Journal of High School Science

A dependency parser for code-switched Telugu-English

Putta A

Submitted: March 14, 2024, Revised: version 1, April 24, 2024, version 2, April 27, 2024
Accepted: April 27, 2024

Abstract
This study addresses a notable gap in the availability of Code Switched Telugu-English dependency parsers. Conducting research into Code Switched data, especially Telugu-English could be very important, with the prevalence of Code Switching in native Telugu speakers. This is due to the fact that English is widely understood in Telugu-Speaking areas, and since code switching is a prevalent phenomenon in those areas. Through an informal blog dataset, the first Code Switched dependency treebank for Telugu-English was annotated. This treebank is limited to 300 sentences, to serve as an experiment to understand the feasibility of developing more Telugu-English Code Switched technologies. An error analysis was also performed, exploring the common issues faced by dependency parsing models and their possible resolutions. Using the UDify parser, a universal part of speech accuracy of 56.17%, an unlabeled attachment score of 30.61%, and a labeled attachment score of 11.79% was achieved.

Keywords
Code-switching, Code-mixing, Dependencies, Parser, Telugu-English, Annotation, UDify parser, Translation tools, Sentiment analysis, Natural Language Processing

Akshith Putta, Coppell High School, 185 W Parkway Blvd, Coppell, TX 75019, USA.
akshithpt109@gmail.com

Introduction

Code Switching (CS), sometimes called Code Mixing (CM), is the phenomenon in which two or more languages are mixed in written or spoken communication (1-2). CS is frequently observed in multilingual societies, especially in informal settings. Code Switching and Code Mixing are used interchangeably in this paper. CM, which was previously observed most commonly in speech, has become more prevalent with the popularity of social media (3). Code switching can be divided into Intra-Sentential and Inter-Sentential Code-Mixing (4). In this paper, the Intra-Sentential Code Mixed sentences are mostly used, given to mean those sentences which have both Telugu and English words and/or phrases in that sentence. Inter-Sentential Code-Mixing refers to when the code mixing is performed at the end of the sentence. This is observed less frequently in the corpus.

Using the UDify parser's state-of-the-art performance, it was hoped to adapt a dependency parser for a CM Telugu-English corpus. The Universal Dependencies (UD) v2 as the framework was used for the dependency annotations (5). The data was sourced from a blog, provided in (6). This data was used to evaluate model performance and propose solutions. It was expected that for this parser, there would be less accuracy for longer sentences, and challenges in clause selection by the parser. This was observed previously in other papers (7).

In this paper, an experimental annotated corpus of 300 sentences was created (link in appendix), where the matrix language is Telugu, and then the UDify parser was trained on it, in an attempt to create a dependency parser for CM Telugu-English. Creating a dependency parser for Telugu-English Code-Mixed data can significantly enhance translation tools, sentiment analysis, and other Natural Language Processing (NLP) tasks that demand a deep understanding of syntactic relationships. This can, in turn, help create better applications and tools for Code-Mixed Telugu-English text.

Materials and Methods

UD Framework

Universal Dependencies (UD) v2 (5) is the most commonly used system for cross-lingually consistent treebank annotations. This was a development of Universal Dependencies v1 (8). This has been extensively used for NLP applications, especially for the development of the dependency parsers, such as in the CoNLL 2018 shared task (9). UD is uniquely suited for dependency parsing tasks because dependency relations are maintained between words, and the words are not separated into morphemes (5). The syntactic words are kept as is, making it able to be processed by dependency parsers. This view of syntax is considered to be lexicalist (5).

Morphological annotations of a syntactic word refer to the specific labels applied to the word, aiming to categorize its linguistic properties accurately. In the UD system, they consist of a lemma, a part-of-speech tag, and a specific set of features representing lexical and grammatical properties associated with the particular word (5). In the annotation of the Telugu-English CM dataset, only the part-of-speech tags were annotated, since this was the most important morphological annotation (10), and also due to annotator limitations.

