Sandhi Splitting in Tamil and Telugu: A Sequence-to-Sequence Approach Leveraging Transformer Models

Priyanka Dasari[1], **Nagaraju Vuppala**[1], **Mupparapu Sohan Gupta**[1], **Pruthwik Mishra**[2], **Parameswari Krishnamurthy**[1]
[1]IIIT Hyderabad; [2]SVNIT Surat
{dasari.priyanka, nagaraju.vuppala, sohan.mupparapu}@research.iiit.ac.in
pruthwikmishra@aid.svnit.ac.in, param.krishna@iiit.ac.in,

Abstract

Dravidian languages like Tamil and Telugu are agglutinative languages, they form wordforms by combining two or more elements into a single string with morpho-phonemic changes at the point of concatenation, known as *sandhi*. This linguistic feature adds complexity to automatic language processing, making the pre-processing of sandhi words essential for NLP applications. We developed extensive sandhi-annotated corpora of 15K for Telugu and Tamil, focusing on the systematic application of sandhi rules which explains the word formation patterns by showing how lexical and functional categories combine to create composite non-compound words. We implemented compact sequence-to-sequence transformer networks for the automatic sandhi processing. To evaluate our models, we manually annotated Telugu and Tamil IN22-Conv Benchmark datasets (Gala et al., 2023) with sandhi annotations. Our experiments aim to enhance the language processing tasks like machine translation in morphologically rich languages.

1 Introduction

In a text, the identification of individual words is necessary for the computational processing of the text. Due to the high agglutinative nature of Dravidian languages, word identification often becomes complex because of *Sandhi*. *Sandhi*, in linguistics, is a process in which two or more morphemes or word forms unite to form a complex word. It involves the alteration of sounds at word boundaries when two words are combined to form a new word (Uma Maheshwar Rao, 2012). *Sandhi*, as derived from Sanskrit, means 'join together'. It refers to the natural phonetic transformations that occur when two or more words are juxtaposed. These transformations are often guided by a particular language's phonological rules. These phenomena can include merging phonemes (units of sound), omitting phonemes, and adding phonemes to facilitate smooth word transitions.

The task of sandhi splitting becomes complex in agglutinative languages as tokens obtained through tokenization can contain more than one morphological word within them. Shallow parsers (Abney, 2022), which are useful in both text (Collins, 1996), and speech processing domains (Wahlster, 2013) is a task of automatic identification of correlated groups of words or chunks. A shallow parser is not a single module but is a set of modules with a tokenizer, parts-of-speech tagger, and chunker or phrase identifier which are put in a pipeline that can be affected by the sandhi words. Tokenization serves as the initial step before engaging in sandhi splitting. Hence, sandhi splitting must be done as part of the tokenization step, as composite words cause the fusion of tokens. This sequential approach ensures that the text is appropriately structured and analyzed, thus facilitating a deeper understanding of the intricate morphological processes.

We conduct our experiments on sandhi splitting in Tamil and Telugu using the OpenNMT framework, (Open-Source Neural Machine Translation) (Klein et al., 2017) which is a popular open-source toolkit for building and training neural machine translation models. It is a deep learning framework designed specifically for seq2seq modeling tasks, such as machine translation (Bahdanau et al., 2014), text summarization (Nenkova et al., 2011), and speech recognition (Yu and Deng, 2016). We utilized this framework for simulating the task of sandhi splitting as it can be modeled as a Seq2Seq task.

Proceedings of the First Workshop on Challenges in Processing South Asian Languages (CHiPSAL 2025), pages 93–103
January 19, 2025.

This study evaluates the effectiveness of transformer architectures in handling sandhi splitting for Tamil and Telugu texts. We trained our models using the datasets of 15K sandhi annotated sentences per language, which encompasses both splitting and non-splitting sentences to enable the model to discern when to split words. We evaluated our approach using two distinct test sets: a held-out portion of our 15K annotated corpus and the sandhi-annotated IN22-Conv benchmark dataset (Gala et al., 2023). Through this study, we seek to enhance sandhi splitting accuracy in text processing, ultimately contributing to the advancement of NLP applications in morphologically rich languages.

2 Sandhi Splitting in Dravidian languages

Dravidian languages like Tamil and Telugu are agglutinative languages as they form words by attaching various morphemes (meaningful word units) as suffixes to a root or base word (Krishnamurti and Gwynn, 1985). These morphemes can convey grammatical information such as gender, number, person, case markers, tense, aspect, mood, and more. In these languages, nouns can be highly inflected to indicate case and number. And verbs are also richly inflected, with extensive conjugation patterns for tense, aspect, mood, gender, person, and number.

Sandhi can be understood in two ways. Internal sandhi refers to different types of sound changes that occur within words (internally) in the word-formation processes, such as inflection and derivation, whereas external sandhi refers to sound changes that happen between two or more fully-formed word boundaries (externally). In our study compound words are considered as single tokens and can not be split because as they derive new lexical sense on combining. The composite words which are non-compound words in external sandhi do not derive new meanings on combining, these word forms are considered as multiple different tokens and we consider these to be the split points in our model.

