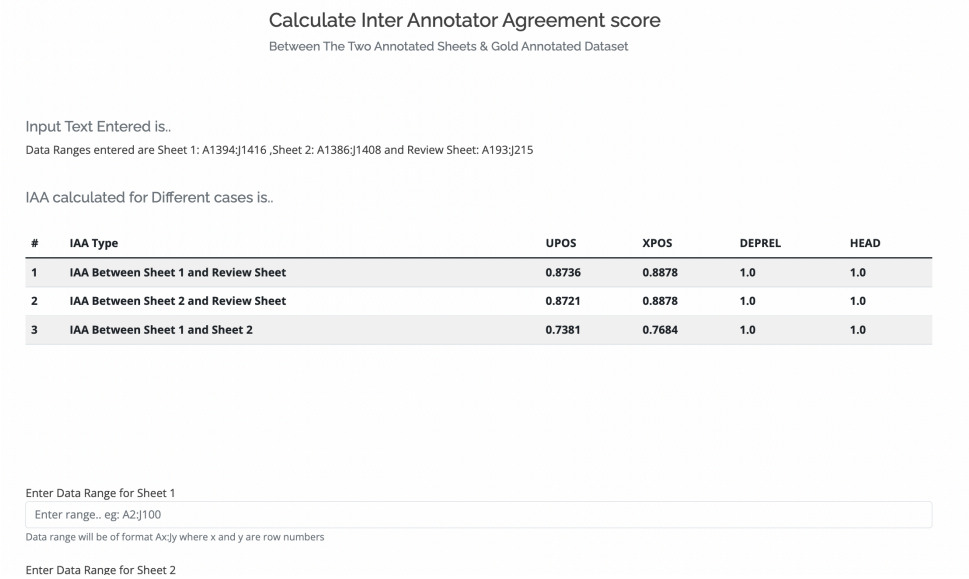


Figure 3.4: CoNLL-U GSheets Functionality: Inter Annotator Score of Similar Sentences

the three sheets as seen in Fig. 3.4. These IAA scores are calculated for the similar sentences across the sheets. Cohen Kappa is used for this qualitative measure. It is a statistical measure of inter-rater agreement for categorical items. Its value is ranges from 0 to 1. Score of 0.81 to 1.00 is interpreted as a perfect agreement, score 0.61 to 0.8 indicates substantial agreement, 0.41 to 0.6 indicates moderate agreement and so on.

- Tag set visualizer - This helps in visualizing and checking the tag set uploaded on the platform

Other features involve curation and freezing of final data. These tools are also being used in the annotation of real data.

Chapter 4

Annotation Guidelines

We decided to follow the universal dependency guidelines as a starting point for our annotation guidelines. Universal Dependencies (UD) is a project that is developing cross-linguistically consistent treebank annotation for many languages, with the goal of facilitating multilingual parser development and research on parsing and crosslingual learning [15]. It started in 2014 and over the years, the community has grown vastly. Currently UD is working with version 2. UD release v2.5 contains 157 treebanks representing 90 languages. Version 2 of UD introduced changes in various morpho syntactic tasks, mostly keeping the tag sets the same. UD provides the basic guidelines on its site [1]. They have mentioned that UD is not necessarily an optimal parsing representation. It is clear that the need for cross-linguistic consistency and perspicuity often runs counter to the requirements of optimal parsability for specific languages [3]. Hence we need to identify language specific requirements to bring optimal parsability. With the same in mind, we decided to form our annotation guidelines. We aimed at understanding the multiword expressions, morphological nature of the language and developing a diverse NER tag set.

[1] https://universaldependencies.org/guidelines

4.1 Tokenization

Tokenization is the identification of each “atomic” unit, representing the very first operation to be performed in NLP [16]. These atomic units are also referred to as ”tokens”. Tokenization can be performed at levels such as byte level, word level, sentence level, subword level and so on. The notion of token must first be defined before computational processing can proceed [17]. UD views the basic units of annotation as the syntactic words and not the orthographic or phonological words [15]. This means the clitics need not be splitted in all cases. The dependency relations hold between words and not morphemes, since UD is based on the lexicalist view of syntax. But we do need to split the compound or multi-word tokens.

We started by performing sentence tokenization on the text corpus to get individual sentences as output. This is necessary for annotating linguistic information such as Part of Speech (POS) and dependency relations [18]. Sentences are formed by splitting on the new line characters and also on sentence delimiters like ’.’, ’!’ and ’?’. We plan to improve this by adding additional features to make it context specific. The next level of tokenization involves splitting the sentence into tokens by splitting on whitespaces and on punctuations like

The next step is identifying and decomposing multi-word tokens. Tamil is a morphologically rich language in which there are syntactic elements such as compound verbs, particles, clitics, etc, that are seen added to a word. These need to be split for further syntactic annotations.

We started by identifying the compound words that need to splitted for further analysis. We found out that not all the compounds occurring in Tamil need to be further broken down. Hence we developed two sections for compounds, one where we need to break them down and one in which no decomposition is needed. The following cases were identified where the compounds needed to be splitted further. Complex predicates consist of different Verb Forms and Auxiliary verbs. The predicate shall be splitted into main and auxiliary verbs appropriately.

E: Kumar was studying

T: குமார் படித்துக்கொண்டிருந்தான்

Transliteration - kumār paṭittukkoṇṭiruntāṉ

When a word is formed by an oblique stem + postposition (like 'பக்கம்'), it may be split into two words.Oblique stem is a modified noun. Postposition is a word or morpheme placed after the word it governs, for example -ward in homeward.

E: He came near the garden

T: தோட்டத்துப்பக்கம் வந்தான்

Transliteration - tōṭṭattuppakkam vantāṉ

When a word is formed by noun + postposition, it shall be splitted into two words.

E: He had no money

T: பணமில்லாமல் இருந்தான்

Transliteration - paṇamillāmal iruntāṉ

There are cases where we don't need to decompose the compound words. Some of these cases are word consist of NOUN-NOUN, word (NOUN) consist of VERB-NOUN, word (VERB) consist of NOUN-VERB, word consist of VERB-VERB, word consist of PREPOSITION-VERB and when a noun is suffixed by derived adverbs like 'ஆக' (aaka).

It is also necessary to segment clitics in some cases. Clitcs are bound forms which are affixed to a word by the phonological process [19]. Clitics can emphasize the meaning of the word and can even express hidden meanings. According to Thomas Lehman, all clitics are post-clitics in Tamil language. Tamil Language has a finite set of clitics. Clitics like உம்(um),தான்(taan),ஓ(oo),ஏ(ee),ஆ(aa),ஆவது (aavatu),ஆம் (aam) give meanings on its own in the sentences. Clitics like மட்டும் (mattum) and கூட (kuuta) are not referred in the Modern Grammar book [19]. The general idea was to split the clitic when separation of clitic does not change the meaning of the sentence.

E: Kumar also came

T:குமா ரும் வந்தான் (குமார் வந்தான்)

Transliteration - kumārum vantāṉ (kumār vantāṉ)

This is then explained by various examples and cases. Situations of not splitting are also highlighted in the annotation guideline.

4.2 Part of Speech (POS) Tagging

Parts of speech (POS), is a piece of information about a word, its morphological structure, inflectional paradigm and its potential role in a clause [20]. These are also referred to as word classes or lexical categories and are usually encoded as a short string, called part-of-speech (POS) tag [21]. POS tagging plays an important role in building models for numerous NLP tasks like information retrieval, text to speech synthesis and speech recognition.

There are also some popular tagsets available for Tamil along with the annotated corpora like Amrita [22] and BIS5. UD provides 17 POS tags for annotating the UPOS field in CoNLL-U format. These are considered sufficient for any natural language. Additional information to the POS token is expected to be encoded as a morphological feature in UD in the FEATS field. But there are difficulties that are encountered in this task are insufficient data, inherent POS ambiguities and unknown words, which are the major tagging failures in many cases [23]. Understanding sub categories in these abstract 17 POS tag classes can further help in building an optimal parser. We developed a hierarchical tag set to solve some of these problems.

