Morphological Disambiguation of Classical Sanskrit

Oliver Hellwig

University of Düsseldorf, Germany

Abstract. Sanskrit, the "sacred language" of Ancient India, is a morphologically rich Indo-Iranian language that has received some attention in NLP during the last decade. This paper describes a system for the tokenization and morphosyntactic analysis of Sanskrit. The system combines a fixed morphological rule base with a statistical selection of the most probable analysis of an input text. After an introduction into the research history and the linguistic peculiarities of Sanskrit that are relevant to the task, the paper describes the present architecture of the system and new extensions that increase its accuracy when analyzing morphologically ambiguous forms. The algorithms are tested on a gold-annotated data set of 3.587.000 words.

1 Introduction

Sanskrit, an Old Indo-Aryan (OIA) language, whose first texts date back to around 1.500 BCE, has produced one of the largest premodern text corpora in the world. Taking an oversimplifying approach, there are two relevant linguistic layers of Sanskrit. The earlier layer contains the Vedic corpus that may have been created between 1.500 and the middle of the first millenium BCE and that has probably preserved a spoken form of Sanskrit.[1] The later layer of Classical Sanskrit is written in a language that is largely regulated by the famous grammar of Pāṇini (details in Section 2). The term "Sanskrit" only refers to this classical stage of the language throughout this paper.

While the oldest Vedic layer has been the subject of numerous detailed linguistic studies, the later layer of Classical Sanskrit, which contributes the vast majority of transmitted texts, has been studied only scarcely from a linguistic point of view. There are several reasons for this inequally distributed research interest, some of them originating in the fascination of traditional philology for the old, "authentic" layers of the language. In addition, the early codification of Sanskrit in the grammar of Pāṇini has led to the assumption that Classical Sanskrit is mainly interesting for the content it produced, but not for its linguistic features. This view will not hold stand when the numerous interactions with other South-Asian languages are taken into account (Section 3). Another obstacle that prevented the large-scale linguistic study of Classical Sanskrit is

[1] Bloch gives an introduction into the linguistic history of Sanskrit [3]. More details about the Vedic layer are found in [35].

the mere size of the literature it produced. A computer based approach will facilitate access to this corpus and its underlying linguistic structures, and may be helpful in moving scholarly attention to the numerous linguistic peculiarities of Classical Sanskrit. This paper describes a lexical and morphological parser for Classical Sanskrit whose output allows a strictly corpus-based, data driven approach to this language.

The paper is organized as follows. Section 2 gives an overview of formal systems of Sanskrit grammar and modern NLP related research in processing Classical Sanskrit. Section 3 summarizes some of the central issues that complicate the automatic analysis of Sanskrit. Section 4 describes the basic architecture of the tokenizer and morphological analyzer. Extensions of the basic system and improvements in the accuracy and coverage of the morphological analysis are reported in Section 5. Section 6 summarizes the paper.

2 Previous research and resources

Sanskrit has a long tradition of formal language description and analysis that predates any modern Western attempts in this field by millenia. This tradition was started by the grammarian Pāṇini, who probably lived around 350 BCE in Northwestern India (a general overview in [30]). His grammar Aṣṭādhyāyī ("eight [*aṣṭan*] chapters [*adhyāya*]") provides an extremely concise description of a late Vedic level of Sanskrit and may reflect a dialect of Sanskrit spoken at his time (details of the discussion in [5]). This conciseness was made possible by introducing methods such as thematic roles, rewrite rules, abstract derivation levels, and pre-concepts of phonemes and morphemes, all of which are crucial for contemporary linguistics [14].[2] Pāṇini's methods of language description were expanded and refined by his followers in works such as the Mahābhāṣya (Patañjali, 150 BCE) or the Siddhāntakaumudī (16. c. CE; refer to Scharfe for a history of grammatical research in India [30]).

Given the sophisticated formal methods that are provided by the Indian grammatical tradition and that were "rediscovered" in Western linguistics only in the 20th century, it is not surprising that some researchers try to transpose the Pāṇinian system of morphosyntactic analysis, more or less directly, into an NLP tool. These approaches frequently face problems with overgeneration (refer to Kulkarni's study from the area of phonetics [18]) and with the order in which the rules of the Aṣṭādhyāyī need to be applied.[3] Mishra handles these problems

[2] In a recent research project, the Aṣṭādhyāyī has been fully annotated on the morphological, lexical and word-semantic level to make it easier accessible for Western researchers without knowledge of Sanskrit [26]. A web platform that gives access to this database is available at `http://panini.phil-fak.uni-duesseldorf.de/panini/`.

[3] The rules of the Aṣṭādhyāyī are not given in the order in which they need to be applied for generating a valid Sanskrit word. Instead, it is generally assumed that their order minimizes the resulting rule base. The Indian grammar uses the concept of *anuvṛtti* ("following") rules for regulating the order in which rules and their elements are applied. These rules are not part of the text of the Aṣṭādhyāyī, but are recorded

by reformulating the rules of the Aṣṭādhyāyī in terms of set theory [22]. Huet [13] and Kulkarni [17] combine formal methods from the Aṣṭādhyāyī with a statistical scorer. Mittal estimates the probability of Sandhi splits from a parallel corpus of sandhied and unsandhied texts [23].[4] He combines a finite state automaton built from the parallel corpus, estimations of word frequencies, and a morphological analyzer with a scoring function that calculates a joint lexical and phonological weight for a given analysis of a Sanskrit sentence. The author reports that his system selects the best split of a given Sanskrit string in 92.8% percent of all cases. Hellwig presents a statistical lexical and morphological analyzer [9], but misses the opportunity to give reliable performance data of this system [11].

In recent years, NLP has become increasingly interested in the processing of morphologically rich languages, and Sanskrit fits well into this extended scope of research. From among the more popular languages, Hebrew has similar challenges for NLP as Sanskrit. According to Adler, Hebrew has a rich and highly ambiguous morphology, and its morphological tag set is by far more comprehensive than that of English [2]. Shacham and Wintner combine results of classifiers that are each specialized on a subset of all morphological tags for Hebrew, and report an improvement over the baselines found in former papers on the topic [31]. Yuret describes an algorithm for the morphological disambiguation of Turkish, which has a similar proportion of morphologically ambiguous forms as Sanskrit [36]. By training decision lists with local, non-lexical features, the author achieves a disambiguation accuracy of nearly 96% on a small test set. Lee proposes a graphical model for the joint morphological analysis and parsing of morphologically rich languages such as Latin and Czech [20], and obtains slight improvements over a baseline generated by separate training on the two tasks.

