Compound Type Identification in Sanskrit : What Roles do the Corpus and Grammar Play?

Amrith Krishna, Pavankumar Satuluri, Shubham Sharma, Apurv Kumar and Pawan Goyal
Department of Computer Science & Engineering
Indian Institute of Technology, Kharagpur, WB, India
amrith@iitkgp.ac.in

Abstract

We propose a classification framework for semantic type identification of compounds in Sanskrit. We broadly classify the compounds into four different classes namely, *Avyayībhāva*, *Tatpuruṣa*, *Bahuvrīhi* and *Dvandva*. Our classification is based on the traditional classification system as mentioned in the ancient grammar treatise *Aṣṭādhyāyī* by Pāṇini, written 25 centuries back. We construct an elaborate feature space for our system by combining conditional rules from the grammar *Aṣṭādhyāyī*, semantic relations between the compound components from a lexical database *Amarakoṣa* and linguistic structures from the data using Adaptor Grammars. Our in-depth analysis of the feature space highlights the inadequacy of *Aṣṭādhyāyī*, a generative grammar, in classifying the data samples. Our experimental results validate the effectiveness of using lexical databases as suggested by Kulkarni and Kumar (2013) and put forward a new research direction by introducing linguistic patterns obtained from Adaptor grammars for effective identification of compound type. We utilise an ensemble based approach, specifically designed for handling skewed datasets and we achieve an overall accuracy of 0.77 using random forest classifiers.

1 Introduction

Compounding is a productive process of vocabulary expansion in languages where two or more nouns are used together to generate a new lexeme. Compound analysis is computationally challenging primarily due to three factors: i). compounds are highly productive in nature, ii). the relation between the components is implicit and iii). the correct interpretation of a compound is often dependent on contextual or pragmatic features (Kim and Baldwin, 2005). For example, 'houseboat' and 'boathouse'[1] are compounds formed from the same pair of nouns, 'house' and 'boat', but do not mean the same. Similarly, the relation between 'olive' and 'oil' in 'olive oil' does not hold between 'baby' and 'oil' in 'baby oil'.

Identifying the head of a compound can lead to significant improvements in semantic analysis tasks like Machine Translation, Question Answering etc. (Weller et al., 2014; Tiedemann, 2005). The head of a compound, in general is indicative of the referent(s) of the compound, in addition to determining the syntactic properties of the compound. For example, in 'paleface' paraphrased as 'a person who has a pale face', the head of the compound is an external entity. Here a word to word translation of the components would yield undesirable results. In 'bittersweet', both the stems 'bitter' and 'sweet' are the heads of the compound. In both 'houseboat' and 'boathouse', the final component forms the head.

On our empirical investigation of the Digital Corpus of Sanskrit (DCS)[2], we find a rich use of compounds with a presence of about 198,000 unique compound words occurring 373,000 times in a corpus of 2.5 million tokens (after stop-word removal). This is almost double in comparison to languages like German, which report 5.5-7% of corpus presence of the compounds (Schiller, 2005; Baroni et al., 2002). In DCS, 75% of the vocabulary consists of compounds, as against 47% vocabulary share

[1]http://wikidiff.com/houseboat/boathouse

[2]http://kjc-sv013.kjc.uni-heidelberg.de/dcs/

(Baroni et al., 2002) of compounds in German. We also find that 43% of the 66,000 lemmas in the corpus vocabulary were part of the compound formation as compared to 3-4% in English (Séaghdha and Copestake, 2013). In Sanskrit literature, especially in poetry, use of long compounds with multiple components is common. In DCS, more than 41 % of compounds have 3 or more components. For example, "*pravaramukuṭamaṇimarīcimañjarīcayacarcitacaraṇayugalaḥ*" is an exocentric compound used from the text "*Pañcatantram (kathāmukham)*" which translates to "The pair of whose feet was covered with a stream of rays originating from the gems in wreaths of eminent noble kings". This compound is composed of 9 components (Krishna et al., 2016).

Aṣṭādhyāyī, an ancient grammar treatise on Sanskrit, discusses the generation of four broad classes of compounds, namely, *Avyayībhāva*, *Tatpuruṣa*, *Bahuvrīhi* and *Dvandva*. We propose a classifier model to identify the semantic type of Sanskrit compounds i.e. one of the four classes. We find that the aforementioned notion of 'head' in compounds is discriminative as per this categorization. For our classification task, we successfully combine features extracted from rules in *Aṣṭādhyāyī*, taxonomy information and semantic relations inferred from *Amarakośa* ontology (Nair and Kulkarni, 2010), and linguistic structural information from the data using Adaptor grammar (Johnson et al., 2006). We perform an in-depth analysis of the performance of the system and highlight where the existing rules of *Aṣṭādhyāyī*, a generative grammar, are inadequate in classifying the data samples and show how additional features help us improve the performance of the classifier, by using our results obtained on a held-out dataset.