We decided to follow a hierarchical structure as it brings flexibility to its usage for various applications and can be freely adjusted for the same [23, 24]. Hierarchical tags will help to understand the further distribution of POS categories in Tamil language. The first level of hierarchy is kept the same as 17 POS tags provided by UD, to support cross lingual learning. The sub classifications are provided in XPOS category as it can contain

language specific information. There is a maximum of 3 levels of hierarchy followed in some cases. These proposed tags are aimed to provide better accuracy in tagging and reduce the ambiguous cases along with help of morphological features.

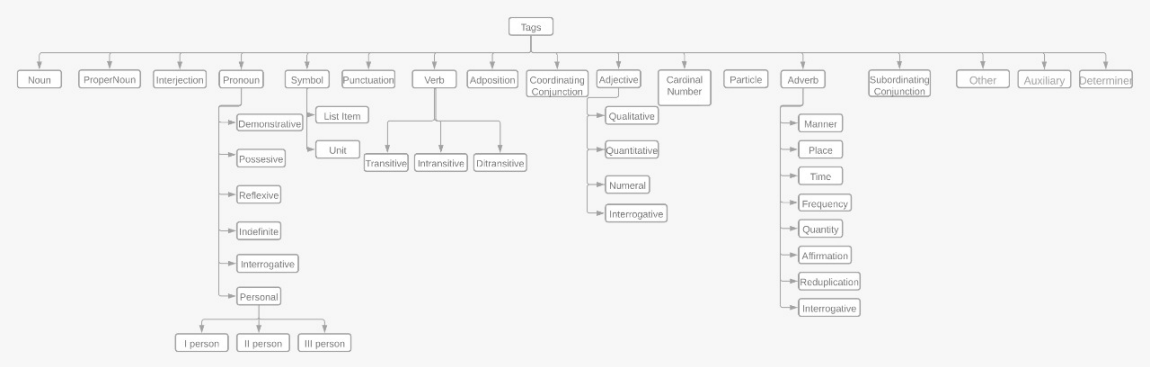


Figure 4.1: Proposed Hierarchical POS tag set

Fig. 4.1 shows the proposed tag set. Some of these tags also hold the morphology knowledge and are aimed to provide a better understanding of the word just using the POS tags. We acknowledge that there will be some redundancy in morphological annotations due to this. But the tag set alone can be used for detailed POS taggings in other tasks. Some of the further classifications of these 17 UD tags are as follows -

- Verbs were classified as Transitive, Intransitive and DiTransitive. This further classification helped in identifying the nature of clauses in the annotations.
- Pronouns were classified as possessive, interrogative, demonstrative, etc.
- Reduplication tag was introduced in adverbs along with other tags, for identifying words that express feelings like kalakala.
- Numeral Adjective was introduced as a subtag of Adjective to cover the Ordinal Numbers, which before required morphology features annotation for its complete definition.
- Symbols were further classified into Lists and Unit for helping in the NER annotations.

- Other tag were classified as Unknown and Foreign Words, to better understand the distribution of Other tag.

We developed annotation guidelines providing the detailed description and examples for each tag, and highlighting the confusion cases.

4.3 Named Entity Recognition (NER)

Named entity recognition (NER) is the task of identifying and classifying important nouns and proper nouns in a text [25]. These important nouns and proper nouns are called named entities. A named entity are words or phrases that are named real-world objects, such as a person, location, organization, product, and so on. It can be abstract or have a physical existence. NER plays a crucial role in various NLP applications such as Machine Translation, Information Extraction and Question Answering. Hence the accuracy of NER tasks can have a huge impact on these applications.

There are many challenges present in the NER task including lexical and semantic ambiguities. Malarkodi identified challenges related to NER for Tamil language [26]. They included lack of capitalisation in Tamil, noise in data, nested entities, name variations and spelling variations. Tamil has an agglutinative nature which often leads to increase in the number of entities in accordance to the increase in the case markers. Some examples are Koyilil with the suffix ”il” and marthandavarmanukku with the suffix ”kku”. There are also cases where common words like thamarai (lotus), malar (flower) can also be the names of the persons. These cases require inspection of the context to make an informed decision. There are other ambiguous cases where common nouns may be used as a proper noun in the context. Free order nature of Tamil text can also be a challenge in the model development stage. We developed a hierarchical tag set for the NER task.

We went ahead with the hierarchical approach for the same reasons as in the case of the POS task. It brings flexibility to the task and it can be further adapted for other tasks. These tags were formed after researching the hierarchical tag set proposed for

Tamil language by Sobha Lalitha Devi, Ontonotes flat NER tag set [27], and Sekine's Extended Named Entity Hierarchical tag set [28]. Sekine proposed an extensive tag set containing 150 tags in total. These detailed classifications were introduced to cover all the possible named entities. It has a high and diverse number of third level tags. But in a low resource setting like Tamil, data is generally scarce given the amount of annotated data currently available. Also some confusion cases were identified in this tag set like confusion for tag for a cryptocurrency or a man made lake. Ontonotes on the other hand defines 18 types of named entities. This simplification and minimisation in tags counts results in skipping some of the named entities that are seen in the corpora like natural objects, nominals, units. Sobha introduced the hierarchical tag set designed for Tamil along with the annotated corpora. We found some overlaps in the third level of hierarchy in the tags and felt that these can be reduced further in numbers. Since in this ever evolving world, new tags need to be introduced or definitions need to be updated for new entities like cryptos, etc. We incorporated various attributes of these tag sets along with some new additions to develop our tag set.

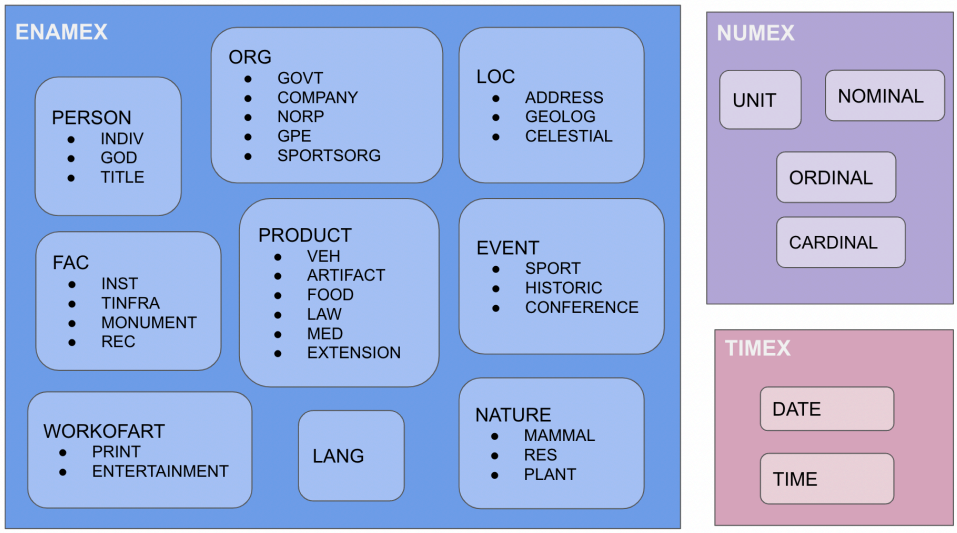


Figure 4.2: Proposed Hierarchical NER tag set

We have proposed a hierarchical tag set having three levels of hierarchy as shown in the Fig. 4.2. We have 3 tags in the first level of hierarchy, 15 tags in the second level of hierarchy and 31 tags in the third level of hierarchy. This is done to reduce the number of

tags at each higher level which can help in better model training in NLP. The first level consists of ENAMEX (name entity expressions), NUMEX (numerical entity expressions) and TIMEX (temporal entity expressions). This is an abstract high level distinction to select entities based on only names, numeral or temporal class. The second level consisting of 15 tags was designed taking motivation from OntoNotes tags. We combined some of its tags as a part of a single category like NORP and GPE were introduced as the subtags for ORG tag in our tagset. We also introduced some new categories in NATURE and NOMINAL in our tag set to cover the missed out named entities. We aimed to minimize the number of tags at second level in the thought of having more samples for each abstract category for better training of deep learning models. We also wanted to keep the categories at this level in accordance with OntoNotes so that transfer learning approaches can be applied for some of these universal categories. The third level consisted of 31 tags. These were developed to cover as many cases as possible in the parent tag. A fine grained division of categories was overlooked to aid the annotators and keep the annotations unambiguous. The tags at third level are grouped in categories based on some similarity in context with other tags. We acknowledge that grouping of categories like LAW and MED in PRODUCT, or GOD in PERSON are debatable and require validation from other people and annotations. These tags at third level can be further divided into categories for developing a fine grained NER classifier and it is considered as a future work depending on the number of retrieved entities after annotation.