Basic computational resources were missing almost completely when the first versions of the system presented in this paper were created. The relational lexicographic database (Section 4.1) had to be extracted from a digitized version of the dictionary of Monier-Williams [24]. Although the majority of entries could be converted automatically by using regular expressions, preparing a usable database version of this dictionary still required considerable manual correction due to inconsistencies in the formatting of the original dictionary and of its digitized text version. Moreover, the Monier-Williams was designed as a typical reference-oriented printed dictionary. This means that it recorded numerous compounds with purely compositional meaning (e.g., *mahāgiri*, "high mountain", consisting of the adjective *mahā* "high" and the noun *giri* "mountain"). Such lexicon entries may be useful in printed editions, but complicate the lemmatization that focuses on decomposed primitives.

Similar problems are also encountered when building computational processors for other premodern Indian languages as, for example, for Pāli [15] or for Old Marathi, for which a morphological analyzer is currently developed. Lack of

– and heavily discussed – in the commentary literature; refer to [5, 187ff.] for details about rule order in the Aṣṭādhyāyī, and to [27] for the proof of minimality in a subset of Pāṇinian rules.

[4] Refer to page 3.3 for the phonological phenomenon of Sandhi.

resources further pertains to large, consistently formatted digital text corpora[5] and to gold annotated training and test data from any level of linguistic analysis, although the research community has begun to compile such resources on a small scale during the last years [32].

3 Linguistic background and challenges

Sanskrit poses a number of challenges for the automatic linguistic analysis, some of which are not found in the languages typically examined in current NLP. This section gives an overview of the most important of these phenomena.

3.1 Morphology

Sanskrit has a rich and partly ambiguous morphology. Nouns and adjectives inflect for three numbers (SG., DU., and PL.), eight cases (NOM., ACC., VOC., GEN., DAT., INS., ABL., LOC.), and a frequent and productive stem form (CO.) that is used to create compounds.[6] The finite forms of the verbal system differentiate between present (indicative present, imperative, imperfect), future (future, conditional, periphrastic future), perfect, and seven classes of aorists. Each of the tenses builds forms for three persons, three numbers, and the active and medium voices. These forms are supplemented by infinite forms such as fully declinable participles of the present, future, perfect, and of the past, and indeclinable forms (infinitive, absolutive). With a few exceptions, the derivation of nominal and verbal forms from their respective stems is regular. The complex, interacting rules that guide these derivational processes are formulated in the Aṣṭādhyāyī.

While the rich inflexion of Sanskrit, in principle, promotes a reliable morphological analysis of Sanskrit texts, there are a few high-frequency morphemes

[5] The GRETIL web repository (`http://gretil.sub.uni-goettingen.de/`) contains less than 20 million strings. Several of the texts are not usable for automatic processing due to excessive formatting of their editors, as described in Section 3.6.

[6] The following abbreviations are used in this paper: NOM.: nominative; ACC.: accusative; INS.: instrumental; DAT.: dative; GEN.: genitive; LOC.: locative; VOC.: vocative; CO.: compound; SG.: singular; DU.: dual; PL.: plural; MSC.: masculine; FEM.: feminine; NEU.: neuter; IND.: indeclinable; PRES.: present; IMPF.: imperfect; PERF.: perfect tense; PROH.: prohibitive (a kind of imperative that is only used in negated phrases); PASTPART.: past participle, frequently with a passive sense; PRESPART.: present participle

Ambiguities in a morphological analysis are expressed by a regex-style notation, with | denoting the operator OR and round brackets a set of options. So, (NOM.|ACC.|VOC.)PL.NEU.means that a form is a neuter plural either in nominative or accusative or vocative.

The plus operator + is used to separate elements of compounds, the ampersand sign & to indicate Sandhi at word boundaries (Section 3.3).

Further abbreviations: `tri`: trigram based model for morphological disambiguation; `crf`: Conditional Random Fields; `me`: Maximum Entropy

whose phonetic ambiguity complicates the morphological tagging. Most important among them is the word final sequence *-am*, which marks NOM., ACC., and VOC.SG. of the noun class *-a* neuter, ACC.SG. of the noun class *-a* masc. and ACC.SG. of nominal stems that end with consonants. Such morphological ambiguities gain in importance through the phenomenon of *bahuvrīhi* compounding, which can change the gender of a nominal compound (refer to Section 3.2). In addition, the phonetic process of Sandhi can "obfuscate" the output forms of the inflectional endings (Section 3.3).

3.2 Compounding

Although compound formation does not seem to interfere with morphological analysis at first view, the long ranges covered by many compounds, e.g., in philosophical and scientific texts, complicate the estimation of transition weights for sequence based algorithms. In addition, Sanskrit knows a class of compounds called *bahuvrīhi* ("(a person who has) much rice") that produce adjectives with a possessive meaning.[7] During this compounding process, the original nominal compound is transformed into an adjective, and the gender of the final member of the compound is adopted to the gender of the noun the *bahuvrīhi* refers to. In example 1, the phrase *pāpakarmabhiḥ* can be interpreted as a "default" compound ("by bad actions", INS.PL.NEU., *tatpuruṣa* interpretation) or, more correctly, as a *bahuvrīhi* adjective refering to the anaphoric masculine pronoun *taiḥ*. During adjectivization, the gender of the compound is changed from the original neuter to the masculine of the pronoun:

(1) *niḥśeṣo* *hi*
without remainder-NOM.SG.MSC. because-IND.
kṛto *vaṃśo* *mama*
make-PASTPART.NOM.SG.MSC. family-NOM.SG.MSC. my-GEN.SG.
taiḥ *pāpakarmabhiḥ*
they-INS.PL.(MSC.|NEU.) bad-CO.+actions-INS.PL.(MSC.|NEU.)
"Because my family was completely destroyed by these bad persons (lit.: by them, who have bad actions), ..." (Mahābhārata, 13.31.25)[8]

3.3 Phonetic rules (Sandhi)

Sanskrit uses a large set of euphonic rules called Sandhi ("connection"), whose formulation is another major contribution of the Aṣṭādhyāyī. Most of these rules

[7] Note that the word *bahuvrīhi* is itself an example of a *bahuvrīhi* compound. In its "default interpretation" as a so-called *tatpuruṣa* ("his man", an instance of relational compounding) compound, it means just "much rice".

