Free as in Free Word Order: An Energy Based Model for Word Segmentation and Morphological Tagging in Sanskrit

Amrith Krishna[1], Bishal Santra[1], Sasi Prasanth Bandaru[2]*, Gaurav Sahu[3]*
Vishnu Dutt Sharma[4]†, Pavankumar Satuluri[5]† and Pawan Goyal[1]
[1]Dept. of Computer Science and Engineering, IIT Kharagpur
[2]Flipkart, India [3]School of Computer Science, University of Waterloo
[4]American Express India Pvt Ltd [5]Chinmaya Vishwavidyapeeth
amrith@iitkgp.ac.in, bsantraigi@gmail.com,
pawang@cse.iitkgp.ernet.in

Abstract

The configurational information in sentences of a free word order language such as Sanskrit is of limited use. Thus, the context of the entire sentence will be desirable even for basic processing tasks such as word segmentation. We propose a structured prediction framework that jointly solves the word segmentation and morphological tagging tasks in Sanskrit. We build an energy based model where we adopt approaches generally employed in graph based parsing techniques (McDonald et al., 2005a; Carreras, 2007). Our model outperforms the state of the art with an F-Score of 96.92 (percentage improvement of 7.06%) while using less than one tenth of the task-specific training data. We find that the use of a graph based approach instead of a traditional lattice-based sequential labelling approach leads to a percentage gain of 12.6% in F-Score for the segmentation task.[1]

1 Introduction

Sanskrit, a morphologically-rich and a free word order language (Kulkarni et al., 2015), poses a series of challenges even for automation of basic processing tasks such as word segmentation. The recent surge in the digitisation efforts for archiving the works ranging from the pre-classical to modern times (Hellwig, 2010-2016) has led to a growing demand for such tools (Goyal et al., 2012; Huet, 2006). We propose a structured prediction approach that jointly solves the word segmentation and morphological tagging tasks for Sanskrit.

The computational problems arising from the mechanical treatment of Sanskrit fall somewhere between speech recognition and the analysis of written text (Huet, 2005). For instance, consider Figure 1a which shows all the phonetically valid word splits for a Sanskrit poetic verse[2]. The written representation in Sanskrit is actually a phonemic stream (Huet, 2005). The constructions often undergo phonetic transformations at the juncture of successive words, similar to what one observes in connected speech (Morris et al., 2004; Shieber and Tao, 2003). These transformations obscure the word boundaries and often modify the phones at these word boundaries. In Sanskrit, these transformations get reflected in writing as well. This is primarily due to the presence of an advanced discipline of phonetics in Sanskrit which explicitly described euphonic assimilation as *sandhi* (Goyal and Huet, 2016). For instance, words prefixed with numbers 14 and 15 in Figure 1a are valid candidates in spite of the phonetic differences they posses from that of the original sentence.

Sanskrit is rich with syncretisms (Crystal, 2011) and homonyms. For example, the surface form '*satī*', prefixed with numbers 6 and 10, are homonyms, while the root '*satya*' generates identical surface form for three different morphological classes leading to syncretism (1 to 3 in Figure 1a). Hence, in addition to segmentation, the morphological analysis of the segmented word forms will be critical for reducing the ambiguity in further downstream tasks such as syntactic analysis. The sentence construction in the language follows weak non-projectivity (Havelka, 2007) permitting the words to have a relatively free word order structure (Kulkarni et al., 2015). The language is all the more lenient for poetic constructions (Scharf et al., 2015; Kulkarni et al., 2015), where arranging the words to adhere to metri-

*Work done while the authors were at IIT Kharagpur

†Part of the work was done while the authors were at IIT Kharagpur

[1]The code and the pretrained edge vectors (§3) used in this work are available at https://zenodo.org/record/1035413#.W35s8hjhUUs

[2]A saying from *subhāṣitam* text: One should tell the truth, one should say kind words; one should neither tell harsh truths, nor flattering lies; this is a rule for all times.

Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing, pages 2550–2561
Brussels, Belgium, October 31 - November 4, 2018.

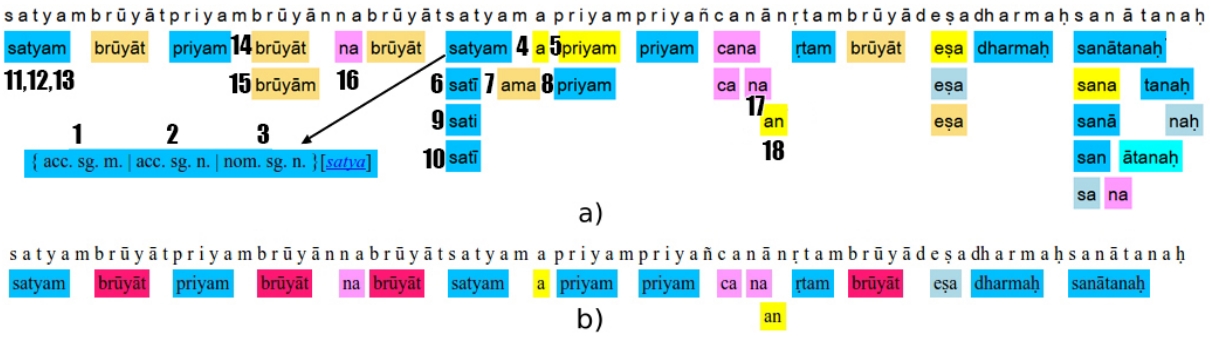

Figure 1: a) All the phonetically valid segmentations (link) for '*satyaṃbrūyātpriyaṃbrūyānnabrūyātsatyamapriyaṃpriyaṃcanānṛtambrūyādeṣadharmaḥsanātanaḥ*' from (*subhāṣitam*) as output by Sanskrit Heritage Reader (SHR) and b) correct segmentation selected from the candidate space.

cal constraints is a bigger concern (Melnad et al., 2013). Hence, the whole input context is desirable when making each prediction at the output (Bahdanau et al., 2015), even for preliminary tasks such as segmentation in Sanskrit (Reddy et al., 2018).