2 Compounds in Sanskrit

The compounds in Sanskrit exhibit numerous regularities that characterise them (Gillon, 1991). Compounds in Sanskrit are concatenative in nature, with a strict preference for the ordering of the components. A generated compound is treated as a fully qualified word (pada), such that the compound is subject to all the inflectional and derivational modifications applicable to nouns. Affixation occurs at the end of the compound similar to languages like that of Greek and not within the components (Ziering and van der Plas, 2016; Gillon, 1991). Any compound can be analysed by decomposing it into two immediate component nouns.

Linguists in Sanskrit have deeply discussed exceptions for the aforementioned regularities leading to different categorisations and further sub-categorisations of the compound types (Kulkarni and Kumar, 2013; Gillon, 2009). We only consider the four broad categorisations of the compounds. We now explain four classes of compounds and discuss various discriminative aspects about the broad level classes that we can abstract out from the generated forms and use in our system. In Sanskrit Grammar, compounds are classified into four general categories, namely, *Avyayībhāva*, *Tatpuruṣa*, *Bahuvrīhi* and *Dvandva*.

1. *Avyayībhāva* Compounds - In *Avyayībhāva* compound, the first component is an indeclinable or *avyaya*, which generally forms the head of the compound. The compound so generated will also become an indeclinable. For instance, in '*upakṛṣṇam'* (near to *Kṛ̇śṅa*), the word '*upa*' (near) is an indeclinable and the second component '*kṛṣṇa*' bears an inflectional affix, but the compound becomes an indeclinable.
2. *Tatpuruṣa* Compounds or Determinative compounds - They are endocentric compounds in which the second component is generally the head of the entire compound. For example, the phrase '*rājñaḥ puruṣaḥ*' (King's man) yields *rājapuruṣaḥ*. The second component, '*puruṣaḥ*' forms the head in the canonical paraphrase and hence the head of the compound (Gillon, 1991). The relation between the components is marked by the genitive case inflection of the first component *rājñaḥ*. *Tatpuruṣa* compounds constitute a distinctive sub-categorization, namely, Descriptive compounds or *Karmadhāraya*. In *karmadhāraya* compounds, one of the components needs to be an adjective and it is observed that generally the adjective comes as the first component. For example, in '*nīlameghaḥ*' (blue cloud) the first component, '*nīla*' (blue), is qualifying the head word, '*megha*' (cloud).
3. *Bahuvrīhi* Compounds or Exocentric Compounds - When the components in the compound refer to some external entity, say a person or an object, we call it a *Bahuvrīhi* compound. Here, the referent of the compound becomes the head of the compound. For example, '*pītāmbaraḥ*' is paraphrased as '*pītām ambaraṃ yasya saḥ*'. Here the words *pītam* (yellow) and *ambaram* (cloth) together form the

compound referring to the Lord Vishnu. In absence of the paraphrase, the referent or headword often needs to be inferred from the context in which the compound is used. However, the gender differences between the final component and that of the compound is a convenient heuristic that can be used to identify the compound type in some of the cases (Goyal and Huet, 2013).

4. Dvandva or Copulative compounds - They are conjunctive compounds where the components are compounded to show the collectiveness of the individuals. The components involved may be nouns or adjectives. Since the components share a conjunctive relation, often multiple components are compounded together in a single process. In *Dvandva* compounds, the compound generally assumes the gender of the final component. But deciding the final component can be tricky especially in a free word order language like Sanskrit. For example, a *Dvandva* compound formed from the paraphrases '*pitā ca mātā ca*' and '*mātā ca pitā ca*' (mother and father) will always be of the form '*mātāpitarau*', which is in masculine due to the masculine noun *pitā* (father), but will never be '*pitāmātarau*', which should be in feminine gender. The formation of the latter is prohibited in the grammar, thereby eliminating the possibility of a conflict.

It is often observed that the same pair of components can generate compounds belonging to different semantic types. For example, *pītāmbaraḥ* (Lord Vishnu) and *pītāmbaram* (yellow cloth) are *Bahuvrīhi* and *Tatpuruṣa* compounds respectively, formed from the same components, *pīta* and *ambaram.* Here the gender of the compounds becomes a discriminative feature. In general, the stem '*ambara*' is in neuter and hence in *Tatpuruṣa* compounds, the compound also maintains the neuter gender. But, for *Bahuvrīhi* compounds, the gender is based on the referent, which in this case is masculine.