Additionally, the lemmatization did not help predictions of the UDify parser, as will be described below. For the feature set of lexical and grammatical properties, it was not possible to effectively develop new properties for CM Telugu-English. Similar to the lemmas, these annotations do not help the predictions of the UDify models. This occurs due to there being such a large and varied set of features so that the training examples are so few that the dependency parser cannot effectively utilize this data in a meaningful way. However, with a large treebank utilizing the UD features and also language specific features, these features could significantly help the predictions of UDify.

Syntactic annotations are the dependency links used between the words to denote a relationship between them (5). For this project, the basic format was used, which forms a dependency tree rooted in one word, which branches out to encompass all the words in the sentence, either as 'branches' or 'leaves'. The enhanced format was not used, but it gives the ability to add additional UD annotations on top of the basic format. The root is most frequently the main clause predicate (5), which was very commonly the case in the annotations. For UD's syntactic analysis, priority is given to predicate-argument and modifier relations between content words; this was done to make cross-lingual grammatical relations more transparent and consistent (5). Additionally, there are dependency relations for specific function words, like determiners (DET), auxiliaries (AUX) and others. There are also other special non-syntactic dependencies; for example: multiword expressions, disfluencies, ellipses, and other special cases. A complete list of all part-of-speech (PoS) tags, features, and syntactic relations can be seen in (5). For storage of UD annotations, the CoNLL-U file format is used, which is a modified form of the widely used CONLL-X file format from (11).

Annotation

The INCEpTION semantic annotation platform was used to make annotations (12). This platform is similar to what has been previously used in other papers. This is the platform that was used for the part-of-speech (PoS) and Dependency Annotations.

The goal had been to create an annotated corpus of 300 sentences to train a dependency parser on. The data was first obtained in the form of a text file and was then converted into a CoNLL-U file, which was the file formatted for INCEpTION and UDify. Most of the effort was focused on shorter and more grammatical sentences to reduce the number of non-trivial choices, with an objective to optimize the corpus for better performance.

When compared to the sentences in the Telugu UD corpus, the CM sentences consisted of more tokens, on average, as was the nature of the dataset provided. It was necessary to make non-trivial choices in the annotations, as is usual for such CM annotation tasks (13), due to the nature of a multilingual dataset. This was due to the corpus being from an informal source, and annotation of informal or spoken language has been more challenging than more standard/written language (14). This was also observed when contrasted with the grammar book examples that were taken for the Telugu Universal Dependencies corpus. The starting guidelines for the annotation work were challenges of annotating a Code-Switching Treebank (14). Since Telugu was the Matrix

Language, Telugu UD Guidelines were used as extensively as possible.

Annotating this Code-Mixed corpus was made easier due to UD being a universal annotation framework. Therefore, a Telugu treebank was used for reference in the annotations (15). Informal sentences, especially those from social media are often ungrammatical and often do not have the perfect UD annotations that encapsulate all of the information. This is because UD does not have adequate guidelines for multilingualism within a treebank. Figures 1 and 2 show examples of sentence(s) incorporating only Telugu words, or a combination of Telugu and English words respectively.

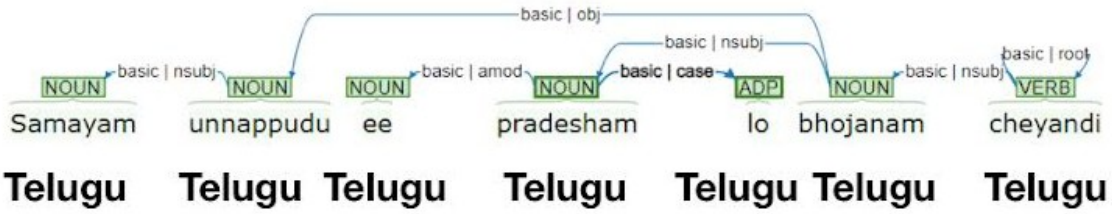

Figure 1. Example of Sentence incorporating only Telugu words.