1. **Telugu**

   īMṭikoccāḍu = īMṭiki + occāḍu (*He came home*)
   home + come-PST.SG.3.M

2. **Tamil**

   camayttukkoṭuttān̲ = camayttuk + koṭuttān̲ (*Cooked and gave*)
   cook + give-PST.SG.3.M

Further detailed explanations of the various types of sandhi and the rules for sandhi splitting are provided in Appendix **A**, which formed the basis for constructing our datasets.

Sandhi Splitting is the process of splitting a given composite word into its constituent word forms[1]. This explicit study on splitting composite words is essential as it necessitates a deep understanding of morphological distinctions, syntactic variations, and semantic clarity. Composite words are usually formed with multiple word forms with different roots including suffixes and their grammatical features. Along with morphological distinctions, they exhibit various syntactic relationships that affect overall sentence structure and meaning. They may also carry multiple lexical senses and splitting them into constituent parts clarifies their individual contributions to the overall context. These linguistic aspects are interconnected and essential for understanding the complexity of language.

3 Challanges in downstream tasks:

Machine Translation (MT) systems often struggle with accurate translations, particularly when handling morphologically complex languages. One of such challenge arises from sandhi formations - where two distinct words combine with phonological changes at their boundaries. This phenomenon is especially prevalent in Dravidian languages like Telugu and Tamil. Table[1] compares translation outputs for the same sentences under two conditions: before and after applying sandhi splitting.

As shown in Table[1], in Telugu-English translation, the Telugu sandhi word 'māyiṃṭikocci - *to come our home*' is transliterated in translation output as 'Maintikochi'.

[1]the first wordform is termed as W1, the second wordform as W2, and subsequent words are termed W3... Wn.

Language	Sandhi Splitting	Source	Google-Translate
Telugu-English	**Before**	మాయింటికొచ్చి నాలుగు జానపద గేయాలు పాడేసి వెళ్ళవా? 'māyiṃṭikocci nālugu jānapada gēyālu pāḍēsi veḷḷavā? '	Shall we sing four folk songs in Maintikochi?
	After	మా ఇంటికి వచ్చి నాలుగు జానపద గేయాలు పాడేసి వెళ్ళవా? 'mā iṃṭiki vacci nālugu jānapada gēyālu pāḍēsi veḷḷavā? '	*Will you **come to** **my house** and sing four folk songs and leave?*
Telugu-Tamil	**Before**	లక్ష్మీదేవి, చోళరాజుకు కూతురైయ్యింది. 'lakṢmīdēvi, cōḷarājuku kŪturayyiṃdi '	லட்சுமி தேவி சோழராஜாவின் மகள் ஆவார். 'Laxmi devi will become Chola Raja's daughter.'
	After	లక్ష్మీదేవి, చోళరాజుకు కూతురు అయ్యింది 'lakṢmīdēvi, cōḷarājuku kŪturu ayyiṃdi' .	லட்சுமி தேவி சோழராஜாவின் மகள் ஆனார். *'Laxmi devi **became** the daughter of Chola Raja.'*

Table 1: Differences in translation outputs in before and after sandhi splitting

However, after sandhi splitting, 'māyiṃṭikocci' is splitted into 'mā iṃṭiki vacci' gives the accurate translation. Here, the sandhi word combines; pronoun (mā- *our*) + noun (iṃṭiki- *home*) + verb (vacci - *to come*), providing the correct syntactic structure for a meaningful translation.

In Telugu-Tamil translation example, the Telugu word 'kŪturayyyiṃdi - *became daughter*' is translated incorrectly in the future tense as 'makaḷ āvār - *will become daughter*'. After sandhi splitting 'kŪturu ayyiṃdi' produces the correct translation. In this case, the sandhi word combines the noun (kŪturu - *daughter*) with verb (**ayyiṃdi** - *became*), accurately conveying the intended past tense in translation.

4 Related Work

A significant amount of research has focused on sandhi splitting in Indian languages. However, most of this research has concentrated on internal sandhi phenomena. Our study, in contrast, addresses the splitting of external sandhi words where two or more fully formed composite words are combined. Among the traditional approaches, a rule-based sandhi splitter was developed as part of a morphological analyzer and spell checker tool for Telugu (Uma Maheshwar Rao, 2012). However, this method fails to split new words not covered by the predefined rules.

Kuncham et al. (2015) employs a statistical method to perform sandhi splitting in Telugu and Malayalam languages. The testing results indicated an accuracy of 89.07% for Telugu and 90.50% for Malayalam, demonstrating the effectiveness of this approach. This methodology comprised two main components: segmentation and word generation, both of which utilized Conditional Random Fields (CRFs) as a key tool. Devadath et al. (2014) devise a hybrid method that leverages the phonological changes occurring when words are joined together in the context of external sandhi. This approach combines the statistical identification of split points with the application of predefined character-level linguistic rules. As a result, their system currently achieves an accuracy rate of 91.1%. The study by Devadath and Sharma (2016) addresses issues related to the Malayalam Dependency Treebank in the context of external sandhi explaining the challenges posed by external sandhi in the syntactic annotation of Malayalam sentences. Their experiment uses a statistical parser to empirically validate the improvements made to the treebank, stating that even the separation of a single type of external sandhi significantly enhances overall parsing accuracy.