[8] From a purely grammatical point of view, the sentence can also be translated as "... destroyed by these bad actions." Numerous references of the *bahuvrīhi* solution with unambiguous case endings (e.g., in NOM.PL.MSC.) make the proposed interpretation much more plausible.

combine two phonemes into one or two other phonemes to produce a "smoother pronounciation".[9] Sandhi occurs inside of words during the derivational process and at the boundary between words during the construction of the sentence from the inflected lemmata. For an example of how word boundary Sandhi works, consider the three inflected Sanskrit words *tān* (ACC.PL.MSC. of pronoun *tad*, "they"), *cet* (IND., "if"), and *jayati* (3rdSG.PRES. of verb *ji*, "to win"). The word-final *n* of *tān* and the initial *c* of *cet* produce the Sandhi *ṃśc*, while the word-final *t* of *cet* and the initial *j* of *jayati* produce *jj*. Using these two Sandhis, the three separate forms are merged into a single string *tāṃścejjayati* ("if (s)he overcomes them").

Sandhi tends to "obfuscate" morphological terminations. This phenomenon is, for instance, observed in the phrase *pāṇḍavā api* (NOM.PL.MSC. of the noun *pāṇḍava*, "name of a famous family", and *api*, IND., "also"), where the original word final letter *ḥ* of *pāṇḍavāḥ* has been elided through boundary Sandhi. A morphological analyzer must be able to reconstruct the pre-sandhied form *pāṇḍavāḥ* based on the right context *api*, before it starts the actual morphological analysis.

While, theoretically, Sandhi must be used whenever applicable, real texts show a lot of divergence from this rule. Sanskrit epics, for example, do not adhere strictly and consistently to these rules ("Epic Sandhi").[10] Much more frequently, however, deviations from these rules may have been caused by errors of the author or the scribe of a text due to an insufficient knowledge of these euphonic rules. It should be needless to emphasize that Sandhi rules complicate the automatic analysis of Sanskrit massively, because they introduce ambiguity into tokenization (refer to Section 4) and tend to overgenerate possible analyses of a string. A working system used for real texts must be able to cope with the full set of standard Sandhi rules, but also with irregular situations as found in the epics, and it should not interrupt analysis when a Sandhi rule has not been applied.

3.4 Word order

Word order is another problematic area, because it is explored intensively in NLP models for English, while its role in South-Asian languages is far less prominent. Staal claims that there are virtually no rules for word order in Sanskrit [33], without supporting his theory with quantitative data. Gillon, who claims that unmarked Sanskrit sentences show a tendency for verb-finality [8], has certainly arrived at a more realistic picture of word order in Sanskrit prose texts. Although prose texts seem to prefer a certain word order, it is by far not as strictly

[9] Though slightly outdated, the grammar of Stenzler still provides a good introduction into Sanskrit Sandhi rules [34, 3ff.].

[10] Refer to [25, 1ff.] for a detailed linguistic description with several examples. Brockington locates the epics, especially the Mahābhārata, in a continuum "of dialects and language registers from classical or Pāṇinian Sanskrit at one end to colloquial MIA [Middle Indo-Aryan] at the other" [4, 83] and makes this linguistic situation responsible for the irregular application of Sandhi in epic texts.

regulated as, for instance, in English. Features exploring the order of words may, therefore, not contribute as significantly to the accuracy of the morphological analysis as they do for English [7].

3.5 The lexicon

Classical Sanskrit has built up an extremely rich vocabulary that covers topics as diverse as religion, philosophy, science (grammar, medicine, mathematics), poetry, and popular tales. Apart from OIA terms and their derivatives, Sanskrit included words from several "substrate languages" such as Dravidian, Middle (Prākṛts, Apabhraṃśa) and even New Indo-Aryan languages.[11] Because Sanskrit was used as a literary language by most authors, and because most authors were well aware of its long-standing literary tradition, one can observe a tendency to revive older strata of the vocabulary and to incorporate them into new texts.[12] From the perspective of NLP, the richness of the vocabulary contributes to the sparsity of lexicographic data, and complicates the estimation of lexical parameters of ML models.

3.6 Orthography

A closely connected area is the general lack of a strict orthography and the missing reliability of punctuation. Many algorithms for language analysis require full sentences as the basic input units. Sanskrit uses a vertical line called *daṇḍa* ("staff", "stick") for marking breaks in the metrical structure and in the general discourse structure of texts. However, the *daṇḍa*, which is rendered by a backslash (/) in this paper, frequently does not coincide with the termination of complete sentences. Example 2 contains two *daṇḍa*s that mark the end of the metrical units. Here, only the second *daṇḍa* after *adhitiṣṭhati* coincides with the termination of a full sentence, while the first one separates the verbal form *uvāca* from its subject (Mahābhārata, 13.28.10):

(2) *... gardabhī putragṛddhinī /*
... she-ass-NOM.SG.FEM. son-CO.+caring-NOM.SG.FEM. *daṇḍa*
uvāca mā śucaḥ putra
say-3rdSG.PERF. not-IND. worry-PROH.SG. son-VOC.SG.MSC.
caṇḍālastvādhitiṣṭhati /
Caṇḍāla-NOM.SG.MSC.&you-ACC.SG.&ride-3rdSG.PRES. *daṇḍa*
"The she-ass, who was sorried about her son, said: 'Don't worry, son! A Caṇḍāla[13] is riding on you.' "

[11] Emeneau describes the basic parameters of the interaction between Indo-Iranian and Dravidian languages [6]. A quantitative overview of the major influences that is based on Mayrhofer's etymological dictionary [21] is given in [10].
[12] A quantitative evaluation of the reuse of Pāṇinian vocabulary is presented in [12].
[13] A member of a low caste

Algorithms for detecting the true sentence breaks in Sanskrit have not yet been developed. NLP systems must either rely on sentences with manually marked borders, which involves time consuming manual annotation, or must be able to handle the lack of orthographic information appropriately. To increase the readability, the term "sentence" will, nevertheless, denote a sequence of strings that is terminated by a *daṇḍa* in this paper. Such a "sentence" may thus contain parts of a sentence or of sentences, a full sentence, or several concatenated sentences.

At an even more basic level, traditional editions of Sanskrit texts insert blank spaces between non-mergable strings sparingly, if at all. On the contrary, Western editors frequently even resolve boundary Sandhis to increase the readability of the text, thereby producing syllable sequences that are invalid from an Indian point of view. Therefore, the text of the second sentence in Example 2 could also be written as *uvācamāśucaḥputracaṇḍālastvādhitiṣṭhati* (traditional Indian style) or *uvāca mā śucaḥ putra caṇḍālas tvā adhitiṣṭhati* (Western style).