We also listed the cases of confusion with explanations and examples. Some of these identified cases of confusion are listed below -

- NORP vs GPE - NORPs are generally adjectival, nationalities while countries are GPEs. Religions, political parties, and other named groups are ORGs, but their members are NORP
- INST (institute) vs MONUMENT - Monuments are places that are known as a tourist spot and have a historic and archeological significance. Institutional spaces are more associated with commerce and business aspects. They exist for operational

purposes.

- ORG v. PRODUCT - Makes, models, and versions are PRODUCTS; the company that produces them is ORG
- CARDINAL v. DATE/TIME - Dates and Times MUST have explicitly-mentioned units of time, otherwise they should be marked CARDINAL (even if the implicit unit seems obvious)
- FAC v. LOC - Facilities are man-made, Locations are natural There are few exceptions like man made lakes are placed in GEOLOG in LOC.

4.4 Morphology

The term "morphology" refers to the study of the structural relationships between different parts or aspects of the object of study, in a number of scientific disciplines [29]. The process of dividing a word into its constituent morphemes is known as the morphological analysis [30]. It also involved assignment of grammatical information like case, gender, animacy and so on to the word or morphemes. It also involved assignment of lexical information to a particular lemma or lexeme [31]. Morphemes are referred to as the smallest unit of meaning that a word in a particular language can be divided into. Morphology analysis is also the reverse process of morphological generation in which the word is built from its constituents. Morphological generation holds high importance in various other tasks in NLP like machine translation and information retrieval [30].

Tamil is a morphologically rich language and an agglutinative language. The set of morphemes are mostly seen as suffixes in the word. But, there are few cases where they are present as prefixes also [32]. Though there can be a large number of morphemes present in the word which should make the task of identifying them complex, these morphemes are generally seen to occur in structured and templatic mannar [33]. Morphology in Tamil is mainly concatenative and derivations that are also possible by means of adjectivalization, adverbialization and nominalization [34]. The POS tag also helps in further analyzing

the features associated with a word. The NOUN generally shows features like Gender, Animacy, Case, Definite, while the VERB shows features like VerbForm, Mood, Tense, Voice and so on. We follow UD guidelines to analyze the feature set for Tamil language.

UD represents the morphological specification of the word using 3 representations. These consist of lemma, POS tag and the set of features. In this section we would be focusing on specifying these features for the Tamil language. These features are used to specify the grammatical and lexical properties that are associated with the word form. These features are used to provide additional pieces of information about the word that may or maynot be present in the POS tag, which can help in understanding the morphosyntactic properties of the word with better accuracy. These are mentioned in the FEATS column of the CoNLL-U format annotations. They are annotated in the form 'Name=Value' like Animacy=Inan|Gender=Neut. UD defines three categories for these features for better identification [2] -

- Lexical features - These are considered as attributes of the lemmas or lexemes.
- Inflectional features - These are considered as the attributes for the word form rather than lemmas. There are further two categories of nominal and verbal features. The labels Nominal and Verbal are used as approximate categories only. Even the boundary between lexical and inflectional features is sometimes blurred: for example, gender is a lexical feature of nouns but an inflectional feature of adjectives or verbs.
- Layered features - Some features may be marked more than once for the same word. It is mentioned that the exact meaning of these annotated values depends on the language itself. For example, possessive adjectives, determiners and pronouns may have two different values of Gender and two of Number like Gender=Masc|Gender[psor]=Fem.

After analyzing the current two treebanks of Tamil on UD, we decided to analyze the lexical and inflectional features in context of Tamil language in our guidelines intialially. We listed down examples from the MWTT treebank, for each of the feature and value pairs. The remarks and glossed example sentences were also written in the guidelines. The

[2] https://universaldependencies.org/u/overview/morphology.html

following are some of the observations from analysis that are mentioned in the guidelines also -

- Inv feature value for the feature Number does not occur in Tamil as the non-defaults for noun does not occur in Tamil
- Dual and Tri values of Number are also not seen in Tamil language.
- Definite feature is not used in MWTT or TTB as there are no definite articles in Tamil. The only indefinite article present is which can be marked using Ind value of the feature. Rest of the values are not used.
- Degree of comparison is typically an inflectional feature of some adjectives and adverbs. Degree feature in Tamil is not used as the adjective on its own does not have a comparative or superlative form and makes use of the modifiers to express comparison.
- Since the articles do not occur in Tamil, PronType value Art can be avoided in cases like 'a boy came' with NumType=Card.
- Foreign feature should be used for code-switching or quoting in foreign languages and not be used for legitimate loanwords.
- Reflex feature is used to refer to the subject of the clause. It is applicable for pronouns and determiners and is marked as YES when it is reflexive.

4.5 Dependency Parsing

Dependency parsing is the method in which syntactic analysis is performed on the words present in the natural language texts. It examines a sentence's grammatical structure and find the relations or links between head words and modifier words [35]. Considering the sentence "big brown dog", the word "dog" determines it as the noun phrase and is considered as the head of the modifier words "big" and "brown". Understanding the grammatical structures of the natural language text helps in analyzing the language variations

and the social interactions. This syntactic analysis of the sentence plays an important role in other NLP tasks like semantic role labeling, machine translation and relation extraction [36]. Tamil is a free order language and the dependency tree structure is considered a cutting edge approach for parsing it [35].

In Dependency parsing, various tags represent the relationship between two words in a sentence. Universal Dependencies (UD) explains a simple typology of 3 kinds of phrasal units [37]:

1. Nominals: The primary means for referring to entities. They consist of a noun, proper noun or pronoun.

2. Clauses: It referes to the events. They comprise of a predicate along with its core argument dependents. It may also be extended with oblique modifiers. These core arguments are mostly nominals, and the oblique modifiers can be adverbial modifiers or nominals. There are complex cases, in which subordinate clauses can comprise of both core arguments and oblique modifiers. The function words that express grammatical information like voice, mood, tense, aspect and so on, may be seen as the predicate.

3. Modifiers: They are the canonical attributive modifiers of nominals, clauses, and other modifiers

In the annotation for the dependency parsing, the relation, referred to as the dependency relation is typed between the pair of words. There is a special relation called root, that is used to annotate word that is not dependent on any other word in the sentence in consideration.. The set of pairs of the dependents and heads, covering every word in the sentence, forms a rooted tree that helps in understanding the syntactic structure. We followed the annotation guidelines of UD for the dependency parsing task, analyzing the 37 dependency relations in the context of Tamil language. There are also cases of enhanced dependency relations which result in formation of the directed graph, which is left as a future work in our annotation guidelines.

The fields of interest in the CoNLL-U format for universal dependencies are HEAD, DEPREL and DEPS. We have explained and given examples for these fields in the Appendix A.