The word splits in Figure 1 are based on the analysis by a lexicon driven analyser, Sanskrit Heritage Reader (SHR)[3]. A total of 1,056 combinations can be formed from the word splits, such that each of those combinations is a solution which covers the entire input. We call such a solution as an '*exhaustive segmentation*'. Out task is to find an '*exhaustive segmentation*', which is also semantically valid. Figure 1b shows the semantically valid solution for the sentence.

We propose our structured prediction framework as an energy based model (LeCun et al., 2006). Considering the free word-orderness, morphological richness and the phenomena of *Sandhi* in Sanskrit, we adopt a graph based treatment for a given input sentence as shown in Figure 2. All the word splits, as predicted by SHR, are treated as the nodes in the graph. Every pair of nodes that can co-occur in at least one '*exhaustive segmentation*'[4] forms directed edges in both the directions. By construction, any subset of nodes that forms a maximal clique will be an '*exhaustive segmentation*'. We formalise our task as the search for a maximal clique. The graph structure eliminates the sequential nature of the input, while the greedy maximal clique selection inference policy of ours can take the entire input context into consideration. We hypothesise that both of these will be beneficial for processing constructions in Sanskrit.

The major contributions of our work are:

1. We propose the first model that performs both word segmentation and morphological tagging for Sanskrit as suggested by Krishna et al. (2017); the combined task reports an F-Score of 90.45.
2. We obtain an F-Score of 96.92 for the word segmentation task, an improvement of 7.06% over the state of the art, a seq2seq model with attention (Reddy et al., 2018).
3. We achieve the results with less than one-tenth of the training data that Reddy et al. (2018) uses, a desirable outcome for a low resource language such as Sanskrit. The pre-training in the form of morphological constraints to form edge vectors enables this.
4. We propose a scheme that uses the Path Ranking Algorithm (Lao and Cohen, 2010) to automate the feature selection and the feature vector generation for the edges. This eliminates the need for manual feature engineering.

2 Proposed Architecture

Given an input construction, we obtain our search space of possible word splits using SHR as shown in Figure 1. The search space represents all the possible exhaustive segmentations with possible gaps and overlaps between the word splits in each of the exhaustive segmentation (Kudo, 2006; Oerder and Ney, 1993; Wolf and Woods, 1977).[5] In such a setting, representing the search space as a lattice (Kudo, 2006; Smith et al., 2005) has been

[3]Available at http://sanskrit.inria.fr/, SHR is a lexicon-driven segmenter which produces all the valid word splits. An interface is provided for manual selection of a solution (Goyal and Huet, 2016)

[4]For instance, segments 6 and 7 in Figure 1a are connected, while 6 and 9 are not.

[5]The word splits in an exhaustive segmentation often overlap and sometimes leave gaps by virtue of Sandhi. For examples, please refer the §1 supplementary material.

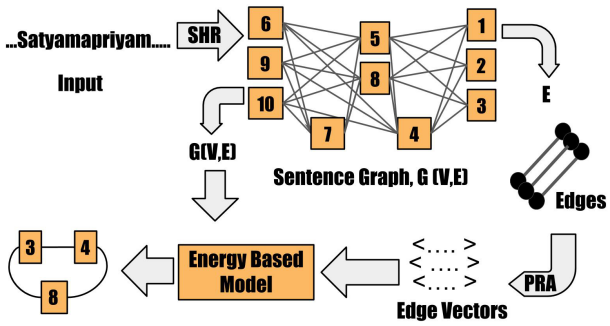

Figure 2: Architecture for the proposed model. Word-splits for '*satyamapriyam*', a sub-sequence of the sentence in Figure 1a are considered here. The nodes are numbered from 1 to 10 and are marked with the same in Figure 1a. For simplicity, we assume that words in nodes 4 to 10 have only one morphological analysis each.

a popular approach for fusional languages (Goldberg and Tsarfaty, 2008; Cohen and Smith, 2007; Hatori et al., 2012). In a lattice, there will be edges only between the adjacent word splits of an exhaustive segmentation. We deviate from this norm in a minor yet fundamental way. In the search space, we choose to add edges between every pair of word splits that are part of a single exhaustive segmentation. Henceforth, we will refer this structure as the sentence graph G. We then employ our minimum cost maximal clique finding energy based model on the sentence graph G. Figure 2 shows the proposed architecture of the model. It shows the sentence graph G for '*satyamapriyam*', a sub-sequence of the sentence in Figure 1a.

Our current design choice results in a denser graph structure as input and a computationally expensive inference. Such a choice requires justification. Currently, there exist digitised versions of texts which spans over a period of 3000 years categorised into pre-classical literature (1500 BCE - 100 BCE), classical literature (300 CE - 800 CE) and modern literature (900 CE to now). Hellwig (2009) assert that the assumption that Sanskrit syntax has remained unchanged over an interval of over 3000 years is not valid. Kulkarni et al. (2015) notes that the constructions in prose generally follow weak non-projectivity (Havelka, 2007; Maier and Lichte, 2011). Kulkarni et al. (2015) also observes that constructions in verses violate weak non-projectivity especially with the adjectival and genitive relations. A large number of texts are written in verses or to complicate things further, they are written as a combination of prose and verse. A lack of consensus among the experts on a common set of rules for works across the different time spans, and the enormous effort in categorising constructions based on their writing styles, motivated us to use this graph construction scheme which is agnostic to word order.

Graph Formation:[6] In Figure 1a, identical surface-forms with the same root are grouped together and displayed as a single entity. But we consider, every unique combination of root, morphological class and the word position in SHR as a separate node in $G(V, E)$. Hence the surface-from *satyam*, appears as 6 separate nodes numbered from 1-3 and 11-13 in Figure 1a. Here the nodes 1-3 differ from each other in terms of their morphological classes. The nodes 1 and 11 differ only in terms of their position owing to its repeated occurrence in the input. The position information is opaque to our proposed system and is used only in forming the nodes for the sentence graph. During the inference, we consider all the pairwise potentials as contexts for each of the prediction made in the search space. The edges in our model should capture the likeliness of two nodes to co-occur in the final solution. Hence, every pair of nodes that can co-occur in an 'exhaustive segmentation' forms two directed edges, one each at either of the directions.