4 Architecture of the system

This section gives an overview of how a Sanskrit sentence is analyzed in the proposed system. Because the core functionality of the tagger has been described in [9], this section only summarizes the central components (4.1) and processing steps (4.2). Section 4.3 gives a short evaluation of the algorithm for joint tokenization and lemmatization.

4.1 Basic components

The system consists of five core components.

1. The lexical database stores lemmata, their grammatical categories, meanings, word semantic information, and inflected verbal forms. The database currently contains 174.190 distinct lemmata with 313.725 meanings and 104.811 connections to a word semantic repository that is derived from OpenCyc [1].
2. The corpus stores the Sanskrit texts along with their lexicographic, morphological and word semantic gold annotations.[14] There are 273 texts in the corpus database, 69 of which are completely annotated. The texts contain 2.674.000 strings that are split into 3.587.000 lexical tokens with morphological gold annotations. The corpus data are used to train statistical models for lexico-morphological analysis and disambiguation. As can be seen in Table 1, the corpus mainly contains texts from the epic-Purāṇic traditions[15] and

[14] As these data are only checked by one annotator and have not been adjudicated, they should rather be called semi-gold annotations.

[15] The Mahābhārata and the Rāmāyaṇa are the two central epic texts written in Sanskrit. The term Purāṇa ("old (story)") denotes a group of works dealing with virtually everything; refer to Rocher for an introduction [29].

from science, including medicine, alchemy, and gemmology. The type-token-ratios (TTR)[16] vary strongly between the topic levels, with the highest value not surprisingly found in lexicography.

3. The linguistic models comprise (1) a hard-coded rule base for Sandhi resolution and for determining morphological categories of nouns, and (2) learned parameters of the statistical algorithms that are created using the training data extracted from the corpus. Sanskrit nouns are inflected by adding terminations to the roots of words in a similar way as in Latin or Ancient Greek. Morphological analysis of nominal forms is performed by removing possible inflectional suffixes from a string at runtime, and looking up the remaining word root in the noun section of the lexical database. Inflected verbal forms have a special role in the system. They are synthesized automatically for each (prefixed) verbal root, checked manually, and then stored in the verb section of the lexical database along with their morphological information (tense, mode, person, number). At runtime, morphological analysis of verbal forms is performed by looking up an input string in the verb section of the database, and returning the associated morphological information, if it is found. Although Sanskrit verbal forms can be analyzed using a rule based system (refer, for instance, to [22]), the current solution was chosen to speed up the creation of the initial system, because a thorough handling of the verbal forms would have required a formalization of large ranges of the Aṣṭādhyāyī.
4. The tag set consists of tags for indeclensibles, nouns and verbal forms. The indeclensible tag covers particles, interjections, and conjunctions. The noun tags represent substantives and adjectives in one of the three genders, three numbers and eight cases, plus the 3 respective stem forms. The tags for the verbal system differentiate between present, future, and past tenses in the three persons and numbers. The verbal tags are thus less fine-grained than the nominal ones: While each morphological category of nouns is mapped to its own tag, tags for verbal forms focus on person and number distinction, but differentiate only roughly between the tenses (refer to Section 3.1 for the tense system of Sanskrit). Because the morphological ambiguity in the verbal system is much lower than in the nominal one, this design decision reduces the number of tags and simplifies the task of automatic morphological analysis.
5. The linguistic processor uses the models and the lexical database to analyze a sentence. The resulting analysis can be checked manually (creation of gold-annotated data) and stored in the corpus to increase the size of the training database.

[16] The TTRs found in the third column of Table 1 are obtained by calculating the TTRs for each text, and then averaging these values over the topic levels. Because text lengths have not been used as normalizing factors, the TTRs of underrepresented topic levels such as *śruti* or Buddhist literature are most probably too high.

Topic	Perc.	TTR
Buddhist	1.51	0.34
darśana ("philosophy")	1.62	0.3322
dharma ("law")	4.07	0.4095
Grammar	0.52	0.314
Epic-Purāṇic	55.33	0.2145
Lexicography	2.13	0.5966
Poetry	4.88	0.4308
Religious	4.37	0.414
Science	23.8	0.347
śruti (late Vedic texts)	1.78	0.3576

Table 1. Composition of the corpus, grouped by topics. Perc.: percentage of lemma tokens in texts with a given topic in relation to the number of all lemma tokens in the corpus; TTR: averaged type-token-ratio for each topic.

4.2 Tokenization and morphological analysis

The algorithm that performs tokenization, lemmatization, and morphological analysis works in two main steps. The first step tries to generate the correct lemmatization of the input text, which includes Sandhi resolution and compound splitting. The second step performs a fine-tuning of the morphological analysis of the highest scoring lemmatization obtained in the first step. The disambiguation methods dealt with in this paper are part of the second step.

In the first step, each input string s is scanned from left to right (refer to [9] for details). If a (combination of) phoneme(s) at position i in s is found in the list of possible Sandhi results, s is tentatively split at i, and its left part is analyzed lexically and morphologically after its final Sandhi has been undone. If the left part is a valid Sanskrit form, the right part of s is analyzed in the same way. If this recursive algorithm reaches the end of the string, all proposed analyses are inserted in a hypothesis lattice. After the full input sentence has been processed, Viterbi decoding is used to find the most probable sequence of lexical tokens in the resulting lattice.

For Viterbi decoding, all morpho-lexical analyses LM_j are extracted for each possible split string j. The split string *vanam*, for example, produces the three analyses LM_{j1} = (NOM.SG.NEU., *vana*, "forest"), LM_{j2} = (ACC.SG.NEU., *vana*, "forest"), and LM_{j3} = (VOC.SG.NEU., *vana*, "forest"). Although the first step is concerned with lemmatization, morphological information is included at this point because it helps to distinguish between different lexical derivations of ambiguous surface strings such as *te*, which can be derived from *tvat* ("you", DAT. or GEN.SG.) or *tat* ("this", NOM.PL.MSC.). The decoding process uses the probabilities of bigrams (LM_{j-1}, LM_j) whose frequencies are estimated from the annotated corpus and smoothed using the method proposed by Kneser and Ney [16]. To make analysis paths of different lengths comparable to each other, the

sums of logarithmized transition probabilities resulting from Viterbi decoding are divided by the lengths l of their paths, which is equivalent to taking the lth root from the unlogarithmized path probabilities. When T denotes the set of all possible tokenizations t of a given sentence, the winning tokenization of a sentence fulfills the condition $\underset{t \in T}{\operatorname{argmax}} \frac{1}{|t|} \left(\sum_{i=1}^{|t|} \log p(LM_i | LM_{i-1}) \right)$.

