A Sandhi Splitter for Malayalam

Devadath V V **Litton J Kurisinkel** **Dipti Misra Sharma** **Vasudeva Varma**
Language Technology Research Centre
International Institute of Information Technology - Hyderabad, India.
{devadathv.v,litton.jkurisinkel}@research.iiit.ac.in,{dipti,vasu}@iiit.ac.in

Abstract

Sandhi splitting is the primary task for computational processing of text in Sanskrit and Dravidian languages. In these languages, words can join together with morpho-phonemic changes at the point of joining. This phenomenon is known as Sandhi. Sandhi splitter splits the string of conjoined words into individual words. Accurate execution of sandhi splitting is crucial for text processing tasks such as POS tagging, topic modelling and document indexing. We have tried different approaches to address the challenges of sandhi splitting in Malayalam, and finally, we have thought of exploiting the phonological changes that take place in the words while joining. This resulted in a hybrid method which statistically identifies the split points and splits using predefined character level linguistic rules. Currently, our system gives an accuracy of 91.1% .

1 Introduction

Malayalam is one among the four main Dravidian languages and 22 official languages of India. It is spoken in the State of Kerala, which is situated in the south west coast of India . This language is believed to be originated from old Tamil, having a strong influence of Sanskrit in its vocabulary. Malayalam is an inflectionally rich and agglutinative language like any other Dravidian language. The property of agglutination eventually leads to the process of sandhi.

Sandhi is the process of joining two words or characters, where morphophonemic changes occur at the point of joining. The presence of Sandhi is abundant in Sanskrit and all Dravidian languages. When compared to other Dravidian languages, the presence of Sandhi is relatively high in Malayalam. Even a full sentence may exist as a single string due to the process of Sandhi. For example, *അവനാരാണ്*(avanaaraaN) is a sentence in Malayalam which means *"Who is he ?"*. It is composed of 3 independent words, namely അവൻ(avan (*h*e)), ആര്(aar(*w*ho)) and ആണ് (aaN(*i*s)). However, ambiguous splits for a word is very less in Malayalam. Sandhis are of two types, Internal and External. *Internal Sandhi* exists between a root or a stem with a suffix or a morpheme. In the example given below,

പറ$(para)$ + ഉന്നു$(unnu)$ = പറയുന്നു$(para\mathbf{yu}nnu)$

Here പറ(para) is a verb root with the meaning "to say" and ഉന്നു(unnu) is an inflectional suffix for marking present tense. They join together to form പറയുന്നു(para**yu**nnu), meaning "say"(PRES). *External sandhi* is between words. Two or more words join to form a single string of conjoined words.

ചെയ്യും$(ceyyu\mathbf{M})$ + എങ്കിൽ$(enkil)$ = ചെയ്യുമെങ്കിൽ (ceyyu**me**nkil)

ചെയ്യും(ceyyu**M**) is a finite verb with the meaning "will do" and എങ്കിൽ(enkil) is a connective with meaning "if". They join together to form a single string ചെയ്യുമെങ്കിൽ(ceyyu**me**nkil).

For most of the text processing tasks such as POS tagging, topic modelling and document indexing, External Sandhi is a matter of concern. All these tasks require individual words in the text to

D S Sharma, R Sangal and J D Pawar. Proc. of the 11th Intl. Conference on Natural Language Processing, pages 156–161,
Goa, India. December 2014.

be identified. The identification of words becomes complex when words are joined to form a single string with morpho-phonemic changes at the point of joining. More over, the sandhi can happen between any linguistic classes like, a noun and a verb, or a verb and a connective etc. This leads to misidentification of classes of words by POS tagger which eventually affects parsing. Sandhi acts as a bottle-neck for all term distribution based approaches for any NLP and IR task.

Sandhi splitting is the process of splitting a string of conjoined words into a sequence of individual words, where each word in the sequence has the capacity to stand alone as a single word. To be precise, Sandhi splitting facilitates the task of individual word identification within such a string of conjoined words.

