Multicultural Education

Volume 7, Issue 11, 2021

A Study of aindra School of Sanskrit Grammar in the Light of Paninian Framework in Natural Language Processing

Kaustav Sanyala, Partha Paula

Article Info

Article History

Received:
April 12 , 2021

Accepted:
November 14, 2021

Keywords :
Grammar, Natural Language Processing, Computational Sanskrit, Paninian Framework, Linguistics

DOI:
10.5281/zenodo.5700833

Abstract

Before the era of Panini's Grammar, aindra School of grammar was the most popular tradition to provide a language model for Sanskrit. This tradition used merely the structural analysis of the words to construct the framework of a language. Though this tradition is extinct today, Panini's grammar provides a pathway to reconstruct the aindra School of grammar. In this paper we have reviewed the existing major works of Computational Sanskrit in the field of Natural Language Processing (NLP) and proposed a possible approach which can further be implemented to apply Paninian framework only in order to reconstruct a lost school of grammar of Sanskrit linguistics.

Introduction

Among the eight most prominent traditions of Sanskrit Grammar [1] the *aindra* School of Sanskrit grammar has served the purpose of protecting the meaning of the Vedas for a long time before Panini provided his work. The vast application of the *aindra* School of grammar in the Vedic period was due to the specific structure of the text's composition. The *aindra* tradition of grammar was capable to provide a language model for Sanskrit language entirely on the morphological base [2]. Later on this framework gave rise to one of the oldest living language, Tamil. Burnell showed in details in his work the remarkable resemblance of the Tamil language model with the *aindra* framework. In the later Vedic era when the Vedic literature was on rise on the basis of the *samhitAs*, the morphological language model didn't suffice the need of the scholars. Broader aspects of the language had to be covered and then only Paninian framework came into existence. In the *pratyAhAra sUtras* the general phonology was introduced. In the *sUtra-pATha* many more aspects like the *kArakas*, *vibhaktis, samAsa, sandhi* etc. were introduced. In the *dhAtupATha* section the roots were compiled and so on. But it will be inappropriate to assume that the *aindra* framework lacks completeness on its own. The morphological analysis of the language proposed by the scholars of the *aindra* tradition did provide a language model that can suffice all the needs to analyze and understand the Vedic literature. Moreover Panini in his extensive work provided a very logical gateway to reconstruct the root framework of the *aindra* Grammar. Among a number of certain procedures in this paper we have considered the *sandhi prakaraNa* of the Paninian framework in order to reconstruct the morphological language model for the Sanskrit language. In the later sections of this paper after providing the brief overview of the Paninian and the *aindra* framework we will primarily review the existing works and then will propose an algorithm which will not only be capable to produce the morphological construct for the language but will also provide a tool to analyze the Paninian semantics as well in a critical manner.

1. Related Works

1.1 Paninian and the aindra Framework

The six limbs of the Vedas, i.e. the *vedAngas*, serve three major purposes with respect to maintaining the integrity of the root text (*samhitAs*) [3]. They have different aspects and serve the purposes thus:

- Maintaining the Identity: *SikShA* and *Canda.*
- Maintaining the Rites: *kalpa* and *jyotiSha*.
- Maintaining the Meaning: *nirukta* and *vyAkaraNa*.

The ideology of the *aindra* grammar was focused on the etymological aspects of the Vedas and the language construct provided by it served the purposes of *nirukta* and *vyAkarana* on its own [2]. The language-model proposed by this tradition included a great deal of involvement of Yaska's '*Nirukta*' texts, which are none other

than word-level analysis of the language. That could be one of the reasons, in that era the study of grammar was termed as the discipline of words (*SabdAnushasanam*)[3].
Moreover Paninian framework provided broader aspects of the language [5], namely:
- Semantic rules (*sUtra*)
- Roots (*dhAtu*)
- Categorization of vocabulary (*gaNa*)
- Categorization of suffixes (*uNa*)
- Gender studies (*linga*)