4.3 Evaluation of the tokenization

The quality of the joint tokenization and lexical disambiguation was assessed by calculating the Levenshtein edit distance between gold sequences of lexemes from the corpus and the corresponding silver sequences of lexemes generated by the system. For testing, a set of 10.000 sentences was drawn randomly from the corpus. The parameters of the tokenizer were re-estimated from the remaining part of the corpus, and the retrained model was applied to the 10.000 holdout sentences.

As can be seen in Table 2, the model produces the correct lexical tokenization for 94.4% of the holdout sentences, and one error for 3.3% of them. To get an idea of how the topics of the texts influence tokenization accuracy, the numbers of edit operations were grouped by the texts from which the sentences were drawn, and some of the most voluminous texts were labeled with the same coarse-grained domain tag set that was used for creating Table 1. Table 3 shows that tokenization works well for texts from the epic-Purāṇic literature. These texts are mostly written in an easy, unpretentious style, they share a large core vocabulary, and they constitute one of the thematic focus areas of the corpus (refer to Table 1). A similar picture emerges for the Āyurvedic (medical) texts, which are, however, linguistically and especially lexicographically much more demanding than the epic-Purāṇic literature, as indicated by the higher type-token-ratio shown in Table 1. The alchemical texts, in contrast, show higher error rates, although the alchemical tradition has actually developed from Āyurveda. This fact can possibly be explained by rare lexemes used in these texts (e.g., in the Rasādhyāya, a late text showing strong influences from NIA languages) and by the low literary quality of many alchemical texts. The Aṣṭādhyāyī produces the worst tokenization score in the evaluation, because the text uses Sanskrit as a metalanguage for encoding its grammar, and some of these metalinguistic phenomena are not handled by the regular processing pipeline of the program to avoid overgeneration. In addition, Table 1 shows that the grammatical texts constitute the smallest thematic section of the corpus. Adding more texts from the grammatical tradition may improve the unsupervised tokenization of the Aṣṭādhyāyī.

The evaluation has shown that the lemmatization produces acceptable results for texts from domains for which enough training data is available. The morphological analysis resulting from the first step of the decoding process, however, is frequently wrong. Therefore, another level of morphological Viterbi decoding is added for the highest-scoring lexical path, using only smoothed trigrams of morphological tags, but no lexical information. Column 3 (`tri`) of Table 5 re-

	Number of edits			
	0	1	2	≥ 3
≤ 5	14.47	0.19	0.26	0.04
$6-10$	75.63	2.91	1.43	0.28
$11-15$	3.58	0.16	0.17	0.01
≥ 16	0.75	0.04	0.04	0.03
$\sum$	94.43	3.3	1.9	0.36

Table 2. Length of sentences (rows) and numbers of edit operations needed to transform silver in gold tokenization (Levenshtein), tested on a holdout set of 10.000 sentences. Values in percent of all 10.000 sentences.

Text	Number of tested sent.	Number of edits 0	1	2	≥ 3	Domain
Mahābhārata	3071	94.86	3.35	1.69	0.1	Epic-Purāṇic
Rāmāyaṇa	774	95.61	2.71	1.68	0	Epic-Purāṇic
Liṅgapurāṇa	395	93.92	4.05	1.52	0.51	Epic-Purāṇic
Suśrutasaṃhitā	347	98.27	0.86	0.86	0	Science (med.)
Aṣṭāṅgahṛdayasaṃhitā	289	94.12	2.77	2.42	0.69	Science (med.)
Ānandakanda	243	92.18	4.94	2.88	0	Science (alchem.)
Bṛhatkathāślokasaṃgraha	189	95.77	4.23	0	0	Poetry (narr.)
Carakasaṃhitā	172	95.93	2.33	1.16	0.58	Science (med.)
Rasaratnākara	163	93.87	3.07	1.84	1.23	Science (alch.)
Rājanighaṇṭu	141	91.49	4.96	2.84	0.71	Lexicography
Manusmṛti	122	95.08	3.28	0.82	0.82	*dharma*
Viṣṇusmṛti	75	90.67	2.67	5.33	1.33	*dharma*
Hitopadeśa	58	93.1	3.45	3.45	0	Poetry (narr.)
Rasendracintāmaṇi	36	91.67	5.56	2.78	0	Science (alch.)
Aṣṭādhyāyī	35	57.14	14.29	22.86	5.71	Grammatical
Rasādhyāya	24	87.5	8.33	4.17	0	Science (alch.)

Table 3. Number of edit operations for individual texts; refer to Table 1 for the topic labels. Abbreviations: med.: medical (Āyurveda); alchem.: alchemical (*rasaśāstra*); narr.: narrative

ports the global accuracy of this approach, grouped by the number of different proposals per lexical item. As could be expected, the accuracy of this approach decreases with the number of morphological options per word. The next section describes experiments for improving the performance of the system for ambiguous morphological analyses.

5 Improvements and evaluation

As noted in Section 4.2, words in the best lexical path can be annotated with more than one morphological solution. Column 2 of Table 5 shows that such ambiguous solutions occur for approximately $100\% - 58\% = 42\%$ of all words in the test set. Resolving these ambiguities is, therefore, crucial for the accurate morphological tagging of Classical Sanskrit. This section describes a set of experiments that aim at improving the morphological analysis of ambiguous cases. Subsection 5.1 sketches the feature extraction. Subsection 5.2 describes which ML models are used for learning. Section 5.3 describes how the test and training sets are created. Results and error analysis are presented in 5.4

5.1 Features

The features are built from the two pieces of information that are available after the first stage of linguistic analysis has been completed: (1) the lexical information about each word in the highest scoring path, and (2) the morphological solutions that the analyzer has proposed for each lexeme in this path. The proposals for the morphological analysis are merged into an ordered set of distinct morphological classes. This merged set is used as a single feature M of the word under consideration. If, for example, the analyzer has detected that a word can be (NOM.|ACC.)PL.(MSC.|FEM.), these four proposals are unfolded into the single feature ACC.PL.FEM.|ACC.PL.MSC.|NOM.PL.FEM.|NOM.PL.MSC., using alphabetical ordering of the names of the morphological tags.