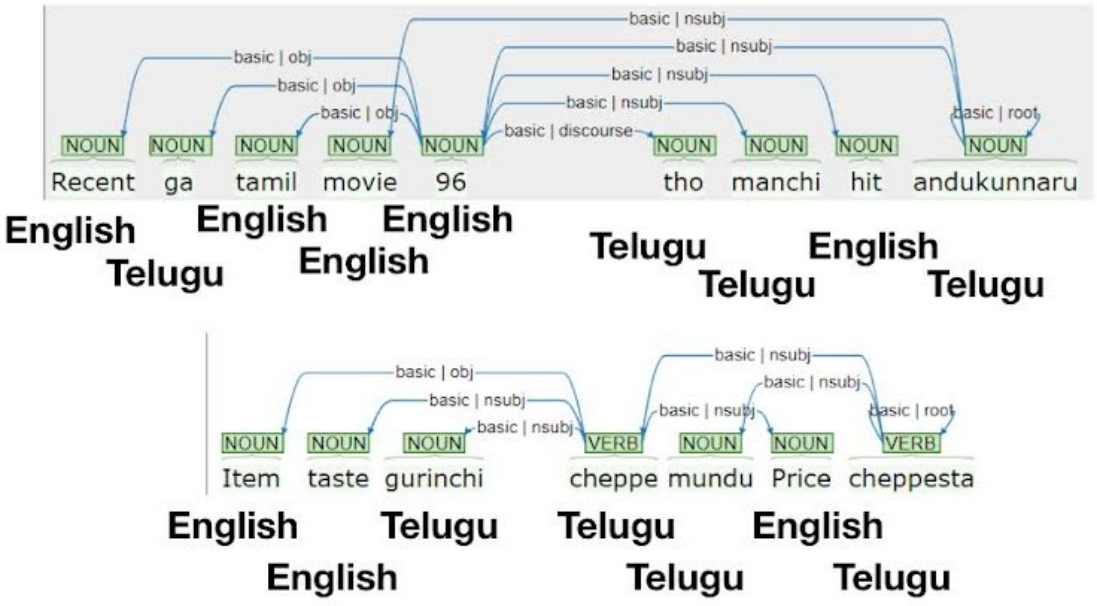

Figure 2. Examples of sentences used in this study incorporating both Telugu and English words.

As noted in (6), some of the most common challenges, in performing Dependency annotation tasks with Code Mixed data involved misspelled words and Romanization variability.

Since this data was obtained from a blog dataset, misspelled words occured quite frequently, presenting a significant challenge in building a spell agnostic dependency parsing model. Currently, there is not much work in building such spell agnostic models which can effectively use CM data. However, UDify has shown promise by being effective for languages that it has not been trained on, such as Faroese, Naija, or Sanskrit. This was observed in (10) when evaluating the UDify parser.

Another challenge for the dependency parsing model was to capture and remove the romanization variability. This variability occurs because Telugu words can be romanized in different ways, which may lead to the model incorrectly creating Part-Of-Speech and Dependency Annotations. Also, this can result in a need for more data, because the model may process the data such that the same Telugu words translate to different words in the Romanized form. This is one of the major reasons why more data could significantly improve the performance of the parser.

UDify

UDify is a multilingual multi-task machine learning model for NLP applications. This model can predict the universal part of speech tags (UPOS), morphological features, lemmas, and dependency trees simultaneously. This model has great promise, since its predictions are very accurate on the UD datasets. This parser is unique in that the pre-trained model does well on datasets it hasn't seen before. This model tries to make a multilingual parser which can effectively predict dependencies even on low-resource languages, with the added benefit of scalability over many languages. This special capability is the main reason why UDify was chosen for this project, since the total annotated dataset only consisted of 300 sentences.