Energy Based Model (EBM) Architecture: Our approach is inspired from the graph based parsing approaches employed generally for dependency parsing (McDonald et al., 2005b; Carreras, 2007) and follows a likewise structured prediction paradigm (Taskar et al., 2005). Specifically, we use an EBM where we model our joint task as search for a maximal clique with minimum energy. Learning consists of finding an energy function that associates lower energies to cliques with increasing similarity to the correct clique. The correct clique configuration will have the lowest energy (LeCun et al., 2006). The inference policy, a maximal clique selection algorithm, is used to make the predictions.

We follow an approach similar to the arc-factored approaches in graphs (McDonald et al., 2005b; Carreras, 2007), where the total energy of a maximal clique, $T_i = (V_{T_i}, E_{T_i})$, is decomposed as the summation of energies of its

[6] Our graph construction approach is explained using finite state methods in §1 of the supplementary material

edges (Ishikawa, 2011).

$$\mathcal{S}(T_i) = \sum_{e_{pq} \in E_{T_i}} \mathcal{S}(\vec{e}_{pq})$$

where, $V_{T_i} \subseteq V, E_{T_i} \subseteq E$. The edges are featurised. For an edge $e_{pq} \in E$, the features are represented as a vector, denoted by $\vec{e}_{pq}$. For a given edge, the energy function, $\mathcal{S}(\cdot) : [0, \infty)^{|\vec{e}|} \rightarrow (-\infty, \infty)$, takes the edge feature vector and produces a scalar value as the energy assignment.

Loss Function and Training: We use Hinge Loss (Altun et al., 2003; Taskar et al., 2003) as our loss function. The hinge loss is formulated such that it increasingly penalises those cliques, sampled by our inference algorithm, with increasingly more number of wrong segmentation candidates. We minimise the hinge loss L which is defined as

$$L = max(0, \mathcal{S}(T_{GT}) - \underset{T_i \in A^Q}{\text{argmin}}(\mathcal{S}(T_i) - \Delta(T_i))$$

Here, A^Q denotes the set of all the unique maximal cliques and T_{GT} denotes the maximal clique corresponding to the ground truth.

The margin $\Delta(T_i)$ is defined as $\Delta(T_i) = |V_{T_i} - V_{GT}|^2$. We minimise the given loss function using gradient descent method. The network parameters are updated per sentence using back-propagation. The hinge loss function is not differentiable at the origin. Hence, we use the subgradient method to update the network parameters (Socher et al., 2010; Ratliff et al., 2007). We use a multi-layer perceptron network with a single hidden layer and a leaky ReLU activation function at the hidden layer for the training.

Inference Policy: For the maximal clique selection, we use a greedy heuristic approach inspired from Prim's algorithm (Prim, 1957). The policy is described in Algorithm 1.

In Algorithm 1, we start the clique selection with a single node. At any given instance, we loop through the nodes in the graph which are not yet part of the clique. We add a vertex v to the clique if the cumulative score of all the edges from v to every vertex that is already in the clique is the minimum. We discard all the nodes which are conflicting with vertex v. "Conflicting" nodes are any pair of nodes which are not connected by an edge between them. This follows from the construction of the graph G, as the non-connectivity between the nodes implies that they are proposed as alternative

Algorithm 1: Greedy maximal clique selection heuristic

1 **for** *each node* v_i *in* V **do**
2 Initialize a graph $K_i(V_{K_i}, E_{K_i})$ with $K_i = G$ such that $V_{K_i} = V$ and $E_{K_i} = E$. Initialise a vertex set V_{T_i} with v_i as the only element in it. Remove all the vertices which are conflicting with v_i from K_i.
3 Add the vertex $v_j \in (V_{K_i} - V_{T_i})$ to V_{T_i}, such that in K_i, the sum of edge weights for the edges starting from v_j to all other vertices in V_{T_i} is minimum.
4 Remove all the vertexes which are conflicting with v_j from V_{K_i}.
5 Repeat steps 3 - 4 till $V_{K_i} - V_{T_i} = \varnothing$
6 **end**

word suggestions in G. As guaranteed by our sentence graph construction, we obtain the maximal clique (*exhaustive segmentation*) when there exist no more vertices to loop through. We perform this for every node in the graph G. From all the cliques so obtained we select the maximal clique with the least score. The approach does not guarantee enumeration of all the cliques, but it is guaranteed that every node will be covered by at least one maximal clique. The heuristic can be seen as a means of sampling some potential minimum energy maximal cliques for the learning task. Energy based models do not require proper normalisation of the solution space (LeCun et al., 2006), a choice that enables the use of the heuristic.

During inference, the greedy clique selection heuristic is performed for every node in G. Though the run-time for this inference is polynomial, it can still be computationally expensive. But, in practice we find that our inference procedure results in faster output for graphs with > 19 nodes in comparison to the exponential time Bron-Kerbosch algorithm (Tomita et al., 2006; Bron and Kerbosch, 1973) for clique enumeration (McDonald et al., 2005a). We further improve the run time of our inference procedure by paralleling the clique selection procedure for each node on a separate thread.

3 Feature Generation for the Edges

Given two non-conflicting nodes in G, there exists a pair of directed edges, one each in either direc-

tion. For every edge in the sentence graph G, we need to generate features that capture the distributional information between the candidate nodes that the edge connects. Similar in spirit to Bilmes and Kirchhoff (2003) and Krishna et al. (2016), we condition the distributional information based on different morphological constraints to enrich the context. The distributional information is obtained from a morphologically tagged corpus $\mathcal{C}$. Let $\mathcal{V}_w$, $\mathcal{V}_m$ and $\mathcal{V}_r$ be the vocabulary of the inflected word forms, morphological classes and the roots respectively in the corpus. Let $\mathcal{V} = \mathcal{V}_w \cup \mathcal{V}_m \cup \mathcal{V}_r$. For each $n_i, n_j \in \mathcal{V}$, the conditional probability is calculated as $P_{co}(n_j|n_i) = \frac{count(n_j,n_i)}{count(n_i)}$. Here $count(\cdot)$ represents the count of co-occurrence between the entries in the corpus. Also, let MC be the set of morphological constraints used for conditioning the distributional information. Now, for each $n_i, n_j \in \mathcal{V}$ and each $g_k \in MC$, we can obtain the feature value as follows:

$$P_{g_k}(n_j|n_i) = -log(P_{co}(n_j|g_k) \times P_{co}(g_k|n_i))$$

We use the following scheme for feature generation. A node in G essentially contains three attributes, namely, the root, the morphological class and the inflected word form. A feature uses corpus evidence of exactly one of the three attributes. For instance, consider two candidate nodes o_1, o_2 in G with (o_{1w}, o_{1m}, o_{1r}) and (o_{2w}, o_{2m}, o_{2r}) as the respective 3-tuple attributes. Now, one such possible feature value for the edge from o_1 to o_2 can be calculated as $P_{g_1}(o_{1r}|o_{2w})$ where $g_1 \in MC$ and $o_{1r}, o_{2w} \in \mathcal{V}$. Hence, features for a directed edge connecting two different nodes in G can be generated in $3 \times |MC| \times 3$ ways.