Sandhi splitting is a different type of word segmentation problem. Languages like Chinese (Badino, 2004), Vietnamese (Dinh et al., 2008) Japanese, do not mark the word boundaries explicitly. Highly agglutinative language like Turkish(Oflazer et al., 1994) also need word segmentation for text processing. In these languages, words are just concatenated without any kind of morpho-phonemic change at the point of joining, whereas morpho-phonemic changes occur in Sanskrit and Dravidian languages at the point of joining (Mittal, 2010).

2 Related works

Mittal (2010) adopted a method for Sandhi splitting in Sanskrit using the concept of *optimality theory*(Prince and Smolensky, 2008), in such a way that it generates all possible splits and validates each split using a morph analyser. In another work, statistical methods like Gibbs Sampling and Dirichlet Process are adopted for Sanskrit Sandhi splitting (Natarajan and Charniak, 2011).

To the best of our knowledge, only two related works are reported for the task of Sandhi splitting in Malayalam. Rule based compound word splitter for Malayalam (Nair and Peter, 2011), goes in the direction of identifying morphemes using rules and trie data structure. They adopted a general approach to split both external and internal sandhi which includes compound words. But splitting a compound word which is conceptually united will lead to the loss of linguistic meaning. In an other work (Das et al., 2012), which is a hybrid approach for sandhi splitting in Malayalam, employs a TnT tagger to tag whether the input string to be split or not and splits according to a predefined set of rules. But this particular work does not report any empirical results. In our approach, we identify precisely the point to be split in the string using statistical methods and then applies predefined set of character level rules to split the string.

3 Our Approach

We have tried various approaches for sandhi splitting and finally arrived at a hybrid approach which decides the split points statistically and then splits the string in to words by applying pre-defined character level sandhi rules.

Sandhi rules for external sandhi, are identified from a corpus of 400 sentences from Malayalam literature, which includes text from old literature(texts before 1990) as well as modern literature. In comparison with text from old literatures, modern literature have very less use of sandhi. By analysing the text, we were able to identify 5 unambiguous character level rules which can be used to split the word, once the split point is statistically identified. Sandhi rules identified are listed in the **Table 1**.

R	Rule	Example
1	$(CSC)V_s = (CSC)S + V$	വാക്കില്ല = വാക്ക് + ഇല്ല word(Noun)+no(verb)
2	$(യ/വ)V_s = V$	പേടിയാണ് = പേടി + ആണ് fear(Noun) + is(verb)
3	$മV_s = ം + V$	പണമില്ല = പണം + ഇല്ല money(Noun) + no(verb)
4	$(ര/ല/ള/ന/ണ)V_s = (ർ/ൽ/ൾ/ൻ/ൺ) + V$	മുകളിലാണ് = മുകളിൽ + ആണ് above(Loc)+is(verb)
5	$CV_s = (CS) + V$	ആണെന്ന് = ആണ് + എന്ന് is(verb)+that(quotative)
6	just split	അവൻവന്നു = അവൻ + വന്നു He(Noun)+Came(verb)

C=Consonant, V=Vowel, V_s=Vowel symbol, S=Schwa

Table 1: Sandhi rules

Rules in **Table 1** are given in a particular order corresponding to their inherent priority which avoids any clash between them. Every character level Sandhi rule is based on a consonant and a vowel. *Rule* **5** is the most general rule for Sandhi that we could identify in Malayalam which states that a group of a consonant and a vowel can be