To accomplish this impressive scalability for many languages, UDify uses the Bidirectional Encoder Representations from Transformers (BERT) (16) to take advantage of the large amounts of data provided from the UD datasets. The overall model architecture for Udify can be found in reference 10 and is reproduced in Figure 3. BERT is a language representation model that utilizes a Transformer (self-attention) network to have excellent parsing performance when fine-tuning its contextual embeddings. Within the BERT framework, there are two steps: pre-training and fine-tuning. Pre-training consists of unsupervised learning over unlabeled data, where it performs different pre-training tasks. Subsequently, fine-tuning consists of supervised learning, where the model is initialized with pre-trained parameters, and then these parameters are trained on labeled data from downstream tasks (16). Each of these downstream tasks is a fine-tuned model, although starting with the same pre-trained parameters (16).

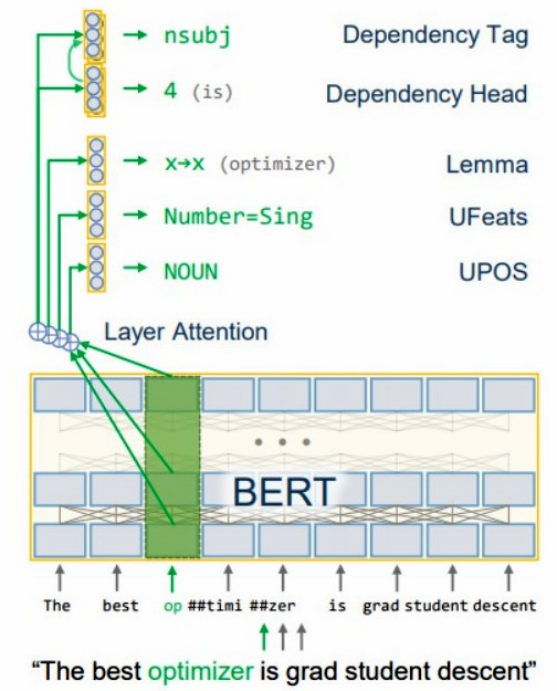

Figure 3. Model architecture for Udify (reproduced from reference 10)

BERT's model architecture is a multi-layer bidirectional Transformer encoder based on the Transformer from (17). A practical example of BERT's model architecture can be found in Figure 4, reproduced from reference 16. The implementation in (16) and in (17) are very similar. Here the working of BERT will be described in detail. The pre-training consists of two unsupervised tasks. The first one is the Masked Language Model (MLM), a deep bi-directional model. To train the MLM, the hidden vectors are input to an output softmax function over the vocabulary. The second pre-training task is Next Sentence Prediction (NSP). This mainly focuses on the relationship between two sentences. Since this is not useful to the context of this study, this will not be described in detail. For UDify, the fine-tuning is performed by training the model on the entire UD Dataset. Prior to this step, BERT is regularized in UDify. This includes input masking - a process which selectively applies masks on text imputs to filter out irrelevant information, increased dropout in neural network layers – which shrinks the squared norm of the weights and reduces overfitting, weight freezing – reducing certain neurons' influence on the decision making process by freezing specific weights in the fully connected layer during the back propagation process, discriminative fine tuning – tuning each layer with different learning rates to improve its accuracy, and layer dropout – a mask that nullifies the contribution of some neurons towards the next layer and reduces overfitting. (10)