Now, if we consider a compound like *nīlotpalam*, which contains two components *nīla* and *utpala*, the compound maintains the same final form in the case of both *Tatpuruṣa* and *Bahuvrīhi*, leading to ambiguity in semantic type identification. To resolve this conflict, either the canonical paraphrase or the context of usage is necessary. The potential conflict in disambiguation is often expressed between the compounds of *Bahuvrīhi* and specifically *karmadhāraya* compounds. Similarly, for compounds where the first component denotes a negation marker, there can be conflicts between *Tatpuruṣa* and *Bahuvrīhi* classes. The specific sub-categories are called as *Nañ-Tatpuruṣa* and *Nañ-Bahuvrīhi* compounds respectively. For instance, the compound '*aputraḥ*' is paraphrased as '*na putraḥ*'(not a son) in the case of *Tatpuruṣa* and '*avidyamānaḥ putraḥ yasya saḥ*'(having no son) in the case pf *Bahuvrīhi*. *Tatpuruṣa* compounds can conflict with *Avyayībhāva* compounds as well. For example in '*ativanam*', the compound consists of two components viz '*ati*' and '*vanam*'. Here the first component '*ati*' is an indeclinable, a strong characteristic of *Avyayībhāva* compounds. But, there exists a sub-categorisation of *Tatpuruṣa*, where the first component is an indeclinable. The paraphrase of '*ativanam*' in the case of *Avyayībhāva* is '*vanasya atyayaḥ*' (past the forest) and '*vanam atikrāntaḥ*' (having passed the forest) in the case of *Tatpuruṣa*.

The aforementioned instances show the challenges involved in identifying the semantic type of a compound. Sometimes, the task is non-trivial even for humans and human cognition often relies on the context in which the compound is used or on the world knowledge about the entities involved .

3 Method

In our current work, we treat the problem as follows. When given a compound word decomposed into two immediate components of the compound, we identify the semantic type of the given compound and classify it into one of the four classes as discussed in Section 2. We build a feature-rich multi-class classifier and analyse the effectiveness of the features for the classification task. In *Aṣṭādhyāyī*, the generation of a compound is assumed to start with the canonical paraphrase of the compound. The noun declensions, modifiers and relation words in the paraphrase are then elided to form the compound. In our current settings, we only consider the compound and its individual split components. In this section, we describe the various features used for this classification task.

3.1 Rules from *Aṣṭādhyāyī*

Kulkarni and Kumar (2013) provides a categorisation of the rules in *Aṣṭādhyāyī* which are useful for compound analysis. Table 1 provides a summary of the type of rules that we employ in our system. The

Rule Type	Rule	Example
Type 1: Lexical lists	*Aṣṭādhyāyī* Rules like A.2.1.40 enlist specific lists of stems to be used as a component in compound formation	**akṣaśauṇḍaḥ** - *śauṇḍa* is listed in the rule A.2.1.40
Type 2: Morphological Properties	Rules like A.2.1.25 use inflectional suffix, derivational suffix etc. of the components in paraphrase as conditions for compounding	*kṛta* in the compound **svayamkṛta** bears a derivational suffix *ta*.
Type 3: Semantic property of the component	Rules like A.2.1.45 state specific properties of objects as conditions for compounding, e.g., part of day.	Stem *pūrvāhṇa* (forenoon) in **pūrvāhṇakṛta** denotes a part of day.
Type 4: Semantic relations between the components	Rules like A.2.1.57 check for specific relations between the components, e.g., Modifier - Modified relation	**nīlotpalam** - *nīla* (blue) describes the second component utpalam (lotus).

Table 1: Various rule types in *Aṣṭādhyāyī* for compound analysis (Kulkarni and Kumar, 2013). A.2.1.40 etc. indicate the rule numbers in the book.

type 1 rules are lexical lists which contain lists of nouns and indeclinable that appear as a component in the compound. Type 2 considers the morphological properties of the components. Inflectional affixes are indicators of the case of the noun, gender and plurality. In our work, we utilise string patterns at the end of the components to infer inflectional and derivational affixes used. Obtaining the exact noun declensions from the final forms is not always deterministic as the same affix might be used for representing multiple noun declensions for a given word. Additionally, the current parsers in Sanskrit do not include analysers for derivational affixes. On an empirical analysis over a dataset of 8000 labelled compounds, we find that a little above 4000 of 10000 unique compound components are recognised by the Sanskrit Heritage Reader unambiguously (Goyal and Huet, 2016). This is primarily due to the fact that the parsers are lexicon driven, and also due to the absence of derivational suffix analysers. The last two rule types are semantic in nature. Rule type 3, i.e., rules that check for semantic property of the component, is captured using manually curated lists of lexicons such as list of rivers, parts of day and night, etc. It essentially contain word lists stated outside of *Aṣṭādhyāyī*. The last type of rule looks into the possible relations between the components. where we utilise the lexical database *Amarakoṣa*.

3.2 Relations from Lexicons

Lexical databases with annotated semantic networks are beneficial in identifying the semantic compatibility between individual nouns and hence can be used in compound analysis (Kim and Baldwin, 2005; Séaghdha and Copestake, 2013). We utilise '*Amarakoṣa*', an ancient dictionary which covers about 11580 words (9031 unique lemmas) in altogether 4035 *synsets*. With efforts from Nair and Kulkarni (2010), Amarakoṣa is digitised, forming a semantic network explicitly labelled with semantic relations between the words. The lexicon primarily consists of six relations, of which three of the relations, namely, 'part-whole','is a kind of' and 'master-possession', are useful in identifying *Tatpuruṣa* compounds. Two of the three remaining relations, namely, 'child-parent' and 'husband-wife', are helpful in identifying *Dvandva* compounds. An additional advantage with *Amarakoṣa* is that we get gender information about the individual nouns from the e-lexicon, which is a discriminative factor in identifying *Bahuvrīhi* compounds as mentioned in Section 2. For each component, the gender, head word and the corresponding word with which the component bears the relation, are used as features. We consider all the six relations in *Amarakoṣ*a between the compound components.