In our annotation guidelines, we identified 30 dependency relations applicable for our Tamil corpus. These are explained thoroughly in our guidelines along with the glossed examples in Tamil. The examples are glossed as follows: tamil tokens, english transliterations, glossing, english translations and remarks. We have also highlighted and explained the confusion cases. The remaining 7 universal dependency relations like expl, dislocated, clf, parataxis, goeswith, reparandum, dep are not seen in our Tamil corpus and are not given examples in this document. We made the language specific observations and some of them are listed below:

- The determiners in Tamil are demostratives in nature. இந்த (this) and அந்த (that), are being annotated with the det relation.
- Article ஒரு is annotated with the NUM POS tag and the nummod relation rather than the det relation.
- Possessive pronouns are treated with the nmod relation
- It is difficult to distinguish between an argument and adjunct and should be taken care at the semantic level. The relation obl:arg and obl can also be used to annotate it at the dependency level, though the former is preferred.
- Multiwords can be annotated with different relations like flat which is used for exocentric (headless) semi-fixed multiword expressions like names (Hillary Rodham Clinton) and dates (24 December). It contrasts with fixed, which applies to completely fixed grammaticized (function word-like) multiword expressions (like in spite of), and with compound, which applies to endocentric (headed) multiword expressions (like apple pie).
- SOV nature of word order of Tamil can result in the cases where orphan dependency relation is needed, though it is rare. One example of this cases is Vijay won gold

and Ajay bronze. Verb ”win” in Tamil will be at the right and gold will be attached to Vijay by an orphan relation.

- Case markersmarkers can be elided in the case of Tamil which change the POS tag for the word in different cases. One example is உருண்டையான -> உருண்டை + ஆன , since ஆன is elided here which makes the word behave as an adjective rather than a noun. These cases needed to be handled with care.

- Tamil uses light verbs with infinitives to express an activity such as causation, performed on the full verb expressed in the infinitive, which is closer to the grammatical function. The light verbs seen are vay, viḍu, and paar. In our corpus, the light verbs are considered as aux to avoid confusion.

- There are often confusion in the cases for ccomp and xcomp. The ccomp relation is used for expressing the relation between verbs in a reported speech while xcomp is mainly used for compound verbs.

Chapter 5

Models

We also planned to develop machine learning and deep learning model for various tasks involved in the morpho-syntactic analysis. We performed a literature review to understand the developments in this field and also to develop the baseline models.

NLP Tasks	Task Type	Models Used
a. POS Tagging b. NER Tagging	Sequential Labelling	CRFS, HMMs, Deep Learning models involving LSTM, GRU, transformer, etc
a. Morphological Analysis b. Lemmatization	Rule Based	Finite-State Morphological Analyser, Deep learning models
a. Dependency Parsing	Tree Parsing	Deep learning , graph based, or transition-based models

Table 5.1: Summary of Names of Models used in Different NLP task

Table. 5.1 presents the abstract list of the different models used for different morpho-syntactic analysis. However, most of them can be converted to seq2seq labeling task like POS, NER. We performed a comparative study on the open-source tokenizers for the Tamil language which can be found in Appendix D.

5.1 Related Work

There have been multiple approaches ranging from machine learning to deep learning, used for NER tagging. The scarcity of annotated benchmark datasets for the NER task in Tamil has resulted in relatively less progress in terms of model developments and evaluations. This also means that there is an opportunity for a lot of improvement and contribution in this domain. Most of the mentioned approaches can also be applied to POS tasks as both NER and POS are sequence labeling tasks. [26] trained CRFs for NER tagging as they are beneficial for sequential. They followed a hierarchical tagset consisting of 106 tags. They used a variety of features containing individual words, POS tags, noun phrase, verb phrase, combination of word and POS, and combination of word, POS, and chunk. They found an increase in the accuracy by working with just the root word which overcomes the problem of agglutination and post-processing heuristic rules which overcome the issue of nested entities. This helped in achieving a score of 70%. Srinivasagan makes use of both Rule-based and Hidden Markov models in succession [38]. Six rules were designed for the rule-based approach, which can be used to solve problems caused by agglutination and other ambiguities in NER. This approach helped in achieving a high F-score. [39] uses a chain classifier method, working with a hierarchical tag set of 3 levels. They were able to achieve a precision score of 42 for the outer level and 36.31 for the inner level. In the research [41][40], a hybrid Naive Bayes algorithm was applied to the extracted dataset from the Forum for Information Retrieval Evaluation (FIRE) Tamil documents. This classifier achieved an F-score of 83.54. The features were extracted using Regex, and morphological and contextual features were then extracted using POS tagger.

With the advent of word embeddings and feature extractors, deep learning models like RNN, LSTM, GRU, and so on are developed for NER tasks. A pipeline comprising feature extraction, feature selection, dimensionality reduction using SVD, and the application of a deep learning model can be a good start to test out these famous deep learning architectures [41]. The feature extraction can be done using Regex, a morphological analyzer, and a content feature extractor. There are models which require minimal to no data

pre-processing and feature engineering. A state-of-the-art model comprising bidirectional LSTM, CNN, and CRF is developed for the sequence labeling task [42]. CNN is used to compute the character embeddings for character-level representation, which are then passed through BiLSTMs and the output of BiLSTMs to CRF to decode the best label sequence. In the research [43], a dynamic layered model is proposed for hierarchical or nested NER prediction tasks, comprising stacked layers of Bi-LTSM and CRFs. It gives state-of-the-art results for two English NER datasets.

These deep learning approaches are also applied to various levels of morphosyntactic analysis. These deep learning approaches work on embeddings of the word, characters, and so on. In the research [44], they emphasize the importance of the sentence-level context in the initial character and word-based representations and achieve this by introducing a meta-BiLSTM-model that learns to combine their states. For each word, context-sensitive characters and word-based encodings are concatenated and put through another BiLSTM to form a combined context-sensitive encoding which then can be used by the feed-forward network. They showed state-of-the-art results on the CoNLLU 2017 shared task for POS tagging and morphology. For morphology, they considered a single morphological bundle as a single tag which reduced the problem of sequence labeling task. This approach opens the possibilities of using the models created for NER or POS, for the morphological task.

Dependency parsing can be implemented through searching strategies over parsing trees, graph-based and transition based. There are many deep learning approaches being used for this task over the years. In [45], it is considered as a seq2seq task where encoder-attention-decoder is used to predict the head for every word. This is then fed into the BiLSTM-CRF to predict the dependency relation. The tree like structure is imposed using a beam search with tree constraint method in the decoder side. It gave competitive results for English and Chinese benchmark treebanks. This approach can also be employed in our case, while the same model can be tested for other tasks too.

We aim to develop a complete pipeline for morphosyntactic analysis like in the case of open source dependency parser for Tamil, ThamizhiUDp [46]. It is composed of tokenizer and lemmatizer implemented using Stanza [47], POS tagger ThamiziPOSt [48],

morphological parser Thamizhimorph [49], and the dependency parser. To deal with the problem of scarcity of annotated data for each task, a direct deep learning approach was not viable. The author has used different methods for each task. ThamziMorph is based on the Finite State Transducer (FST) models, lexicons, meta-morphological rules. These meta-morph rules can also be used to generate pseudo labels in semi supervised learning approaches. ThamziPOSt, is developed using Stanza and uuparser trained on Amrita Tag set to predict UD POS tags and gave a F1 score of 93.27 for unseen data. The dependency parser leverages the information from other languages by multilingual training and it gave a Labelled Assigned Score (LAS) score of 62.39.

5.2 Developments

Value of Coefficient A	Value of Coefficient B	Accuracy
0.5	0.5	0.807
0	1	0.004
1	0	0.806
0.2	0.8	0.803

Table 5.2: Accuracy of Probabilistic Model for Different values of coefficient a and B

We implemented the baseline models for POS tags using the MWTT treebank. We started by creating probabilistic models. Two matrices were created. In the first matrix, the co-occurrence of the tags are stored, where Mat[i,j] denotes the number of times tag "j" came immediately after tag "i". In the second matrix, the co-occurrence of the tag and word are stored. This is referred as emission probabilities. The probability for a tag for a word is calculated as follows:

$$\text{UPOS Tag} = \underset{tag}{a\ rgmax}\{A * P(tag|word) + B * P(prevtag|tag)\}$$

, where A and B are coefficients.