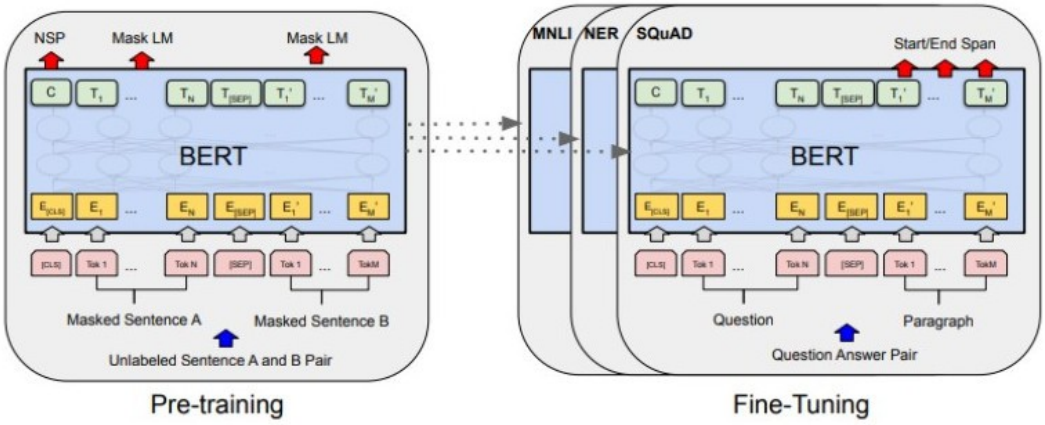

Figure 4. A practical example of BERT's model architecture (reproduced from reference 16)

UDPipe

UDPipe 2.0 (18) was used as a comparison to the UDify developed parser's performance.

It was trained and then used to produce predictions. UDPipe (version 2.0) is a pipeline which can perform sentence segmentation, tokenization, POS tagging, lemmatization and dependency parsing. For the purposes of testing, only the dependency parsing feature was utilized. This was a deep biaffine parser from (19). The process commenced by employing contextualized embeddings generated by bidirectional recurrent neural networks, preceded by the addition of an artificial ROOT word preceding the sentence's start. These embeddings were then nonlinearly transformed into representations of arc-head and arc-dep, which were fused *via* biaffine attention to generate a distribution for each word, indicating the likelihood of all other words being its dependency head. Ultimately, an arborescence, or directed spanning tree, with the highest probability was derived using the Chu-Liu (20)/Edmonds (21) algorithm.

Model Training

To train the model, the corpus was split into the train, test, and validation sets and used this data in the UDify parsing model (10). As a benchmark, the UDPipe parser, which had been state-of-the-art before the introduction of the UDify parser, was selected for use.

Additionally, the treebank/language-specific part-of-speech tags (XPOS) were not utilized because XPOS tags were not used to train the UDify model (10), due to the variability that was present in the various datasets.

The dataset was randomly split along a 70:20:10 ratio for training, testing, and validation. The batch size was changed from 32 to 16 in the UDify system, due to the smaller dataset available. This was manually changed, due to more common hyperparameter tuning

methods (Grid Search, Random Search, and Bayesian Optimization) requiring unreasonably long times to be implemented. Other than this, the other hyperparameters were left at their default values, Changing these other hyperparameters would not have had a significant influence on the model's performance because the change to the treebanks was negligible. Table 1 shows the hyperparameters used in the model.

Table 1. Hyperparameters used in the UDify parsing model

Hyperparameter	Value
Dependency tag dimension	256
Dependency arc dimension	768
Optimizer	Adam
β1, β2	0.9, 0.99
Weight decay	0.01
Label smoothing	0.03
Dropout	0.5
BERT dropout	0.2
Mask probability	0.2
Layer dropout	0.1
Batch size	16
Epochs	80
Base learning rate	0.001
BERT learning rate	5 X 10-5
Learning rate warm-up steps	8000
Gradient clipping	5.0

Results

The 2018 CoNLL Shared Task evaluation script (22) was used to evaluate the results of the Udify model. Using this script, the Universal Part of Speech (UPOS), Labeled Attachment Score (LAS) and Unlabeled Attachment Score (UAS) were obtained (Tables 2 and 3). The UDify model's F1 score was found to be significantly greater than that of UDPipe, which may be due to the UDify model (Augmented with Telugu-English Corpus) being fine-tuned on BERT in addition to using UDPipe. The UDify model was 12% better than the UDPipe model in predicting the UPOS, 15% better in predicting the UAS, and 6% better in predicting the LAS.