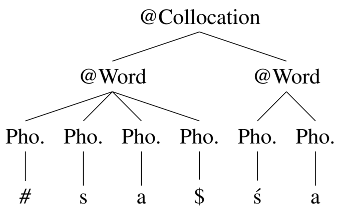
3.3 Variable Length Character n-grams

We capture semantic class specific linguistic regularities present in our dataset using variable length character n-grams and character n-gram collocations shared between compounds. In order to learn the character n-grams, we use Adaptor grammars (Johnson et al., 2006), a non-parametric Bayesian approach towards learning productions for a probabilistic context free grammar (PCFG).

The grammar obtained from the Adaptor Grammar (AG) is a probabilistic context free grammar, where the productions form a set of fixed non-terminals and the probabilities for the productions to be invoked are learnt from the data. In Adaptor Grammar, a skeletal context free grammar is defined as shown in Listing 1a, where the set of non-terminals to be adapted is fixed a priori and will be a subset of the entire set of non-terminals in the skeletal grammar. For each of the adapted non-terminal, marked with a '@', the grammar learns a distribution over trees rooted at each of the adapted non-terminal (Zhai et al., 2014). We learn grammars G1, G2 and G3 with the same skeletal structure in Listing 1a, but with different data samples belonging to *Tatpuruṣa*, *Bahuvrīhi* and *Dvandva* respectively. We did not learn a grammar for *Avyayībhāva*, due to insufficient data samples for learning the patterns. We use a '$' marker to indicate the word boundary between the components and a '#' symbol to mark the beginning and ending of the first and the final components respectively. We also learn a grammar G4, where the entire dataset is taken together along with additional 4000 random pair of words from the DCS corpus, where none of the words appeared as a compound component in the corpus.

```
@Collocation -> Word+
@Word -> Phoneme+
Phoneme -> {Sanskrit Alphabet, $, #}
```

(a)



(b)

Listing 1: a) Skeletal grammar for the adaptor grammar (Johnson et al., 2006). b) Derivation tree for an instance of a production '#sa$ śa' for the non-terminal @collocation

Every production in the learned grammars has a probability to be invoked, where likelihood of all the productions of a non-terminal sums to one. To obtain discriminative productions from G1, G2 and G3, we find conditional entropy of the production with that of G4 and filter only those productions above a threshold. We also consider all the unique productions in each of the Grammars in G1 to G3. We further restrict the productions based on the frequency of the production in the data and the length of the sub-string produced by the production, both of them were kept at the value of three.
We show an instance of one such production for a variable length character n-gram collocation. Here, for the adapted non-terminal @Collocation, we find that one of the production finally derives '#sa$ śa', which actually is derived as two @Word derivations as shown in the Listing 1b. We use this as a regular expression, which captures some properties that need to satisfied by the concatenated components. The particular production mandates that the first component must be exactly *sa*, as it is sandwiched between the symbols # and $. Now, since *śa* occurs after the previous substring which contains $ the boundary for both the components, *śa* should belong to the second component. Now, since as per the grammar both the substrings are independent @word productions, we relax the constraint that both the susbtrings should occur immediately one after the other. We treat the same as a regular expression, such that *śa* should occur after *sa*, and any number of characters can come in between both the substrings. For the particular susbtring, we had 22 compounds, all of them belonging to *Bahuvrīhi*, which satisfied the criteria. Now, compounds where first component is '*sa*' are mostly *Bahuvrīhi* compounds, and this is obvious to Sanskrit linguists. But here, the system was not provided with any such prior information or possible patterns. The system learnt the pattern from the data. Incidentally, our dataset consisted of a few compound samples belonging to different classes where the first component was '*sa*'.

3.4 Other Features

We look for specific patterns that check for the lexical similarity between components. For example, consider the compound *bhāvābhāvau* where the final component *a-bhāva* is the negation for the first component *bhāva*. The prepositions '*a*' and '*an*' represent negation of entities. We identify those compounds, where the first and second components differ only by *a* or *an*. This heuristic has its own limitations, as not all negations are marked by the markers. We also use Jaro-Winkler distance, an edit distance variant, between both the components as an additional feature to capture the lexical similarity between the

Word	Component Position	Compound Class
iti	First	*Bahuvrīhi*
sva	First	*Tatpuruṣa*
manāḥ	Final	*Bahuvrīhi*
mātā	First	*Dvandva*
dharmā	Final	*Bahuvrīhi*

Table 2: Sample of filtered words and their position in the compound.