split into a consonant and a vowel. *Rule* **1** is a special case of *Rule* **5**, which is framed in order to treat the special case of compound of consonants with a schwa in between. This rule is being introduced , because the consonants specified in rules **2, 3** *and* **4** can come as the last consonant in the case of a compound of consonants. This will lead to an improper split of the string by any of the rules **2, 3** *or* **4**. So every word, with an identified split point should be primarily checked against rule 1 to avoid wrong split by other rules in the case of a sandhi containing a compound of consonants. *Rule* **2** is to handle the process of phonemic insertion came after the process of Sandhi. In Malayalam, these extra characters ാ/ു can be inserted between words after sandhi in certain context. *Rule* **3** and *Rule* **4** are to handle the case of phonemic variation caused due to the process of Sandhi. *Rule* **3** enforces that, the letter 'മ' with a vowel symbol becomes 'ം' and a vowel, while *Rule* **4** enforces that, the characters, 'ര/ല/ള/ന/ണ' with a vowel symbol becomes *chillu*[1], 'ര്/ല്/ള്/ന്/ണ്' and a vowel.

The hybrid approach utilises the phonological changes (Klein et al., 2003) due to the presence of sandhi. The remaining part of the paper explains this approach in detail, with theoretical formulation, Experimental Results and Error analysis.

When the words are conjoined, they undergo phonological changes at the point of joining. These phonological changes can be evidential in identifying the split point in the given string of conjoined words. For example, words Wx and yZ are conjoined to form a new string $Wx^{'}y^{'}Z$. As a part of this process, substring x in the original string Wx has undergone a phonological change to become $x^{'}$ and the substring y in the original string yZ has undergone a phonological change to become $y^{'}$. We try to identify the split point between $x^{'}$ and $y^{'}$ in $Wx^{'}y^{'}Z$ using $x^{'}$ and $y^{'}$ as the evidence. Going forward, we use "S_p" to denote *S*plit Point and "N_{sp}" to denote Non Split Point. $P(S_p|x^{'}, y^{'})$ will give the probability of a character point between $x^{'}$ and $y^{'}$ within a conjoined string to be a split point. A character point will be classified as split point if,

$$P(S_p|x^{'}, y^{'}) > P(N_{sp}|x^{'}, y^{'}) \quad (1)$$

[1]A *chillu* is a pure consonant which can stand alone independently without the help of vowels

As per our observation, the phonological changes of x to $x^{'}$ and of y to $y^{'}$ are independent of each other. So,

$$P(S_p|x^{'}, y^{'}) = P(S_p|x^{'}) * P(S_p|y^{'}) \quad (2)$$

To produce the values of $x^{'}$ and $y^{'}$ for each character point, we take k character points backwards and k character points forward from that particular point. For example, the probability of the marked point $PointX$ in the string given in the **Figure 1** below to be a split point is given by **equation (3)**

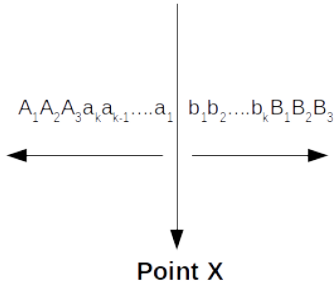

Figure 1: Split point

$$P(S_p|a_1a_2...a_k) * P(S_p|b_1b_2...b_k) \quad (3)$$

Where $a_1a_2...a_k$ is $x^{'}$ and $b_1b_2...b_k$ is $y^{'}$. The value of k is experimentally optimized.

The split points of different agglutinated strings in training data are annotated in the following format.

$$WordN = i_1, i_2, i_3, ...i_z$$

This indicates that the string $WordN$ needs to be splitted at "z" character indexes $i_1, i_2, i_3, ...i_z$ within the string.

We employ two Orthographic Tries (Whitelaw and Patrick, 2003) to statistically capture the phonological differences in $x^{'}$ and $y^{'}$ for split points and non-split points. A mould of orthographic tries used is given in **Figure 2**. The trie in **Figure 2** is trained with character sequences $c_1c_2c_3$, $c_1c_2c_4$, $c_1c_3c_4$, $c_1c_3c_5$. In the first trie, the path from root to k nodes represents the string $a_1a_2...a_k$. Each node $i(1 \leq i \leq k)$ in the path stores the number of occurrences of split points and non-split points in the entire training data which are preceded by $a_ia_{i-1}...a_1$. In the second trie, a path from root to k nodes represent