Table 5.2 shows the accuracy calculated for UPOS tag prediction. It can be seen that probability of a tag for word is more significant and greatly influence the accuracy of the predictor.

ML Algorithm A	Accuracy in Case 1	Accuracy in Case 2
Decision Tree Classifier	0.887	0.899
Logistic Regression	0.815	0.823
Random Forest Classifier	0.863	0.867
SVM	0.839	0.875

Table 5.3: Accuracy of ML Models for the Two Cases

We also tested some popular machine learning algorithms for the UPOS prediction task. The result can be seen in the Table 5.3. There are two cases for which the accuracy is calculated. In first case, model just learns the token and tag in the training set. In second case, the relation between previous and current tag is also considered.

Chapter 6

Future Work

This is the foundation stone for a big NLP project. The dataset is extracted, curated, and ready to use. But topics like the annotation platform, guidelines, and models have scope for more work to make a significant and useful contribution to the NLP community.

New features like dependency tree visualization, metadata checks, translation validity, and so on can be implemented to make it more useful for CoNLL-U format annotations.

Further areas for the annotation guidelines can be explored covering the complex topics like enhanced dependency graph relations for dependency parsing, layered features for morphology, meta morphology rules, other cases of compound word decomposition, and so on. This can result in annotation guidelines version 2.

Model development is still at an early stage. The state-of-the-art models, mentioned in the related work, can be developed and tested with current Tamil treebanks or for other languages. Since the scarcity of the annotated corpus is one of the major issues in Tamil NLP, semi-supervised approaches and multilingual learning techniques can also be explored.

Chapter 7

Conclusion

There is a big scope for improvement and room for contribution in Tamil NLP. We identified the need for a bigger and more diverse treebank for Tamil for the morphosyntactic analysis. We extracted the data from namely 5 sources, consisting of both formal and informal instances which gave a better representation of the real-world Tamil textual instances.The dataset is curated in the form of paragraphs to assist in later tasks like semantic role labeling, discourse, and so on.

Detailed annotation guidelines were developed for each task in the morpho-syntactic analysis, facilitating in efficient and accurate annotations. Universal Dependencies guidelines were tested for the Tamil language and changes were introduced for uncovered cases. We also kept the basic fields of annotations like UPOS and DEPREL as per UD standards to further facilitate the scope of multilingual learning. Hierarchical tag sets for NER and POS were introduced for flexibility in application in different scenarios.

During the research, the need for an accessible, collaborative, and easy-to-use CoNLL-U based annotation tool led to the development of the CoNLL-U GSheets. It helped in producing the release-ready annotated data at an efficient speed.

We performed a literature review of various state-of-the-art developments for each task. We also tested the existing Tamil treebanks with some baseline models. We identified and set up a pipeline for the complete dependency parser. This research laid the foundation of

a complete ecosystem that will be immensely beneficial for the NLP for Tamil. Necessary future works were also listed in the research.

Bibliography

[1] K. Chowdhary, "Natural language processing," *Fundamentals of artificial intelligence*, pp. 603–649, 2020.

[2] S. S. Sundari, "Nlp towards revival and development of tamil."

[3] J. Nivre, "Towards a universal grammar for natural language processing," in *International conference on intelligent text processing and computational linguistics*. Springer, 2015, pp. 3–16.

[4] A. Kunchukuttan, D. Kakwani, S. Golla, A. Bhattacharyya, M. M. Khapra, P. Kumar *et al.*, "Ai4bharat-indicnlp corpus: Monolingual corpora and word embeddings for indic languages," *arXiv preprint arXiv:2005.00085*, 2020.

[5] A. Gupta, M. Ayyar, A. K. Singh, and R. R. Shah, "raiden11@ iecsil-fire-2018: Named entity recognition for indian languages." in *FIRE (Working Notes)*, 2018, pp. 174–186.

[6] P. Krishnamurthy and K. Sarveswaran, "Towards building a modern written tamil treebank," in *Proceedings of the 20th International Workshop on Treebanks and Linguistic Theories (TLT, SyntaxFest 2021)*, 2021, pp. 61–68.

[7] T. Baldwin, P. Cook, M. Lui, A. MacKinlay, and L. Wang, "How noisy social media text, how diffrnt social media sources?" in *Proceedings of the Sixth International Joint Conference on Natural Language Processing*, 2013, pp. 356–364.

[8] B. Wilie, K. Vincentio, G. I. Winata, S. Cahyawijaya, X. Li, Z. Y. Lim, S. Soleman, R. Mahendra, P. Fung, S. Bahar *et al.*, "Indonlu: Benchmark and resources for evalu-

ating indonesian natural language understanding," *arXiv preprint arXiv:2009.05387*, 2020.

[9] P. Stenetorp, S. Pyysalo, G. Topić, T. Ohta, S. Ananiadou, and J. Tsujii, "Brat: a web-based tool for nlp-assisted text annotation," in *Proceedings of the Demonstrations at the 13th Conference of the European Chapter of the Association for Computational Linguistics*, 2012, pp. 102–107.

[10] S. M. Yimam, I. Gurevych, R. E. de Castilho, and C. Biemann, "Webanno: A flexible, web-based and visually supported system for distributed annotations," in *Proceedings of the 51st Annual Meeting of the Association for Computational Linguistics: System Demonstrations*, 2013, pp. 1–6.

[11] M. Neves and J. Ševa, "An extensive review of tools for manual annotation of documents," *Briefings in bioinformatics*, vol. 22, no. 1, pp. 146–163, 2021.

[12] F. Tyers, M. Sheyanova, and J. Washington, "Ud annotatrix: An annotation tool for universal dependencies," in *Proceedings of the 16th International Workshop on Treebanks and Linguistic Theories*, 2017, pp. 10–17.

[13] J. Heinecke, "Conllueditor: a fully graphical editor for universal dependencies treebank files," in *Proceedings of the Third Workshop on Universal Dependencies (UDW, SyntaxFest 2019)*, 2019, pp. 87–93.

[14] E. Maguire, A. González-Beltrán, P. L. Whetzel, S.-A. Sansone, and P. Rocca-Serra, "Ontomaton: a bioportal powered ontology widget for google spreadsheets," *Bioinformatics*, vol. 29, no. 4, pp. 525–527, 2013.

[15] J. Nivre, M.-C. de Marneffe, F. Ginter, J. Hajič, C. D. Manning, S. Pyysalo, S. Schuster, F. Tyers, and D. Zeman, "Universal dependencies v2: An evergrowing multilingual treebank collection," *arXiv preprint arXiv:2004.10643*, 2020.

[16] B. Habert, G. Adda, M. Adda-Decker, P. B. de Marëuil, S. Ferrari, O. Ferret, G. Illouz, and P. Paroubek, "Towards tokenization evaluation," in *Proceedings of LREC*, vol. 98, 1998, pp. 427–431.

[17] J. J. Webster and C. Kit, "Tokenization as the initial phase in nlp," in *COLING 1992 Volume 4: The 14th International Conference on Computational Linguistics*, 1992.

[18] B. Jurish and K.-M. Würzner, "Word and sentence tokenization with hidden markov models." *J. Lang. Technol. Comput. Linguistics*, vol. 28, no. 2, pp. 61–83, 2013.

[19] E. Annamalai and S. B. Steever, "Modern tamil," *The dravidian languages*, pp. 100–128, 1998.

[20] S. Moeller, L. Liu, and M. Hulden, "To pos tag or not to pos tag: The impact of pos tags on morphological learning in low-resource settings," in *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, 2021, pp. 966–978.

[21] D. Zeman, *The World of Tokens, Tags and Trees.* Ústav formální a aplikované lingvistiky, 2018.