Classifiers	P	R	F	A
Random Forests	0.76	0.75	0.74	0.74
Extreme Random Forests (ERF)	0.76	0.75	0.74	0.75
Gradient Boosting Methods (GBM)	0.62	0.54	0.53	0.54
Adaboost Classifier	0.71	0.69	0.69	0.69

Table 3: Precision (P), Recall (R), F-Score (F) & Accuracy (A) for the competing systems on held-out dataset.

components. We find that the mean Jaro-Winkler distance between components of *Dvandva* compounds (0.48) is higher than that of other compounds (0.31 - 0.38). We also consider the last three characters of the second component, where the second component bears the nominal inflections of the compound word. We also used a handful of specific suffix patterns based on the entropy score of the patterns in discriminating the classes. The patterns are indicative of the affix information. We finally filtered 34 words and patterns by manual inspection, that had lower entropy score as well as there is a linguistic motivation for their inclusion. Table 2 shows a sample of such filtered words; we skip the linguistic motivation behind the filtering of each lemma due to space constraints.

4 Experiments

4.1 Dataset

We obtained a labelled dataset of compounds and the decomposed pairs of components from the Sanskrit studies department, UoHyd[3]. The dataset contains more than 32000 unique compounds. The compounds were obtained from ancient digitised texts including *Śrīmad Bhagavat Gīta*, *Caraka saṃhitā* among others. The dataset contains the *sandhi* split components along with the compounds. With more than 75 % of the dataset containing *Tatpuruṣa* compounds, we down-sample the *Tatpuruṣa* compounds to a count of 4000, to match with the second highest class, *Bahuvrīhi*. We find that the *Avyayībhāva* compounds are severely under-represented in the data-set, with about 5 % of the *Bahuvrīhi* class. From the dataset, we filtered 9952 different data-points split into 7957 data points for training and the remaining as held-out dataset. For all the features mentioned in Section 3, we have considered data points which are in the training set and we have not considered data from the held-out in calculating any of the features, including Adaptor grammar.

4.2 Results

Probably due to a large feature space of 2737 features we employ, and an imbalanced dataset, the performance of the classifier models like SVM and decision tree were near to chance with SVM making no predictions to the *Avyayībhāva* class. We use ensemble based approaches for our system and the results are presented in Table 3. The results presented in the table are predictions over held-out data, where the classifier was trained with the entire training data. We find that the Extreme Random Forests (ERF) (Geurts et al., 2006; Pedregosa et al., 2011) gives the best performance amongst all the compared systems in Table 3. The performance of the Random Forests and the ERF were almost similar with reported performance measures varying only from the third decimal point. Table 4b shows the result for the ERF classifier over training data when trained with 10 fold cross validation. The class-wise precision and recall for the model over held out dataset is presented in Table 4a. We find that the classifier fares poorly for *Avyayībhāva* and *Dvandva*, primarily due to sparsity in the data as they both amount to about 5% and 33% of the other two classes respectively.

To measure the impact of different types of features we have incorporated, we train the classifier incrementally with different feature types as reported in Section 3. We report the results over the held-out

[3]http://sanskrit.uohyd.ac.in/scl/

Class	P	R	F
A	0.92	0.43	0.58
B	0.85	0.74	0.79
D	0.69	0.39	0.49
T	0.68	0.88	0.77

(a)

Class	P	R	F
A	0.85	0.48	0.61
B	0.84	0.76	0.80
D	0.94	0.25	0.39
T	0.75	0.85	0.80

(b)

Class	P	R	F
A	0.84	0.67	0.74
B	0.88	0.73	0.79
D	0.69	0.61	0.65
T	0.72	0.87	0.79

(c)

Table 4: Classwise Precision (P), Recall (R) and F-Score (F) results for three different setups. a) on held-out data (Accuracy - 0.75). b) with 10-fold cross validation over training data (Accuracy - 0.79). c) Easy ensemble on held-out data (Accuracy - 0.77). A, B, D and T represent the classes *Avyayībhāva*, *Bahuvrīhi*, *Dvandva* and *Tatpuruṣa* respectively.

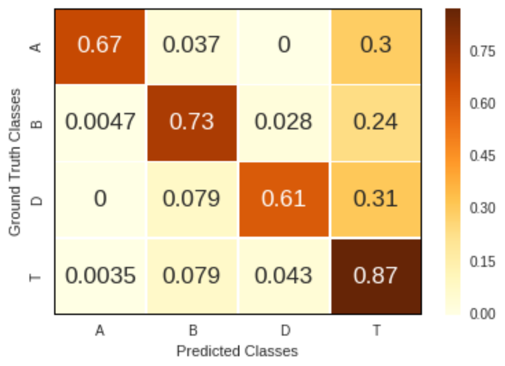


Figure 1: Confusion Matrix heat map for the easy ensemble classifier

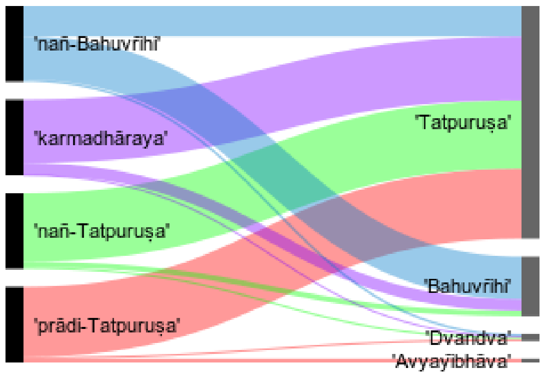


Figure 2: Alluvial graph showing the classification outcome for specific sub-classes