In addition, the tuned UDify model performed significantly better than its untuned counterpart across all metrics (Table 3).

Table 2. UDPipe results

Metric	F1 score
UPOS	44.18
UAS	15.38
LAS	6.15

Table 3. Udify results with Telugu-English Corpus

Metric	Tuned model F1 score	Un-tuned model F1 score
UPOS	56.17	45.72
UAS	30.61	17.49
LAS	11.79	8.85

Discussion

Error analysis

An error analysis was performed on the model's results, aiming to diagnose the most frequently occurring issues or errors in the dependency parsing model and propose solutions to address them. The model created dependency links which did not operate in accordance with the UD guidelines. This became apparent through the model's inability to associate the dependency annotations with the correct parts of speech. Incorporating syntactic rules into dependency parsers may improve their functionality for low-resource languages and help resolve semantic role errors. This approach can accelerate the training process and provide substantial assistance given the limited number of sentences available for training in this instance. For example, setting the root annotation to apply for verbs as a predetermined weight in the model, which can be changed during training, would be helpful.

Furthermore, it was challenging to classify idiomatic expressions and compounds, which are very prevalent in this Telugu-English code switched corpus. The parser was inaccurate in processing long sentences, especially in classifying various clauses. Consequently, there were frequent occurrences in which annotations crossed dependency arcs, which violates the acyclicity constraint of trees. This was indicative of the model not accurately capturing dependencies between words that were far apart in the sentence. This may be due to the fact that the UDify model has not been tested extensively on Code-Mixed data. For the specific parts of speech, the model exhibited the largest inaccuracy with the ‘case’ dependency annotation. This is an example of the challenges that can be resolved with the solution proposed in the previous paragraph.

Accuracy

The model’s performance was not as high as was hoped, possibly due to the using mixed

batches on an unbalanced training set, which skewed the model toward predicting larger treebanks more accurately, as was noted in (10). However, smaller treebanks like the Telugu-English dataset were not as easy to predict. This challenge could be resolved with a larger treebank that could be used to fine-tune BERT alongside all Universal Dependencies (UD) treebanks.

Conclusion

With the possibility of developing a preliminary parser for Telugu-English Code Switched data, more sophisticated Natural Language Processing applications for CM Telugu-English can now be envisioned. The first Telugu-English experimental corpus with dependency annotations was used to augment the UDify dependency parser. Adapting a parser to predict dependencies for code-switched languages poses significant challenges due to challenges such as romanization variability and misspelled words. Aiming to address this limitation in the UDify model, future plans involve generating a larger corpus of code-mixed (CM) sentences to train UDify. This expanded training set may lead to significantly improved performance.

References

1. Myers-Scotton C. Social Motivations For Codeswitching: Evidence from Africa. Oxford, New York: Oxford University Press; 1995. 192 p. (Oxford Studies in Language Contact).

2. Poplack S. Code-Switching (Linguistic). In: Smelser NJ, Baltes B, editors. International Encyclopedia of the Social and Behavioral Sciences. 2001. p. 2062–5. https://www.academia.edu/26412938/Code_switching_Linguistic

3. Rijhwani S, Sequiera R, Choudhury M, Bali K, Maddila CS. Estimating code-switching on twitter with a novel generalized word-level language detection technique. In: Proceedings of the 55th annual meeting of the association for computational linguistics (volume 1: long papers) [Internet]. 2017 [cited 2023 Dec 7]. p. 1971–82. https://aclanthology.org/P17-1180/

4. Zirker K. Intrasentential vs. Intersentential Code Switching in Early and Late Bilinguals. Theses and Dissertations [Internet]. 2007 Jun 18; https://scholarsarchive.byu.edu/etd/927

