Language Resources and Evaluation (2023) 57:1173–1206
https://doi.org/10.1007/s10579-023-09636-5

# Data-driven dependency parsing of Vedic Sanskrit

**Oliver Hellwig**[1,2] · **Sebastian Nehrdich**[1,3] · **Sven Sellmer**[1,4]

Accepted: 13 January 2023 / Published online: 10 February 2023


**Abstract**
This paper describes the first data-driven parser for Vedic Sanskrit, an ancient Indo-Aryan language in which a corpus of important religious and philosophical texts has been composed. We report and critically discuss experiments with the input feature representations, paying special attention to the performance of contextualized word embeddings and to the influence of morpho-syntactic representations on the parsing quality. In addition, we provide an in-depth discussion of the parsing errors that covers structural traits of the predicted trees as well as linguistic and extra-textual influence factors. In its optimal configuration, the proposed model achieves 87.61 unlabeled and 81.84 labeled attachment score on a held-out set of test sentences, demonstrating good performance for an under-resourced language.

**Keywords** Vedic Sanskrit · Dependency parsing · Low-resource languages · Contextual embeddings

## 1 Introduction

Vedic Sanskrit (VS)—or short, Vedic—used in the 2nd and 1st millennium BCE, is one of the oldest transmitted Indo-European (IE) languages and the historical predecessor of Classical Sanskrit (CS) as well as of most Modern Indo-Aryan languages.

✉ Oliver Hellwig
Oliver.Hellwig@uni-duesseldorf.de

Sebastian Nehrdich
nehrdich@uni-duesseldorf.de

Sven Sellmer
sellmer@uni-duesseldorf.de

[1] Institute for Linguistics, Heinrich-Heine-Universität Düsseldorf, Düsseldorf, Germany

[2] Department of Comparative Language Science, University of Zürich, Zurich, Switzerland

[3] Khyentse Center for Tibetan Buddhist Textual Scholarship, Universität Hamburg, Hamburg, Germany

[4] Institute of Oriental Studies, Adam Mickiewicz University, Poznan, Poland

There exists a rich corpus of texts composed in Vedic, which is crucial for understanding the cultural history of South Asia and for reconstructing the early development of the IE languages. While, however, the computational processing of CS has seen significant progress in the last decade, much less work has been done on resources and NLP tools for VS. The present paper addresses this issue by presenting the first data-driven dependency parser for Vedic, which is used for extending an existing treebank of this language (Hellwig et al. 2020). We do not aim at presenting a new parsing architecture. Instead, we concentrate on discussing choices made when designing the parser, most notably the selection of input features and their static or contextualized representation. In addition, we present an in-depth, linguistically motivated discussion of the parsing errors. These contributions are meant to form the foundation for a parsing algorithm more targeted to the peculiarities of Vedic and similar premodern languages such as Ancient Greek and Latin.

The results discussed in this paper are also relevant for the wider field of parsing morphologically rich languages (MRLs) because Vedic has a rich system of fusional morpho-syntactic features. Parsing MRLs has encountered increasing interest in the NLP community over the last decade (see Tsarfaty et al., 2013 and esp. the survey in Tsarfaty et al., 2020). Many MRLs, including Vedic, share a number of basic syntactic traits that set them apart from languages using a reduced morphology, such as English or Chinese. Most importantly, they have a rich repertoire of morpho-syntactic markers that indicate the syntactic relations in a sentence and thereby can support a dependency labeler in finding the correct parse. The morpho-syntactic expressiveness, however, often comes along with a low degree of configurationality, implying, among others, free word order and the use of discontinuous constituents (see Sect. 4.4.2).

Dependency parsing of VS involves several domain- and annotator-related issues. Firstly, the Vedic corpus has been composed over a period of at least one millennium and contains texts from different literary genres. VS is therefore a good test case for studying domain effects on a diachronic linguistic axis and with regard to genres (see Sect. 4.4.8). Secondly, the number of potential annotators for a Vedic Treebank (VTB) is small, so that the standard approach, which involves adjudicating multiple annotations of the same sentence, is practically not viable (for an example of good practice in this regard see Berzak et al., 2016). In addition, active speakers of VS are missing and many syntactic phenomena and content-related issues are by far less well understood than for modern languages. As individual annotators tend to form idiosyncratic annotation decisions (see Biagetti et al., 2021 for a study of VS), a parser of VS must be able to learn from partly idiosyncratic annotation schemes. Thirdly, the extant Vedic corpus contains around 3 million words (see Sect. 3.1) and may therefore not be large enough for pretraining contextualized word embeddings, which have boosted the performance of many downstream NLP tasks (see e.g. Kulmizev et al., 2019). Adding data from the corpus of CS mitigates the issue of data sparsity, but these later texts come from different cultural and linguistic domains (think of the difference between Middle and Modern English). As similar issues are encountered with other premodern languages (see e.g. Passarotti, 2019, Sect. 4.2, for Latin), we perform a systematic evaluation of how state-of-the-art contextualized word embeddings influence the performance of the parser. Certain linguistic

characteristics of (Vedic) Sanskrit such as sandhi (see Hellwig & Nehrdich, 2018) and its high morphological complexity complicate the annotation process. It is therefore even more important to use high-quality morpho-syntactic input data when parsing VS; we obtain this data from the gold annotations in the Digital Corpus of Sanskrit (DCS, see Sect. 3.3). This situation is contrary to that of many other languages, where predicted (silver) morpho-syntactic data is typically used as input for the parsing process (see Sect. 4.3 for a comparative evaluation). Our experiments suggest that parsers trained with lexical and morpho-syntactic gold annotations are at least competitive with contextualized models when only limited text corpora are available.

Our paper thus makes the following main contributions:

- We present and discuss the first data-driven syntactic parser for Vedic.
- The experiments described in Sects. 4.2 and 4.3 provide quantitative evidence that the lack of large corpora needed for pretraining contextualized model can be counterbalanced by the use of gold input data.
- We provide an in-depth discussion of the errors made by the parser.

After an overview of related research (Sect. 2) and the available data (Sect. 3), Sect. 4 describes the experimental setup and presents the evaluation of contextual embeddings. Individual types of parsing errors are discussed in Sect. 4.4. Section 5 summarizes the paper. In addition, we publish a new, significantly extended version of the VTB as compared to its state described in Hellwig et al. (2020). This treebank and the code of the parser are available under a Creative Commons license at https://github.com/OliverHellwig/sanskrit.

## 2 Related research

Modern dependency parsing methods can be broadly categorized into transition- (Nivre, 2003) and graph-based parsers (McDonald, 2006). Transition-based parsers build the dependency tree incrementally by a series of actions. A simple classifier is trained on local parser configurations and guides the parsing process by scoring the possible actions at each step. This approach is very efficient since the time-complexity is usually linear. Graph-based parsers on the other hand maximize a particular score by searching through the space of possible trees, given a sentence. The search-space is encoded as a directed graph and the score of a possible tree is calculated by a linear combination of the scores of local sub-graphs. Methods from graph theory such as the maximum spanning tree (MST) are then used to find the highest scoring among all possible trees. Recently, the application of neural networks and continuous representations has led to a substantial performance gain for transition-based (Chen & Manning, 2014; Ballesteros et al., 2015; Weiss et al., 2015; Kiperwasser & Goldberg, 2016) as well as graph-based parsers (Kiperwasser & Goldberg, 2016; Dozat & Manning, 2017). These current state-of-the-art parsers are still either transition- or graph-based, but the differences in their error distributions decrease constantly due to the convergence of neural architectures and feature representations.

The comparative experiments in McDonald and Nivre (2011) and Kulmizev et al. (2019) show systematic differences between transition- and graph-based models. In our analysis of their results we noticed that, according to Table 1 in McDonald and Nivre (2011), their graph-based model performs better than a transition-based one for morphologically rich IE languages (mean labeled attachment score/LAS[1] + 1.127), whereas for IE languages with a comparatively low amount of inflection it even performs worse (mean LAS −0.47). In addition, the graph-based model of Kulmizev et al. (2019) yields much better results than the transition-based one for languages of the SOV type (mean LAS +1.82). Since Vedic is both a morphologically rich IE language and has a preference for SOV, we decided to use a graph-based architecture.

We adapt the biaffine model that achieved a state of the art UAS (see Footnote 1) on all CoNLL 09 languages (Dozat and Manning 2017) and that performs better on non-projective dependencies than transition-based neural models (e.g. Andor et al., 2016)—a relevant feature for Vedic where non-projectivity plays an important role (see Sect. 4.4.2). We further adapt DCST (Rotman & Reichart, 2019) which enhances the biaffine parser by adding deep contextualized self-training. More recent models that outperform these two architectures exist, for example Zhou and Zhao (2019) and Mrini et al. (2020). However, the head-driven phrase structure grammar that they apply requires constituency annotation in addition to dependency annotation, which is currently not available for VS. Recent research (Che et al., 2018; Kulmizev et al., 2019) has shown that dependency parsers benefit from the addition of deep contextual embeddings such as ELMo (Peters et al., 2018) and BERT (Devlin et al., 2019), and such embeddings are also used in current state of the art models such as Mrini et al. (2020). We therefore evaluate how the addition of deep contextual embeddings affects the performance in the case of VS.

Syntactic parsing of Sanskrit has met with increasing interest in recent years. Kulkarni (2021) describes a rule-based dependency parser that uses semantic and structural principles of the Pāṇinian system of grammar and Śābdabodha for pruning the arcs of an initially fully connected graph. While the use of the Pāṇinian grammar is an appealing solution, Vedic texts contain phenomena that are not (fully) compatible with this system (e.g. sentences without finite verbs) so that an application to VS does not seem to be promising. In addition, this parser requires information about the case frames of verbs. While case frames are available for several frequent verbs in CS (Sanka, 2015), such a resource does not exist for the highly variegated verbal system of Vedic which abounds in hapax legomena with unclear meaning. Applying this parser to Vedic would therefore require a substantial amount of data collection and validation. More closely related to the present paper is the survey of data-driven dependency parsing of CS given by Krishna et al. (2020). The authors compare the performance of YAP (More et al., 2019), biaffine (Dozat & Manning, 2017), DCST (Rotman & Reichart, 2019) and L2S (Chang et al., 2016) on the Sanskrit Treebank

[1] The labeled attachment score gives the proportion of arcs whose heads and labels are predicted correctly, while the unlabeled attachment score (UAS) only considers the heads; see e.g. Kübler et al. (2009).

Corpus (STBC, see Kulkarni, 2013), a small treebank of works composed mainly in the 20th century CE. On this treebank, the graph-based neural models (biaffine, DCST) perform significantly better than YAP or L2S, and the pretraining step of DCST gives another 2% advance as compared to the biaffine model (UAS 80.95; LAS 72.86). Notably, the authors report substantially lower scores when they apply the models trained on the STBC to the *Śiśupā lavadha*, a metrical text composed in the 7th or 8th century CE. In this cross-domain application, DCST, being the best model, only achieves 40.02 UAS and 35.7 LAS. While the authors explain this drop primarily by the metrical form of the *Śiśupā lavadha*, one should also consider that the texts in the STBC are composed, so to say, in Neo-Sanskrit, a regulated form of the language which follows a strict SOV word order that is not found in the majority of texts composed before the 19th century, and whose vocabulary differs from that used in CS texts. Similar considerations apply to the results reported for the EBM model (Krishna et al., 2021) because EBM uses largely the same linguistic rules for pruning the search space that also regulate Neo-Sanskrit. The good performance that EBM shows on the STBC (UAS 85.32, LAS 83.93) is therefore not surprising.

Computer-based analyses of texts in premodern languages often face specific challenges, some of which also apply to the Vedic corpus. Recent reports on two important cuneiform languages (Sukhareva et al., 2017; Bansal et al., 2021) show that, due to data sparsity and the difficult nature of the texts, sometimes even the problem of POS tagging is far from being solved. On the other extreme, we have Latin, where the amount of raw textual material is large enough to achieve a state of the art POS tagging result by training a BERT model (Bamman & Burns, 2020). Universal Dependencies (UDs) list about a dozen treebanks of premodern languages on their website,[2] among which the cases best comparable to the situation of Vedic are probably Latin and Ancient Greek (see also Passarotti, 2019; Celano, 2019). Challenges arising when annotating ancient texts have also been discussed by Bamman et al. (2010) for Ancient Greek and Biagetti et al. (2021) for Vedic. Examples of error analysis following the dependency parsing of Ancient Greek texts are given by Mambrini and Passarotti (2012) and Majidi and Crane (2014).