data. At first we train the system with only *Aṣṭādhyāyī* rules and features discussed in Section 3.4. We find that the overall accuracy of the system is about 59.34%. We do not report the accuracy of the system when we use *Aṣṭādhyāyī* rules alone as it was not sufficient to cover all the samples. Then we augmented the classifier by adding features from *Amarakoṣa* as described in Section 3.2. We find that the overall accuracy of the system has increased to 63.81%. Notably, the precision for *Dvandva* and *Bahuvrīhi* increased by absolute values 0.15 and 0.06 respectively. We then add the Adaptor grammar features to the feature set and Table 4a presents the result of the system with the entire feature set. We perform feature ranking based on entropy measure. We try to iteratively drop the least ranked features in steps of 50, till 1700 of 2737 features are dropped. We find that the accuracy does not change much, but mostly drops from the reported accuracy of 0.75 by a maximum of 0.38% (0.747).
To handle the imbalanced data set, we employed easy ensemble (Liu et al., 2009) approach. In easy ensemble approach, we form multiple training sets, where each of the set is a subset of the original training set such that the data samples in the minority classes remain intact whereas the majority classes are under-sampled to a defined proportion. In effect, we have multiple training sets where the data samples in the majority class are distributed across the subsets. Now with each of the subset, we run ERF classifier and average out the results. As can be seen from Table 4c, this approach gives consistent results across the four classes, with significant improvements in F-Score for *Dvandva* and *Avyayībhāva* classes. We further look into specific cases of compound classes which get misclassified. Figure 1 shows the confusion matrix heat-map for our best performing system, the easy ensemble classifier. From the heat-map we can observe that most of the mis-classifications go to *Tatpuruṣa*, resulting in a lower precision of 0.72 for *Tatpuruṣa*. It can also be noted that there are no *Dvandva* and *Avyayībhāva* mis-classifications. Figure 2 represents classification of the specific cases of sub-types as discussed in Section 2. *Avyayībhāva* and *Tatpuruṣa* can potentially be conflicting as there exists specific types of *Tatpuruṣa* where the first component can be an *avyaya*. We find that 6 data samples of *Tatpuruṣa* have been misclassified into *Avyayībhāva* and all the 6 data points have their first component as an *avyaya*. From Figure 1, it is already clear that majority of mis-classifications in *Avyayībhāva* go to *Tatpuruṣa*. Out of 70 mis-classifications of

Tatpuruṣa to *Bahuvrīhi*, 38 belong to *karmadhāraya* class. Also only 2 of the *karmadhāraya* compounds got mis-classified to a different class other than *Bahuvrīhi*. 83.81 % of the *karmadhāraya* compounds got correctly classified into *Tatpuruṣa*. *Nañ-Bahuvrīhi* and *Nañ-Tatpuruṣa* are also potentially conflicting cases, and we find that in *Nañ-Tatpuruṣa*, 8 of 11 mis-classifications happen to *Bahuvrīhi* class and in *Nañ-Bahuvrīhi* 13 of 14 mis-classifications happen to *Tatpuruṣa* class. But in all the aforementioned cases, the majority of the data samples got correctly classified.

5 Related Work

Semantic analysis of compounds has attracted much traction from the computational linguistics community, especially on languages like English, German, Italian, Afrikaans and Dutch (Verhoeven et al., 2014). Lexical databases like Wordnet (Kim and Baldwin, 2005) and Wikipedia (Strube and Ponzetto, 2006) were extensively used to infer semantic relations between the components in a compound. Effectiveness of verb-semantics and word sense disambiguation of the components involved were also studied (Kim and Baldwin, 2006; Kim and Baldwin, 2013). Séaghdha (2009) defines wordnet kernel functions for identifying the relational similarity between the components. Works like Séaghdha and Copestake (2013) use corpus-based approaches where co-occurrence measures between the components are utilised. Nastase et al. (2006) combine both the corpus-based approaches and lexical database based approaches for semantic analysis of compounds. Ziering and van der Plas (2016) presents a corpus-based approach for splitting of German compounds The authors augment the model by incorporating distributional information in Ziering et al. (2016). Botha et al. (2012) builds a language model by using a hierarchical Bayesian model where the models for head word and the other component are conditioned differently.
The *samarthāhnika* (Joshi, 1968) gives a detailed account of the discussion involved in the Indian tradition on the semantic compatibility of constituents and the compositionality of the meaning of a compound. Pataskar (1996) has discussed the use of the *Dvandva* compounds in relation to their case endings and how *Pāṇini* dealt with the *sūtras* in *Aṣṭādhyāyī*. Bhandare (1995) has discussed the structural and semantic aspects of *Dvandva* compounds. Mahavir (1986) has discussed various transformations that take place on the canonical paraphrase of a compound (*vigrahavākya*) to generate the final form. Gillon (2009) proposes an extended phrase structure syntax to represent the underlying constituent structure of the compounds. Kumar (2012) has described the computational analysis of Sanskrit compounds in his doctoral dissertation. Goyal and Huet (2013) describes various morphological phenomena involved in the generation and analysis of *Avyayībhāva* compounds. Pavankumar (2015) built a Sanskrit compound generator, adhering to the tradition followed in *Aṣṭādhyāyī*, as a part of his doctoral dissertation.