[22] M. Anand Kumar, V. Dhanalakshmi, K. Soman, and S. Rajendran, "A sequence labeling approach to morphological analyzer for tamil language," *International Journal on Computer Science and Engineering*, vol. 2, no. 06, pp. 1944–1951, 2010.

[23] G. Lee, J. Cha, and J.-H. Lee, "Hybrid pos tagging with generalized unknown-word handling," in *International workshop on information retrieval with Asian languages (IRAL).* Citeseer, 1997, pp. 43–50.

[24] K. Sarveswaran and S. Mahesan, "Hierarchical tag-set for rule-based processing of tamil language," 2016.

[25] B. Mohit, "Named entity recognition," in *Natural language processing of semitic languages.* Springer, 2014, pp. 221–245.

[26] C. Malarkodi, P. R. Rao, and S. L. Devi, "Tamil ner-coping with real time challenges," in *Proceedings of the Workshop on Machine Translation and Parsing in Indian Languages*, 2012, pp. 23–38.

[27] R. Weischedel, S. Pradhan, L. Ramshaw, M. Palmer, N. Xue, M. Marcus, A. Taylor, C. Greenberg, E. Hovy, R. Belvin *et al.*, "Ontonotes release 4.0," *LDC2011T03, Philadelphia, Penn.: Linguistic Data Consortium*, 2011.

[28] S. Sekine, K. Sudo, and C. Nobata, "Extended named entity hierarchy." in *LREC*, 2002.

[29] A. Álvarez and T. Ritchey, "Applications of general morphological analysis," *Acta Morphologica Generalis*, vol. 4, no. 1, 2015.

[30] S. Menaka, V. S. Ram, and S. L. Devi, "Morphological generator for tamil," *Proceedings of the Knowledge Sharing event on Morphological Analysers and Generators (March 22-23, 2010), LDC-IL, Mysore, India*, pp. 82–96, 2010.

[31] J. P. Jayan, R. Rajeev, S. Rajendran *et al.*, "Morphological analyser and morphological generator for malayalam-tamil machine translation," *International Journal of Computer Applications*, vol. 13, no. 8, pp. 0975–8887, 2011.

[32] K. Sarveswaran, G. Dias, and M. Butt, "Using meta-morph rules to develop morphological analysers: a case study concerning tamil," in *Proceedings of the 14th International Conference on Finite-State Methods and Natural Language Processing*, 2019, pp. 76–86.

[33] T. Lehmann, "A grammar of modern tamil. pondicherry institute of linguistics and culture," 1993.

[34] L. Ramasamy and Z. Žabokrtskỳ, "Tamil dependency parsing: results using rule based and corpus based approaches," in *International Conference on Intelligent Text Processing and Computational Linguistics*. Springer, 2011, pp. 82–95.

[35] C. A. Kumar, A. Maharana, S. Murali, B. Premjith, and S. Kp, "Bert-based sequence labelling approach for dependency parsing in tamil," in *Proceedings of the Second Workshop on Speech and Language Technologies for Dravidian Languages*, 2022, pp. 1–8.

[36] P. Qi, T. Dozat, Y. Zhang, and C. D. Manning, "Universal dependency parsing from scratch," *arXiv preprint arXiv:1901.10457*, 2019.

[37] M. Toska, J. Nivre, and D. Zeman, "Universal dependencies for albanian," in *Proceedings of the Fourth Workshop on Universal Dependencies (UDW 2020)*, 2020, pp. 178–188.

[38] K. G. Srinivasagan, S. Suganthi, and N. Jeyashenbagavalli, "An automated system for tamil named entity recognition using hybrid approach," in *2014 International Conference on Intelligent Computing Applications*, 2014, pp. 435–439.

[39] N. Abinaya, N. John, B. H. Ganesh, A. M. Kumar, and K. Soman, "Amrita_cen@ fire-2014: named entity recognition for indian languages using rich features," in *Proceedings of the Forum for Information Retrieval Evaluation*, 2014, pp. 103–111.

[40] R. Srinivasan and C. Subalalitha, "Automated named entity recognition from tamil documents," in *2019 IEEE 1st International Conference on Energy, Systems and Information Processing (ICESIP)*. IEEE, 2019, pp. 1–5.

[41] P. S. Kathiravan and R. Saranya, "Named entity recognition (ner) for social media tamil posts using deep learning with singular value decomposition," EasyChair, Tech. Rep., 2021.

[42] X. Ma and E. Hovy, "End-to-end sequence labeling via bi-directional lstm-cnns-crf," *arXiv preprint arXiv:1603.01354*, 2016.

[43] M. Ju, M. Miwa, and S. Ananiadou, "A neural layered model for nested named entity recognition," in *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)*, 2018, pp. 1446–1459.

[44] B. Bohnet, R. McDonald, G. Simoes, D. Andor, E. Pitler, and J. Maynez, "Morphosyntactic tagging with a meta-bilstm model over context sensitive token encodings," *arXiv preprint arXiv:1805.08237*, 2018.

[45] Z. Li, J. Cai, S. He, and H. Zhao, "Seq2seq dependency parsing," in *Proceedings of the 27th International Conference on Computational Linguistics*, 2018, pp. 3203–3214.

[46] K. Sarveswaran and G. Dias, "Thamizhiudp: A dependency parser for tamil," *arXiv preprint arXiv:2012.13436*, 2020.

[47] P. Qi, Y. Zhang, Y. Zhang, J. Bolton, and C. D. Manning, "Stanza: A python natural language processing toolkit for many human languages," *arXiv preprint arXiv:2003.07082*, 2020.

[48] K. Sarveswaran and G. Dias, "Building a part of speech tagger for the tamil language," in *2021 International Conference on Asian Language Processing (IALP)*, 2021, pp. 286–291.

[49] K. Sarveswaran, G. Dias, and M. Butt, "Thamizhimorph: A morphological parser for the tamil language," *Machine Translation*, vol. 35, no. 1, pp. 37–70, 2021.

Appendix A

CoNLL-U format Data Annotation

CoNLL-U[1] is an annotation schema for describing linguistic features across diverse languages. There are 3 types of lines used in this format::

- Blank lines: They are used marking sentence boundaries.
- Comment lines: They start with the hash (#) They may display metadata like sentence id, sentence text, sentence translation to other language (mostly to English), transliteration, source and so on.
- Word lines: They containing the annotation of a word/token in 10 fields separated by single tab characters (ID, WORD, LEMMA OR STEM, UPOS, XPOS, FEATS, HEAD, DEPREL, DEPS and MISC).

ID	WORD	LEMMA	UPOS	XPOS	FEATS	HEAD	DEPREL	DEPS	MISC
# sent_no = 20									
# sent_id = news_3528_A93CD94E_2									
# text = தமிழோசை ஒய்வதில்லை.									
# text_en = Tamilosai does not rest.									
# translit = tamiḻōsai ōyvathillai									
# source = theekkathir_news_articles									
1	தமிழோசை	தமிழோசை	PROPN	PROPN	Animacy=Inan|Gender=Neut|Case=Nom|Number=Sing|Person=3	2	nsubj	_	Entity = B_LANG
2-3	ஒய்வதில்லை	_	_	_	_	_	_	_	SpaceAfter=No
2	ஒய்வது	ஒய்வது	VERB	TR	Gender=Neut|Animacy=Inan|Case=Nom|Number=Sing|VerbForm=Inf|Mood=Ind|Tense=Fut|Voice=Act	0	root		
3	இல்லை	இல்	DET	DET	Definite=Def	2	det		
4	.	.	PUNCT	PUNCT	PunctType=Peri	2	punct	_	_

Figure A.1: Data Annotated in CoNLL-U Format

Fig. A.1 shows a snapshot of a sentence annotated in CoNLL-U format. There are

[1] https://universaldependencies.org/format.html

some constraints applied to the fields in the word lines. The fields must not be empty. The fields other than FORM, LEMMA and MISC should not contain space characters. Unspecified values in all fields expect ID are denoted using underscore (_).