Vempaty and Nagalla (2011) introduce a method that employs finite state automata

to identify possible words within compound words in Telugu. It is built using the syllables of base words, enabling the recognition of candidate words within compound structures. Nair and Peter (2011) design an algorithm aimed at breaking down Malayalam compound words into a series of morphemes by employing multiple levels of finite state automata. Shree et al. (2016) have adopted an approach to the internal sandhi splitting technique in the Kannada language. Gupta and Goyal (2009) conduct Sandhi-Vicheda on Hindi words and assess their software using a dataset of over 200 words through their rule-based algorithm. The studies highlighted employ a variety of approaches, including rule-based methods, statistical techniques, finite state automata, and hybrid approaches combining rules and statistics for tackling the sandhi splitting problem in different Indian languages like Telugu, Malayalam, Kannada, and Hindi.

More recent studies indicate that neural network approaches perform well for sandhi splitting implemented for Sanskrit. The work by Hellwig (2015) is the first in formulating the problem as a neural sequence labeling task, and this was further improved upon by Hellwig and Nehrdich (2018). Hellwig and Nehrdich (2018) introduce end-to-end neural network models that tokenize Sanskrit words by jointly separating compound words and resolving phonetic merge (sandhi) cases. These models do not require hand-crafted features or external linguistic resources, operating solely on parallel data of raw and segmented text.

Additionally, Reddy et al. (2018); Aralikatte et al. (2018) propose Seq2Seq models for this task. Aralikatte et al. (2018) treat word segmentation as a multi-task problem, using a shared encoder with two decoders, where one decoder predicts the split locations and the other generates the characters in the split words. Unlike earlier rule-based or statistical methods, these studies demonstrate the application of neural architectures like sequence labeling and Seq2Seq models to tackle the sandhi splitting problem in an end-to-end fashion directly from data.

5 Need for Sandhi Splitting

This explicit focus on the sandhi split of composite words is essential for several key reasons:

1. **Morphological Distinctions:** Composite words are often formed by joining distinct roots, each with its own grammatical and morphological properties. By splitting these words, the language retains clarity in morphological structure, enabling a precise understanding of the constituent roots and their individual roles within the word.

2. **Syntactic Variations:** In composite words, the constituent word forms can display various syntactic relationships. These relationships help in understanding the overall sentence structure and meaning. Splitting these word forms helps disambiguate syntactic relationships and ensure the sentence retains its intended syntax.

3. **Semantic Clarity:** Composite words may convey multiple, distinct lexical senses when examined as separate components. Splitting them into their constituent parts enables a clearer understanding of the individual lexical meanings they contribute to the overall context. Combining multiple roots and grammatical features often leads to nuanced and context-specific meanings. Sandhi rules help in preserving and revealing these semantic nuances, ensuring that the intended message is conveyed accurately.

6 Implementation

6.1 Data Collection and Annotations

Experiments are conducted on Tamil and Telugu sandhi data using Transformer models at both sentence and character levels to capture morphological patterns effectively. For each language, we developed a dataset comprising 15,000 manually annotated sandhi sentences. Tamil data was sourced from diverse online repositories, including Wikipedia and the Tamil Nadu Tourism website[2], while

[2] https://www.tamilnadutourism.tn.gov.in/tamil

Telugu data was gathered from Wikimedia and publicly available datasets on GitHub[3]. The models were trained and evaluated on both levels to compare their performance and efficacy in handling sandhi variations.

We created two test sets to evaluate model performance on sandhi detection and splitting. The first set was randomly sampled from the 15,000 annotated sentences, while the second set is from the IN22-Conv benchmark dataset (Gala et al., 2023), chosen over FLORES-200(NLLB Team, 2022) for its higher frequency of sandhi splits. In each test set, we formatted the data so that each line in the source side consists of a single sentence, while the target side contains either the single equivalent word or its sandhi form(s), with components separated by a '+' symbol when applicable. The annotations are carried in accordance with sandhi guidelines as mentioned in Appendix **A**. To ensure accuracy, the annotations were thoroughly reviewed and verified by expert linguists proficient in the respective languages. Our curated datasets include both sandhi and non-sandhi sentences, allowing the model to distinguish contexts where splitting is required from those where it is not, promoting a balanced understanding of natural language structure.

Splits of Dataset	No. of sentneces
Train	10000
Valid	3000
Test	2000
IN22-Conv	1503

Table 2: Statistics of Dataset

Splits	Tamil	Telugu
Train and Valid	3200	4002
Testset	853	809
IN22-Conv	145	374

Table 3: Statistics of sandhi split

The dataset is divided into training, validation, and test sets. The statistics for each split are presented in Table[2], while the number of sandhi occurrences within each split is shown in Table[3]. Careful measures have been taken to ensure that the test set remains entirely separate from the training data, eliminating any risk of data leakage and ensuring unbiased predictions.

6.2 Experiments

We employ the Transformer architecture (Vaswani et al., 2017), a state-of-the-art deep learning model with proven success across numerous natural language processing tasks. Transformer's capacity to capture contextual information makes it especially suitable for sandhi splitting. Our model follows the typical encoder-decoder structure: the encoder processes input text, and the decoder predicts sandhi splits upon detecting sandhi words. We implement the Transformer model using the PyTorch-based OpenNMT toolkit (Klein et al., 2018).