## 3 Vedic Sanskrit and the Vedic corpus

### 3.1 Structure and size of the Vedic corpus

The texts of the Vedic corpus were originally composed orally and were written down only at a much later date (Falk, 1993; Renou, 1947). In fact, the corpus as we have it today represents only a small part of the original oral material (for a detailed overview see Gonda, 1975, 1977; Olivelle, 1998). Traditionally it is divided into five

[2] https://universaldependencies.org.

major groups which can be briefly characterised as follows in terms of their content and style[3]:

- Saṃhitās (15th–9th century BCE): lit. 'collections', namely of metrical hymns addressed to various deities, and of ritual and magical formulas (in verse and prose).
- Brāhmaṇas (9th–7th century BCE): voluminous prose texts mostly containing explanations and discussions of rituals.
- Āraṇyakas (8th–6th century BCE): prose texts of ritualistic and philosophical character.
- Upaniṣads (7th–2nd century BCE): theological and philosophical treatises, the oldest of which are composed in prose, the younger ones in verses.
- Vedāṅgas (6th century BCE–3rd century CE): Apart from numerous treatises on technical topics such as phonetics and metrics, this group comprises three types of texts, composed in a special, extremely condensed style: the Gṛhyasūtras, Śrautasūtras (manuals of domestic and of solemn rituals), and Dharmasūtras (compendiums of law and customs).

From a linguistic point of view, the Vedic corpus can be divided into layers that only partly overlap with the traditional text groups just mentioned. The division used in this paper has been adopted from Kümmel (2000, 5f.):

(1) Early Vedic [= 1-RV]: the Ṛgveda-Saṃhitā,
(2) Old Vedic [= 2-MA]: the metrical portions of the Atharvaveda- and Yajurveda-Saṃhitās ('Mantra language'),
(3) Middle Vedic [= 3-PO]: the prose portions of the Saṃhitās, and the older parts of the Brāhmaṇas, Āraṇyakas, and Upaniṣads,
(4) Young Vedic [= 4-PL]: the younger parts of the Brāhmaṇas, Āraṇyakas (both prose), and Upaniṣads (partly prose, partly verse),
(5) Late Vedic [= 5-SU]: the Sūtra texts of the Vedāṅgas (prose).

Data-driven parsing and pretraining methods require large training corpora. This raises the question whether the Vedic corpus is large enough for successfully pretraining (contextualized) language models. To our knowledge, no reliable estimation of its word count has been published so far because the euphonic merging of separate words into one string (the so-called external sandhi, lit. 'connection'; see Sect. 3.2) prevents a straightforward word count in Sanskrit texts. In order to obtain an approximate number of the words in the extant Vedic corpus, we collect publicly available digital versions of Vedic texts and count the characters in the resulting 72 files. For 14 of these files, the DCS (see Sect. 3.3) contains a complete lexical and morpho-syntactic annotation that allows to establish the true, 'unsandhied' number

[3] The absolute dates of the Vedic texts are highly uncertain, so the chronological figures given below are only meant to serve as a rough orientation; for a more detailed relative chronology see Kümmel (2000) and Witzel (1989).

**Table 1** Size and composition of the Vedic corpus, split by the main genres of the Vedic literature (column 1; see Sect. 3.1)

| Group | Words (est.) | Words (morph.) | Words (dep.) | Sens. (dep.) |
|---|---|---|---|---|
| Saṃhitā | 847,158 [28] | 333,516 [45] | 44,666 [35] | 6512 [39] |
| Brāhmaṇa | 869,560 [28] | 208,392 [28] | 35,489 [28] | 4707 [28] |
| Āraṇyaka | 46,317 [2] | 19,675 [3] | 3455 [3] | 582 [4] |
| Upaniṣad | 52,551 [2] | 40,260 [5] | 13,415 [11] | 1747 [11] |
| Śrautasūtra | 657,010 [21] | 38,513 [5] | 12,445 [10] | 1107 [7] |
| Gṛhyasātra | 279,181 [9] | 87,686 [12] | 16,854 [13] | 1837 [11] |
| Dharmasātra | 154,158 [5] | 12,283 [2] | 205 [0] | 33 [0] |
| Other | 154,021 [5] | 518 [0] | 0 [0] | 0 [0] |
| | $\sum$ : 3,059,956 | 740,843 | 126,529 | 16,525 |

Column 2 gives an estimation of the size of the respective group in words (see Sect. 3.1), and columns 3–5 report the current state of the annotation in the DCS (column 3) and the Vedic Treebank (columns 4–5; see Sect. 3.3). The numbers in square brackets give the rounded proportion of a text category with regard to the total count in the respective column (lowermost row)

of words in these files. We fit a linear model that predicts the number of words ($y_i$) given the number of characters ($x_i$) and an intercept term $a$ on the basis of the 14 lexically annotated files, i.e. $y_i = a + \beta x_i$. The model shows an almost perfect linear fit ($R^2 = 0.994$) and is therefore used for estimating word counts in the remaining 58 files for which no (complete) lexical annotation is available. The estimated counts reported in column 1 of Table 1 show that Brāhmaṇas and the old metrical Saṃhitā texts are the dominant text categories of the Vedic corpus, closely followed by the expositions of the solemn rituals (Śrautasātras). The true size of the extant Vedic corpus is slightly higher than indicated by our estimate of about 3 million words because some texts have not been digitized (completely) so far. It also becomes apparent from Table 1 that the annotation of the Vedic corpus is biased because the Saṃhitās and Upaniṣads have significantly more lexical, morphological and syntactical annotations than would correspond to their proportion in the corpus. This over-representation correlates with their importance in the scholarly discourse. Neither the DCS nor the VTB are therefore balanced samples from the extant Vedic literature, but rather reflect scholarly preferences.

### 3.2 Linguistic, syntactic and orthographic traits of Vedic

Vedic is a MRL, with all eight Proto-IE noun cases, three numbers, numerous verbal categories and declension classes, both for nouns and verbs (see e.g. Burrow, 1955). The morphology abounds in sound changes because of ablaut phenomena and of euphonic changes occurring between (nominal and verbal) stems and endings. A notable feature of Vedic word formation are the ample possibilities of nominal compounding (see Sect. 4.4.6). In addition to a considerable vocabulary of nouns, verbs and various pronouns, Vedic also possesses a large number of particles, many of

which are used with high frequency. The word order is extremely free, so that Vedic in this respect is often compared to languages like Warlpiri (e.g. Reinöhl, 2020), although certain preferred word order patterns can be distinguished (Delbrück, 1888, Chap. 2 ).

Tokenization of (Vedic) Sanskrit is complicated because individual words are sometimes merged due to external sandhi. Although algorithms and tools for tokenization and related tasks are available for CS (Goyal & Huet, 2016; Hellwig & Nehrdich, 2018; Krishna et al. 2021), they either require manual correction of the results (Goyal & Huet, 2016), only perform word splitting, but no lexical and morphological analysis (Hellwig & Nehrdich, 2018) or are tuned for CS (Krishna et al., 2021). The obvious solution to this problem is an end-to-end system that generates syntactic parses from raw Vedic text, featuring word segmentation as one of its intermediate steps (see e.g. the model proposed by Hashimoto et al., 2017). As such a system is currently not available, we use texts with presegmented words throughout this paper. Moreover, Vedic does not feature grammatical sentence boundary markers, and the scribes who first wrote down the Vedic texts did not provide reliable orthographic markers. Traditionally, the texts are structured by single or double vertical strokes (*daṇḍa*s, i.e. 'staffs'), that are used to indicate the end of verse lines, paragraphs, sentences and other textual units. Usage of these signs largely does not follow general rules so that a sequence of strings terminated by a *daṇḍa* can contain one or several sentences or only a part of a sentence. Sentence segmentation is, consequently, another, far from trivial, task that has to be accomplished by the parser, and by the human annotators of the gold data (see Biagetti et al., 2021). In the experiments performed in this paper we therefore work with presegmented sentences.

### 3.3 Digital resources for (Vedic) Sanskrit

Two important repositories of digitized Vedic texts are hosted by the Universities of Göttingen (GRETIL[4]) and Frankfurt (TITUS[5]). Due to sandhi and the fusional morphology of VS (see Sect. 3.2), texts from these repositories cannot be used directly for training a dependency parser, because the texts need at least to be split into words for further processing. The files in CoNLL-U format made available by the DCS (Hellwig, 2010-2021) provide manually validated lexical and morpho-syntactic annotations, but cover only about one quarter of the extant Vedic corpus (see column 3 of Table 1). In spite of their limited coverage, we therefore use the DCS data for those (pre-)training steps that require linguistically annotated input.

The syntactic gold annotations for our experiments come from an extended version of the VTB (Hellwig et al., 2020; Biagetti et al., 2021), a treebank of VS annotated following the UD standard (Nivre et al., 2016). When compared with its state described in Biagetti et al. (2021), the amount of data has increased by about 80,000 words, and it now features substantial text samples from all parts of the Vedic corpus

[4] http://gretil.sub.uni-goettingen.de/gretil.html.

[5] http://titus.fkidg1.uni-frankfurt.de/framee.htm?/index.htm.

(see Table 1, columns 4 and 5). Because the current version of the VTB covers several new textual domains, we also revised the annotation guidelines, paying special attention to linguistic phenomena that preferably occur in late Vedic texts such as the frequent citation of mantras (see Sect. 4.4.7) or argument sharing.

Because there are no native speakers, the inter-annotator agreement (IAA) for treebanks of ancient languages tends to be lower than for those of modern languages. Getting an idea of the IAA on the VTB is important for the purpose of this paper because this value can be thought of as bounding the expected accuracy of a parser: when the annotators strongly diverge in their decisions, the parser cannot be expected to produce perfect prediction scores. This risk is real in the present case, because many Vedic texts leave ample space for contending interpretations in terms of grammar, syntax and content. For our IAA experiments, two authors of this paper re-annotate (1) 50 sentences annotated by one former contributor to the VTB ('annotator 3') and (2) 50 sentences annotated by each other. Each sentence has at most 15 words, and examples from the Rigveda are excluded due to the many disputed passages of this text. We obtain overall values of 89.3% for unlabeled (UAA) and 86.8% for labeled attachment agreement (LAA; see Kübler et al., 2009). This is a marked improvement over the results of the evaluation done for the previous version of the VTB (Biagetti et al., 2021, Sect. 4), although the figures are not completely comparable as the Rigveda is excluded in the present evaluation.[6] A detailed evaluation reveals that most of the disagreement in our annotation can be tracked down to two sources. First, disagreement between the two authors of this paper arises in the annotation of the *Śvetāśvatara-Upaniṣad*, a notoriously difficult text. Second, the two authors of this paper often disagree with annotator 3, especially about the attachment and function of particles. When excluding these two problematic areas from the evaluation, we obtain 93.1% UAA and 89% LAA, which may reflect the degree of disagreement in annotations recently added to the VTB. Note that the true value of agreement for the complete VTB is certainly below this optimistic estimate as the treebank has grown over years and annotations were usually not adjudicated due to lack of manpower.

## 4 Experiments

In this section, we give an in-depth assessment of the parser performance. After a brief overview of the experimental settings in Sect. 4.1, Sect. 4.2 studies the influence of contextualized embeddings and Sect. 4.3 reports the results of a feature ablation experiment and the scores of an optimal model. The subsequent sections discuss various sources of errors.

[6] The figures of 87.4% UAA and 80.6% for ancient Greek given by Bamman et al. (2010) are in the same range but cannot be directly compared to our results because they have been calculated in a different manner. A very high UAA of 96% is reported for presegmented Coptic by Zeldes and Abrams (2018).

### 4.1 Experimental settings

For all experiments described in this section, we apply the biaffine architecture of Dozat and Manning (2017) with the addition of a character based CNN (CharCNN) that uses the individual characters of each inflected form as input (see Rotman & Reichart, 2019; Zhang et al., 2015). We use the implementation of Rotman and Reichart (2019), DCST,[7] but do not apply the pretraining steps for the evaluation of the lexical embedding strategies and the feature ablation due to time constraints (also see Sect. 4.3 for a comparison with the full DCST model). The main extension of this model is that we integrate a larger number of categorical input features. In the same way as in the biaffine model, these features are represented as continuous, randomly initialized embeddings. We use embedding dimensions of 100 for lexical and all other features. Because our parser targets the middle and late Vedic language, Rigvedic data are not included in the following experiments if not specified otherwise. We consider the following input features for the dependency parser:

*Morpho-syntax:* Case, number and gender of each word, as provided by the VTB CoNLL-U files. These features are fully specified for nouns, adjectives, non-personal pronouns and verbal nouns with a nominal inflection (e.g. participles of various tenses). Personal pronouns have case and number information. We evaluate atomic features and their joint representations (see Gupta et al., 2020).
*Verbal nouns:* Verbal nouns convey syntactic information. Participles of the present, future, aorist and perfect stems as well as infinitives typically occur in active constructions, whereas gerunds and the so called past participle are regularly used in active as well as passive constructions and thus require different case frames. We therefore evaluate a combination of the tense and the type of verbal nouns.
*Inflected word forms:* Some graph based parsers (e.g. DCST) use the character representation of each word as input. Since sandhi (see Sect. 3.2) prevents the straightforward extraction of words from digital texts, we take recourse to the 'unsandhied' word forms stored in the VTB CoNLL-U files. If this information (due to reasons connected with the history of the DCS) is missing, we reconstruct the word form by looking it up in the complete DCS including its non-Vedic parts (note that the inflectional morphology of VS and CS differs only in minor points) and by applying a set of heuristic rules. This approach typically retrieves more than 99% of the missing word forms. The word forms are used as input for the CharCNN and for pre-training fastText embeddings.
*Punctuation:* Apart from (double) *daṇḍa*s, which have only a limited value for sentence segmentation (see Sect. 3.2), the DCS CoNLL-U data provide additional punctuation marks that demarcate clauses and were added by the annotators of the DCS (Hellwig, 2016). We use these values as additional input features in our experiments.