We automate the process of feature generation and feature selection using the framework of Path Ranking Algorithm (Lao and Cohen, 2010). Formalising our approach using PRA leads to an efficient and scalable implementation of our scheme. In PRA, enumerating and generating all the possible features needs to be performed only for a sampled set of data pairs from the corpus. By using a supervised feature selection approach, a relevant sub-set of features is filtered. This is a one-time process (Gardner and Mitchell, 2015). During inference, the feature values are computed only for the filtered features. [7]

Morphological Constraints: MC is defined as the set of grammatical category combinations, each combination falling into one of the following two descriptions. a) *Complete combination*, i.e., a morphological class – In Sanskrit, similar to Czech (Smith et al., 2005), a morphological class represents a certain combination of grammatical categories. For instance, a noun is represented by case, gender and number. Hence, the combination 'genitive-masculine-singular' forms a morphological class. b) *Partial combination* - A combination of grammatical categories, which can form a morphological class by adding one or more categories to it. For instance, 'genitive-masculine' is a partial combination that denotes all the possible (three) morphological classes which differ from each other only in terms of the category 'number'. However, 'genitive-present tense' is not a 'valid' combination as it can never form a valid morphological class. The evidence for a partial combination in the corpus $\mathcal{C}$ can be obtained by summing the evidence of all the morphological classes which it denotes. We obtain a total of 528 morphological constraints. A filtered set of 1500 features (out of 4752) is used in our model. Mutual Information Regression (Kraskov et al., 2004) with the word to word co-occurrence probability as label is used for feature selection.[8]

4 Experiments

Dataset: We use the Digital Corpus of Sanskrit (DCS) (Hellwig, 2010-2016), a morphologically tagged corpus of Sanskrit, for all our experiments. DCS contains digitised works from periods that span over 3000 years and contain constructions written in prose or poetry. Identifying sentence boundaries in Sanskrit constructions is a challenge of its own (Hellwig, 2016). DCS currently has split the corpus into more than 560,000 text lines, all of which need not be following explicit sentence boundaries. Krishna et al. (2016) identify 350,000 constructions from the DCS fit for the segmentation task. They use a separate set of 9,577 constructions from the DCS, called as 'DCS10k', and use it as the test set. They ignore the remaining text lines from DCS due to ambiguities either in the provided tagging or alignment with SHR (Krishna et al., 2017). We use the 350,000 constructions used in Krishna et al.

[7] The edge vector formation is explained in terms of Metapaths (Sun, 2010) in §3 of the Supplementary.

[8] For different settings we experimented with, for the vector generation, refer to §4 of the supplementary material.

(2016) as the corpus $\mathcal{C}$ (§3) for the generation of our edge vectors. 'DCS10k' was neither used in the training of our model, nor in the generation of edge vectors. Reddy et al. (2018) report their results on a subset of DCS10k containing 4,200 sentences, which we will refer to as 'DCS4k'.

We experiment with the following systems:

SupervisedPCRW: Proposed in Krishna et al. (2016), this model also uses the graph output from SHR. Using PCRW (Lao and Cohen, 2010; Meng et al., 2015), feature vectors for edges are generated using hand-crafted morphological constraints. Starting with the longest word in the graph, the prediction is performed by greedily selecting the candidates as per the edge weights.

EdgeGraphCRF: This is a second order CRF Model (Müller and Behnke, 2014; Ishikawa, 2011) which uses the sentence graph structure G as the input to the system. Every node is represented with fastText (Bojanowski et al., 2017) word embeddings trained under default settings. The edges are featurised with the PRA vectors (§3). We used 1-slack structured SVM for training. For the binary class problem, QPBO (Rother et al., 2007) inference approach provided the best results.

Seq2Seq - Reddy et al. (2018) uses an Encoder-Decoder framework with LSTM cells for the segmentation task. We consider two models from the work, namely, '*segSeq2Seq*' and '*attnSegSeq2seq*' as our baselines. The later which uses attention (Bahdanau et al., 2015) is the current state of the art in Sanskrit word segmentation.

Lattice-EBM: An energy based sequence labelling model, where the input is a lattice (Wolf and Woods, 1977) similar to that of Kudo (2006). The model can be seen as a special case of Graph Transformer Networks (LeCun et al., 1998, 2007). In the lattice structure, the candidate links only to its adjacent nodes in an exhaustive segmentation. We also generate edge vectors for the dummy nodes that act as the start and end markers in the lattice. During prediction, we have to find the best path from the lattice which minimises the sentence score. Here, we consider two variants of Lattice-EBM. *L-EBM-Vanilla* uses the discriminative forward training approach (Collobert et al., 2011) with the standard hinge loss. The second variant *L-EBM-Beam*, uses multi-margin loss (Edunov et al., 2018), instead of the hinge loss. Here, we employ beam search to generate multiple candidates as required by the loss.

Tree-EBM: The baseline model works exactly the same as the proposed model where the only difference between both is in the inference algorithm used. Tree-EBM has an inference that searches for a Steiner Tree (Takahashi, 1980) from the input graph $G(V, E)$, the structure of which is described in §2.[9] The inference procedure outputs a spanning tree that covers a subset of the nodes from G. This raises a challenge while estimating the loss as, unlike in the case of a clique, there can be multiple rooted tress that spans the subset of nodes in the ground truth. In this model, we augment the loss L_{tree} such that the Steiner tree with the least energy that spans the nodes in ground truth is chosen.

$$L_{tree} = max(0, \operatorname*{argmin}_{T_m \in A^G} \mathcal{S}(T_m) - \operatorname*{argmin}_{T_i \in A^Q} (\mathcal{S}(T_i) - \Delta(T_i))$$

Here A^G represents set of all the trees that spans the nodes in the ground truth.