His huge corpus of semantic rules is divided into several teachings (*upadeSa*) that serve solely one purpose at a time, namely:
- *pratyAhAra*: Phonetics
- *sUtrapATha*: Collection of semantics
- *dhAtupATha*: Collection of roots
- *vArtikpATha*: Collection of primary commentaries
- *gaNapATha*: Collection of vocabulary sections
- *uNAdipATha*: Collection of suffixes
- *lingAnuSAsanam*: Collection of gender based vocabulary
- *Agama*: Study of letter

Among these sections the word-level analysis leads us to the root framework of the *aindra* framework. On the basis of the purposes served Bhattoji Dikshita classified the semantic rules provided by Panini in his celebrated text *Siddhanta Kaumudi*. The sections are known as *prakaraNa* [6]. For example:
- *sanJa prakaraNa*: Definition
- *sandhi prakaraNa*: Concatenation of words
- *samAsa prakaraNa*: Complex word analysis
- Study of the suffixes
- Study of the gender of words etc.

Among these classifications some focus on the word level analysis and thus from those rules it is possible to recreate a rough framework for the lost tradition of *aindra* School of grammar. In the following section we shall discuss the existing works in the field of NLP which cover the previous mentioned domains of the Paninian framework.

1.2 kAraka as a tool of Knowledge Representation

For the knowledge representation purpose Briggs[7] took the *kAraka prakaraNa* into account. In his paper he has presented equivalence between the semantic nets and the *kAraka* relations to represent knowledge. It is true that throughout *aShTAdhy*, Panini had managed to develop several definite algorithms to form the structural base of Sanskrit language. Briggs took the *kAraka* relation and just represented it in a standard structure of semantic nets for knowledge representation. The *kAraka* relations actually falls under the noun phrase generation phase with twenty one suffixes known as the sup *vibhakti* [8]. The words used in a sentence are known as the *pada*. According to Panini they can either have the sup suffixes or the ting suffixes (सुप्तिङन्तं पदम् (1.4.14)) [5], where the first category produces noun phrases and the later one the verb phrases. These sup suffixes when added with the words produce the *kAraka* relations. Briggs's paper focused on the relational aspect only of the *kAraka prakaraNa*, excluding the noun phrase generation detailing. The problem may arise here in five cases:
- *karaNa kAraka* dual number
- *sampradAna kAraka* dual number
- *apAdAna kAraka* dual number
- *sampradAna kAraka* plural number
- *apAdAna kAraka* pural number

First three have the same suffix '*bhyAm'* and the later two have the same suffix '*bhyas'* [8]. From a sentence in Sanskrit it will need the complete analysis of the sentence to deduce the *kAraka* relation with the other agents present if the above mentioned cases occur. For example,
- देवेभ्यः नमः। (salutations to the Gods)
- देवेभ्यः प्राप्तमिदम्। (this is got from the gods)

In both the cases the *kAraka* relations are different but the pada 'देवेभ्यः' is the same for the suffix similarity. So for the same word the representation of knowledge in both the sentences will vary for the change in the *kAraka* relational anomaly. This angle is somehow not mentioned in Briggs's paper. However a different approach is seen in the work of Bhavin Panchal, Vishvajit Bakrola and Dipak Dhabi [9]. Instead of the kAraka relations, they used the dhAtupATha and used their morphological and syntactic encoding properties. Being a distinct section of aShTAdhyAyI, the framework that the dhAtupATha provides seems a more complete and efficient aspect in the case of knowledge representation.

On the other hand Manish Kumar and Manish Dua [10] used kAraka representation for the parsing purpose for the free-order languages. Their approach to map Stanford parser (used to get the dependencies for fixed-order languages) to the kAraka relations of Panini is a remarkable application of Paninian framework of kAraka relations in the field of NLP. This adaptation of Stanford parser in the Paninian framework allowed them to create a parallel English-Hindi Treebank, which can be treated as a better approach to apply kAraka relation in NLP than mere knowledge representation. But the catch in this scenario is the consideration of Hindi language, not Sanskrit directly. So the ambiguity in the suffix application explained previously couldn't be resolved in this case either.

1.3 The Paninian Model

While introducing Paninian Parser, Akshar Bharati, et.al provided a very well-structured parser involving the basic semantic rules of kAraka and verb along with the active lexicons [11]. To design a core parser for a free-order language they considered two steps:

- identifying the kAraka relations in a sentence and
- identify the sense of the word.