[7] https://github.com/rotmanguy/DCST.

*Text-historical layers:* We flag each sentence in the VTB CoNLL-U files with a historical period according to the scheme proposed by Kümmel (2000) (see Sect. 3) and use these flags as additional input features.

In order to regularize the training process of the parser and to avoid overfitting, we augment the training data by randomly concatenating up to four, not necessarily subsequent sentences from the training set. We make the root of the first sentence the root of the new concatenated sentence and connect the root nodes of the subsequent sentences with the first root using the non-UD label `senconj`. As a side effect, this strategy enables the model to learn how to split texts without presegmented text lines (see Sect. 3.2), because clauses labeled with `senconj` can be split into individual sentences during decoding. This feature is especially useful when applying the parser to texts with unreliable sentence boundary annotation.

### 4.2 Word representation strategies

Inspired by Sandhan et al. (2021), we evaluate how the following word representation strategies influence the parsing accuracy: a CharCNN, Word2Vec (Mikolov et al., 2013), fastText (Bojanowski et al. 2017), ELMo (Peters et al., 2018), RoBERTa (Liu et al., 2019) and XLM-RoBERTa (Conneau et al., 2020). CharCNN, which uses the individual characters of each inflected word form as input, serves as a baseline.[8] The Word2Vec model was pretrained on the lemmata provided by the DCS (see Sect. 3.3; ca. 4,800,000 tokens; settings: continuous bag of words (CBOW) model, window size: 8, embedding size: 100). fastText was pre-trained on the GRETIL corpus (see Sect. 3.3; 5m lines/237 Mib). We decided to train and apply fastText to the inflected forms instead of the lemmata, because, due to subword modeling, fastText promises a better performance on inflected forms of MRLs. Word2Vec on the other hand creates token-level representations and is therefore more suited to be applied to the lemmata. While Word2Vec and fastText were pretrained on a mixture of VS and CS, ELMo was only pretrained on the 26 Mib of Vedic data on which the estimates in Table 1 are based, a restriction due to hardware constraints. Since it turned out that the performance of this ELMo model is clearly inferior to that of a RoBERTa model that uses only Vedic (RoBERTa-Vedic), we assume that training ELMo with Vedic and non-Vedic data together will not lead to a performance gain over RoBERTa-Vedic-GRETIL. Default settings for the bidirectional language model were applied during training.[9] We trained two versions of RoBERTa from scratch: one based only on Vedic data (RoBERTa-Vedic) and one based on Vedic and Classical data (RoBERTa-Vedic-GRETIL). For both versions we applied the default parameters given in Liu et al. (2019) with a vocabulary size of 32,000. We also evaluate the performance of a pretrained XLM-RoBERTa model (XLM-100, see Conneau et al., 2020) that was trained, among other languages, on

[8] We adapted the CharCNN from Rotman and Reichart (2019), see also Zhang et al. (2015).
[9] https://github.com/allenai/allennlp-models/blob/main/training_config/lm/bidirectional_language_model.jsonnet.

**Table 2** Evaluation of lexical embedding strategies with different amounts of training data and different levels of additional information (None = embeddings only; POS: with POS tags; All: all available information, see Sect. 4.1)

| | | UAS | | LAS | |
|---|---|---|---|---|---|
| Features | Model | 1000 | 5000 | 1000 | 5000 |
| None | CharCNN | 51.5 | 65 | 38.5 | 54.1 |
| | fastText | 57.5 | 70 | 45.5 | 60.3 |
| | Word2Vec | 51.7 | 62.5 | 37.1 | 49.1 |
| | ELMo | 63.8 | 69.8 | 49.6 | 57.8 |
| | RoBERTa-Vedic | 63.3 | 71.2 | 49.2 | 59.8 |
| | RoBERTa-Vedic-GRETIL | 64.8 | **73.40*** | 52.4 | **63.50**** |
| | XLM-100 | 58.5 | 64.5 | 44 | 52 |
| | XLM-100-Vedic | **66.30.** | 72.8 | **53.8** | 62.5 |
| +POS | CharCNN | 59.7 | 71.4 | 46.2 | 61 |
| | fastText | 64.9 | 74.4 | 54 | 65.8 |
| | Word2Vec | 59.7 | 68.2 | 44.9 | 55.2 |
| | ELMo | 64 | 70.5 | 49.6 | 58.6 |
| | RoBERTa-Vedic | 65.1 | 72.8 | 50.9 | 62.1 |
| | RoBERTa-Vedic-GRETIL | 67.6 | **75.20*** | 55.6 | **66** |
| | XLM-100 | 62.2 | 69.4 | 48.6 | 58.3 |
| | XLM-100-Vedic | **67.9** | 74.6 | **55.9** | 65.1 |
| All | CharCNN | 69.2 | 78.1 | 58.4 | 69.8 |
| | fastText | **71** | **79.50.** | **61** | **72** |
| | Word2Vec | 70.4 | 79.0 | 60.4 | 71.5 |
| | ELMo | 63.9 | 71.1 | 49.7 | 59.1 |
| | RoBERTa-Vedic | 66.1 | 76.6 | 53.6 | 68.1 |
| | RoBERTa-Vedic-GRETIL | 70.8 | 78.5 | 60 | 70.6 |
| | XLM-100 | 68.9 | 77.1 | 57.4 | 68.3 |
| | XLM-100-Vedic | 70.7 | 78.6 | 60.3 | 70.7 |

The best result per combination of training set size and feature type is printed bold, the second best in italics. Asterisks and dots after the best result give the significance level of a $\chi^2$-test that compares this result with the second best one (notation: . (single dot): 0.05; *: 0.01; **: 0.001)

Vedic and non-Vedic Sanskrit.[10] Due to hardware constraints, we could only evaluate the performance of the XLM-100 model in its base configuration, although Fig. 4 in Conneau et al. (2020) suggests that a better performance for low-resource languages can be expected when a model with more capacity is trained. We also evaluate the performance of XLM-100 after it was further pretrained on Vedic data

[10] The Sanskrit dataset used in that publication is accessible here: https://metatext.io/datasets/cc100-sanskrit. It contains about 340 Mib of Vedic, Classical and contemporary Sanskrit, occasionally intermixed with text from other languages.

(XLM-100-Vedic). For all embedding models except for Word2Vec we use the inflected 'unsandhied' forms as inputs.

In order to assess how the models react under different conditions of data-availability, we report results for experiments with 1000 and 5000 sentences. We apply a cross-validation scheme with fivefolds, using 80% of the data for training and 20% for the evaluation. In order to make the results comparable, we evaluate all models on the same train/test splits. We further evaluate all models in three settings: first using only lexical information ('None' in Table 2), second with the universal POS tags enabled ('+POS'), and third with the complete ensemble of linguistic information (see Sect. 4.1) enabled ('All'). Notably, we include the CharCNN for all models in the third setting. We choose these three settings in order to emulate different levels of availability of (gold) annotated data. The influence that this gold information exerts on the results is evaluated separately in Sect. 4.3.

The results of our experiments are shown in Table 2. In general, all models benefit from the addition of the POS tags, and adding the full ensemble of linguistic information gives a further boost in performance. Equally, having more sentences available for training leads to an overall increase in performance. Static embeddings (CharCNN, fastText, and Word2Vec) are inferior in performance when no POS tags and/or linguistic information are available. With detailed linguistic information static models perform slightly better than contextual ones. The results also show that contextual models benefit from the addition of non-VS training data.

When no POS tags or linguistic information is available (setting 'none', first compartment of Table 2), the three non-contextual models perform significantly worse than the contextual models. This difference is especially pronounced with only 1000 sentences of data. Among the non-contextual models, fastText performs clearly better than the rest, while Word2Vec produces the lowest LAS. This shows that using inflected forms instead of lemmata is important for obtaining good LAS when no further linguistic information is available. When we increase the size of the training set to 5000 sentences, fastText begins to outperform, in terms of LAS, all contextual models with the exception of RoBERTa-Vedic-GRETIL and XLM-100-Vedic.

Adding POS tags (setting '+POS', second compartment of Table 2) reduces the gap between static and contextual models. Word2Vec becomes more competitive with the contextual models: It consistently outperforms XLM-100 and ELMo and it also performs better than RoBERTa-Vedic with the exception of the UAS for 1000 sentences. Among the contextual models, XLM-100-Vedic remains the best performing model for 1000 sentences, as does RoBERTa-Vedic-GRETIL for 5000.

When all linguistic information is considered (third compartment of Table 2), fastText outperforms all other models although the difference to the respective second best model, which is now Word2Vec, is statistically significant only for the UAS with 5000 sentences. XLM-100-Vedic and RoBERTa-Vedic-GRETIL remain the best contextual models with very similar performance. ELMo clearly falls behind with a significant gap to all other models. It is noteworthy that XLM-100 performs much better in this setting than in the previous two settings, even outperforming RoBERTa-Vedic.

We conclude that contextual models have an advantage over non-contextual ones. This advantage, however, vanishes when enough linguistic information is available.

While all models benefit from the addition of such information, this effect is weakest for ELMo. This is in line with observations made in Straka et al. (2019), Sect. 4.2, where ELMo is outperformed by BERT (Devlin et al. 2019) in dependency parsing. The finding that ELMo performs worse than other types of embeddings in the setting 'All' seems to be in contrast to a recent paper according to which ELMo outperforms several other contextualized embedding types on a wide range of semantic tasks in CS (Sandhan et al., 2021). We hypothesize that the contradictory results are first due to the nature of the tasks (semantics vs. syntax) and, second, to the fact that the authors did not consider a scenario where additional linguistic information was available to their model.

The results of RoBERTa-Vedic show that it is difficult to achieve competitive performance when only Vedic data is considered during training. Conneau et al. (2020) have observed that "a few hundred MiB of text data is usually a minimal size for learning a BERT model". Presumably, 26 Mib of Vedic text are not enough to train a competitive model, especially in the light of the linguistic complexity of Vedic (see Sect. 3.2). As the performance of RoBERTa-Vedic-GRETIL shows, adding non-Vedic Sanskrit data improves the performance although the model is not able to outperform the static ones in the setting 'All'. The performance gains of the static fastText model in this setting are especially obvious for the labels `iobj` and `compound` where fastText shows substantial performance gains when compared to RoBERTa-Vedic-GRETIL. It seems plausible that case information contributes most to these gains. It thus becomes apparent that non-VS data can support the training process, which mirrors the observation made in Lample and Conneau (2019), Table 4, where a Nepali language model shows significantly lower perplexity when additional Hindi data is considered during training. The poor results for XLM-100 show that the performance of contextual models on the very specific Vedic domain is hampered when numerous other languages are included during training. This is in line with Conneau et al. (2020), Fig. 2, which shows an effect of dilution on low-resource languages when more than 15 languages are considered. In addition, most of the data used for training XLM-100 are of much younger age and from completely different socio-cultural domains—a statement that even pertains to the Sanskrit and Latin subcorpora. Further finetuning of the XLM-100 model on Vedic data helps to boost its performance so that it becomes competitive with, but not better than RoBERTa-Vedic-GRETIL. We conclude that pretraining on 100 languages does not lead to a better performance over pretraining on Sanskrit data alone.