The first set of experiments is conducted on sentence-level data, where the source sentences are non-sandhi split sentences, and the target sentences are sandhi-annotated, with sandhi words marked using the '+' symbol. We refer to this as the *Sentence Level Model* (SLM).

In a variation of this experiment, we used *subword-nmt* (Sennrich et al., 2016) to apply byte-pair encoding (BPE) on the training data, creating the *Sentence Level Subword Model* (SLSM). This approach allowed us to evaluate the impact of subword tokenization on the model's performance.

In the character-level experiment, we have arranged the sentence-level data such that every single character is separated by a space. This structure enables the model to process the text at a granular, character-by-character level, which is particularly useful for capturing detailed morphological patterns in sandhi splits. We refer to this setup as the *Character Level Model* (CLM).

In addition to our custom models, we fine-tuned the mt5-small model (Xue, 2020) for the task of sandhi splitting in Tamil and Telugu . mT5 (multilingual T5) is a variant of the T5 (Text-To-Text Transfer Transformer) architecture specifically designed to handle 101 languages, including low-resource ones. Fine-tuning on our annotated dataset refines the model's focus on the sandhi splitting task, improving its ability to identify and split sandhi words accurately. Results are discussed

[3]https://github.com/AnushaMotamarri/Telugu-Books-Dataset?tab=readme-ov-file

Gloss	*Mom, let's go for a movie tomorrow.*		
Input Type	**Sentence Level**	**Sentence Level Subword Model**	**Character Model**
Telugu	అమ్మా, రేపు సినిమాకి వెళ్దాం.	అ@@ మ్మా@@ , రే@@ పు సి@@ ని@@ మా@@ కి వె@@ ళ్@@ దా@@ ం.	అ మ ్మ ా , # ర ేప ు # స ి న ి మ ా క ి # వ ె ళ ్ద ా ం .
Tamil	அம்மா, நாம் நாளைக்கு சினிமா பார்க்கப் போகலாமா?	அ@@ ம்@@ மா@@, ந@@ ாம@@ ்ந@@ ாள@@ ை க்க@@ ூச@@ ினை@@ ிம@@ ாப@@ ார@@ ்க்க@@ப@@ ்போ@@க@@ லாம@@ ா@@?	அ ம ் ம ா , # ந ா ம ் # ந ா ள ை க ் க ூ # ச ி ன ி ம ா # ப ா ர ் க ் க ப ் # ப ோ க ல ா ம ா ?

Table 4: Illustration of the sample input format used for the models.

Testset	**Character**		**Sentence**		**Sentence-subword**		**mT5**	
Metrics	**Tamil**	**Telugu**	**Tamil**	**Telugu**	**Tamil**	**Telugu**	**Tamil**	**Telugu**
Precision	0.8076	0.7832	0.7715	0.7351	0.8124	0.7639	0.8045	0.7568
Recall	0.7384	0.731	0.7423	0.6924	0.7532	0.745	0.7862	0.731
F1 Score	0.7714	0.7562	0.7566	0.7133	0.7818	0.7544	0.8132	0.7437
Accuracy	**0.8205**	0.7485	0.7902	0.7319	0.7841	**0.7832**	0.7917	0.7742

Table 5: Evaluation metrics on Testset

IN22-Conv	**Character**		**Sentence**		**Sentence-subword**		**mT5**	
Metrics	**Tamil**	**Telugu**	**Tamil**	**Telugu**	**Tamil**	**Telugu**	**Tamil**	**Telugu**
Precision	0.7532	0.7413	0.7398	0.6774	0.7856	0.7559	0.7583	0.7553
Recall	0.7421	0.6754	0.7123	0.6342	0.7521	0.7221	0.7245	0.7218
F1 Score	0.7476	0.7075	0.7256	0.6551	0.7685	0.7383	0.7408	0.7384
Accuracy	0.7595	0.6912	0.7451	0.6653	**0.7793**	**0.7625**	0.7632	0.7496

Table 6: Evaluation metrics on IN22-Conv

in next section 6.3. Table [4] illustrates the sample input format sentence taken from the IN22-conv test set used for three models: sentence-level, sentence-level subword, and character-level. For the mT5 model, the standard sentence-level data format is utilized during training.

6.3 Results

Our experiments treat sandhi splitting as a Seq2Seq task, evaluating our models using precision, recall, f1 score, and accuracy. We define true positives as correctly split sandhis and true negatives as accurately identified non-sandhis. False positives count when non-sandhis are incorrectly predicted as sandhis, while false negatives represent instances where fewer sandhi splits are predicted than expected. Our performance evaluation is based on these definitions.

Across all models, Tamil character-level models achieve higher accuracy, with 82% as shown in Table [5]. This is due to the character-level nature of sandhi, where conjugation occurs within the characters of words. On the IN22-Conv test set, sentence-level subword models outperform others, as shown in Table [6] due to their ability to capture broader contextual information and handle diverse sandhi constructions effectively. We also considered a rule-based sandhi splitter for Telugu (Uma Maheshwar Rao, 2012) as a baseline model, which resulted in 65% accuracy on our test set and 59% on the IN22-Conv set. This method struggles to split new words not covered by the predefined rules, limiting its performance compared to the learned models.