6 Conclusion

In this work, we built an automated classifier for identifying the semantic type of a compound in Sanskrit. With an ensemble based classifier approach, we tackle the challenge of an imbalanced dataset and our system effectively classifies data into appropriate semantic classes. We successfully incorporate rules from the ancient grammar treatise *Aṣṭādhyāyī*, lexical relations from *Amarakoṣa* and we also learn linguistic structures from the data using adaptor grammars. In our work, we show the improvement in performance after incorporating each of the aforementioned feature types. We also discuss the specific cases of conflicts between the semantic types.
Our primary motivation for this work was to understand the computational challenges involved in automated means of classifying compounds in Sanskrit. Our work can be seen as an extension in the line of works suggested in Kulkarni and Kumar (2013), and ours is the first such system for Sanskrit to incorporate semantic relations in taxonomy as well as class specific linguistic structures for the task. Results from our system demonstrate the effectiveness of a lexical database for the task and that it is a promising direction to be explored. We can extend the current system by incorporating other lexicons such as Indowordnet (Sinha et al., 2006) along with *amarakoṣa*. The improvement gained by using adaptor grammar productions look promising, as the grammar was not exposed to the data from the held-out dataset and yet was able to classify the data samples into appropriate classes. We will be further investigating the utility of Adaptor grammar in defining skeletal grammars as per the rules mentioned in Gillon (2009) and

some of the conditional rules in *Aṣṭādhyāyī* itself. From multiple instances discussed in Section 2, the role of context in determining compound type is evident. But such systems should be designed only after giving enough thought on solving the obvious resource constraints that the language currently faces.

Acknowledgements

The authors would like to thank Mr. Nishant Nikhil and Ms. Survi Makharia, IIT Kharagpur, for their contributions towards the implementation of the framework.

References

Marco Baroni, Johannes Matiasek, and Harald Trost. 2002. Wordform-and class-based prediction of the components of german nominal compounds in an aac system. In *Proceedings of the 19th international conference on Computational linguistics-Volume 1*, pages 1–7. Association for Computational Linguistics.

V. V. Bhandare. 1995. Structural and Semantic Aspects of the Dvandva Compound. *Annals of the Bhandarkar Oriental Research Institute*, 76(1-4):89–96.

Jan Botha, Chris Dyer, and Phil Blunsom. 2012. Bayesian language modelling of german compounds. In *Proceedings of COLING 2012*.

Pierre Geurts, Damien Ernst, and Louis Wehenkel. 2006. Extremely randomized trees. *Machine learning*, 63(1):3–42.

Brendan Gillon. 1991. Sanskrit word formation and context free rules. *Toronto Working Papers in Linguistics*, 11.

Brendan S. Gillon. 2009. Tagging Classical Sanskrit Compounds. In Amba Kulkarni and Gérard Huet, editors, *Sanskrit Computational Linguistics 3*, pages 98–105. Springer-Verlag LNAI 5406.

Pawan Goyal and Gérard Huet. 2013. Completeness analysis of a sanskrit reader. In *Proceedings, 5th International Symposium on Sanskrit Computational Linguistics. DK Printworld (P) Ltd*, pages 130–171.

Pawan Goyal and Gérard Huet. 2016. Design and analysis of a lean interface for sanskrit corpus annotation. *Journal of Language Modelling*, 4(2):145–182.

Mark Johnson, Thomas L Griffiths, and Sharon Goldwater. 2006. Adaptor grammars: A framework for specifying compositional nonparametric bayesian models. In *Advances in neural information processing systems*, pages 641–648.

S. D. Joshi. 1968. *Patañjali's Vyākaraṇa - Mahābhāṣya Samarthāhnika (P. 2.1.1).* Centre of Advanced Study in Sanskrit, University of Poona, Poona.

Su Nam Kim and Timothy Baldwin. 2005. Automatic interpretation of noun compounds using wordnet similarity. In *International Conference on Natural Language Processing*, pages 945–956. Springer.