Our observations match the results presented in Wu and Dredze (2020), where the performance of multilingual and monolingual BERT models for low-resource languages on the tasks of named entity recognition, part of speech tagging and dependency parsing is evaluated. The authors observe that for languages with a corpus size of less than 0.044–0.088 GB (raw dump of the Wikipedia archive file), using no pretrained language model at all is in general better than using multilingual BERT, with the performance of monolingual BERT models for such small corpora being even worse. Straka et al. (2019) also evaluate the performance of multilingual BERT for dependency parsing of various languages. For two corpora of Ancient Greek, contextual embeddings do not improve parsing accuracy, in contrast to most

**Table 3** UAS results of the feature ablation experiments for morpho-syntactic and other input features

| Removed feature | 1000 | 5000 |
|---|---|---|
| Characters | −1.27* | −0.76* |
| fastText representation | −1.63** | −1.71** |
| UD-POS | −1.05 | −0.63* |
| Joint representation | −1.01 | −3.01** |
| Case | −1.32 | −0.49 |
| Number | 0.03 | 0.12 |
| Gender | 0.23 | 0.1 |
| Verbal nouns | −0.98 | 0.02 |
| Punctuation | −0.61 | −0.22* |
| Diachronic layers | −0.24 | −0.18 |

other languages.[11] Similarly, contextual embeddings provide only marginal improvements for Latin parsing. These combined observations lead to the conclusion that for ancient languages with limited resources, using contextual embeddings does not necessarily improve the parsing accuracy, especially when enough other linguistic information is available.

### 4.3 Feature ablation

In order to evaluate how the input features influence the performance of the parser, we carry out a feature ablation experiment. Based on the results of Sect. 4.2, we use fastText of the inflected forms as the lexical embedding, apply the same fivefold cross-validation scheme as in Sect. 4.2 and run the experiment with 1000 and 5000 sentences. As a baseline we use a setting where character embedding, fastText representation and all linguistic features as well as punctuation and diachronic layer information are enabled. The results in Table 3 show that removing the joint representation (see Sect. 4.1) and the lexical fastText representation decreases the performance most clearly, whereas removing number, gender and even case has no statistically significant effects. The effect of the joint representation is most obvious for 5000 sentences where data sparsity is less severe than in the 1000 sentence setting. We conclude that when the joint representation is available, atomic morpho-syntactic features do not support the model performance in a significant way. This is contrary to the results reported in Gupta et al. (2020), where composite models outperformed joint models on the task of Sanskrit morphological tagging. With the exception of number, gender and the joint representation, the negative effects of removing a feature are stronger for 1000 than for 5000 sentences. This observation suggests that fine-grained linguistic information is more beneficial when less training data is

[11] Note that in the case of Ancient Greek Straka et al. (2019) did not use any Ancient Greek data for the training of the BERT model, while we could use a small corpus of VS. In addition, they did not employ gold POS tags and lemmatized words for dependency parsing, but utilized the output of UDPipe 2.0 (Straka, 2018); see Sect. 4.3 for the effect of this setting.

**Table 4** Results of training the best performing configuration of the biaffine parser with the full dataset and augmentation

| Model | UAS | LAS |
|---|---|---|
| Biaffine Word2Vec + gold annotation | **87.63** | 81.68 |
| DCST + gold annotation | 87.61 | **81.84** |
| Biaffine Word2Vec + silver annotation | 84.87 | 79.34 |
| Biaffine XLM-100-Vedic + silver annotation | 84.97 | 79.18 |

Upper half: gold POS and morpho-syntax; lower half: silver POS and morpho-syntax

available. The character embedding has a significant impact in both scenarios. Since VS is a morphologically rich language, we assume that the character embedding is able to encode lexical properties that are not accessible via the other features. From among the remaining features, only punctuation has a small positive effect on the parsing performance.

For the best performing model, we use all available sentences of the VTB, yielding 16,272 sentences after removing duplicates. 80% are held back for training, 10% are used for validation and 10% for testing. After augmentation (see above, Sect. 4.1), the training data consists of 41,052 sentences. Samples from the Rigveda are used for training, but not for validation and testing. Since the effect of adding atomic case, number and gender features is not conclusive, we decided to use all available features for this setting. We slightly modify the hyper-parameters used in Rotman and Reichart (2019): 100 epochs, a batch size of 32, a learning rate of 0.002 and dropout probabilities of 0.33. With these settings, our model reaches 87.63 UAS and 81.68 LAS without pretraining. When we apply the DCST pretraining step, the model reaches 87.61 UAS and 81.84 LAS, showing that this kind of pretraining, while substantially increasing the computation time, does not lead to decisive performance gains.

Contrary to most other studies, our parser uses gold POS and morpho-syntactic information, since this data is available for the Vedic subcorpus of the DCS. In order to evaluate to which degree this gold information influences the parser performance, we repeat the ablation experiment using predicted (silver) tags (results in Table 4). To obtain these tags, which are not provided by the DCS, we train two separate linear classifiers on top of the XLM-100-Vedic-model, which is among the strongest of the contextual models evaluated in our ablation study (see Table 2). These classifiers receive the manually validated split word forms provided by the DCS as input, and are trained to predict the part-of-speech and the morpho-syntactic information of each word. The classifiers reach accuracy rates of 97.1% for POS and 94.1% for morpho-syntactic tagging on the same held-out set that we use for evaluating the dependency parser. When using a static word embedding model in combination with these silver annotations, the best performing configuration of the biaffine parser reaches a performance of 84.87 UAS and 79.34 LAS, which is clearly below the performance of the biaffine parser with gold annotation. When we add the contextual embedding model XLM-100-Vedic to this configuration, the performance increases slightly to 84.97 UAS, while LAS decreases to 79.18. The results in Table 4 thus show that gold POS and morpho-syntactic information improves the

**Table 5** Results of a linear regression that tests the joint influence of the sentence lengths, the diachronic layers (see the list of abbreviations on p. 6) and of their combinations (Length: 3-PO etc.) on the sentence-wise LAS; $R^2 = 0.0249$

| Coefficient | Estimate | $t$ | p-value |
|---|---|---|---|
| Intercept | 0.8726 | 239.32 | $< 0.001$ |
| Length | −0.0121 | −13.66 | $< 0.001$ |
| Length: 3-PO | 0.0093 | 9.84 | $< 0.001$ |
| Length: 4-PL | 0.0069 | 7.95 | $< 0.001$ |
| Length: 5-SU | 0.0056 | 6.31 | $< 0.001$ |

Column 'Estimate' reports the values of the weight of each covariate as estimated by the model, 't' is the result of a t-test applied to this estimate, and column 'p-value' reports the statistical significance of this test (low values are good)

parsing accuracy; and that, even when silver annotation is used, the parsing accuracy for Vedic does not improve consistently by adding contextual embedding models.

### 4.4 Error analysis

This section provides an in-depth analysis of the errors made by the parser. We start with structural features as explored by McDonald and Nivre (2007) and others and then move forward to examining linguistic and text-historical issues that have an impact on the performance.

#### 4.4.1 Structural traits of the dependency tree

It has repeatedly been reported that the parsing accuracy depends on the length of a sentence (see e.g. McDonald & Nivre, 2007). This effect can also be observed in Vedic, as the sentence-wise LAS decreases significantly for increasing sentence lengths.[12] The only peculiarity worth mentioning here is that very short sentences of two or three words have labelled attachment scores of only 87.5 and 86.2, although one may expect them to be easy to parse. A closer inspection of the relevant POS patterns shows, however, that these low scores are mainly caused by purely nominal identity statements. This type of sentences is problematic for the parser as well as for human readers, as the direction of the identification and thus the root assignment is often questionable (see the discussion in Sect. 4.4.4).

The sentence lengths interact differently with the diachronic layers of the Vedic corpus (see Sect. 3.1 for a presentation of the five layers distinguished used in this study). To understand these interactions, we fit a linear model that predicts the sentence-wise LAS using an additive combination of an intercept term, the length of the sentence and the interaction of its length with the chronological layer. When we apply a t-test to assess if the coefficient estimates of this model differ from zero, we obtain highly significant p-values for all model coefficients (details in Table 5). Most

[12] A linear model that predicts the LAS conditioned on the sentence lengths has highly significant coefficients: Intercept: 87.6065, $t = 250.98$, p 0.001; Length: −0.6209, $t = -15.74$, p 0.001.

**Table 6** Influence of the sentence complexity score (Eq. 1) on the sentence-wise LAS

| Coefficient | Estimate | $t$ | p-value |
|---|---|---|---|
| Intercept | 0.8247 | 297.82 | $< 0.001$ |
| Complexity | −0.0158 | −4.31 | $< 0.001$ |
| Length | −0.0219 | −5.72 | $< 0.001$ |
| Complexity:Length | −0.0047 | −2.14 | 0.032 |

Refer to Table 5 for the interpretation of the columns

notably, the two prose layers (3-PO, 4-PL) and even the elliptic Sūtra texts (5-SU) obtain higher scores than the earlier metrical texts (2-MA), which forms the calibration for the interaction terms and is therefore not listed separately in Table 5. Such a result makes sense because long sentences in early metrical texts often contain long adpositions, placed at the left and right periphery of sentences, whose syntactic connection with the rest of the sentence is open to discussion.

We further hypothesize that some issues with long sentences are due to complex paths leading from the roots to the leaves of the dependency trees (see Gulordava & Merlo, 2015). We define a measure of complexity $c$ for each sentence $s$. For each word $w_i \in s$ of length $|s|$, we determine the depth $l_i$ of its incoming edge, i.e. the number of nodes between the word and the root node of the sentence (see e.g. Husain & Agrawal, 2012, p. 7); note that $l_i = 1$ for the root node itself for the sake of consistency. $c$ is now defined as:

$$c = 1 - \frac{|s|}{\sum_{w_i \in s} l_i}. \tag{1}$$

If all nodes are directly connected to the root, $c$ evaluates to 0, whereas it approaches 1 for increasing edge depths. We fit a linear model that predicts the sentence-wise LAS from the z-standardized values of $c$ and the z-standardized sentence lengths, using complete interaction between the predictors. The estimates of the model coefficients in Table 6 show that the main effects would have a comparable influence on the LAS when considered in isolation. The interaction of the main effects increases this negative effect on the LAS.

Branching is another feature that influences the attachment scores. Like other old IE languages (see Lehmann, 1974), Vedic shows a preference for left-branching constructions (i.e. the dependent is found to the left of its head). This tendency is confirmed when we perform a binomial test of the count data, which produces an estimate of $\pi = 0.609, p < 0.001$. When we split the counts of correctly and wrongly labelled assignments by the branching directions, left-branching constructions obtain a higher average attachment score and a $\chi^2$ test of the resulting $2 \times 2$ table yields a highly significant test statistic of $\chi^2(1) = 686.1, p < 0.001, v = 0.093$. The distribution of the branching directions is, however, biased by the UD convention that requires enumerations to be connected to their leftmost element using `conj` (see Rehbein et al., 2017) for a critical discussion of such entropy-increasing encoding schemes). In addition, we label the frequently interspersed mantras (see Sect. 4.4.7) with `flat` in right-to-left chaining annotation. Both conventions artificially increase

**Table 7** Proportions of sentences with at least one non-projective (row 1), OV (vs. VO) constructions (row 2) and third person pronouns functioning as objects (row 3), grouped by the five historical layers of the VTB

| Feature | 1-RV | 2-MA | 3-PO | 4-PL | 5-SU |
|---|---|---|---|---|---|
| Non-projectivity | 37.2 | 33.3 | 19.3 | 21.2 | 13.6 |
| OV | 64.1 | 71.8 | 97.1 | 95.2 | 93.7 |
| Zero arguments | 2.6 | 4.7 | 12.5 | 13.6 | 3.9 |

All indicators show the diachronic development towards a more configurational setting of Vedic Sanskrit

**Table 8** Coefficient estimates of a linear model calculated to predict LAS based on the number of non-projective attachments ($R^2 = 0.020$)

| Coefficient | Estimate | Standard error | $t$ | p |
|---|---|---|---|---|
| Intercept | 86.15 | 0.39 | 221.15 | < 0.001 |
| Non-projective | − 2.1 | 0.43 | -5.03 | < 0.001 |
| Length | − 0.48 | 0.04 | − 11.02 | < 0.001 |
| Non-projective:Length | 0.06 | 0.03 | 1.88 | 0.060 |

the number of right-branching constructions and may thereby decrease the parsing accuracy.

### 4.4.2 Non-projective constructions and non-configurationality

The oldest layer of VS features central traits of a non-configurational language in the sense defined by Hale (1983): it has a free word (or clause) order, discontinuous NPs and null arguments (Ponti & Luraghi, 2018; Reinöhl, 2016; Keydana & Luraghi, 2012). The presence of such features is less well studied for the middle and late Vedic periods, but observations from other IE languages suggest that the degree of configurationality increases over time. As the resulting changes in the syntax of VS may affect the efficacy of a parsing algorithm, we perform the three tests for non-configurationality proposed by Keydana and Luraghi (2012) by collecting statistics about non-projective constructions, object–verb (OV) vs. verb–object (VO) order and the frequency of third person pronouns[13] in the five diachronic layers of the VTB (see Simonenko et al., 2018 for a broader set of metalinguistic influence factors). We consider a subgraph of a dependency tree as non-projective if its yield does not form an interval (Kuhlmann & Nivre, 2006).