Before extracting the features from the training set, frequency thresholds are applied to remove lexical items L with a total frequency of less than 10 and combined morphological solutions M that occur with a total frequency of less than 50 in the training corpus. If only one morphological solution is proposed for a word (58% of all cases), this solution is assumed to be the correct one, and the true morphological class of this word is replaced by a dummy variable to reduce the complexity of the training process.[17] The full feature vector v_i for the word at position i is the union of all lexical and morphological features of words with a maximal distance of 3 from i (context window).[18] Consider, as an example, the trivial sentence *sa vanaṃ gacchati* ("he goes into

[17] The one-solution case predicts the correct morphological category in about 99.8% of all cases. The errors are caused by irregular word forms.

[18] The parameter 3 for the window size was chosen after comparing disambiguation results for window sizes between 1 and 7. Window sizes above 3 did not consistently increase the accuracy, but required higher training times.

the forest") with a context window of size 1. The first step of the analysis has proposed the following highest scoring sequence: (*sa* = (NOM.SG.MSC., lemma *tad*, "he")), (*vanam*[19] = ((NOM.|ACC.|VOC.)SG.NEU., *vana*, "forest")), (*gacchati* = ((3rdSG.PRES.|(LOC.SG.(MSC.|NEU.), PRESPART.)), *gam*, "to go")). The first word has the local features L=*tad* and M=(NOM.SG.MSC.), the second word has L=*vana* and M=(NOM.SG.NEU.|ACC.SG.NEU.|VOC.SG.NEU.), and the third one L=*gam* and M=(3rdSG., past tense|LOC.SG.MSC.|LOC.SG.NEU.). The target classes, on which the classifiers are trained, are the correct morphological tags according to the gold standard for the ambiguous solutions, or the dummy variable X in case of unambiguous solutions. The full feature vectors and the target classes for each word are given in Table 4, where the numeric subscripts denote the distance from the respective focus word at position i. These full vectors are the input for the ML methods that are described in Section 5.2.

	1	2	3
Word	*sa*	*vanam*	*gacchati*
Local features L	*tad*	*vana*	*gam*
Local features M	NOM.SG.MSC.	(NOM.|ACC.|VOC.)SG.NEU.	3rdSG.PRES. ...
Full feature vector	{X}	$\{L_{-1}$ = tad, L_0 = vana, L_{+1} = gam, M_{-1} = NOM.SG.MSC., M_0 = (NOM. ..., M_{+1} = 3rdSG.PRES. ...}	$\{L_{-1}$ = vana, L_0 = gam, $L_{+1} = \emptyset$, M_{-1} = (NOM. ..., M_0 = 3rdSG.PRES. ..., $M_{+1} = \emptyset\}$
Target class	X	ACC.SG.NEU.	3rdSG.PRES.

Table 4. Features and target classes for the sentence *sa vanaṃ gacchati*. X denotes the dummy variable used for words with only one morphological analysis. (NOM. ...: local features of *vanam*, i.e. (NOM.|ACC.|VOC.)SG.NEU.; 3rdSG.PRES. ...: local features of *gacchati*, i.e. 3rdSG.PRES.|(LOC.SG.(MSC.|NEU.))

5.2 Models

The ML models that are used to resolve the morphological ambiguities must be able to handle high-dimensional feature vectors of varying size from a nominal scale. The size of the feature vectors is variable because (1) all lexical and morphological context information of words with an unambiguous morphological analysis is replaced by the dummy variable X, and (2) lexical and morphological context information can be pruned away when their frequencies are below the thresholds given in Section 5.1. Two models that fulfill these requirements are Maximum Entropy Classifiers (ME, [28])[20] and Conditional Random Fields

[19] The final Sandhi *ṃ* has been transformed into the pausa form *m*.

[20] Used in the Java implementation of the OpenNLP package; settings: smoothing factor: 0.001, 100 iterations.

(CRF, [19]).[21] While ME is trained and evaluated on single words, CRF is a sequential model that takes information about the preceding word into account. Because the local morphological and lexical context presumably influences the analysis of a word, it may be expected that the sequential CRF performs better than the non-sequential ME, even if they are trained with the same data.

5.3 Test design

Training and test sets are constructed by first extracting those sentences from the corpus whose gold analysis contains between 2 and 20 lexical items. Longer sentences are excluded to limit the time needed for data creation. The resulting set S consists of approximately 475.000 sentences. Each sentence in S is tokenized by using the first step of the analysis algorithm described in Section 4.2. If the lexical silver annotation proposed after the first step is identical with the lexical gold annotation from the corpus, the features described in Section 5.1 are extracted from the sentence, and the sentence is added, along with its features, to a set F. This set F is split randomly into a training set containing 95% and a test set containing 5% of all sentences in F.

It should be kept in mind that the final test and training sets contain only sentences that have been analyzed correctly on the lexical level. This restriction was introduced to simplify the creation of the test data, but it could have a negative effect when the final system is confronted with real-world sentences whose first-step lemmatization contains errors. Another caveat concerns the testing method. Because considerable time is needed for training the models, no cross-validation of the results is performed for this paper.

5.4 Evaluation and error analysis

Table 5 shows the accuracy rates of all three tested classifiers, i.e. the number of correctly classified items divided by the number of all items in the respective category, depending from the number of morphological categories proposed by the system. As could be expected, the two sequence based algorithms (`tri`, `crf`) consistently outperform the `me` model, although this model is also trained with context features. Among the sequence based classifiers, `crf` is superior to `tri`.

The column called `fallback` shows that it is possible to improve over the accuracy of `crf` for some frequent classes when the decisions of `tri` and `crf` are merged. As `crf` outputs a probability p along with its decision, a threshold for p that maximizes the accuracy of the `crf` result is searched on a holdout set. If the probability for a solution from the test set is below this treshold, the output of `crf` is replaced with that of `tri`. A simple majority voting with all three classifiers `tri`, `crf` and `me` does not increase the accuracy (last column in Table 5).

[21] Used in the C++ implementation from `http://www.chokkan.org/software/crfsuite/`; settings: L2 regularization: 2.0, one-dimensional architecture.