Each field of the 10 fields carry an important piece of information for morpho-syntactic analysis. The following briefly discuss the definition of each field:

1. ID: It represents the token id starting with value 1 for first token. In case of multi-words, non overlapping range of interger ids are displayed seprerated by a '-'.
2. WORD: It holds the token itself which may be a word or a punctuation symbol.
3. LEMMA: It is for the stem or lemma of the word form.
4. UPOS: It is used for annotating the POS tag from the Universal POS tag set.
5. XPOS: It is used for annotating language specific POS tag and can also be used for heirachical tag set. In case of use, appropriate guidelines need to be provided.
6. FEATS: It is used for annotating morphological features which can be used from the UD feature set.
7. HEAD: It represents the head of the word denoted by the ID of the head word or 0 in case of the root word.
8. DEPREL: It represents the dependency relations between the current token and the head token.
9. DEPS: It is used for enhanced dependency graph in the form of a list of head-deprel pairs.
10. MISC: It covers other pieces of information like transliteration of the word and lemma, information about space after punctuation marks and so on.

In our case, we follow a hierarchical tag set for POS whose first level is in accordance with UD POS tag set. The leaf tag is represented in the XPOS field in an appropriate format. We provide descriptive guidelines for the same purpose. We also make use of the

MISC field to store the information like NER tags of our proposed hierarchical tag set. Rest of the things are in accordance with UD guidelines and tag sets.

Appendix B

Sample Distribution of Extracted Sources

We discussed about the contribution of different sources in the curated dataset in the Section 2.3. These percentages are a result of the amount of data that extracted from different sources. For sources like movie reviews, only a limited amount of data could be extracted. Table B.1 shows the amount of data extracted from different sources.

Source	No of Paragraphs	Ratio
Wikipdeia Articles	158424	87.31
Ebooks	12173	6.72
Movie Reviews	2920	1.61
News Articles	7722	4.26

Table B.1: Distribution of Extracted Data From Different Sources

Wikipedia was the largest source of data from which around 1.5 lac paragraphs of maximum 4 sentences were extracted. Since we aimed for a diverse dataset, we also scraped data from other sources. The second largest source of data was ebooks from which around 13 k paragraphs were extracted. We identified other ebook sources that can provide additional data in case of need. The third largest contributors were news articles which resulted in around 8 k paragraphs. The last identified source were movie reviews which were taken from a Kaggle repository. This provided with around 3k paragraphs.

We also have a fixed set of simple grammar sentences of 500 count that will be used in the dataset. Since we wanted to represent real life Tamil text instances, we gave similar weightage to all the sources. Sources with abundance data were given slightly more weightage to complete the goal of 10,000 sentences. The decision of the amount of contribution from each source is influenced from a practical standpoint of availability and diversity of data.

Appendix C

Comparitive Study of Existing Tamil Treebanks

Currently, there are two open-source treebanks for Tamil available on the Universal Dependencies website. The first one is the TTB treebank consisting of 600 sentences. The second one is the MWTT treebank consisting of 534 simple grammar sentences. MWTT was formed with the motivation to provide more accurate and consistent annotations than the TTB treebank. General statistics about these treebanks is shown in Table C.1.

Statistics	*TTB*	*MWTTB*
Sentences	600	534
Words	8635	2536
Tokens	9581	2584
Unique Lemmas	2024	445

Table C.1: General Statistics of the two Tamil Treebank

We analyzed the two treebanks to understand the distribution of tags in the present treebanks. The idea was to understand different tags occurring Tamil language by studying their frequency. We also wanted to identify the underrepresented cases so that we can give more examples for them in our treebank. This study is done using the code 'treebank_stat_analysis' present in the "comparitive_study" folder present on Github of UD and producing some statistics from the treebanks too. The counts are normalized as

the counts of tokens vary a lot in the two treebanks.

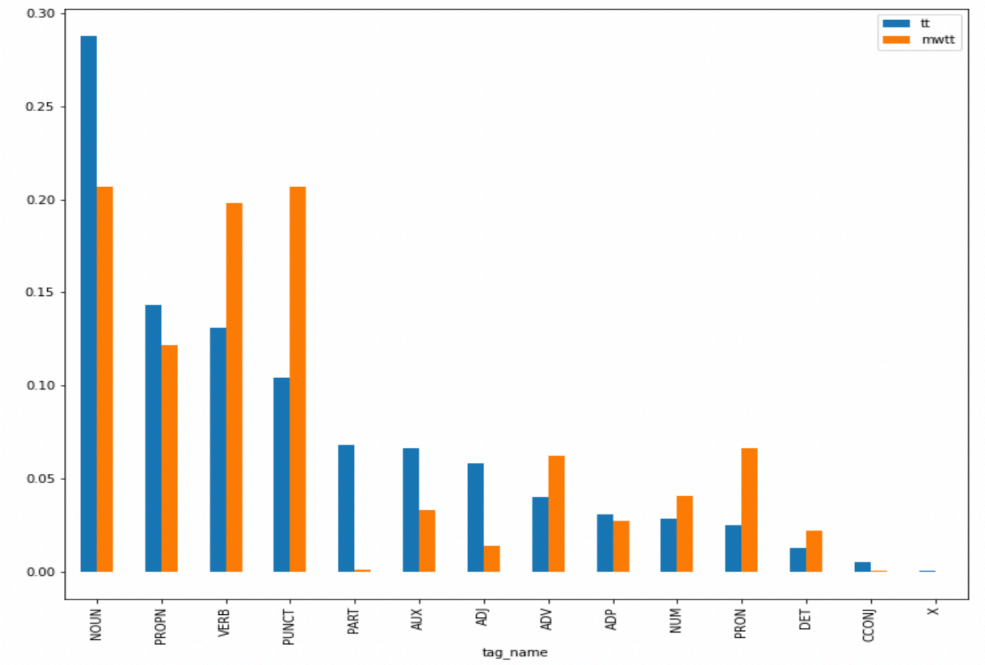


Figure C.1: Distribution of POS tags in the two Tamil treebanks

Fig. C.1 shows the count of different POS tags in the two treebanks. There are a total of 14 different tags appearing in the TTB treebank while MWTT has 13 unique treebanks. The one extra treebank used in TTB is X with count 1. The tag X is used very restrictively, mostly in cases where a real part-of-speech category cannot be assigned for the word. Noun, pronoun, verb, and punctuations are the most occurring tags in both the corpus. Conjunction and determinants are the least occurring tags. Particle, PART is seen to be of low count in MWTT as particle tag words in Tamil are case markers and postpositional suffixes which need not be tokenized. It is represented in the morphological features.