In order to implement the idea they considered studying Hindi language and used integer programming to build a constraint parser. Using the lakShaNa chart they deduced the essence of the same word used in different contexts in sentences.

The core idea of their works is to mostly computationally implement Panini's grammar in the free-order Indian languages, mostly Hindi. Akshar Bharati and Rajeev Sangal [12] proposed the levels of Paninian model to analyze a sentence. There they mentioned kAraka and vibhakti as two independent levels of a sentence apart from the meaning and the desire of the speaker. We couldn't agree with the idea as it directly contradicts the actual essence of Panini's grammar [3]. The meaning of the sentence follows the desire of the speaker and the kAraka and vibhakti follows the meaning. The idea is also established by Adi Shankaracharya in his non-dualism commentaries [13]

Using the Paninian model exclusively Bhavin Panchal, et.al compiled their wok in [14]. In this work they possibly for the first time brought to light the idea of upasarga (prefix) and pratyaya (suffix) in the field of NLP. For word-level analysis of Sanskrit they for the first time considered the core idea of word formation taking the word and disintegrate it into the root, prefix and suffix. After the split, their work to translate it into English is remarkable in its own. This work somehow can be considered as a step towards the *aindra* School, as their approach of machine translation is solely word-level analysis only.

1.4 The Extensive Sanskrit Language Analysis

One of the most extensive works in analyzing Sanskrit has been done by Pawan Goyal, et.al [15]. We know this already that the idea of Turing Machine was inspired from Panini's pratyAhAra rules only [16]. Goyal, Arora and Behera reversed the application and with the help of a DFA they implemented the core idea of the utsarga apavAda approach. This work allowed them to break the internal and external sandhis in a sentence and parse them accordingly. Thus they were able to create semantic nets for multiple classes from a complete paragraph. The brilliance of their work should further be noted for they did their work on Sanskrit language only. Unlike most of the researchers in the field of computational Sanskrit, their work was not limited to the kAraka relation to parse but to explore wider aspects of the language.

Another approach has made its position strong enough in Vishvajit Bokarola and Jitendra Nasriwala's work [17]. In this work, they considered the root words again and the word generation application of them from the semantic rules by Panini himself. The brilliance of that approach is, it focuses on an advance idea of NLP, i.e. the dictionary-independent machine translation. The word generation rules of Panini concise the generation of millions of words making Sanskrit vocabulary rich and efficient. This paper applied this idea in the field of NLP making machine translation more efficient.

2. Proposed Method

As we have seen so far most of the works in the field of computational Sanskrit has been done on either the knowledge representation or to find out the relation among the words in a sentence. Moreover work to find out the accuracy of Panini's rules over free-word-order Indian languages such as Hindi has been also done extensively. As we have said previously Panini's grammar provide a pathway to the lost *aindra* tradition and here we shall try to explore the field. Panini's grammar provides rules to disintegrate words on the basis of various factors like *sandhi, samAsa, pratyaya* etc. *aindra* School of grammar explores that very field [2]. On the basis of the formation of words it analyzes the complete language. Among all the categories of the words we have taken the sandhi prakaraNa to disintegrate. The reasons to take sandhi into account are:

- sandhi in Sanskrit plays a major role to form words
- sandhi is used between two words to form a new word, as well as two words in a sentence for shortening purpose
- in-word sandhis give rise to a huge number of vocabulary

- in-word sandhis can be disintegrated to find out the root of a word
- in-word sandhi disintegration can lead us to the actual grammatical meaning of the word without considering any external reference