Overall, Tamil models achieve higher accuracy than Telugu models, likely because Telugu, being more agglutinative, requires larger datasets to capture its complex linguistic intricacies. Furthermore, the greater number of sandhi splits in Telugu as in

Table [3] increases the challenge. The higher number of true negatives, representing correctly predicted non-sandhi sentences, also contributes to the improved performance of Tamil models.

7 Observations and Limitations

Our findings indicate that the effectiveness of sandhi splitting techniques varies between languages, with Telugu resulting in lower accuracies as it exhibits more sandhi words naturally as shown in Table [3].

In some cases, the model tended to overgenerate or hallucinate outputs. However, it performed well in sandhi splitting for functional category contexts, as detailed in Section B. For example, in Telugu, the model correctly split combinations like అదేమిటంటే 'adēmiṭaṃṭē' (*what it means*) into అది 'adi' (*it*') + ఏమిటి 'ēmiṭi' (*what*) + అంటే 'aṃṭē' (*means*) and మానేద్దమని 'mānēddamani' (*to quit*) into మానేద్దాము 'mānēddāmu' (*quit*) + అని 'ani' (*particle*). Similarly, in Tamil, examples like பிறகெங்கே 'piṟakeṅke' (*then where*) into பிறகு 'piṟaku' (*then*) + எங்கே 'eṅke' (*where*) and இலங்கையாகும் 'ilaṅkayyākum' (Srilanka) into இலங்கை 'ilaṅkay' (*Srilanka*) + ஆகும் 'ākum' (*be-copula*).

Functional categories are typically finite, shorter (mono or bisyllabic), and easier for the model to generalize, especially when paired with longer, non-monosyllabic lexical categories. However, the model faced difficulties with more complex lexical category combinations, such as in Telugu, ప్రోత్సహిస్తారనుకుంటావా? 'prōtsahistāranukuṃṭāvā?' (*Do you think they will encourage?*) should be split as ప్రోత్సహిస్తారు 'prōtsahistāru' (*encourage*) + అనుకుంటావా? 'anukuṃṭāvā' (*do you think*) — and in Tamil, ஆட்டமிழந்து 'āṭṭamiḻantu' (*lost the game*) split as ஆட்டம் 'āṭṭam' (*game*) +இழந்து 'iḻantu' (*lost*). These longer and different lexical category combinations posed greater challenges for the model, making accurate splitting more difficult.

To address this, more high-quality data with richer sandhi splits is needed and due to limited computational resources couldn't fine-tune the model for more epochs. We attempted to train a language model, specifically, Llama-3.1-8B (Touvron et al., 2023), which requires substantial datasets and higher computational power. However, our efforts using the comparatively smaller dataset did not yield higher accuracies, as the outcomes were very low.

8 Conclusion and Future work

This study examined the challenges of sandhi splitting in Dravidian languages, focusing on Tamil and Telugu. Future work will extend this research to other Dravidian languages, such as Kannada and Malayalam, and fine-tune additional large language models (LLMs) with more data to address issues of overgeneration and hallucination. Additionally, by fine-tuning machine translation (MT) systems using sandhi-split annotated datasets, we aim to assess their performance on existing benchmarks. This approach will enhance translation evaluation metrics and contribute to the advancement of MT systems for more accurate translations across Dravidian languages.

References

Steven P Abney. 2022. Principle-based parsing. In *12th Annual Conference. CSS Pod*, pages 1021–1021. Psychology Press.

Rahul Aralikatte, Neelamadhav Gantayat, Naveen Panwar, Anush Sankaran, and Senthil Mani. 2018. Sanskrit sandhi splitting using seq2 (seq)^ 2. *arXiv preprint arXiv:1801.00428*.

Dzmitry Bahdanau, Kyunghyun Cho, and Yoshua Bengio. 2014. Neural machine translation by jointly learning to align and translate. *arXiv preprint arXiv:1409.0473*.

Michael Collins. 1996. A new statistical parser based on bigram lexical dependencies. *arXiv preprint cmp-lg/9605012*.

VV Devadath, Litton J Kurisinkel, Dipti Misra Sharma, and Vasudeva Varma. 2014. A sandhi splitter for malayalam. In *Proceedings of the 11th International Conference on Natural Language Processing*, pages 156–161.

VV Devadath and Dipti Misra Sharma. 2016. Significance of an accurate sandhi-splitter in shallow parsing of dravidian languages. In *Proceedings of the ACL 2016 Student Research Workshop*, pages 37–42.

Jay Gala, Pranjal A Chitale, A K Raghavan, Varun Gumma, Sumanth Doddapaneni, Aswanth Kumar M, Janki Atul Nawale, Anupama Sujatha,

Ratish Puduppully, Vivek Raghavan, Pratyush Kumar, Mitesh M Khapra, Raj Dabre, and Anoop Kunchukuttan. 2023. Indictrans2: Towards high-quality and accessible machine translation models for all 22 scheduled indian languages. *Transactions on Machine Learning Research*.