The results presented in Table 7 show a clear development of the three indicators from a non-configurational setting in early Vedic towards sentences with a stricter

[13] The use of third person pronouns (third test) is meant as a proxy for zero object constructions. According to Keydana and Luraghi (2012) the increasing use of such pronouns indicates that zero object constructions become dispreferred.

word order (row 2 of Table 7) and less discontinuous constituents (row 1) in the later layers of the Vedic literature. Especially noteworthy are the high proportions of sentences with non-projective constructions in the early metrical texts which should be compared with e.g. 23% for modern Czech as reported by Kuhlmann and Nivre (2006). The third indicator rises until the late prose level, but drops unexpectedly in the Sūtra period. We hypothesize that this behavior is due to the brevity of the sūtra style which requires argument sharing. It must be underlined that these results need to be considered as preliminary because it is not clear to what degree they are correlated to the genre differences between the older metrical and the younger prose texts (see Hock, 1997a, 2001).

In order to assess how discontinuous constructions affect the parsing accuracy, we fit a linear model that predicts the sentence-wise LAS given its z-standardized number of dependents with non-projective attachment while controlling for z-standardized sentence lengths. The coefficient estimates in Table 8 show that, when considered as an isolated main effect, the number of non-projective attachments exerts a larger negative effect on the LAS than the sentence length, while their interaction just captures the obvious fact that longer sentences have a higher chance of having multiple non-projective attachments. As non-projective constructions are not equally distributed over the Vedic corpus, we fit a second linear model that additionally controls for the diachronic layer (details not reported here). Somehow surprisingly, the coefficients of the interaction terms suggest that the most severe problems with non-projective constructions are found in the early Mantra literature and the late Sūtra texts whereas non-projectivity exerts almost no influence on the LAS in old prose (3-PO). Upon closer inspection it becomes, however, apparent that the unexpected correctness of non-planar attachments in the old prose is largely due to (long) intersecting edges in the fifteenth book of the *Atharvaveda*, a prose section that is characterized by highly repetitive passages easily memorized by the parser. When we exclude these passages from the evaluation, the trend in the old prose conforms to the overall picture.

#### 4.4.3 Syntactic constructions

Turning now to the question which syntactic constructions are especially error-prone, we calculate precision, recall and F-score for all UD labels.[14] The results in Table 9 (sorted by decreasing F-scores) show a high variability of the F-scores, the values of which are moderately correlated with the amount of training data available per label (results of Kendall's correlation test: $\tau = 0.393$, $T = 303$, $p = 0.002$). Among the highest scoring labels in Table 9, we find subordinating (`mark`) and coordinating conjunctions (`cc`) as well as the root label, whose high error contribution (see column 6 of Table 9) is mainly due to issues in identity statements (see

[14] A word is labeled correctly if its predicted label as well as its predicted head are the same as in the gold standard. Precision is calculated by dividing the number of correct predictions by the number of predicted labels of a given type. Recall is calculated by dividing the number of correct predictions by the count of this label in the gold standard.

**Table 9** Precision, recall and F-score for individual UD labels, sorted by decreasing F-scores

| Label | P | R | F | Count | Error contr. |
|---|---|---|---|---|---|
| root | 91.3 | 91.3 | 91.3 | 12,745 | **6.3** |
| mark | 89.7 | 90.0 | 89.9 | 4059 | 2.3 |
| flat | 89.8 | 88.6 | 89.2 | 6140 | 4 |
| nummod | 87.4 | 87.4 | 87.4 | 682 | 0.5 |
| vocative | 85.2 | 87.0 | 86.1 | 438 | 0.3 |
| obj | 84.0 | 87.4 | 85.6 | 7122 | **5.1** |
| advmod | 84.3 | 84.4 | 84.3 | 6076 | **5.4** |
| cc | 83.7 | 84.4 | 84.1 | 1916 | 1.7 |
| nsubj | 82.8 | 83.2 | 83.0 | 8894 | **8.5** |
| fixed | 79.0 | 86.3 | 82.5 | 563 | 0.4 |
| obl | 81.4 | 81.4 | 81.4 | 6633 | **7.0** |
| discourse | 81.7 | 80.3 | 81.0 | 4795 | **5.4** |
| det | 78.6 | 78.7 | 78.7 | 2838 | 3.4 |
| advcl | 79.1 | 77.9 | 78.5 | 3303 | 4.2 |
| amod | 74.6 | 80.5 | 77.4 | 1837 | 2 |
| aux | 78.5 | 76.1 | 77.3 | 163 | 0.2 |
| nmod | 75.4 | 77.9 | 76.6 | 4667 | **5.9** |
| iobj | 76.3 | 76.5 | 76.4 | 1126 | 1.5 |
| cop | 76.0 | 71.7 | 73.8 | 769 | 1.2 |
| case | 71.8 | 72.4 | 72.1 | 731 | 1.2 |
| ccomp | 65.4 | 76.5 | 70.5 | 1649 | 2.2 |
| compound | 68.7 | 66.3 | 67.5 | 703 | 1.4 |
| conj | 68.0 | 66.2 | 67.1 | 5597 | **10.8** |
| acl | 66.5 | 65.1 | 65.8 | 3385 | **6.8** |
| xcomp | 65.8 | 60.6 | 63.1 | 1125 | 2.5 |
| appos | 68.4 | 56.4 | 61.8 | 165 | 0.4 |
| csubj | 67.5 | 53.5 | 59.7 | 256 | 0.7 |
| orphan | 64.6 | 52.3 | 57.8 | 2367 | **6.4** |
| parataxis | 65.3 | 49.3 | 56.2 | 649 | 1.9 |
| dislocated | 50.0 | 14.3 | 22.2 | 35 | 0.2 |

The column 'Error contrib.' records how much the given label contributes to the overall error; values of more than 5% are printed bold

Sect. 4.4.4). Another label with a high error contribution is `discourse`. Vedic texts use more than two dozen particles, some of which are extremely frequent (see Delbrück, 1888 for the RV and Dunkel, 2014 for a recent survey). In many cases it is not easy to determine the function and scope of a given particle, with the general label `discourse` and the more specific ones `advmod` and `cc` being available. In addition, some particles tend to occur in chains, and there is no scholarly consensus about which combinations should be treated as fixed and which functions should be attributed to such multi-word entities. These uncertainties are also reflected by the comparatively high error contribution of `advmod`. The high error contribution of

**Table 10** Top five errors from a label confusion matrix

| Head correct | | | Head wrong | | |
|---|---|---|---|---|---|
| Gold | Silver | Frequency | Gold | Silver | Frequency |
| orphan | ccomp | 270 | nsubj | root | 405 |
| obl | obj | 256 | root | nsubj | 345 |
| obj | obl | 184 | acl | conj | 126 |
| obj | nmod | 105 | conj | acl | 124 |
| acl | amod | 93 | conj | nsubj | 120 |

Left: Arcs correct, labels wrong. Right: Arcs and labels wrong

`conj` is due to problems with the right-to-left direction of coordinating structures (see also Husain and Agrawal, 2012, 9ff. and Sect. 4.4.1 of this paper). One exemplary case in which coordinating structures are not recognized are multiple verbal arguments of the same type. While we label such constructions in bouquet annotation, the parser tends to connect each element directly to the verb, though often using the correct label. Similar effects have, for instance, been reported for papyrological Ancient Greek (Keersmaekers, 2019, Sect. 4.4).

Table 10 gives a more detailed overview of some problematic constructions. The left half of this table records the top five entries of a confusion matrix constructed from those instances in which the arc assignment was correct, but the labelling failed. Leaving aside the confusion of `orphan` and `ccomp`, which is mainly due to inconsistencies in the annotation of citations (see Sect. 4.4.7), a substantial number of errors is caused by obliques being labelled as objects and vice versa. This is largely due the fact that words in accusative case, though primarily functioning as direct objects, can also denote oblique arguments, preferably the goal of a motion and the duration of an action. There exists a certain amount of disagreement among the annotators about how individual instances need to be labelled, because the size of the Vedic corpus often does not allow to perform passivization tests. A lexicon of verbal arguments would be helpful in these cases, but is currently not available for VS as the discussion in Sanka (2015) concentrates on the classical language. The confusion of `nmod` and `obj` which contributes to the low score of the label `obj` (see Table 9) mainly occurs in compounds the final members of which have a verbal notion (also see Sect. 4.4.6). One of the most prominent cases are compounds ending in the deverbal noun *kāma-* 'desire, wish'. We observed clear preferences of individual annotators for labelling the dependents of *kāma-* either as `nmod` or as `obj`, without any other special clue that could explain these divergent decisions. We are currently in the process of harmonizing these and related instances. The last entry in the left half of Table 10 also relates to compounds. The DCS is not fully consistent in its treatment of compounds. While many of them are split into their constituents, others, especially those with irregular internal sandhi or with a non-compositional meaning, are given without further internal analysis. While the DCS provides the POS tag ADJ for unanalyzed exocentric compounds, which would result in the obvious annotation `amod`, many annotators choose the syntactic label `acl` in such cases.

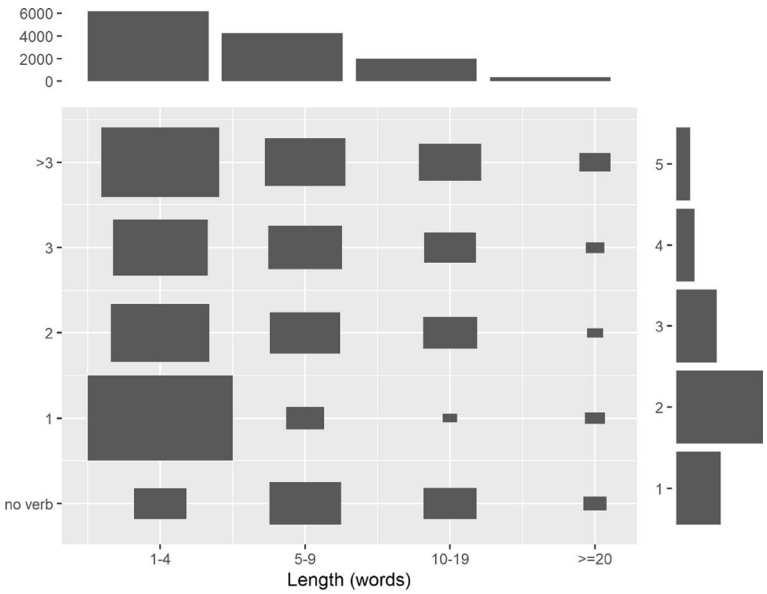


**Fig. 1** Hinton diagram of the sentence-level LAS conditioned on the number of verbal forms (y-axis) and the length class of the sentence (x-axis). The smallest entry corresponds to 79.6% and the largest one to 96%. The marginal histograms summarize the distributions on the two axes

The right half of Table 10 contains the most frequent errors in constructions in which neither the arc nor the label were predicted correctly. Here, the dominating phenomenon is the root assignment in nominal sentences which is responsible for the top two entries (also see Sect. 4.4.4). Issues with `acl` and `conj` typically occur when two words with the same morpho-syntactic information are labelled as coordinated although one is actually an adnominal modifier of the other (and vice versa). The final entry (`conj`, `nsubj`) is another instance of the problem of argument assignment (see p. 21). Here, the parser attaches multiple coordinated subjects individually to their governing verbal head.

### 4.4.4 Nominal sentences without a copula

Vedic texts abound in (elliptic) nominal sentences which are hard to understand even for a human reader. We hypothesize that such constructions are also considerably harder to parse, as already mentioned in Sect. 4.4.3. Figure 1 gives a Hinton diagram of the sentence-level LAS, grouped by the lengths of the sentences (x-axis) and the number of verbal forms, both finite and infinite, found in each sentence (y-axis). While the accuracy generally decreases with increasing sentence lengths (see Sect. 4.4.1), Fig. 1 also shows that the presence of verbal forms has a strong influence on the accuracy, because the lowest accuracy scores are quite consistently observed for sentences lacking any verbal form (see the bottom-most row in Fig. 1),

**Table 11** LAS of words in the OOV frequency class ($LAS_0$), grouped by their POS tags

| POS | $LAS_0$ | $LAS_{1+}$ | Frequency | p |
|---|---|---|---|---|
| NOUN | 62.6 | 75.8 | 1702 | < 0.001 |
| ADJ | 51.9 | 67.6 | 966 | < 0.001 |
| VERB | 84.3 | 85.2 | 896 | 0.229 |

$LAS_{1+}$ is the LAS of all words that occur at least once in the training set

a type of sentences which is comparatively frequent throughout the Vedic literature (see the right marginal histogram in Fig. 1). Especially problematic are short nominal sentences, which are typically identity statements and in which root assignment is the main problem. Consider the two word sentence *Aitareyabrāhmaṇa* 1.1.14.5: *saṃvatsaraḥ prajāpatiḥ* which consists of the two nominatives *saṃvatsaraḥ* 'the year' and *prajāpatiḥ* 'the (god) Prajāpati'. Contentwise, it is not immediately clear which of the two words has to be considered the subject, and which the predicate, neither is the grammar of decisive help here, as Vedic has no rigid word order (see the discussion in Hock, 2013, Sect. 4.1.2). As a consequence, the annotator has to make a decision based on context and pragmatics, mainly taking into consideration which of the two elements is the rheme (or topic) and which is the theme (or comment) of the utterance: the former then becomes the subject, the latter, the predicate (see Gren-Eklund, 1978; also see Viti, 2010 for a critical reappraisal of the theme-rheme approach of the Prague school, with a special focus on early Vedic).