No. of solutions	Proportion	tri	crf	me	fallback	majority
1	58.04	-	-	-	-	-
2	15.74	92.04	93.12	87.56	93.79	93.22
3	9.09	82.48	88.52	82.1	88.56	87.43
4	9.38	77.89	82.94	79.15	82.78	82.56
5	2.98	89.56	91.42	87.18	91.85	91.65
6	1.76	85.82	89.8	82.76	90.76	88.84
7	0.69	85.7	89	86.55	88.88	88.63
8	0.25	76.49	83.44	78.15	84.77	79.14
9	1.51	83.03	84.21	75.03	85.39	83.54
≥ 10	0.54	84.45	86.47	80.72	88.18	85.54

Table 5. Number of proposed morphological categories and accuracy of the tested classifiers. Refer to Footnote 6 for the abbreviations

Table 6 gives a more detailed evaluation of precision, recall, and F-score for the most frequent target tags in the test set. Two observations are relevant. First, `crf` generates the best solutions for most tags, as indicated by the numbers printed in bold. For some cases, such as the notoriously difficult NOM.SG.MSC. (second row), one can observe a strong increase in P, R, and F when compared to the values of `tri` and `me`. Second, there are large differences between the target tags that can be explained by the underlying morphological ambiguities and their statistical distributions. The most frequent tag CO.MSC., for example, is identical with VOC.SG.MSC. in many cases. Nevertheless, the chance of confounding the two forms is low, because vocatives are comparatively rare in the corpus. Similarly good results are achieved for ambiguous verbal forms, most of which have one "dominant" interpretation. So, the form *uvāca* ("I/he said") is almost exclusively used as 3rd SG.PERF. (and not 1st SG.PERF.), and *gacchati* is mostly used as 3rd SG.PRES. of the root *gam* ("to go") and not as LOC.SG.MSC. of the present participle of *gam*. On the contrary, the rates for (NOM.|ACC.)SG.NEU. and ACC.SG.MSC. are low, because the frequent noun classes on *-a* have the same endings for these three forms. This situation is further complicated by *bahuvrīhi* formation (Section 3.2) and sentences extending over *daṇḍa* boundaries (Section 3.6). *bahuvrīhi* formation is also responsible for errors in classifying forms such as INS.SG.MSC. and NEU. (not in the table), because it can change their genders during the compounding process.

6 Summary and perspectives

The paper has described a system for joint tokenization, lemmatization and morphological analysis of Sanskrit, and it has reported performance rates for the tokenization task (Section 4.3, Tables 2 and 3). By using `crf` as an additional sequential classification layer, it is possible to improve the analysis of

		crf			tri			me		
Tag	Prop.	P	R	F	P	R	F	P	R	F
Co.Msc.	14.34	98.52	**99.62**	**99.07**	**98.8**	97.38	98.08	93.54	98.88	96.14
Nom.Sg.Neu.	11.22	**80.92**	**88.07**	**84.34**	73.2	79.09	76.03	75.83	84.29	79.84
Acc.Sg.Neu.	9.11	**81.94**	**76.24**	**78.99**	72.44	69.13	70.75	71.54	74.7	73.09
Nom.Pl.Msc.	7.00	93.91	**98.36**	**96.08**	**94.4**	95.54	94.97	89.79	95.34	92.48
Acc.Sg.Msc.	4.47	**84.4**	79.87	**82.07**	83.08	**80.01**	81.52	75.14	75.96	75.55
3.Sg.Past	3.09	98.9	**99.8**	**99.35**	**99.28**	99.41	99.34	93.51	97.65	95.54
Gen.Sg.Msc.	2.90	90.02	**97.02**	**93.39**	**92.36**	93.96	93.15	89.02	91.67	90.33
Loc.Sg.Neu.	2.86	92.21	89.09	90.62	**93.51**	**90.29**	**91.87**	86.74	85.64	86.19
Nom.Sg.Msc.	2.76	**92.66**	**96.65**	**94.61**	92.31	92.71	92.51	89.41	91.69	90.54
Loc.Sg.Msc.	2.44	85.25	91.09	88.07	**87.57**	**91.25**	**89.37**	82.87	83.83	83.35
3.Sg.Pres.	2.44	**98.28**	**99.09**	**98.68**	96.22	98.68	97.43	92.78	97.53	95.1
Nom.Sg.Msc. (v.n.)	2.38	**82.39**	**93.16**	**87.44**	80.81	85.65	83.16	76.81	87.51	81.81
Co.Fem.	2.37	94.31	**95.75**	**95.02**	**94.82**	88.62	91.62	92.12	91.33	91.72
Nom.Sg.Fem.	2.19	**92.47**	**91.2**	**91.83**	84.36	89.46	86.84	87.21	85.61	86.4
Ins.Sg.Msc.	2.08	89.39	**93.02**	**91.17**	**90.27**	90.79	90.53	86.33	85.66	85.99
Ins.Pl.Msc.	2.08	87.89	**95.55**	91.56	**92.64**	92.64	**92.64**	86.57	86.74	86.65

Table 6. Precision (P), recall (R) and F score (F) for the three classifiers and the most frequent morphological tags (frequency $\geq 2\%$ of all ambiguous cases in the test set). The highest values for P, R and F per line are printed in bold. Prop.: Proportion of this gold tag in all gold tags of ambiguous cases. v. n.: verbal noun

morphologically ambiguous forms. It should be emphasized that the numbers reported in this paper are only valid for the test set. They will probably be lower in unsupervised analysis, because the input for building the features (Section 5.1) may contain errors. Future research should concentrate on improving the quality of the features with which the model is trained, and on integrating more linguistic tasks such as syntactic parsing into the model, as proposed for Latin by Lee [20]. On the engineering side, deep neural learning models co-trained on several tasks should be tested for a morphologically complex language such as Sanskrit.

From a more general point of view, the linguistic analysis of Sanskrit opens perspectives in two areas. First, Sanskrit is a typical representative of resource-poor languages – both in its linguistic embedding in South Asia and in its status as a classical language that is not spoken anymore. Solutions found for the linguistic analysis of Sanskrit may, therefore, be applicable both for other South Asian languages and for similar studies in classical European languages. Second, NLP has been focussing stronger on morphologically rich languages with a weakly regulated word order during the past few years, and it may profit from insights gained from the study of "off-track" languages such as Sanskrit.