Clique-EBM: The proposed model. The EBM model uses the maximal clique selection heuristic for the inference.

Tasks and Evaluation Measures: We use the macro-averaged Precision (P), Recall (R), F1-score (F) and also the percentage of sentences with perfect matching (PM) as our evaluation metrics. We evaluate the competing systems on the following two different tasks.

Word Prediction Task ***(WPT)*** - The word segmentation task is evaluated based on the correctness of the inflected word forms predicted. This was used in Krishna et al. (2016).

Word++ Prediction Task ***(WP3T)*** - This is a stricter metric for the evaluation of the joint task of segmentation and morphological tag prediction. It evaluates the correctness of each of the inflected word form, lemma and its morphological tag.

4.1 Results

Table 1 provides the results for the best performing configurations for each of the systems. The results for WPT on DCS4k and WP3T on DCS10k for each of the systems are shown in Tables 1.A and 1.B. The proposed model, Clique-EBM (System 8), outperforms all the other models across all the 4 metrics on both the tasks. Clique-EBM shows an

[9] The inference procedure is given in §2 of the supplementary material

No:	System	P	R	F	PM
1	segSeq2Seq	73.44	73.04	73.24	29.2
2	SupervisedPCRW	76.30	79.47	77.85	38.64
3	EdgeGraphCRF	79.27	81.6	80.42	35.91
4	L-EBM-Vanilla	86.38	85.49	84.29	53.62
5	L-EBM-Beam	86.38	85.77	86.07	60.32
6	AttnsegSeq2Seq	90.77	90.3	90.53	55.99
7	Tree-EBM	89.44	92.35	90.87	61.72
8	**Clique-EBM**	**96.18**	**97.67**	**96.92**	**78.83**

Table 1.A: WPT on DCS4k

System	P	R	F	PM
EdgeGraphCRF	76.69	78.74	77.7	31.82
LatticeEBM-Vanilla	76.88	74.76	75.8	27.49
LatticeEBM-Beam	79.41	77.98	78.69	31.57
Tree-EBM	82.35	79.74	81.02	32.88
Clique-EBM	**91.35**	**89.57**	**90.45**	**55.78**

Table 1.B: WP3T on DCS10k

Table 1: Performance evaluation of the competing systems in ascending order of their F-Score.

improvement of 7.06% and 40.79% in terms of F-score and *perfect matching* from the current state of the art (System 6) in WPT. Currently there exists no system that predicts the morphological tags for a given word in Sanskrit. For WP3T, Clique-EBM has shown a percentage increase of 11.64% and 69.65% in F-Score and the perfect matching score from Tree-EBM, the next best system.

All the systems except 1 and 6 in Table 1.A use the linguistically refined output from SHR as their search space to predict the final solution. Out of which 3, 4, 5, 7 and 8 use the edge vectors, which encodes the morphological constraints refined using PRA (Lao and Cohen, 2010), generated by the method discussed in §3. As a result these systems require $<10\%$ training data than required by system 6. System 3 was trained on 10,000 sentences, while systems 4 and 5 were trained on 9,000 sentences after which the models got saturated. Systems 7 and 8 were trained on 8,200 sentences which is 7.66% of the training data (107,000) used in System 6. In terms of training time, Reddy et al. (2018) reports a training time of 12 hours on a GPU machine, while systems 7 and 8 take a training time of 8 hours on an Intel Xeon CPU based machine with 24 cores and 256 GB RAM.[10] For systems 4 and 5 it takes roughly 4 hours to train on the same machine. There was no training involved for the prediction of segmentations in system 2.

In systems 4 and 5, the inference is performed sequentially from left to right[11]. The use of beam search with multi margin in System 5 resulted in marginal improvements (<2) in terms of F-Score to that of system 4. Further, the improvement in the results saturated after a beam size of 128. System 3 being a second order CRF model (Ishikawa, 2011), does not take the entire sentence context into account. In fact, about 85.16% of the sentences predicted by the model from DCS10K do not correspond to an '*exhaustive segmentation*'. Prediction of an '*exhaustive segmentation*' is guaranteed in all our EBM models 4, 5, 7 and 8 (also in system 2) by virtue of the inference procedure we use. Both systems 7 and 8, along with System 6 which uses attention, consider the entire input context when making each prediction. System 8 considers all the pairwise potentials between the nodes while making a prediction, but System 7 does not (Steiner tree vs. maximal clique).

Figure 3 reports the performance of the systems 2, 5, 6 and 8 where the sentences in DCS4k[12] are categorised based on the number of words presented in the segmented ground-truth solution. Our proposed system Clique-EBM performs the best across all the lengths with an exception towards shorter constructions of 2 words or less. Interestingly, both the sequential models (systems 5 and 6) show a decreasing trend as the number of words increases, while the Clique-EBM model shows an increasing trend with a larger length, which might indicate that more context helps the model. In fact, the greedy yet non-sequential approach used in Krishna et al. (2016) outperforms both the sequential models at longer lengths. The average length of a sentence in DCS is 6.7 (Krishna et al., 2016), the share of sentences with seven or more words is 62.78%.

4.2 Fine Grained Analysis on Clique-EBM[13]

Pruning the edges in sentence graph G: Constructions in Sanskrit follow weak non-projectivity (with exception to verses), implying that they adhere to the principle of proximity (Kulkarni et al., 2015). By proximity we expect that the words in a phrase go together, without being interrupted by a syntactically unrelated word.

[10] Please refer §4 of the supplementary for wall time analysis. System 6, when trained on this CPU based system, did not converge even after 15 hours of training.

[11] We also tried reversing the input effectively enabling the right to left direction but the results were worse than the reported system by an F-Score of 3.

[12] sentences with length more than 12 words are not shown as such sentences appear less than 10 times in DCS4k.

[13] For hyper-parameter settings, and other fine-grained analysis refer to §4 of the supplementary material.