#### 4.4.5 Word frequencies and out-of-vocabulary terms

It has been observed in previous research that the frequencies of words have a substantial influence on individual tagging decisions (see e.g. Tsarfaty et al., 2013; Kolachina et al., 2017). In order to study such effects for our data, we perform a log-linear regression with the word-wise LAS as the predicted variable and the log-frequency of words in the training set of each fold as the predictor. This regression yields a highly significant result (intercept: 69.43, $t = 125.28$, p 0.001; slope: 3.69, $t = 10.26$, p 0.001), and its coefficients show that the LAS increases by about 3.7% for each step of one in the log space. The same positive log-linear trend can also be observed for most POS types (details not reported here), the only exception seemingly being the tag VERB that displays a decreasing trend in the log-linear space. Closer inspection of the relevant cases shows that the decrease of LAS in the high frequency spectrum is mainly due to the two frequent verbs *as* 'be, exist' and *bhā* 'become, be, exist' both of which can function as main verbs ('exist', 'become'), copula and occasionally also as auxiliaries.

As could be expected, words with errors in the OOV class almost exclusively belong to the three open word classes of nouns, adjectives and verbs (see Table 11). When we compare the error rates of OOVs with the non-OOV error rates for the respective POS tag using binomial tests (alternative hypothesis: the LAS for OOV words $LAS_0$ is less than the score $LAS_{1+}$ for non-OOV), OOV nouns and adjectives obtain a significantly lower score than these tags do in general, while this effect

is much less pronounced for verbs. We hypothesize that OOV verbs are less error prone than the other two classes because verbs have a more restricted choice of syntactic functions than, for instance, nouns. In addition, their derivational morphology is often more transparent than that of nouns which gives the character encoder the chance to transfer semantic information from the base word to the derivative. The low labelled scores of adjectives are due to issues with how the DCS encodes lexicographic information (see the discussion of compounds used as adjectives on p. 21) and the unclear linguistic status of adjectives in Vedic.

The frequency range in which the LAS conforms least to the log-linear trend comprises words with frequencies between 100 and 1000 in the training set. A detailed inspection of these words shows no single source for the widely diverging attachment scores. Some of the lowest scoring words in this frequency range are multifunctional particles (see p. 20 of this paper) and quantifiers (e.g. *viśva-* 'all' which can form part of a high-frequency theonym). At the upper end of the LAS spectrum, we mainly encounter verbs with specialized meanings and well defined case frames (e.g. *ālabh* 'touch [a sacrificial animal in order to kill it]', *ah* 'say') whose LAS often comes close to 100%. Apart from the two verbs *as* and *bhā* (see above), the three deictic pronouns *etad*, *idam* and especially *tad* are responsible for most errors in the highest frequency class. These errors are due to the multiple functions these pronouns can perform in Vedic.[15] Apart from their regular use as determiners, (a) they occur as subjects in identity statements (see Sect. 4.4.4), (b) the accusative singular neuter forms of *etad-* and *tad-* often express an adverbial notion of manner, place or time, and (c) *sa*, apparently the nominative singular masculine of *tad-*, is sometimes used as a discourse particle, the so-called *sa*-figé (see Hock, 1997b).

Mere word frequencies provide only a coarse explanation of what is going wrong in parsing, as is evidenced by the low $R^2$ score of 0.01524 of the linear model. We hypothesize that co-occurrences in the training set better predict the parsing accuracy. We therefore compile a lexical co-occurrence matrix $\mathbf{C}$ for each fold $f$ of the training set. Cell $a$, $b$ of the upper triangular matrix $\mathbf{C}$ is set to 1 if the words $a$, $b$ co-occur in any sentence of the training part of $f$. For each sentence in the test part of $f$, we count the number $u$ of word pairs that have a positive entry in $\mathbf{C}$. As a sentence of length $n$ can maximally have $\binom{n}{2}$ positive entries in $\mathbf{C}$, the evaluation metrics $r = u/\binom{n}{2}$ is limited to [0, 1], with 0 meaning no hits in $\mathbf{C}$ and 1 that all word pairs of a sentence also co-occur in sentences of the training part. We fit another linear model with sentence-wise LAS as the predicted variable and $r$ as the predictor, and obtain a fitted model with a slope of 0.19 (t = 29.6, p 0.001) and an intercept of 0.73 (t = 194.9, p 0.001). If this model would perfectly fit the data (which it does not, $R^2 = 0.06303$), a sentence all word pairs of which co-occur in the training set would therefore have 19% more LAS than one without any pairs in the training set.

[15] For the related problem of case syncretism in dependency parsing see Seeker and Kuhn (2013) and Seeker (2016).

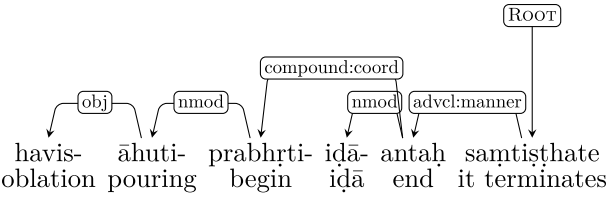


**Fig. 2** Example for compounding at *Āpastamba-Śrautasātra* 7.23.2: '[A ritual discussed in the preceding text] terminates [with a series of ritual actions] that start with pouring an oblation and end with [eating] the iḍā.' Words in square brackets need to be supplemented in thought

**Table 12** Evaluation of compound annotation

| Type | UAS | LAS | Len. | Total | UAS | Type | Total | UAS |
|---|---|---|---|---|---|---|---|---|
| (a) UAS and LAS in compounds (non-terminal and term. members) and in regular text | | | (b) UAS of inner members, split by compound len(gths) | | | (c) UAS of compounds of length 3, split by their internal branching types | | |
| Non-term. | 91.9 | 72.7 | 2 | 1150 | 94.3 | $[w_1\, w_2]\, w_3$ | 41 | 92.7 |
| Term. | 71.5 | 62.1 | 3 | 100 | 89.0 | $w_1\, [w_2\, w_3]$ | 8 | 25.0 |
| Reg. | 84.4 | 78.6 | > 3 | 151 | 52.3 | | | |

### 4.4.6 Compounds

The oldest metrical texts of the Vedic canon use nominal compounding quite sparingly, mostly in the form of two-word compounds with deverbal heads such as *vāja-sṛt* 'running to the prize (of a contest)' (on which see e.g. Wackernagel, 1905, 174ff.). Subsequent historical layers of VS see an increasing use of longer compounds, which can express, among others, coordination or subordination and serve as complex 'adjectival' modifiers of other nouns (for an overview see Lowe, 2015). Because the non-terminal members of compounds usually do not show inflectional endings, determining the compositional structure must largely rely on the semantics and the order of the compounded words which makes this task different from determining the syntactic structure of a 'regular' sentence. The example in Fig. 2 which is taken from a late Vedic manual of the solemn ritual illustrates some of these issues. The basic structure of the compound is a coordination of the two nouns *prabhṛti-* 'start, begin' and *anta-* 'end' that are modified by the nouns *āhuti-* 'pouring' and *iḍā-*, a technical term for food consumed during the sacrifice. The deverbal noun *āhuti-* is further modified by another noun that denotes the substance being poured out (*havis-*) and that we connect with the label `obj` due to the deverbal derivation of its syntactic head. Determining the structure of this compound thus merely relies on the derivation and semantics of the compounded words and on knowledge about the ritualistic context.

The difference between compounds and regular text is reflected in the values reported in Table 12a which contrasts LAS and UAS of (non-)terminal members of compounds with those of words in regular text. The surprisingly high UAS of non-terminal members can easily be explained by the fact that the majority of

Vedic compounds consists of only two words, which makes structure detection a trivial task. This idea finds support in Table 12b, where the UAS for non-terminal members is further split by the lengths of the compounds and which shows that the UAS drops rapidly for longer compounds.[16] This effect can already be observed in compounds of length 3. The three words $w_1, w_2, w_3$ can be bracketed in two ways, namely $[w_1\ w_2]\ w_3$ (left–right chaining) and $w_1\ [w_2\ w_3]$ ($w_1$ and $w_2$ are dependents of the terminal member). Table 12c shows that $[w_1\ w_2]\ w_3$ is both more frequent and less error-prone, whereas $w_1\ [w_2\ w_3]$ seems to be challenging to analyze. This pattern comprises expressions such as *bahu-[parṇa-śākham]* 'many-[leaf-branch]', i.e. '(a plant) that has many leaves and branches' where the decision for the correct bracketing again relies on semantic information about the two coordinated words. This kind of information could only be obtained from much larger corpora than we currently have for (Vedic) Sanskrit. Another relevant point is the LAS of the terminal members (see row 3 of Table 12a). When we compare the label-wise LAS of compound terminals with that of words in regular text using a Fisher test of the respective count data, we find that the LAS of the labels `acl`, `nsubj`, `nmod` and `xcomp` (ordered by their p-values) are lower than those for regular words at a significance level of 10%. Though not significant in a strict sense, this finding nevertheless points to the syntactic flexibility of compounds, which can function as adnonimals (resulting in `acl`) or as nouns (`nsubj`, `xcomp`, `nmod`).

#### 4.4.7 Mantra citations and ellipsis

A peculiarity of many middle and late Vedic texts is the ubiquitous use of mantras, i.e. short citations from early metrical texts such as the *Rigveda*, that accompany the individual steps of a sacrifice. The correct treatment of mantras is highly relevant when parsing Vedic because 7.9% of all words in the current version of the VTB belong to such citations. Mantras complicate the syntactic analysis for two reasons. First, only the first few words of a mantra are cited in many cases, because the authors of the ritual manuals assumed that the participants of a ritual know the relevant mantra collections by heart. This practice results in truncated, elliptic expressions which are difficult to annotate in a dependency framework. Second, most cited mantras were probably composed centuries before the texts citing them. Annotating them as direct speech would mix different historical levels of Vedic and therefore reduce the usefulness of the annotated data for diachronic linguistic studies. We therefore decided to annotate all cited mantras with the dependency label `flat` in chaining annotation.

Deciding whether an utterance is a mantra or regular direct speech requires extrasentential information that a parsing algorithm does not have access to. Since, however, the occurrences of mantras in the major Vedic texts have been collected systematically in Bloomfield's *Vedic Concordance* (Bloomfield, 1906) and a digital version of this resource is available (Franceschini & Bloomfield, 2007), it is possible

[16] Note that the occasional mislabeling of two-word compounds is due to issues in the data preparation routine and especially the limited coverage of Bloomfield (1906); see Sect. 4.4.7.

**Table 13** Results of the layer-wise cross-validation

| Layer | UAS | LAS | Words |
|---|---|---|---|
| 2-MA | 69.2 | 59 | 11,947 |
| 3-PO | 82.9 | 75.9 | 19,782 |
| 4-PL | 77.8 | 69.4 | 34,602 |
| 5-SU | 72.3 | 62 | 24,636 |

See Sect. 3.1 for the abbreviations in column 1

to detect mantra citations automatically before parsing a piece of text. We set the lexical and morpho-syntactic information of words in mantras to uninformative dummy values (e.g. replacing the gold POS tag by the dummy value MANTRA) thereby helping the labeler to annotate mantra sequences with the desired label `flat`.

One fundamental problem of this pre-processing step is that some late Vedic texts were not edited at the time when Bloomfield compiled his concordance so that mantras in such texts are not flagged with the typical dummy values. Consequently, the parsing algorithm tries to label them as direct speech, leading to a substantial divergence from the gold standard where they are labelled as `flat`. This problem is reflected in the LAS of 79% for mantras not recorded by Bloomfield that is clearly below that of 88% for recorded ones. Errors in the last category are mainly due to wrong decisions about how a mantra is connected with the rest of a sentence. While mantras should be connected to some kind of speech verb using `ccomp`, these verbs are often omitted, resulting in arcs labelled with `orphan` which can be problematic for the parser.