Su Nam Kim and Timothy Baldwin. 2006. Interpreting semantic relations in noun compounds via verb semantics. In *Proceedings of the COLING/ACL on Main conference poster sessions*, pages 491–498. Association for Computational Linguistics.

Su Nam Kim and Timothy Baldwin. 2013. Word sense and semantic relations in noun compounds. *ACM Transactions on Speech and Language Processing (TSLP)*, 10(3):9.

Amrith Krishna, Bishal Santra, Pavan Kumar Satuluri, Sasi Prasanth Bandaru, Bhumi Faldu, Yajuvendra Singh, and Pawan Goyal. 2016. Word segmentation in sanskrit using path constrained random walks. In *Proceedings of COLING 2016*.

Amba Kulkarni and Anil Kumar. 2013. Clues from aṣṭādhyāyī for compound type identification. In *5th International Sanskrit Computational Linguistics Symposium (SCLS)*.

Anil Kumar. 2012. An Automatic Sanskrit Compound Processor. In *Proceedings of the International Sanskrit Computational Linguistics Symposium*. Springer-Verlag LNAI 5406.

Xu-Ying Liu, Jianxin Wu, and Zhi-Hua Zhou. 2009. Exploratory undersampling for class-imbalance learning. *IEEE Transactions on Systems, Man, and Cybernetics, Part B (Cybernetics)*, 39(2):539–550.

Mahavir. 1986. Treatment of Samāsa in Pāṇini. *Annals of the Bhandarkar Oriental Research Institute*, 67(1-4):147–158.

Sivaja S Nair and Amba Kulkarni. 2010. The knowledge structure in amarakośa. In *Sanskrit Computational Linguistics*, pages 173–189. Springer.

Vivi Nastase, Jelber Sayyad-Shirabad, Marina Sokolova, and Stan Szpakowicz. 2006. Learning noun-modifier semantic relations with corpus-based and wordnet-based features. In *Proceedings of the National Conference on Artificial Intelligence*, volume 21, page 781. Menlo Park, CA; Cambridge, MA; London; AAAI Press; MIT Press; 1999.

Bhagyalata Pataskar. 1996. Some Observations about the Compound Structure of Aṣṭādhyāyī. *Annals of the Bhandarkar Oriental Research Institute*, 77(1-4):121–131.

Pavankumar. 2015. *Sanskrit Compound Generation: With a Focus on the Order of the Operations (Doctoral Dissertation)*. University of Hyderabad.

F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot, and E. Duchesnay. 2011. Scikit-learn: Machine learning in Python. *Journal of Machine Learning Research*, 12:2825–2830.

Anne Schiller. 2005. German compound analysis with wfsc. In *International Workshop on Finite-State Methods and Natural Language Processing*, pages 239–246. Springer.

Diarmuid O Séaghdha and Ann Copestake. 2013. Interpreting compound nouns with kernel methods. *Natural Language Engineering*, 19(03):331–356.

Diarmuid Séaghdha. 2009. Semantic classification with wordnet kernels. In *Proceedings of Human Language Technologies: The 2009 Annual Conference of the North American Chapter of the Association for Computational Linguistics, Companion Volume: Short Papers*, pages 237–240. Association for Computational Linguistics.

Manish Sinha, Mahesh Reddy, and Pushpak Bhattacharyya. 2006. An approach towards construction and application of multilingual indo-wordnet. In *3rd Global Wordnet Conference (GWC 06), Jeju Island, Korea*.

Michael Strube and Simone Paolo Ponzetto. 2006. Wikirelate! computing semantic relatedness using wikipedia. In *AAAI*, volume 6, pages 1419–1424.

Jörg Tiedemann. 2005. Improving passage retrieval in question answering using nlp. In *Portuguese Conference on Artificial Intelligence*, pages 634–646. Springer.

Ben Verhoeven, Walter Daelemans, Menno Van Zaanen, and Gerhard Van Huyssteen. 2014. Automatic compound processing: Compound splitting and semantic analysis for afrikaans and dutch. *ComAComA 2014*, page 20.

Marion Weller, Fabienne Cap, Stefan Müller, Sabine Schulte im Walde, and Alexander Fraser. 2014. Distinguishing degrees of compositionality in compound splitting for statistical machine translation. In *Proceedings of the First Workshop on Computational Approaches to Compound Analysis (ComAComA 2014)*, pages 81–90.

Ke Zhai, Jordan Boyd-Graber, and Shay B Cohen. 2014. Online adaptor grammars with hybrid inference. *Transactions of the Association for Computational Linguistics*, 2:465–476.

Patrick Ziering and Lonneke van der Plas. 2016. Towards unsupervised and language-independent compound splitting using inflectional morphological transformations. In *Proceedings of the NAACL 2016*.

Patrick Ziering, Stefan Müller, and Lonneke van der Plas. 2016. Top a splitter: Using distributional semantics for improving compound splitting. *The 12th Workshop on Multiword Expressions, ACL 2016*, page 50.