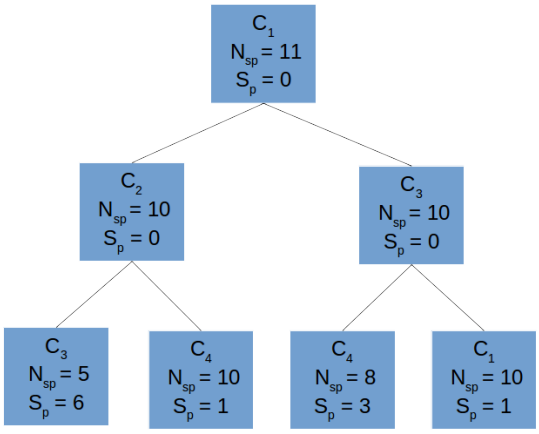

Figure 2: Word trie

the string $b_1b_2...b_k$. Each node $i(1 \leq i \leq k)$ in the path stores the number of occurrences of split-points and non-split points which are succeeded by $b_1b_2...b_i$. The frequencies stored in these trie nodes are used to calculate $P(C|a_1a_2...a_k)$ and $P(C|b_1b_2...b_k)$ where C is S_p or N_{sp}.

As split points are rare within a string $P(C|a_1a_2...a_k)$ and $P(C|b_1b_2...b_k)$ needs to be smoothed. For this purpose, we use the information stored along the depth of the tries for the strings $a_1a_2...a_k$ and $b_1b_2...b_k$ as follows

$$P(C|a_1a_2...a_k) = \sum_{i=initial\ skip}^{k} P(C|a_ia_{i+1}...a_k) \quad (4)$$

$$P(C|b_1b_2...b_k) = \sum_{i=initial\ skip}^{k} P(C|b_ib_{i+1}...b_k) \quad (5)$$

Here the initial-skip decides optimum 'smoothing range' within the string.The value of initial-skip decides the threshold phonological similarity that needs to be considered while smoothing. The experimental optimisation of initial-skip is done. The identified split points are splitted using the applicable sandhi rules.

4 Data and Results

Data Set

We created a dataset which contained 2000 Sandhi bound words for training. Each of the split point within the word are annotated with the within-word index of the corresponding character point. The test data contains 1000 random words, out of which 260 are words with sandhi.

Results

In our experiments, we tried evaluating the accuracy of split point identification, split accuracy of different sandhi rules and overall accuracy of the system. By overall accuracy, we mean percentage of the words with sandhi in which the split is exactly as expected. We have conducted experiments on split point identification with different values of k and initial Skip.

k-S	P	R	F	Accuracy
3-1	85.37	75	79.85	89.6
3-2	84.73	73.26	78.58	88.7
4-1	86.56	76.04	80.96	90.1
4-2	85.48	73.61	79.10	89
5-1	88.37	79.16	83.51	90.9
5-2	85.94	74.30	79.30	88.7
6-1	88.50	80.20	84.15	91.1
6-2	85.59	76.38	80.88	89.2

Table 2: Results

Here *P* implies precision, *R* implies Recall, *F* implies F-measure and *k-S* implies k and Initial skip respectively. k=6 and initial Skip=1 have shown the better result. As per our observation, phonological changes as a part of sandhi would not happen beyond a range of six characters in each of the participating words. So the upper bound for k value is taken as 6.

5 Error Analysis

In Split point identification, most of the incorrectly identified split points are character points between a word and inflectional suffix attached to it. As the system evolve, this error can be rectified by the use of a post-processor which maintains the finite list of inflectional suffixes in the language. Wrong splits in the middle of an actual word are very few in number and will reduce as the size of the training data increases.

Most of the unidentified split points are due to the presence of certain rare patterns. These errors will be reduced with the incorporation of words from diverse texts in the training data.