Elliptic constructions are not restricted to mantra citations, but are widely used as a rhetoric device and as a means for reducing the length of a text, a trend that finds an early culmination in the famous Sanskrit grammar called *Aṣṭādhyāyī*. This trend is also reflected in the proportions of words labelled as `orphan` in the VTB which rises from about 2% in its early layers to 4.5% in the late manuals of the ritual. As can be expected, ellipsis significantly deteriorates the accuracy of the parser. When we compare the LAS of sentences with at least one `orphan` to those without orphans using a t-test, we obtain a highly significant test statistics of $t = -10.484$ (DF = 1720.7, $p < 0.001$). Similar effects can be observed when we additionally control for the sentence length using an ANCOVA with interaction between the two predictors.[17]

### 4.4.8 Diachronic layers

Problems with the cross-domain application of parsers are well known, see e.g. Krishna et al. (2020) for CS, Sorokin et al. (2020) for Russian and Mambrini and Passarotti (2012) for Ancient Greek. The fivefold diachronic structure that we

[17] Coefficients: orphans: $F(1) = 58.4, p < 0.001$; sentence lengths: $F(1) = 200.8, p < 0.001$; interaction: $F(1) = 32.1, p < 0.001$.

impose on our data (see Sect. 3.1) makes it possible to control for time and genres at the same time because this structure was partly deduced from stylistic criteria. For the cross-domain experiments, we use all texts of one layer as the test set and train the parser with the remaining texts. The results in Table 13 show that the only layer that achieves scores comparable to those of the full model (see Sect. 4.3) is the old prose (3-PO). As these texts often consist of short sentences with a clear left-branching structure, this outcome is not really surprising, and analogous considerations apply to the level of late prose. The Sūtra literature (5-SU) and especially the early metrical texts (2-MA), on the other hand, fall far below the scores of the full model. While there exist continuities between the two prose layers and the Sūtra literature on the levels of content, vocabulary and style, the early metrical texts belong to a completely different domain: They feature hymns that address gods and try to cope with the difficulties of life; sentences show a high degree of non-configurational constructions (see Sect. 4.4.2 and esp. Table 7); and their vocabulary contains many rare, semantically unclear words that are not found in any later text. Given the complex interplay of linguistic factors and the limited size of the Vedic corpus, we are sceptical whether domain-invariant architectures as proposed, for instance, by Ganin and Lempitsky (2015) could provide any substantial advantages.

## 5 Summary

This paper has presented the first data-driven parser of Vedic Sanskrit and, more generally, an in-depth evaluation of a modern graph-based parser on an ancient language with limited resources. Apart from making available a substantially extended version of the VTB, which we are planning to integrate in the next UD release, our paper has made two important contributions. First, contextualized embeddings seem not to be able to show their full potential when only a limited corpus is available. Instead, a combination of static embeddings and manually validated morpho-syntactic information achieves clearly better attachment scores. While such a setting may not appear feasible for many modern languages, where large manually annotated resources are not available, one should keep in mind that corpora of ancient languages often originate in philological research environments where linguistic gold data are indispensable for scholarly research. The results of our paper may thus show a viable approach for training good parsing models under limited resources.

Second, the error evaluation in Sect. 4.4 often aligns with results reported in previous research. While many factors influence the parsing accuracy, often to a highly significant degree, it is complicated—or even impossible—to single out the main responsible(s). We are nevertheless convinced that the details provided in this section (e.g. the lexical co-occurrence measure in Sect. 4.4.5) can be useful for designing an error labeler that distinguishes between correct and wrong parses, similar in vein to the model discussed by Bollmann amb Søgaard (2021). Such a labeler can be used for data augmentation (see e.g. McClosky et al., 2006), but also as part of a data processing pipeline that generates reliable input for higher level linguistic studies in VS.

**Acknowledgements** Research for this paper was funded by the German Federal Ministry of Education and Research, FKZ 01UG2121.

**Funding** Open Access funding enabled and organized by Projekt DEAL.



## References

Andor, D., Alberti, C., Weiss, D., Severyn, A., Presta, A., Ganchev, K., Petrov, S., & Collins M. (2016, August). Globally normalized transition-based neural networks. In *Proceedings of the 54th annual meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, Berlin, Germany, 2016 (pp. 2442–2452). Association for Computational Linguistics.

Ballesteros, M., Dyer, C., & Smith, N. A. (2015, September). Improved transition-based parsing by modeling characters instead of words with LSTMs. In *Proceedings of the 2015 conference on empirical methods in natural language processing*, 2015, Lisbon, Portugal (pp. 349–359). Association for Computational Linguistics.

Bamman, D., & Burns, P. J. (2020). Latin BERT: A contextual language model for classical philology. arXiv:2009.10053 [cs.CL]

Bamman, D., Mambrini, F., & Crane, G. (2010). An ownership model of annotation: The Ancient Greek Dependency Treebank. In *Proceedings of the eighth international workshop on treebanks and linguistic theories (TLT8)*, December 4–5, 2009, Milan, Italy.

Bansal, R., Choudhary, H., Punia, R., Schenk, N., Pagé-Perron, É., & Dahl, J. (2021, August). How low is too low? A computational perspective on extremely low-resource languages. In *Proceedings of the 59th annual meeting of the Association for Computational Linguistics and the 11th international joint conference on natural language processing: Student research workshop*, Online, 2021 (pp. 44–59). Association for Computational Linguistics.

Berzak, Y., Kenney, J., Spadine, C., Wang, J. X., Lam, L., Mori, K. S., Garza, S., & Katz B. (2016). Universal dependencies for learner English. In *Proceedings of the 54th annual meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, Berlin, Germany, 2016 (pp. 737–746). Association for Computational Linguistics.

Biagetti, E., Hellwig, O., Ackermann, E., Widmer, P., & Scarlata, S. (2021). Evaluating syntactic annotation of ancient languages. Lessons from the Vedic Treebank. *Old World, 1*, 1–32.

Bloomfield, M. (1906). *A Vedic concordance, being an alphabetic index to every line of every stanza of the published Vedic literature and to the liturgical formulas thereof, that is, an index to the Vedic mantras, together with an account of their variations in the different Vedic books*. Harvard University Press.

Bojanowski, P., Grave, E., Joulin, A., & Mikolov, T. (2017). Enriching word vectors with subword information. *Transactions of the Association for Computational Linguistics, 5,* 135–146.

Bollmann, M., & Søgaard, A. (2021). Error analysis and the role of morphology. In *Proceedings of the EACL*, 2021 (pp. 1887–1900).

Burrow, T. (1955). *The Sanskrit language*. Faber and Faber.

Celano, G. G. A. (2019). The dependency treebanks for ancient Greek and Latin. In M. Berti (Ed.), *Digital classical philology: Ancient Greek and Latin in the Digital Revolution* Age of access? (1st ed., Vol. 10) Grundfragen der Informationsgesellschaft (pp. 279–298). De Gruyter Saur.

Chang, K. W., He, H., Daumé, H., Langford, J., & Ross, S. (2016). A credit assignment compiler for joint prediction. In *Proceedings of the 30th international conference on neural information processing systems, NIPS'16*, 2016, Red Hook, NY, USA (pp. 1713–1721). Curran Associates, Inc.

Che, W., Liu, Y., Wang, Y., Zheng, B., & Liu, T. (2018, October). Towards better UD parsing: Deep contextualized word embeddings, ensemble, and treebank concatenation. In *Proceedings of the CoNLL 2018 shared task: Multilingual parsing from raw text to universal dependencies*, 2018, Brussels, Belgium (pp. 55–64). Association for Computational Linguistics.

Chen, D., & Manning, C. (2014, October). A fast and accurate dependency parser using neural networks. In *Proceedings of the 2014 conference on empirical methods in natural language processing (EMNLP)*, 2014, Doha, Qatar (pp. 740–750). Association for Computational Linguistics.

Conneau, A., Khandelwal, K., Goyal, N., Chaudhary, V., Wenzek, G., Guzmán, F., Grave, E., Ott, M., Zettlemoyer, L., & Stoyanov, V. (2020, July). Unsupervised cross-lingual representation learning at scale. In *Proceedings of the 58th annual meeting of the Association for Computational Linguistics*, 2020, Online (pp. 8440–8451). Association for Computational Linguistics.

Delbrück, B. (1888). *Altindische Syntax*. Verlag der Buchhandlung des Waisenhauses.

Devlin, J., Chang, M. W., Lee, K., & Toutanova, K. (2019, June). BERT: Pre-training of Deep Bidirectional Transformers for language understanding. In *Proceedings of the 2019 conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, 2019 (pp. 4171–4186).

Dozat, T., & Manning, C. D. (2017). Deep biaffine attention for neural dependency parsing. In *5th International conference on learning representations*, 2017 (pp. 1–8).

Dunkel, G. E. (2014). *Lexikon der indogermanischen Partikeln und Pronominalstämme*. Universitätsverlag Winter.

Falk, H. (1993). *Schrift im alten Indien: Ein Forschungsbericht mit Anmerkungen*. Gunter Narr Verlag.

Franceschini, M., & Bloomfield, M. (2007). *An updated Vedic concordance: Maurice Bloomfield's A Vedic Concordance enhanced with new material taken from seven Vedic texts*. Harvard University Press.

Ganin, Y., & Lempitsky, V. (2015). Unsupervised domain adaptation by backpropagation. In *International conference on machine learning*, 2015 (pp. 1180–1189).

Gonda, J. (1975). *Vedic literature: Samhitās and Brāhmanas*. A History of Indian literature; v. 1: Veda and Upanishads; fasc. 1. Harrassowitz.

Gonda, J. (1977). *The Ritual Sūtras*. A History of Indian literature; v. 1: Veda and Upanishads; fasc. 2. Harrassowitz.

Goyal, P., & Huet, G. (2016). Design and analysis of a lean interface for Sanskrit corpus annotation. *Journal of Language Modelling, 4*, 145–182.

Gren-Eklund, G. (1978). *A study of nominal sentences in the oldest Upaniṣads*. Almqvist and Wiksell (Stockholm).

Gulordava, K., & Merlo, P. (2015). Diachronic trends in word order freedom and dependency length in dependency-annotated corpora of Latin and ancient Greek. In *Proceedings of the DepLing*, 2015 (pp. 121–130).

Gupta, A., Krishna, A., Goyal, P., & Hellwig, O. (2020). Evaluating neural morphological taggers for Sanskrit. arXiv:2005.10893 [cs.CL]

Hale, K. (1983). Warlpiri and the grammar of non-configurational languages. *Natural Language and Linguistic Theory, 1*, 5–47.

Hashimoto, K., Xiong, C., Tsuruoka, Y., & Socher, R. (2017). A joint many-task model: Growing a neural network for multiple NLP tasks. In *Proceedings of the EMNLP*, 2017 (pp. 1923–1933).

Hellwig, O. (2010–2021). The Digital Corpus of Sanskrit. http://www.sanskrit-linguistics.org/dcs/index.php

Hellwig, O. (2016). Detecting sentence boundaries in Sanskrit texts. In *Proceedings of the COLING*, 2016 (pp. 288–297).

Hellwig, O., & Nehrdich, S. (2018). Sanskrit word segmentation using character-level recurrent and convolutional neural networks. In *Proceedings of the 2018 conference on empirical methods in natural language processing*, 2018, Brussels (pp. 2754–2763). Association for Computational Linguistics.

Hellwig, O., Scarlata, S., Ackermann, E., & Widmer, P. (2020). The treebank of Vedic Sanskrit. In N. Calzolari, F. Béchet, P. Blache, K. Choukri, C. Cieri, T. Declerck, S. Goggi, H. Isahara, B. Maegaard, J. Mariani, H. Mazo, A. Moreno, J. Odijk & S. Piperidis (Eds.), *Proceedings of the LREC*, 2020 (pp. 5139–5148).

Hock, H. H. (1997a). Chronology or genre? Problems in Vedic syntax. In M. Witzel (Ed.), *Inside the texts—Beyond the texts: New approaches to the study of the Vedas* (pp. 103–126). Harvard University.

Hock, H. H. (1997b). Nexus and 'extraclausality' in Vedic, or '*sa*-figé' all over again: A historical (re)examination. In *Historical, Indo-European, and lexicographical studies. A Festschrift for Ladislav Zgusta on the occasion of his 70th Birthday, ed. by Hans Henrich Hock, Trends in linguistics. Studies and monographs* (Vol. 90, pp. 49–78). Mouton de Gruyter.

Hock, H. H. (2001). Genre, discourse, and syntax in early Indo-European, with emphasis on Sanskrit. In S. C. Herring, P. Van Reenen, & L. Schøsler (Eds.), *Textual parameters in older languages* (pp. 163–196). John Benjamins Publishing.

Hock, H. H. (2013). Some issues in Sanskrit syntax. In P. M. Scharf & G. Huet (Eds.), *Proceedings of the seminar on Sanskrit syntax and discourse structures*, 2013, Paris (pp. 1–52).