Priyanka Gupta and Vishal Goyal. 2009. Implementation of rule based algorithm for sandhi-vicheda of compound hindi words. *arXiv preprint arXiv:0909.2379*.

Oliver Hellwig. 2015. Using recurrent neural networks for joint compound splitting and sandhi resolution in sanskrit. In *4th Biennial workshop on less-resourced languages*.

Oliver Hellwig and Sebastian Nehrdich. 2018. Sanskrit word segmentation using character-level recurrent and convolutional neural networks. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 2754–2763, Brussels, Belgium. Association for Computational Linguistics.

Guillaume Klein, Yoon Kim, Yuntian Deng, Vincent Nguyen, Jean Senellart, and Alexander M. Rush. 2018. Opennmt: Neural machine translation toolkit.

Guillaume Klein, Yoon Kim, Yuntian Deng, Jean Senellart, and Alexander M. Rush. 2017. Opennmt: Open-source toolkit for neural machine translation. In *Proc. ACL*.

Bhadriraju Krishnamurti and John Peter Lucius Gwynn. 1985. *A grammar of modern Telugu*. Oxford University Press, USA.

Prathyusha Kuncham, Kovida Nelakuditi, Sneha Nallani, and Radhika Mamidi. 2015. Statistical sandhi splitter for agglutinative languages. In *Computational Linguistics and Intelligent Text Processing: 16th International Conference, CICLing 2015, Cairo, Egypt, April 14-20, 2015, Proceedings, Part I 16*, pages 164–172. Springer.

Latha Ravindran Nair and S. David Peter. 2011. Development of a rule based learning system for splitting compound words in malayalam language. *2011 IEEE Recent Advances in Intelligent Computational Systems*, pages 751–755.

Ani Nenkova, Kathleen McKeown, et al. 2011. Automatic summarization. *Foundations and Trends® in Information Retrieval*, 5(2–3):103–233.

James Cross Onur Çelebi Maha Elbayad Kenneth Heafield Kevin Heffernan Elahe Kalbassi Janice Lam Daniel Licht Jean Maillard Anna Sun Skyler Wang Guillaume Wenzek Al Youngblood Bapi Akula Loic Barrault Gabriel Mejia Gonzalez Prangthip Hansanti John Hoffman Semarley Jarrett Kaushik Ram Sadagopan Dirk Rowe Shannon Spruit Chau Tran Pierre Andrews Necip Fazil Ayan Shruti Bhosale Sergey Edunov Angela Fan Cynthia Gao Vedanuj Goswami Francisco Guzmán Philipp Koehn Alexandre Mourachko Christophe Ropers Safiyyah Saleem Holger Schwenk Jeff Wang NLLB Team, Marta R. Costa-jussà. 2022. No language left behind: Scaling human-centered machine translation.

Vikas Reddy, Amrith Krishna, Vishnu Dutt Sharma, Prateek Gupta, Pawan Goyal, et al. 2018. Building a word segmenter for sanskrit overnight. *arXiv preprint arXiv:1802.06185*.

Rico Sennrich, Barry Haddow, and Alexandra Birch. 2016. Neural machine translation of rare words with subword units. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1715–1725, Berlin, Germany. Association for Computational Linguistics.

M Rajani Shree, Sowmya Lakshmi, and BR Shambhavi. 2016. A novel approach to sandhi splitting at character level for kannada language. In *2016 International Conference on Computation System and Information Technology for Sustainable Solutions (CSITSS)*, pages 17–20. IEEE.

Hugo Touvron, Thibaut Lavril, Gautier Izacard, Xavier Martinet, Marie-Anne Lachaux, Timothée Lacroix, Baptiste Rozière, Naman Goyal, Eric Hambro, Faisal Azhar, et al. 2023. Llama: Open and efficient foundation language models. *arXiv preprint arXiv:2302.13971*.

Amba P. Kulkarni Parameshwari K. Uma Maheshwar Rao, G. 2012. Telugu spell-checker. *Vaagartha, First edition*.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Ł ukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. In *Advances in Neural Information Processing Systems*, volume 30. Curran Associates, Inc.

Phani Chaitanya Vempaty and Satish Chandra Prasad Nagalla. 2011. Automatic sandhi spliting method for telugu, an indian language. *Procedia-Social and Behavioral Sciences*, 27:218–225.

Wolfgang Wahlster. 2013. *Verbmobil: foundations of speech-to-speech translation*. Springer Science & Business Media.

L Xue. 2020. mt5: A massively multilingual pre-trained text-to-text transformer. *arXiv preprint arXiv:2010.11934*.

Dong Yu and Lin Deng. 2016. *Automatic speech recognition*, volume 1. Springer.

A Appendix

A.1 Types of Sandhi

Sandhi is realized in two ways: Internal Sandhi and External Sandhi. Detailed explanations of Internal sandhi and External sandhi are given here:

Internal Sandhi:
Internal Sandhi, also known as *antar* sandhi, refers to phonological changes that occur within a single word, typically due to morphological processes like inflection and derivation.