References

1. `Opencyc` website (2007), http://www.opencyc.org
2. Adler, M., Elhalad, M.: An unsupervised morpheme-based HMM for Hebrew morphological disambiguation. In: Proceedings of the 21st International Conference on Computational Linguistics. pp. 665–672 (2006)
3. Bloch, J.: Indo-Aryan from the Vedas to Modern Times. Librarie d'Amérique et d'Orient, Paris (1965)
4. Brockington, J.: The Sanskrit Epics. Brill, Leiden (1998)
5. Cardona, G.: Pāṇini. A Survey of Research. Mouton, The Hague - Paris (1976)
6. Emeneau, M.: Dravidian and Indo-Aryan: The Indian linguistic area. In: Language and Linguistic Area. Essays by Murray B. Emeneau, pp. 167–196. Stanford University Press, Stanford (1980)
7. Gillon, B.S.: Review of "Natural Language Processing: A Paninian Perspective" by Akshar Bharati, Vineet Chaitanya, and Rajeev Sangal. Prentice-Hall of India 1995. Computational Linguistics 21(3), 419–421 (1995)
8. Gillon, B.S.: Word order in classical Sanskrit. Indian Linguistics 57(1-4), 1–35 (1996)
9. Hellwig, O.: `SanskritTagger`, a stochastic lexical and POS tagger for Sanskrit. In: Huet, G., Kulkarni, A., Scharf, P. (eds.) Sanskrit Computational Linguistics. First and Second International Symposia. pp. 266–277. Lecture Notes in Artificial Intelligence, 5402, Springer Verlag, Berlin (2009)
10. Hellwig, O.: Etymological trends in the Sanskrit vocabulary. Literary and Linguistic Computing 25(1), 105–118 (2010)
11. Hellwig, O.: Performance of a lexical and POS tagger for Sanskrit. In: Jha, G. (ed.) Proceedings of the Fourth International Sanskrit Computational Linguistics Symposium. pp. 162–172. Springer Verlag, Berlin (2010)
12. Hellwig, O., Petersen, W.: What's Pāṇini got to do with it? The use of *gaṇa*-headers from the Aṣṭādhyāyī in Sanskrit literature from the perspective of Corpus Linguistics. In: Proceedings of the WCS 2015 (forthcoming)
13. Huet, G.: A functional toolkit for morphological and phonological processing, application to a Sanskrit tagger. Journal of Functional Programming 15(04), 573–614 (2005)
14. Kiparsky, P.: On the architecture of Pāṇini's grammar. In: Huet, G., Kulkarni, A., Scharf, P. (eds.) Sanskrit Computational Linguistics, Lecture Notes in Computer Science, vol. 5402, pp. 33–94. Springer, Berlin, Heidelberg (2009)
15. Knauth, J., Alfter, D.: A dictionary data processing environment and its application in algorithmic processing of Pali dictionary data for future NLP tasks. In: Proceedings of the 5th Workshop on South and Southeast Asian NLP. pp. 65–73 (2014)
16. Kneser, R., Ney, H.: Improved backing-off for m-gram language modeling. In: Proceedings of the International Conference on Acoustics, Speech, and Signal Processing. pp. 181–184 (1995)
17. Kulkarni, A., Shukla, D.: Sanskrit morphological analyser: Some issues. Indian Linguistics 70(1-4), 169–177 (2009)
18. Kulkarni, M.: Phonological overgeneration in Paninian system. In: Huet, G., Kulkarni, A., Scharf, P. (eds.) Sanskrit Computational Linguistics. pp. 306–319. Springer, Berlin, Heidelberg (2009)
19. Lafferty, J.D., McCallum, A., Pereira, F.C.N.: Conditional random fields: Probabilistic models for segmenting and labeling sequence data. In: Proceedings of the Eighteenth International Conference on Machine Learning. pp. 282–289 (2001)

20. Lee, J., Naradowsky, J., Smith, D.A.: A discriminative model for joint morphological disambiguation and dependency parsing. In: Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies-Volume 1. pp. 885–894 (2011)
21. Mayrhofer, M.: Kurzgefaßtes etymologisches Wörterbuch des Altindischen. Carl Winter Universitätsverlag, Heidelberg (1982)
22. Mishra, A.: Simulating the Pāṇinian system of Sanskrit grammar. In: Sanskrit Computational Linguistics, pp. 127–138. Springer (2009)
23. Mittal, V.: Automatic Sanskrit segmentizer using finite state transducers. In: Proceedings of the ACL 2010 Student Research Workshop. pp. 85–90. Association for Computational Linguistics, Stroudsburg, PA, USA (2010)
24. Monier-Williams, M.: Sanskṛit-English Dictionary. Munshiram Manoharlal Publishers Pvt. Ltd., New Delhi, 3. edn. (1988)
25. Oberlies, T.: A Grammar of Epic Sanskrit. De Gruyter (2003)
26. Petersen, W., Soubusta, S.: Structure and implementation of a digital edition of the Aṣṭādhyāyī. In: Kulkarni, M. (ed.) Recent Researches in Sanskrit Computational Linguistics. pp. 84–103. D.K. Printworld (2013)
27. Petersen, W.: Zur Minimalität von Pāṇinis Śivasūtras: eine Untersuchung mit Methoden der formalen Begriffsanalyse. Ph.D. thesis, Universität Düsseldorf (2008)
28. Ratnaparkhi, A.: Maximum Entropy Models for Natural Language Ambiguity Resolution. Ph.D. thesis, University of Pennsylvania (1998)
29. Rocher, L.: The Purāṇas., A History of Indian Literature., vol. II, Fasc. 3. Otto Harrassowitz, Wiesbaden (1986)
30. Scharfe, H.: Grammatical Literature. A History of Indian Literature, Volume 5, Fasc. 2, Otto Harrassowitz, Wiesbaden (1977)
31. Shacham, D., Wintner, S.: Morphological disambiguation of Hebrew: A case study in classifier combination. In: Proceedings of the 2007 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning. pp. 439–447. Association for Computational Linguistics, Prague (2007)
32. Shukla, P., Kulkarni, A., Shukl, D.: Geeta: Gold standard annotated data, analysis and its application. In: Proceedings of ICON (2013)
33. Staal, J.: Word Order in Sanskrit and Universal Grammar. Foundations of Language, Supplementary Series, Volume 5, D. Reidel Publishing Company, Dordrecht (1967)
34. Stenzler, A.F.: Elementarbuch der Sanskrit-Sprache. Max Mälzer, Breslau (1872)
35. Witzel, M.: Early Indian history: Linguistic and textual parametres. In: Erdosy, G. (ed.) The Indo-Aryans of Ancient South Asia. Language, Material Culture and Ethnicity, vol. 1, pp. 85–125. Walter de Gruyter, Berlin, New York (1995)
36. Yuret, D., Türe, F.: Learning morphological disambiguation rules for Turkish. In: Proceedings of HLT-NAACL (2006)