In this proposed method we have considered the in-word sandhis to disintegrate because from that as future work, in-sentence sandhi disintegration will be an easier work to implement. Moreover in-sentence sandhis are optional to implement, but in-word sandhis are compulsory. sandhis are classified into three sub-sections namely: 1. ac sandhi, 2. hal sandhi and 3. visarga sandhi. Here we propose an algorithm to train and test the system with the semantic rules for sandhi proposed by Panini.

```
Step 1: Train Module:
 Create three modules for sandhi training ac, hal and visarga.
    Step 1.1: ac Module:
        Step 1.1.1: train with the eight semantic rules
            i.      इको यणचि ।
            ii.     एचोऽयवायावः ।
            iii.    अकः सवर्णे दीर्घः ।
            iv.     आद् गुणः ।
            v.      वृद्धिरेचि ।
            vi.     एङि पररूपम् ।
            vii.    एङः पादान्तादति ।
            viii.   इदुदेद्विवचनं प्रगृह्यम् ।
    Step 1.2: hal Module:
        Step 1.2.1: train with sixteen semantic rules
              i.     स्तोः शचुना श्चुः ।
              ii.    ष्टुना ष्टुः ।
              iii.   शश्छोटि ।
              iv.    चोः कुः ।
              v.     खरि च ।
              vi.    झलां जश् झशि ।
              vii.   नश्चापदान्तस्य झलि ।
              viii.  अनुस्वारस्य ययि परसवर्णः ।
              ix.    यरोऽनुनासिकेऽनुनासिको वा ।
              x.     तोर्लि ।
              xi.    ससजुषो रुः ।
              xii.   अतो रोरप्लुतादप्लुते ।
              xiii.  भोभगोअघोअपूर्वस्य योऽशि ।
              xiv.   एतत्तदोः सुलोपोऽकोरनञ्समासे हलि ।
              xv.    हलि सर्वेषाम् ।
              xvi.   लोपः शाकलस्य ।
    Step 1.3: visarga Module
        Step 1.3.1: train with one rule: खरवसानयोर्विसर्जनीयः ।

Step 2: Test Module
    Step 2.1: run lexical analysis for the given word
    Step 2.2: mark in-word compound letters
    Step 2.3: if the compound letter includes swara (vowel)
        Step 2.3.1: search in ac module
    Step 2.4: else
```

```
        Step 2.4.1: search in hal and visarga module
    Step 2.5: if found
        Step 2.5.1: disintegrate
    Step 2.6: else
        Step 2.6.1: return the input word
```