Husain, S., & Agrawal, B. (2012). Analyzing parser errors to improve parsing accuracy and to inform tree banking decisions. *Linguistic Issues in Language Technology, 7*(1), 1–20.

Keersmaekers, A. (2019). Creating a richly annotated corpus of papyrological Greek: The possibilities of natural language processing approaches to a highly inflected historical language. *Digital Scholarship in the Humanities, 35*(1), 67–82.

Keydana, G., & Luraghi, S. (2012). Definite referential null objects in Vedic Sanskrit and Ancient Greek. *Acta Linguistica Hafniensia, 44*(2), 116–128.

Kiperwasser, E., & Goldberg, Y. (2016). Simple and accurate dependency parsing using bidirectional LSTM feature representations. *Transactions of the Association for Computational Linguistics, 4*, 313–327.

Kolachina, P., Riedl, M., & Biemann, C. (2017, May). Replacing OOV words for dependency parsing with distributional semantics. In *Proceedings of the 21st Nordic conference on computational linguistics*, 2017, Gothenburg, Sweden (pp. 11–19). Association for Computational Linguistics.

Krishna, A., Gupta, A., Garasangi, D., Sandhan, J., Satuluri, P., & Goyal, P. (2020). Neural approaches for data driven dependency parsing in Sanskrit. arXiv:2004.08076 [cs.CL]

Krishna, A., Santra, B., Gupta, A., Satuluri, P., & Goyal, P. (2021). A graph-based framework for structured prediction tasks in Sanskrit. *Computational Linguistics, 46*(4), 785–845.

Kübler, S., McDonald, R., & Nivre, J. (2009). *Dependency parsing*. Morgan & Claypool Publishers.

Kuhlmann, M., & Nivre, J. (2006). Mildly non-projective dependency structures. In *Proceedings of the COLING/ACL*, 2006 (pp. 507–514).

Kulkarni, A. (2013). A deterministic dependency parser with dynamic programming for Sanskrit. In *Proceedings of the second international conference on dependency linguistics (DepLing 2013)*, 2013 (pp. 157–166).

Kulkarni, A. (2021). Sanskrit parsing following Indian theories of verbal cognition. *Transactions on Asian and Low-Resource Language Information Processing, 20*(2), 1–38.

Kulmizev, A., de Lhoneux, M., Gontrum, J., Fano, E., & Nivre, J. (2019). Deep contextualized word embeddings in transition-based and graph-based dependency parsing—A tale of two parsers revisited. In *Proceedings of the 2019 EMNLP*, 2019 (pp. 2755–2768).

Kümmel, M. J. (2000). *Das Perfekt im Indoiranischen. Eine Untersuchung der Form und Funktion einer ererbten Kategorie des Verbums und ihrer Entwicklung in den altindoiranischen Sprachen*. Reichert.

Lample, G., & Conneau, A. (2019). Cross-lingual language model pretraining. In *33rd Conference on neural information processing systems (NeurIPS 2019)*, 2019 (pp. 1–11).

Lehmann, W. P. (1974). *Proto-Indo-European syntax*. University of Texas Press.

Liu, Y., Ott, M., Goyal, N., Du, J., Joshi, M., Chen, D., Levy, O., Lewis, M., Zettlemoyer, L., & Stoyanov, V. (2019). RoBERTa: A robustly optimized BERT pretraining approach. arXiv:1907.11692 [cs.CL]

Lowe, J. J. (2015). The syntax of Sanskrit compounds. *Language, 91*(3), 71–115.

Majidi, S., & Crane, G. (2014). Human and machine error analysis on dependency parsing of Ancient Greek texts. In *Proceedings of the 14th ACM/IEEE-CS joint conference on digital libraries*, JDCL '14, 2014. IEEE Press.

Mambrini, F., & Passarotti, M. (2012). Will a parser overtake Achilles? First experiments on parsing the Ancient Greek Dependency Treebank. In *Eleventh international workshop on treebanks and linguistic theories*, 2012 (pp. 133–144).

McClosky, D., Charniak, E., & Johnson, M. (2006, June). Effective self-training for parsing. In *Proceedings of the human language technology conference of the NAACL, main conference*, 2006, New York City, USA (pp. 152–159). Association for Computational Linguistics.

McDonald, R. (2006). Discriminative training and spanning tree algorithms for dependency parsing. PhD Thesis, University of Pennsylvania.

McDonald, R., & Nivre, J. (2007). Characterizing the errors of data-driven dependency parsers. In *Proceedings of the EMNLP*, 2007 (pp. 122–131).

McDonald, R., & Nivre, J. (2011). Analyzing and integrating dependency parsers. *Computational Linguistics, 37*(1), 197–230.

Mikolov, T., Sutskever, I., Chen, K., Corrado, G. S., & Dean, J. (2013). Distributed representations of words and phrases and their compositionality. In C. J. C. Burges, L. Bottou, M. Welling, Z. Ghahramani, & K. Q. Weinberger (Eds.), *Advances in neural information processing systems* (Vol. 26, pp. 3111–3119). Curran Associates, Inc.

More, A., Seker, A., Basmova, V., & Tsarfaty, R. (2019). 03. Joint transition-based models for morpho-syntactic parsing: Parsing strategies for MRLs and a case study from Modern Hebrew. *Transactions of the Association for Computational Linguistics, 7*, 33–48.

Mrini, K., Dernoncourt, F., Tran, Q. H., Bui, T., Chang, W., & Nakashole, N. (2020, November). Rethinking self-attention: Towards interpretability in neural parsing. In *Findings of the Association for Computational Linguistics: EMNLP 2020*, 2020, Online (pp. 731–742). Association for Computational Linguistics.

Nivre, J., De Marneffe M. C., Ginter F., Goldberg Y., Hajic J., Manning C. D., McDonald R., Petrov S., Pyysalo S., Silveira N., Tsarfaty R., Zeman D. (2003, April). An efficient algorithm for projective dependency parsing. In *Proceedings of the eighth international conference on parsing technologies*, 2003, Nancy, France (pp. 149–160).

Nivre, J., De Marneffe, M. C., Ginter, F., Goldberg, Y., Hajic, J., Manning, C. D., McDonald, R., Petrov, S., Pyysalo, S., & Silveira, N., Tsarfaty, R., & Zeman, D. (2016). Universal dependencies v1: A multilingual treebank collection. In *Proceedings of the tenth international conference on language resources and evaluation (LREC'16)*, 2016 (pp. 1659–1666).

Olivelle, P. (1998). *The early Upaniṣads: Annotated text and translation*. Oxford University Press.

Passarotti, M. (2019). The project of the Index Thomisticus Treebank, In M. Berti (Ed.), *Digital classical philology: Ancient Greek and Latin in the Digital Revolution*, Age of access? Grundfragen der Informationsgesellschaft (Vol. 10, pp. 299–319). De Gruyter Saur.

Peters, M. E., Neumann, M., Iyyer, M., Gardner, M., Clark, C., Lee, K., & Zettlemoyer, L. (2018, June). Deep contextualized word representations. In *Proceedings of the 2018 conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)*, 2018, New Orleans, Louisiana (pp. 2227–2237). Association for Computational Linguistics.

Ponti, E. M., & Luraghi, S. (2018). Non-configurationality in diachrony. *Diachronica, 35*(3), 367–392.

Rehbein, I., Steen, J., Do, B. N., & Frank, A. (2017). Universal Dependencies are hard to parse—Or are they? In *Proceedings of the fourth international conference on dependency linguistics (DepLing 2017)*, 2017 (pp. 218–228).

Reinöhl, U. (2016). *Grammaticalization and the rise of configurationality in Indo-Aryan*. Oxford University Press.

Reinöhl, U. (2020). Continuous and discontinuous nominal expressions in flexible (or "free'') word order languages: Patterns and correlates. *Linguistic Typology, 24*(1), 71–111.

Renou, L. (1947). *Les écoles védiques et la formation du Véda*. Imprimerie Nationale.

Rotman, G., & Reichart, R. (2019). Deep contextualized self-training for low resource dependency parsing. *Transactions of the Association for Computational Linguistics, 7*, 695–713.

Sandhan, J., Adideva, O., Komal, D., Behera, L., & Goyal, P. (2021). Evaluating neural word embeddings for Sanskrit. arXiv:2104.00270 [cs.CL].

Sanka, U. (2015). A study of Kāraka-demand of some Dhātus, based on meaning, following Śābdabodha for machine translation. PhD Thesis, Rastriya Sanskrit Vidyapeetha.

Seeker, W. (2016). Modeling the interface between morphology and syntax in data-driven dependency parsing. PhD Thesis, Universität Stuttgart.

Seeker, W., & Kuhn, J. (2013). Morphological and syntactic case in statistical dependency parsing. *Computational Linguistics, 39*(1), 23–55. https://doi.org/10.1162/COLI_a_00134

Simonenko, A., Crabbé, B., & Prévost, S. (2018). Text form and grammatical changes in Medieval French: A treebank-based diachronic study. *Diachronica, 35*(3), 393–428.

Sorokin, A., Smurov, I., & Kirianov, D. (2020). Tagging and parsing of multidomain collections. In *Proceedings of the international conference "Dialogue 2020"*, 2020 (pp. 670–683).

Straka, M. (2018, October). UDPipe 2.0 prototype at CoNLL 2018 UD shared task. In *Proceedings of the CoNLL 2018 shared task: Multilingual parsing from raw text to universal dependencies*, 2018, Brussels, Belgium (pp. 197–207). Association for Computational Linguistics.

Straka, M., Straková, J., & Hajič, J. (2019). Evaluating contextualized embeddings on 54 languages in POS tagging, lemmatization and dependency parsing. arXiv:1908.07448 [cs.CL]

Sukhareva, M., Fuscagni, F., Daxenberger, J., Görke, S., Prechel, D., & Gurevych, I. (2017, August). Distantly supervised POS tagging of low-resource languages under extreme data sparsity: The case of Hittite. In *Proceedings of the joint SIGHUM workshop on computational linguistics for cultural heritage, social sciences, humanities and literature*, 2017, Vancouver, Canada (pp. 95–104). Association for Computational Linguistics.

Tsarfaty, R., Bareket, D., Klein, S., & Seker, A. (2020). From SPMRL to NMRL: What did we learn (and unlearn) in a decade of parsing morphologically-rich languages (MRLs)? In *Proceedings of the 58th annual meeting of the Association for Computational Linguistics*, 2020 (pp. 7396–7408).

Tsarfaty, R., Seddah, D., Kübler, S., & Nivre, J. (2013). Parsing morphologically rich languages: Introduction to the special issue. *Computational Linguistics, 39*(1), 15–22.

Viti, C. (2010). The information structure of OVS in Vedic. In G. Ferraresi & R. Lühr (Eds.), *Diachronic studies on information structure* (pp. 37–62). De Gruyter.

Wackernagel, J. (1905). *Altindische Grammatik. Band II, 1: Einleitung zur Wortlehre. Nominalkomposition*. Vandenhoeck & Ruprecht.

Weiss, D., Alberti, C., Collins, M., & Petrov, S. (2015, July). Structured training for neural network transition-based parsing. In *Proceedings of the 53rd annual meeting of the Association for Computational Linguistics and the 7th international joint conference on natural language processing (Volume 1: Long Papers)*, 2015, Beijing, China (pp. 323–333). Association for Computational Linguistics.

Witzel, M. (1989). Tracing the Vedic dialects. In C. Caillat (Ed.), *Dialectes dans les littératures indoaryennes* (pp. 97–265). Collège de France.

Wu, S., & Dredze, M. (2020). Are all languages created equal in multilingual BERT? In S. Gella, J. Welbl, M. Rei, F. Petroni, P. S. H. Lewis, E. Strubell, M. J. Seo & H. Hajishirzi (Eds.), *Proceedings of the 5th workshop on representation learning for NLP, RepL4NLP@ACL 2020*, Online, July 9, 2020 (pp. 120–130). Association for Computational Linguistics.

Zeldes, A., & Abrams, M. (2018). The Coptic Universal Dependency Treebank. In *Proceedings of the second workshop on universal dependencies (UDW 2018)*, 2018, Brussels, Belgium (pp. 192–201). Association for Computational Linguistics.

Zhang, X., Zhao, J., & LeCun, Y. (2015). Character-level convolutional networks for text classification. In C. Cortes, N. Lawrence, D. Lee, M. Sugiyama, & R. Garnett (Eds.), *Advances in neural information processing systems* (Vol. 28, pp. 649–657). Curran Associates, Inc.

Zhou, J., & Zhao, H. (2019, July). Head-driven phrase structure grammar parsing on Penn Treebank. In *Proceedings of the 57th annual meeting of the Association for Computational Linguistics*, 2019, Florence, Italy (pp. 2396–2408). Association for Computational Linguistics.