5. Nivre J, de Marneffe MC, Ginter F, Hajič J, Manning CD, Pyysalo S, et al. Universal Dependencies v2: An Evergrowing Multilingual Treebank Collection. In: Calzolari N, Béchet F, Blache P, Choukri K, Cieri C, Declerck T, et al., editors. Proceedings of the Twelfth Language Resources and Evaluation Conference [Internet]. Marseille, France: European Language Resources Association; 2020 [cited 2023 Dec 7]. p. 4034–43. https://aclanthology.org/2020.lrec-1.497

6. Kusampudi SSV, Chaluvadi A, Mamidi R. Corpus Creation and Language Identification in Low-Resource Code-Mixed Telugu-English Text. In: Mitkov R, Angelova G, editors. Proceedings of the International Conference on Recent Advances in Natural Language Processing (RANLP 2021) [Internet]. Held Online: INCOMA Ltd.; 2021 [cited 2023 Dec 7]. p. 744–52. https://aclanthology.org/2021.ranlp-1.85

7. McDonald R, Nivre J. Characterizing the Errors of Data-Driven Dependency Parsing Models. In: Eisner J, editor. Proceedings of the 2007 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning (EMNLP-CoNLL) [Internet]. Prague, Czech Republic: Association for Computational Linguistics; 2007 [cited 2023 Dec 7]. p. 122–31. https://aclanthology.org/D07-1013

8. Nivre J, de Marneffe MC, Ginter F, Goldberg Y, Hajič J, Manning CD, et al. Universal Dependencies v1: A Multilingual Treebank Collection. In: Calzolari N, Choukri K, Declerck T, Goggi S, Grobelnik M, Maegaard B, et al., editors. Proceedings of the Tenth International Conference on Language Resources and Evaluation (LREC'16) [Internet]. Portorož, Slovenia: European Language Resources Association (ELRA); 2016 [cited 2024 Feb 17]. p. 1659–66. https://aclanthology.org/L16-1262

9. Zeman D, Hajič J, Popel M, Potthast M, Straka M, Ginter F, et al. CoNLL 2018 Shared Task: Multilingual Parsing from Raw Text to Universal Dependencies. In: Zeman D, Hajič J, editors. Proceedings of the CoNLL 2018 Shared Task: Multilingual Parsing from Raw Text to Universal Dependencies [Internet]. Brussels, Belgium: Association for Computational Linguistics; 2018 [cited 2024 Feb 17]. p. 1–21. https://aclanthology.org/K18-2001

10. Kondratyuk D, Straka M. 75 Languages, 1 Model: Parsing Universal Dependencies Universally [Internet]. arXiv; 2019 [cited 2023 Dec 18]. http://arxiv.org/abs/1904.02099

11. Buchholz S, Marsi E. CoNLL-X shared task on multilingual dependency parsing. In: Proceedings of the tenth conference on computational natural language learning (CoNLL-X) [Internet]. 2006 [cited 2024 Feb 17]. p. 149–64. https://aclanthology.org/W06-2920.pdf

12. Klie JC, Bugert M, Boullosa B, Eckart de Castilho R, Gurevych I. The INCEpTION Platform: Machine-Assisted and Knowledge-Oriented Interactive Annotation. In: Zhao D, editor. Proceedings of the 27th International Conference on Computational Linguistics: System Demonstrations [Internet]. Santa Fe, New Mexico: Association for Computational Linguistics; 2018 [cited 2023 Dec 18]. p. 5–9. https://aclanthology.org/C18-2002

13. Gerdes K, Kahane S. Dependency Annotation Choices: Assessing Theoretical and Practical Issues of Universal Dependencies. In: Friedrich A, Tomanek K, editors. Proceedings of the 10th Linguistic Annotation Workshop held in conjunction with ACL 2016 (LAW-X 2016) [Internet]. Berlin, Germany: Association for Computational Linguistics; 2016 [cited 2023 Dec 19]. p. 131–40. https://aclanthology.org/W16-1715