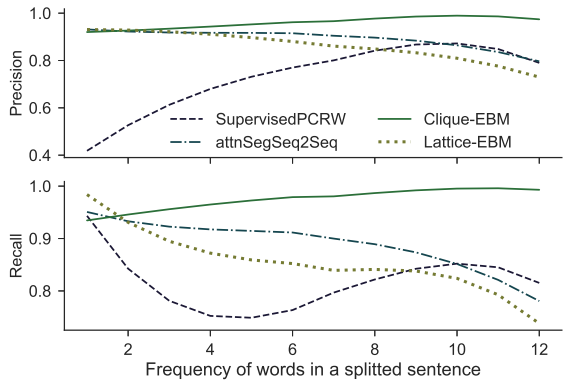

Figure 3: Performance of the competing systems for DCS4k grouped on the word counts in the ground-truth

But the relative ordering between the phrases in a construction and the order of words within a phrase can be free.

For any two words appearing in an exhaustive segmentation, we keep an edge only if both the words overlap within a distance of k characters. We experiment with k = 5, 10, 15 and 20. Hence, for $K = 20$, a word will form edges with all the words that fall within 20 characters to left and 20 characters to right. The average length of an input sequence in DCS10K is 40.79 characters. We do not modify our inference procedure in system 8 other than to take care of the possibility that a clique need not always be returned. Table 2 shows the results for different values of k. Interestingly, the results show a monotonic increase with the increase in context window, and the results with the entire context are still better than those with $k = 20$, even though only marginally. It is interesting to note that, keeping the entire context does not adversely affect the predictions as none of the pruned models outperforms System 8.

The lattice structure can be seen as an extreme case of pruning. We modify System 4 to use a non-sequential inference procedure, adapted from System 7. Here, the start and end markers were removed. Additionally, a given connected node pair has 2 edges, each in either of the directions. We find that the model gives an F-Score of 87.4 which outperforms System 4 by more than three points.

Out of Vocabulary words: As described in §3, the distributional information from DCS is used as the corpus $\mathcal{C}$ for the feature generation. For the case of OOV in roots ($\mathcal{V}_r$), we use add-1 smoothing. But, for the case of OOV in inflections ($\mathcal{V}_w$) we find that using the evidence from corresponding root of the candidate is beneficial. DCS10k has 8,007 roots of which 514 are OOV and 833 occur only 1 to 5 times. The micro-averaged F-Score for these are 57.98 and 72.87, respectively.

k	WPT			WP3T		
	P	R	F	P	R	F
5	90.46	92.27	91.36	83.52	80.48	81.97
10	92.92	95.07	93.98	85.32	84.4	84.86
15	94.85	96.14	95.49	87.67	86.38	87.02
20	**95.23**	**96.49**	**95.86**	**89.25**	**88.62**	**88.93**

Table 2: Performance of Clique-EBM with pruned edges in G.

Type	WPT Recall		WP3T Recall	
	T-EBM	C-EBM	T-EBM	C-EBM
Noun	93.06	96.87	86.14	89.0
Verb	89.14	95.91	87.38	94.42
Compound	89.35	93.52	86.01	91.07
Indeclinable	95.07	97.09	94.93	96.47

Table 3: System performance on the coarse level POS for the competing systems Clique-EBM (C-EBM) and Tree-EBM (T-EBM)

Morphological class specific assessment : Table 3 presents the micro-averaged recall for the words grouped based on their parts of speech (POS) for Clique-EBM and Tree-EBM. Both the systems follow similar trends and the morphological class misprediction is highest among the nouns and compounds (WP3T Recall). It also needs to be noted that the improvement made by Clique-EBM in comparison to Tree-EBM for WP3T was also on prediction of noun and compound morphological classes. Also in Tree-EBM, the mispredictions in compounds were mostly cases of the system getting the compound components confused to one of the morphological classes in nouns.

We find that considering the pairwise potential between all the words in a sentence in Clique-EBM led to improved morphological agreement between the words in comparison to Tree-EBM. In Tree-EBM, the top 5 cases of mispredictions from one morphological class to a particular wrong class were between those classes of nouns that differed in exactly one of the three possible grammatical categories, namely gender, number or declension, that makes up a noun. In Clique-EBM such patterns were not anymore present and more importantly the skewedness in such mispredictions were considerably reduced.[14]

Summarily, our non-sequential method of inference results in better performance in comparison

[14]Please refer to Tables 6 and 7 in the supplementary material for details

to the sequential models. We also find that the sequential models see a drop in their performances when the number of words in a sentence increases. Leveraging the pairwise potentials between every connected nodes while making a prediction improves the performance. The performance gain of Clique-EBM over Tree-EBM illustrates the effectiveness of this approach.

5 Discussion

In Sanskrit, syncretism (Crystal, 2011) leads to ambiguity during morphological analysis. It can further be observed that such common root identical surface forms often have one or more common grammatical categories in their morphological classes (Goyal and Huet, 2016). We find that the first three models in Table 1.B often end up predicting an identical surface-form with an incorrect morphological tag, thus affecting the WP3T scores.[15]. The grammatical categories in a morphological class are indicative of the syntactic roles and the morphological agreement between the words in a construction. We empirically observe that the inference procedure for clique-EBM, which considers the entire input context and pairwise potentials between the candidates, helps in improving the performance of the model. A similar observation regarding incompatible morphological agreements between predicted words was made by Hassan et al. (2018) for their NMT model. The authors introduced an elaborate 2 phase decoder and a KL-Divergence based regularisation to combat the issue.

The energy based model (EBM) we propose is a general framework for structured prediction in Sanskrit. EBMs are widely used for various structured prediction tasks (LeCun et al., 2007; Belanger and McCallum, 2016). Belanger (2017) states that, "CRFs are typically attributed to Lafferty et al. (2001), but many of the core technical contributions of the paper appeared earlier in LeCun et al. (1998)." GTNs (LeCun et al., 1998), in fact, work on a graph based input very similar to that of a lattice, a variant of which we use in L-EBM-Vanilla. For dependency parsing, use of a word-level lattice structure similar to Seeker and Çetinoğlu (2015), where all the homonyms and syncertisms of a given surface-form form a lattice, will potentially result in a reduced candidate space than ours. Additionally, our model currently does not take into account the phrasal nature of compounds in Sanskrit (Lowe, 2015). This can further reduce the edge density in our current graph construction. But, this needs further exploration, as current edge vectors may not be suitable for the task. To generate the possible candidates, we rely completely on the SHR. In case of words not recognised by the lexicon driven SHR, analysis of sentences with a partially recognised portion is still possible. Once a root is added, all its inflections can be generated by the SHR automatically.