1. **Inflections:** Inflected words are base words that undergo grammatical changes to convey different meanings, such as verb conjugations, noun plurals, prepositions or postpositions, and case markers are considered inflectional markers. Tables[7] and [8] explain the inflectional suffixes in Telugu and Tamil.

Wordforms	Inflections	Gloss
kattitō	*katti* + *tō* knife + INS	'with the knife'
guḷlō	*guḍi* + *lo* temple + LOC	'in the temple'
ūḷlu	*uru* + *lu* village + PL	'villages'
cēstāḍu	*ces* + *tā* + *ḍu* come+will -FUT+SG.3.M	'he will do'

Table 7: Inflectional Suffixes of Telugu

Wordforms	Inflections	Gloss
pēn̲āvaykkoṇṭu	*pēn̲āvayk+koṇṭu* pen + INS	'with pen'
*valatupuṟ*attil	*valatupuṟam+il* right side + LOC	'towards right side'
paṟkaḷ	*paḷ+kaḷ* tooth + PL	'teeth'
āṭuvāḷ	*āṭu+vā+ḷ* dance +will -FUT+SG.3.F	'she will dance'

Table 8: Inflectional Suffixes of Tamil

2. **Derivations:** Derivation involves creating new words or modifying the meaning of existing words by adding prefixes, suffixes or infixes. These affixes alter the root word's meaning or grammatical category. Tables [9] and [10] explain the derivational suffixes in Telugu and Tamil.

Wordforms	Derivations	Gloss
aṃdamayna	*aṃdam+ ayna* pleasant + ADJV	'beauteous'
vēgaṃgā	*vēgaṃ+ gā* fast + ADV	'fastly'
cēyavaddu	*cēyu* + *vaddu* do + not-AUX	'do not do'

Table 9: Derivational Suffixes in Telugu

Wordforms	Derivations	Gloss
aṟivān̲a	*aṟivu+ān̲a* Intelligent + ADJ	'intelligent'
nērāka	*nēr+āka* Straight + ADV	'straightly'
pārkkakkūṭātu	*pārkkak* + *kūṭātu* see + not-AUX	'do not see'

Table 10: Derivational Suffixes in Tamil

External Sandhi:
External sandhi, also known as *bahya* sandhi, involves phonological changes that occur at the boundaries of words when they come into contact, either due to word combination or sentence formation as a result of stylistic variation. External sandhi can be divided into two types.

1. **Compound Words:** Compound words are formed by combining two or more complete words to create a new word with a distinct meaning.
   Examples in Telugu:
   (a) *rāmālayaM* 'Rama's temple' = *rāmā* + *ālayaM*
   (b) *paramārdhaM* 'great meaning' = *parama* + *ardhaM*

   Examples in Tamil:
   (a) *matiyavēḷay* ''afternoon'' = *matiya* + *veḷay*
   (b) *tuvaramparuppu* 'toor dal' = *tuvaram* + *paruppu*

2. **Composite words or Non-compound Words:** Composite words, also known as Non-compound words, contain two or

more complete words within them but do not derive new or distinct meanings from this combination. Since these composite words do not derive new meanings on combining, these word forms are considered as two different tokens and hence to be split.

Examples in Telugu

(a) *īMṭikoccāḍu = īMṭiki + occāḍu*
home + come-PST.SG.3.M

(b) *vaccānannāḍu = vaccānu + annāḍu*
come-PST.SG.3 + say-PST.SG.3.M

(c) *atanoccāḍēmō = atanu + occāḍu + ēmō*
'he + come-PST.SG.3.M + emo-ADV

Examples in Tamil

(a) *camayttukkoṭuttāṉ = camayttu + koṭuttāṉ*
cook + give-PST.SG.3.M

(b) *koṭumayceyvāḷ = koṭumay + ceyvāḷ*
torture + do-FUT.SG.3.F

(c) *virayvākappo = virayvāka + pō*
fast -ADV +go

B Contexts for Sandhi Splitting

Sandhi rules allow us to unravel the complexities of composite words, leading to a deeper comprehension of their structure, syntax, and meaning in language analysis and interpretation. These rules for sandhi are made based on the distinction between the lexical and functional categories. lexical categories such as nouns (N), verbs (V), pronouns (PR), adjectives (JJ), and number words, can take inflectional and derivational suffixes and often carry the core meaning in a sentence. Functional categories in Telugu, including quantifiers (QT), particles (RP), quotatives (UT), intensifiers (INTF), negations (NEG), etc., are often considered as closed-class words, and they are more stable in their form and do not readily take inflections or any derivational suffixes. We identify four major contexts in which word forms are conjoined that need to be split. The contexts are:

1. **Lexical Categories + Lexical Categories:**
This combination involves sandhi between two lexical categories of words. Examples are given in Tables [11] and [12].