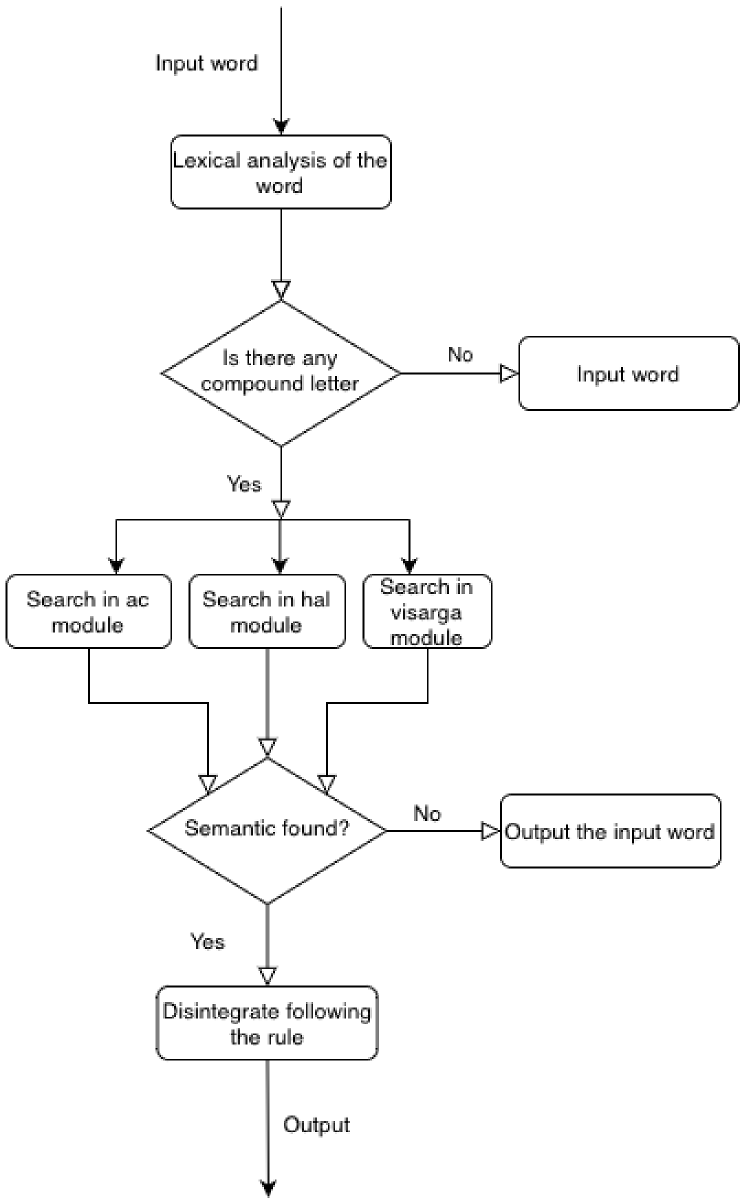

Figure 1Test Module Flow Chart

3. Future Work and Conclusion

By successful implementation of this algorithm we can find out the applied pratyAhAras used in all the semantic rules provided by Panini. This implementation can further modified to decompose the in-sentence sandhis as well. Moreover the result produced by this algorithm can convey the meaning of the words themselves on the basis of the meaning of the output root (dhAtu) only and thus providing a framework to reconstruct the morphological base of the proposed language model on the ideas of the *aindra* School of grammar. The implementation of the algorithm is what we are working on for now and we hope to produce results based on that in coming days.

References

Vopadeva, *Kathasaritsagara.*

A.C. Burnell, On the *Aindra* School of Sanskrit Grammarians Their Place in the Sanskrit and Subordinate Literature, 1875.

N. Mishra, Kashika, Chowkhamba Sanskrit Series Office, Varanasi, 1969

Mundakopanishad, Academy of Sanskrit Research, Melkote, 2005.

N.Mishra, Ashtadhyayi-sutrapaatha, Chowkhamba Orientaia, Varanasi, 2014.

B. Diksita, Vaiyakarana-Siddhanta-Kaumudi, ChowkhambaSanskrit Series Office, Varanasi, 1969.

R. Briggs, Knowledge Representation in Sanskrit and Artificial Intelligence, AI Magazine Volume 6 Number 1 (1985).

P. Dikshit, Shighrabodha Vyakarana, Pratibha Prakashan, 2012.

B. Panchal, et.al. (2018) *An Efficient Approach of Knowledge Representation Using Paninian Rules of Sanskrit Grammar*, In: Sa P., Bakshi S., Hatziygeroudis I., Sahoo M. (eds) Recent Findings in Intelligent Computing Techniques. Advances in Intelligent Systems and Computing, vol 709. Springer, Singapore. https://doi.org/10.1007/978-981-10-8633-5_21

M. Kumar, M. Dua, Adapting Stanford Parser's Dependencies to Paninian Grammar's Karaka relations using VerbNet, Procedia Computer Science 58 (2015) 363-370.

A. Bharati, et.al. Natural Language Processing A Paninian Perspective, PHI Learning Private Limited, ISBN-978-81-203-0921-0.

A. Bharati, R. Sangal, Computational Paninian Grammar Framework, Supertagging: using Complex Lexical Descriptions in Natural Language Processing, ISBN- 978-0-262-01387-1.

M. Mitra, Shankarprasthane Advaitabhavana, Progressive Publishers, ISBN-81-8064-059-0.

B. Panchal, et.al. *An Approach of Splitting Upsarg and Pratyay of Sanskrit Word Using Paninian Framework of Sanskrit Grammar*, Emerging Technologies in Data Mining and Information Security, 2019, Springer.

P. Goyal, et.al. Analysis of Sanskrit Text: Parsing and Semantic Relations.

S. C. Kak, The Paninian Approach to Natural Language Processing, International Journal of Approximate Reasoning, 1987, 1:117-130.

V. Bokarola, J. Nasriwala, (2019) *Computational Representation of Paninian Rules of Sanskrit Grammar for Dictionary-Independent Machine Translation*, Advances in Computing and Data Sciences. ICSCDS 2019. Communication in Computer and Information Science, vol 1046. Springer, Singapore.

Author Information

Sanyala
Sarala Birla University, Ranchi, Jharkhand, India

Partha Paula
Sarala Birla University, Ranchi, Jharkhand, India