[15]Details in §4 of the Supplementary material

6 Conclusion

We proposed a novel approach to tackle word segmentation and morphological tagging problem in Sanskrit. Our model outperforms Reddy et al. (2018), the current state of the art, with an F-Score of 96.92. Reddy et al. (2018) report that the extension of their model to perform morphological tagging is not straightforward, as they learn a new sub-word vocabulary using the sentencepiece model (Schuster and Nakajima, 2012).

The free word order nature of the language motivated us to consider the input to be a graph so as to avoid the sequential processing of input. For the EBM we use, there is no requirement of proper normalisation (LeCun et al., 2006). We benefit from this as we perform a search in the space of complete outputs and there is a combinatorial explosion in the output space for a linear increase in the input space (Doppa et al., 2014). The pretraining of the edge vectors with external knowledge in the form of morphological constraints is effective in reducing the task specific training size (Yang et al., 2017; Andor et al., 2016).

Acknowledgements

We are grateful to Oliver Hellwig for providing the DCS Corpus and Gérard Huet for providing the Sanskrit Heritage Engine, to be installed at local systems. We extend our gratitude to Amba Kulkarni and Rogers Mathew, along with Gérard for helpful comments and discussions regarding the work. We thank the anonymous reviewers for their constructive and helpful comments, which greatly improved the paper. We are indebted to Unni Krishnan T A for his contributions towards implementation of the framework.

References

Yasemin Altun, Mark Johnson, and Thomas Hofmann. 2003. Investigating loss functions and optimization methods for discriminative learning of label sequences. In *Proceedings of the 2003 Conference on Empirical Methods in Natural Language Processing*, Sapporo, Japan. Association for Computational Linguistics.

Daniel Andor, Chris Alberti, David Weiss, Aliaksei Severyn, Alessandro Presta, Kuzman Ganchev, Slav Petrov, and Michael Collins. 2016. Globally normalized transition-based neural networks. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 2442–2452, Berlin, Germany. Association for Computational Linguistics.

Dzmitry Bahdanau, KyungHyun Cho, and Yoshua Bengio. 2015. Neural machine translation by jointly learning to align and translate. In *Proceedings of the Third International Conference on Learning Representation (ICLR)*, San Diego, US.

David Belanger. 2017. *Deep Energy-Based Models for Structured Prediction*. Ph.D. thesis, University of Massachusetts Amherst.

David Belanger and Andrew McCallum. 2016. Structured prediction energy networks. In *Proceedings of The 33rd International Conference on Machine Learning*, pages 983–992, New York, New York, USA. PMLR.

Jeff A. Bilmes and Katrin Kirchhoff. 2003. Factored language models and generalized parallel backoff. In *Companion Volume of the Proceedings of HLT-NAACL 2003 - Short Papers*, Edmonton, Canada. Association for Computational Linguistics.

Piotr Bojanowski, Edouard Grave, Armand Joulin, and Tomas Mikolov. 2017. Enriching word vectors with subword information. *Transactions of the Association for Computational Linguistics*, 5:135–146.

Coen Bron and Joep Kerbosch. 1973. Algorithm 457: finding all cliques of an undirected graph. *Communications of the ACM*, 16(9):575–577.

Xavier Carreras. 2007. Experiments with a higher-order projective dependency parser. In *Proceedings of the 2007 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning (EMNLP-CoNLL)*, pages 957–961, Prague, Czech Republic. Association for Computational Linguistics.

Shay B. Cohen and Noah A. Smith. 2007. Joint morphological and syntactic disambiguation. In *Proceedings of the 2007 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning (EMNLP-CoNLL)*, pages 208–217, Prague, Czech Republic. Association for Computational Linguistics.

Ronan Collobert, Jason Weston, Léon Bottou, Michael Karlen, Koray Kavukcuoglu, and Pavel Kuksa. 2011. Natural language processing (almost) from scratch. *J. Mach. Learn. Res.*, 12:2493–2537.

David Crystal. 2011. *A dictionary of linguistics and phonetics*, volume 30. John Wiley & Sons.

Janardhan Rao Doppa, Alan Fern, and Prasad Tadepalli. 2014. Hc-search: a learning framework for search-based structured prediction. *Journal of Artificial Intelligence Research*, 50(1):369–407.

Sergey Edunov, Myle Ott, Michael Auli, David Grangier, and Marc'Aurelio Ranzato. 2018. Classical structured prediction losses for sequence to sequence learning. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)*, pages 355–364. Association for Computational Linguistics.

Matt Gardner and Tom Mitchell. 2015. Efficient and expressive knowledge base completion using subgraph feature extraction. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing*, pages 1488–1498, Lisbon, Portugal. Association for Computational Linguistics.

Yoav Goldberg and Reut Tsarfaty. 2008. A single generative model for joint morphological segmentation and syntactic parsing. In *Proceedings of ACL-08: HLT*, pages 371–379, Columbus, Ohio. Association for Computational Linguistics.

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

Pawan Goyal, Gérard Huet, Amba Kulkarni, Peter Scharf, and Ralph Bunker. 2012. A distributed platform for sanskrit processing. In *Proceedings of COLING 2012*, pages 1011–1028. The COLING 2012 Organizing Committee.

Hany Hassan, Anthony Aue, Chang Chen, Vishal Chowdhary, Jonathan Clark, Christian Federmann, Xuedong Huang, Marcin Junczys-Dowmunt, William Lewis, Mu Li, et al. 2018. Achieving human parity on automatic chinese to english news translation. *arXiv preprint arXiv:1803.05567*.

Jun Hatori, Takuya Matsuzaki, Yusuke Miyao, and Jun'ichi Tsujii. 2012. Incremental joint approach to word segmentation, pos tagging, and dependency parsing in chinese. In *Proceedings of the 50th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1045–1053, Jeju Island, Korea. Association for Computational Linguistics.