W1+W2	W1	W2
dēvuḍiccina	*dēvuḍu*(N) 'god'	*iccina*(V) 'given'
mēmanukōvaccu	*mēmu*(PR) 'we'	*anukōvaccu*(V) 'thought'
vaccāḍokasāri	*vaccāḍu*(V) 'come-PST.M.SG'	*okasāri*(N) 'once'
aydaMtasthula	*aydu*(N) 'five'	*aMtasthula*(N) 'floors'

Table 11: Lexical Categories (W1) + Lexical Categories (W2) in Telugu

W1+W2	W1	W2
aḷavilirukkum	*aḷavil*(N) 'in amount'	*irukkum*(N) 'be-FUT.3.NEU'
atikamāṉavay	*atikam*(N) ''More''	*āṉavay*(V) 'become-PST.PL.3.NEU'
aticayamākum	*aticayam*(N) 'miracle'	*ākum* (V) 'is'
ataṟkuḷḷākave	*ataṟku*(N) 'for that'	*uḷḷākave*(ADV) ''within''
ārampakālaṅkaḷ	*ārampam*(ADJ) ''begin''	*kālaṅkaḷ*(N) ''seasons''

Table 12: Lexical Categories (W1) + Lexical Categories (W2) in Tamil

2. **Functional Categories + Functional Categories:**
This combination involves sandhi between two functional categories of words. Examples are given in Tables [13] and [14].

W1+W2	W1	W2
lēḍakkaḍa	*lēḍu* 'not'	*akkaḍa* 'there'
ippuḍakkaḍa	*ippuḍu* 'now'	*akkaḍa* 'there'
appuḍaMdarikī	*appuḍu* 'then'	*aMdarikī* 'all'
ekkaḍikaMṭe	*ekkaḍiki* 'where'	*aMṭe* 'means'

Table 13: Functional Categories (W1) + Functional Categories (W2) in Telugu

W1+W2	W1	W2
avvāṟillay	*avvāṟu* 'like that'	*illay* 'not'
piṟakaṅke	*piṟaku* 'then'	*aṅke* 'there'
vēṟellām	*vēṟu* 'something else'	*ellām* 'all'
eppaṭiyeṉṟāl	*eppaṭi* 'how'	*eṉṟāl* 'means'

Table 14: Functional Categories (W1) + Functional Categories (W2) in Tamil

3. **Lexical Categories + Functional Categories:**
This combination is the interaction between lexical categories and functional categories. Lexical categories provide the core meaning, while functional categories modify the sense of the sentence being in

the W2 position. Examples are given in Tables [15] and [16].

W2 Type	W1+W2	W1	W2
Concessive	*koMḍalaynā*	*koMḍalu* (N) 'mountains'	*ayinā* ' even though'
Conditional	*pillalayite*	*pillalu* (N) 'children'	*ayite* 'if'
Quantifiers	*abhivr̥ddhaMtā*	*abhivr̥ddhi* (N) 'development'	*aMtā/aMta* 'all (that much)'
Interogatives	*āhārālēmī*	*āhārālu*(N) 'food'	*ēmī* 'What'
Distals	*vāḷlakkaḍa*	*vāḷlu* (PR) 'they'	*akkaḍa* 'place'
Proximals	*pillalītlA*	*pillalu* (N) 'children'	*iṭlā* 'manner'

Table 15: Lexical Categories (W1) + Functional Category (W2) in Telugu

W2 Type	W1+W2	W1	W2
Concessive	*taṭaṅkalirupp̄iṇum*	*taṭaṇkal* (N) 'obstacles'	*iruppiṇum* 'in spite of'
Conditional	*avaṇiruntāl*	*avaṇ* (PRON) 'he'	*iruntāl* 'if'
Quantifiers	*kavalayyaṇayttum*	*kavalay* (N) 'worries'	*aṇayttum* 'all (that much)'
Interogatives	*paṭuvateṅkē?*	*pāṭuvatu*(N) 'to sing'	*eṅkē?* 'Where'
Distals	*avaṇaṅku*	*avaṇ* (PR) 'he'	*aṅku* 'there'
Proximals	*ivaṇiṅku*	*ivaṇ* (PR) 'he'	*iṅku* 'here'

Table 16: Lexical Categories (W1) + Functional Category (W2) in Tamil

4. **Functional Categories + Lexical Categories:**
In this combination, functional categories are supplementary to Lexical categories, often modifying or specifying the meaning of the lexical category word in the W2 position. Examples are given in Tables [17] and [18].

W1 Type	W1+W2	W1	W2
Quantifiers	*aṃḍarunna*	*aṃḍaru* 'that many /so many (people)'	*unna* (V) 'to be'
Interogatives	*ēmiṭammāyi*	*ēmiṭi* 'What'	*ammāyi* (N) 'girl'
Distals	*akkaḍokaru*	*akkaḍa* 'place'	*okaru* (N) 'one person'
Proximals	*ikkaḍunnavāḷlu*	*ikkaḍa* 'place'	*unnavāḷlu* (N) 'people'

Table 17: Functional Categories + Lexical Categories in Telugu

W1 Type	W1+W2	W1	W2
Quantifiers	*palapēr*	*pala* 'many'	*pēr* 'members'
Interogatives	*ekkāriyam?*	*enta* 'Which'	*kāriyam?* 'matter'
Distals	*aṅkuḷḷōr*	*aṅku* 'there'	*uḷḷōr* 'people-be'
Proximals	*iṅkuḷḷōr*	*iṅku* 'here'	*uḷḷōr* 'people-be'

Table 18: Functional Categories + Lexical Categories in Tamil