Fig. C.2 shows the count of different morphological features and values in the two treebanks. The distributions show similar trend for many features. Ablative value of feature case, benefactive value of case, feminine value of gender, optative value of mood, converb in verbform are some of the morphological values that appear in MWTT and not in TTB. While digit value of numtype, int value of prontype, gerund of verbform, reflex feature are not present in MWTT but appear in TTB. MWTT treebank provides

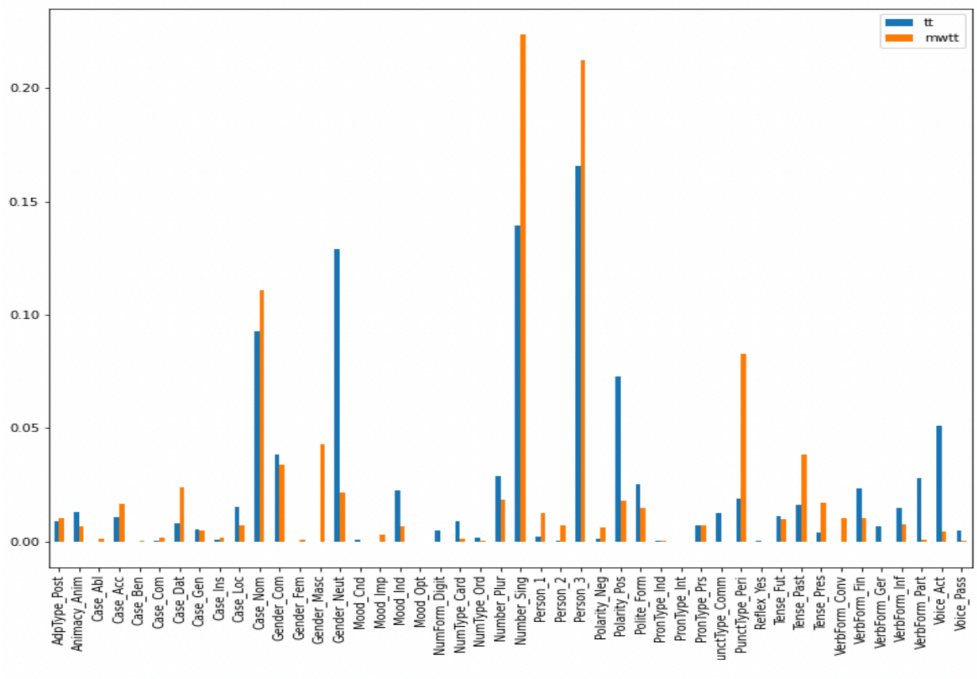


Figure C.2: Distribution of Morphological features in the two Tamil treebanks

more morphological feature diversity. For every morphological feature value, we analyzed the corresponding POS tags appearing. We found more consistency and accuracy of morphological features in the case of MWTT treebank. For example participle value of verbform occurs with ADJ, ADP, ADV, AUX, NOUN, PART, VERB POS tags in TTB while in the case of MWTT, it occurs with only VERB tag.

Fig. C.3 shows the count of different dependency relations in the two treebanks. There are 30 unique dependency relations found in the TTB treebank and 39 unique dependency relations. One of the aims of MWTT treebank creation was to incorporate enhanced dependency relations like nmod:poss, nsubj:nc, nsubj:pass, obl:agent, obl:arg, and so on. Hence it has more dependency relations. Some of the highest occurring dependency relations in the TTB treebank are nmod and obl. [6] observed many inconsistencies in these annotations. Hence in the MWTT tree, these relations are further annotated using enhanced relations which brings more context and consistency to the relations. Some dependency relations like ccomp (used for expressing mostly in a reported speech to express the relation between words), cop, and so on are not represented in the MWTT treebank, often seen in the Tamil language.

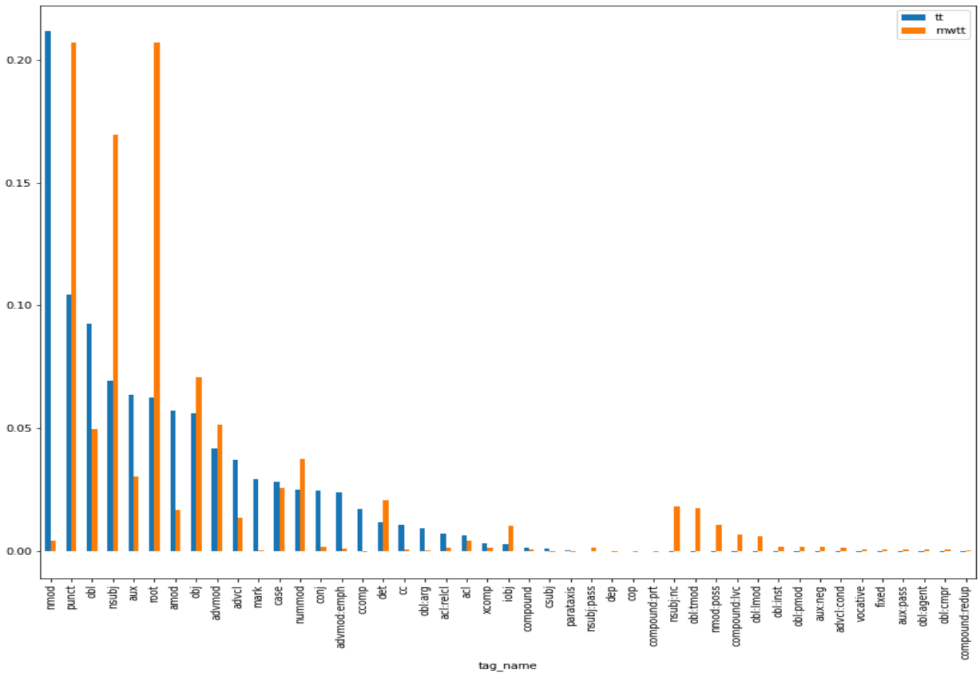


Figure C.3: Distribution of Dependency relations in the two Tamil treebanks

The frequency of many tags in MWTT and TTB is much less than what is required to train better and more accurate models. MWTT treebank is found to be of better consistency and can be used as a starting point for understanding the annotations of UD. But it requires many more annotated samples for better model training. There are cases that require better representation too. The datasets also require diversity to better represent the Tamil language in a real-life context.

Appendix D

Comparitive Study of Tokenization in Tamil

Tokenization is the one of the first step in NLP. There are a lot of open source tools like nltk, textblob, gensim, etc available for tokenization, but most of them are focused on English language, where words are separated by the white spaces or punctuation marks. These situations are also suitable for Tamil along with the identification of multiwords or compound words, which needed to be further tokenized for a better morphogyntactic analysis. In case of Tamil, there are a few tools available online, namely iNLTK and indic library. We analysed their performance on the MWTT treebank for Tamil.

CaseName	*NumberofTokens*
True Tokens	2584
iNLTK tokens	4485
indicNLP tokens	2536

Table D.1: Number of Tokens produced in different cases

There are a total of 2536 words present in the MWTT treebank and the total number of tokens present are 2584. The number of tokens are more than the number of words as some words are identified as multiwords and futher decomposed. Table D.1 displays the number of tokens produced in different cases.

Word	True Tokens	iNLTK Tokens	indicNLP Tokens
கஷ்டப்படுகிறான்	'கஷ்ட' 'படுகிறான்'	'□க' 'ஷ்ட' 'ப்பட' 'ுகிறான்'	'கஷ்டப்படுகிறான்'
இவ்விரண்டு	'*இந்த' 'இரண்டு'	None	'இவ்விரண்டு'
அடுத்தாற்போல	'அடுத்தால்' 'போல'	'□அடுத்த' 'ா' 'ற்' 'போல'	'அடுத்தாற்போல'

Table D.2: Examples of Multi-word Tokenization by Different Tools

Tokenize and trivial_tokenize functions of iNLTK and indicNLP library are used respectively. The following observations are resulted from the analysis -

- The iNLTK library produced a lot more tokens as compared to the other cases. This is because it keeps dividing the word as much as possible by extracting the suffixes and in some case prefix too as seen in Table D.2. Upon further analysis and feedback from the expert, it was found out that the validity of this tokenization is incorrect and unnecessary in many cases. This tool maybe useful for identifying the suffixes, though its accuracy is doubtful and questionable. But it seemed unreliable for performing morphosyntatic analysis using tasks like POS and dependency parsing.
- The indicNLP library produces the same number of tokens as the count of words present in the treebank. It performs trivial tokenization performing split on whitespaces and punctuation marks. It performs no multiword decomposition which is a necessary for further tasks.

We realize the need for a better tokenizer for Tamil. There can be an option for the tokenizer to perform a valid decomposition of the word on the basis of suffixes matching the case of iNLTK. But there should also be the option of performing only trivial tokenizer and multiword decomposition so that other tasks of morphosyntactic analysis can be performed. We acknowledge the fact that multiword decomposition can vary from guideline to guideline. But such a tool can enable the Tamil NLP community to make

larger and more diverse treebanks.