When it comes to rules, we have used only character level rules for splitting the identified split points. Rule 1,4 and 5 go ambiguous at certain rare contexts. To resolve this, certain word level information like POS tags are required. But for an accurate POS tagging, particularly for this disambiguation purpose, a sandhi splitter with a good level of accuracy is inevitable. Our Sandhi splitter can contribute for a better POS tagger. Vice versa, a POS tagger can complement the Sandhi splitter.

Rule	Accuracy
1	92.10
2	96.29
3	100
4	80.64
5	87.71

Table 3: Rule accuracy when k=6 and S= 1

6 Conclusion & Future work

This experiment has given us a new insight that, there exists a character level pattern in the text where the sandhi can be splitted. When compared to a completely rule based approach which is effort intensive, a hybrid approach gives us a more fast and accurate performance with relatively lesser amount of training data. We expect that, this method can be successfully implemented in all other Dravidian languages for the Sandhi splitting. The only language dependent part will be to split the identified split points using language specific character level rules. The system component to split the identified split points can be decided at runtime based on the language on which it is operating on.

There is a possibility of improving the current system into a language independent and fully statistical system. Instead of totally depending on language specific Sandhi rules for splitting, phonological changes after split can be inferred from training data. For Example,

$$Wx^{'}y^{'}Z \Rightarrow Wx + yZ$$

$P(x|x^{'})$ and $P(y|y^{'})$ can be tabulated to predict the phonological changes after split. But an accurate tabulation demands a large amount of annotated training data.

Software

The source code of our Hybrid Sandhi splitter for Malayalam is available at `https://github.com/Devadath/Malayalam_Sandhi_Splitter`

References

[Badino2004] Leonardo Badino. 2004. Chinese text word-segmentation considering semantic links among sentences. In *INTERSPEECH*.

[Das et al.2012] Divya Das, Radhika K T, Rajeev R R, and Raghu Raj. 2012. Hybrid sandhi-splitter for malayalam using unicode. In *In proceedings of National Seminar on Relevance of Malayalam in Information Technology*.

[Dinh et al.2008] Quang Thang Dinh, Hong Phuong Le, Thi Minh Huyen Nguyen, Cam Tu Nguyen, Mathias Rossignol, Xuan Luong Vu, et al. 2008. Word segmentation of vietnamese texts: a comparison of approaches. In *6th international conference on Language Resources and Evaluation-LREC 2008*.

[Klein et al.2003] Dan Klein, Joseph Smarr, Huy Nguyen, and Christopher D Manning. 2003. Named entity recognition with character-level models. In *Proceedings of the seventh conference on Natural language learning at HLT-NAACL 2003-Volume 4*, pages 180–183. Association for Computational Linguistics.

[Mittal2010] Vipul Mittal. 2010. Automatic sanskrit segmentizer using finite state transducers. In *Proceedings of the ACL 2010 Student Research Workshop*, pages 85–90. Association for Computational Linguistics.

[Nair and Peter2011] Latha R Nair and S David Peter. 2011. Development of a rule based learning system for splitting compound words in malayalam language. In *Recent Advances in Intelligent Computational Systems (RAICS), 2011 IEEE*, pages 751–755. IEEE.

[Natarajan and Charniak2011] Abhiram Natarajan and Eugene Charniak. 2011. S3-statistical sam. dhi splitting.

[Oflazer et al.1994] Kemal Oflazer, Elvan Göçmen, Elvan Gocmen, and Cem Bozsahin. 1994. An outline of turkish morphology.

[Prince and Smolensky2008] Alan Prince and Paul Smolensky. 2008. *Optimality Theory: Constraint interaction in generative grammar*. John Wiley & Sons.

[Whitelaw and Patrick2003] Casey Whitelaw and Jon Patrick. 2003. Named entity recognition using a

character-based probabilistic approach. In *Proceedings of the seventh conference on Natural language learning at HLT-NAACL 2003-Volume 4*, pages 196–199. Association for Computational Linguistics.