14. Çetinoğlu Ö, Çöltekin Ç. Challenges of Annotating a Code-Switching Treebank. In: Candito M, Evang K, Oepen S, Seddah D, editors. Proceedings of the 18th International Workshop on Treebanks and Linguistic Theories (TLT, SyntaxFest 2019) [Internet]. Paris, France: Association for Computational Linguistics; 2019 [cited 2023 Dec 19]. p. 82–90. https://aclanthology.org/W19-7809

15. Rama T, Vajjala S. A Telugu treebank based on a grammar book. In: Hajič J, editor.

Proceedings of the 16th International Workshop on Treebanks and Linguistic Theories [Internet]. Prague, Czech Republic; 2017 [cited 2023 Dec 31]. p. 119–28. https://aclanthology.org/W17-7616

16. Devlin J, Chang MW, Lee K, Toutanova K. BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding [Internet]. arXiv; 2018 [cited 2023 Dec 26]. http://arxiv.org/abs/1810.04805

17. Vaswani A, Shazeer N, Parmar N, Uszkoreit J, Jones L, Gomez AN, et al. Attention is all you need. Advances in neural information processing systems [Internet]. 2017 [cited 2024 Feb 19];30. https://proceedings.neurips.cc/paper/7181-attention-is-all

18. Straka M. UDPipe 2.0 Prototype at CoNLL 2018 UD Shared Task. In: Zeman D, Hajič J, editors. Proceedings of the CoNLL 2018 Shared Task: Multilingual Parsing from Raw Text to Universal Dependencies [Internet]. Brussels, Belgium: Association for Computational Linguistics; 2018 [cited 2023 Dec 26]. p. 197–207. https://aclanthology.org/K18-2020

19. Dozat T, Qi P, Manning CD. Stanford's Graph-based Neural Dependency Parser at the CoNLL 2017 Shared Task. In: Hajič J, Zeman D, editors. Proceedings of the CoNLL 2017 Shared Task: Multilingual Parsing from Raw Text to Universal Dependencies [Internet]. Vancouver, Canada: Association for Computational Linguistics; 2017 [cited 2024 Mar 7]. p. 20–30. https://aclanthology.org/K17-3002

20. Chu YJ. On the shortest arborescence of a directed graph. Scientia Sinica. 1965;14:1396–400.

21. Edmonds J. Optimum branchings. Journal of Research of the national Bureau of Standards B. 1967;71(4):233–40.

22. Hershcovich D, Abend O, Rappoport A. Universal Dependency Parsing with a General Transition-Based DAG Parser. In: Zeman D, Hajič J, editors. Proceedings of the CoNLL 2018 Shared Task: Multilingual Parsing from Raw Text to Universal Dependencies [Internet]. Brussels, Belgium: Association for Computational Linguistics; 2018 [cited 2023 Dec 29]. p. 103–12. https://aclanthology.org/K18-2010

Appendix

Link to Treebank
https://drive.google.com/file/d/1KCZTLBnuoSLiD7afHSFjcmZR7_aTkocV/view?usp=sharing

Universal Dependencies labels used in this treebank

Dependency labels	Description
acl	Clausal modifier of noun (adnominal clause)
advcl	Adverbial clause modifier
advmod	Adverbial modifier
amod	Adjectival modifier
appos	Appositional modifier
case	Case marking
cc	Coordinating conjunction
ccomp	Clausal complement
clf	Classifier
compound	Compound
Compound:lvc	Light verb construction
Compound:redup	Reduplicated compounds
Compound:svc	Serial verb compounds
conj	Conjunct
csubj	Clausal subject
dep	Unspecified dependency
det	determiner
discourse	Discourse element
dislocated	Dislocated elements
fixed	Fixed multiword expression
flat	Flat expression
iobj	Indirect object
mark	Marker
nmod	Nominal modifier
nsubj	Nominal subject
nummod	Numeric modifier
obj	Object
obl	Oblique nominal
parataxis	Parataxis
punct	Punctuation
reparandum	Overridden disfluency
root	Root
xcomp	Open clausal complement