Jiri Havelka. 2007. Beyond projectivity: Multilingual evaluation of constraints and measures on non-projective structures. In *Proceedings of the 45th Annual Meeting of the Association of Computational*

Linguistics, pages 608–615, Prague, Czech Republic. Association for Computational Linguistics.

Oliver Hellwig. 2009. Extracting dependency trees from sanskrit texts. In *Sanskrit Computational Linguistics*, pages 106–115, Berlin, Heidelberg. Springer Berlin Heidelberg.

Oliver Hellwig. 2010-2016. *DCS - The Digital Corpus of Sanskrit*. Berlin.

Oliver Hellwig. 2016. Detecting sentence boundaries in sanskrit texts. In *Proceedings of COLING 2016, the 26th International Conference on Computational Linguistics: Technical Papers*, pages 288–297. The COLING 2016 Organizing Committee.

Gérard Huet. 2005. A functional toolkit for morphological and phonological processing, application to a Sanskrit tagger. *Journal of Functional Programming*, 15(4):573–614.

Gérard Huet. 2006. *Themes and Tasks in Old and Middle Indo-Aryan Linguistics*, chapter Lexicon-directed Segmentation and Tagging of Sanskrit. Motilal Banarsidass, Delhi.

Hiroshi Ishikawa. 2011. Transformation of general binary mrf minimization to the first-order case. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 33(6):1234–1249.

Alexander Kraskov, Harald Stögbauer, and Peter Grassberger. 2004. Estimating mutual information. *Physical review E*, 69(6):066138.

Amrith Krishna, Bishal Santra, Pavankumar Satuluri, Sasi Prasanth Bandaru, Bhumi Faldu, Yajuvendra Singh, and Pawan Goyal. 2016. Word segmentation in sanskrit using path constrained random walks. In *Proceedings of COLING 2016, the 26th International Conference on Computational Linguistics: Technical Papers*, pages 494–504. The COLING 2016 Organizing Committee.

Amrith Krishna, Pavan Kumar Satuluri, and Pawan Goyal. 2017. A dataset for sanskrit word segmentation. In *Proceedings of the Joint SIGHUM Workshop on Computational Linguistics for Cultural Heritage, Social Sciences, Humanities and Literature*, pages 105–114, Vancouver, Canada. Association for Computational Linguistics.

Taku Kudo. 2006. Mecab: Yet another part-of-speech and morphological analyzer. *http://mecab. sourceforge. jp*.

Amba Kulkarni, Preethi Shukla, Pavankumar Satuluri, and Devanand Shukl. 2015. How Free is free Word Order in Sanskrit. In *The Sanskrit Library, USA*, pages 269–304.

John D. Lafferty, Andrew McCallum, and Fernando C. N. Pereira. 2001. Conditional random fields: Probabilistic models for segmenting and labeling sequence data. In *Proceedings of the Eighteenth International Conference on Machine Learning*, ICML '01, pages 282–289, San Francisco, CA, USA. Morgan Kaufmann Publishers Inc.

Ni Lao and William W. Cohen. 2010. Relational retrieval using a combination of path-constrained random walks. *Mach. Learn.*, 81(1):53–67.

Yann LeCun, Léon Bottou, Yoshua Bengio, and Patrick Haffner. 1998. Gradient-based learning applied to document recognition. *Proceedings of the IEEE*, 86(11):2278–2324.

Yann LeCun, Sumit Chopra, Raia Hadsell, Marc'Aurelio Ranzato, and Fu-Jie Huang. 2006. A tutorial on energy-based learning. In *Predicting Structured Data*. MIT Press.

Yann LeCun, Sumit Chopra, Marc'Aurelio Ranzato, and Fu-Jie Huang. 2007. Energy-based models in document recognition and computer vision. In *Proceedings of International Conference on Document Analysis and Recognition (ICDAR)*, pages 337–341, Curitiba, Paraná, Brazil. IEEE Computer Society.

John J. Lowe. 2015. The syntax of sanskrit compounds. *Journal of Historical Syntax*, 91(3):71–115.

Wolfgang Maier and Timm Lichte. 2011. Characterizing discontinuity in constituent treebanks. In *Formal Grammar*, pages 167–182, Berlin, Heidelberg. Springer Berlin Heidelberg.

Ryan McDonald, Fernando Pereira, Seth Kulick, Scott Winters, Yang Jin, and Pete White. 2005a. Simple algorithms for complex relation extraction with applications to biomedical ie. In *Proceedings of the 43rd Annual Meeting of the Association for Computational Linguistics (ACL'05)*, pages 491–498. Association for Computational Linguistics.

Ryan McDonald, Fernando Pereira, Kiril Ribarov, and Jan Hajic. 2005b. Non-projective dependency parsing using spanning tree algorithms. In *Proceedings of Human Language Technology Conference and Conference on Empirical Methods in Natural Language Processing*, pages 523–530, Vancouver, British Columbia, Canada. Association for Computational Linguistics.

Keshav S Melnad, Pawan Goyal, and Peter Scharf. 2013. Meter identification of sanskrit verse. *The Sanskrit Library, USA*.

Changping Meng, Reynold Cheng, Silviu Maniu, Pierre Senellart, and Wangda Zhang. 2015. Discovering meta-paths in large heterogeneous information networks. In *Proceedings of the 24th International Conference on World Wide Web*, WWW '15, pages 754–764, Republic and Canton of Geneva, Switzerland. International World Wide Web Conferences Steering Committee.

Andrew Cameron Morris, Viktoria Maier, and Phil Green. 2004. From wer and ril to mer and wil: improved evaluation measures for connected speech recognition. In *Eighth International Conference on Spoken Language Processing*.

Andreas C Müller and Sven Behnke. 2014. Pystruct: learning structured prediction in python. *Journal of Machine Learning Research*, 15(1):2055–2060.

M. Oerder and H. Ney. 1993. Word graphs: an efficient interface between continuous-speech recognition and language understanding. In *1993 IEEE International Conference on Acoustics, Speech, and Signal Processing*, volume 2, pages 119–122.

Robert Clay Prim. 1957. Shortest connection networks and some generalizations. *Bell Labs Technical Journal*, 36(6):1389–1401.

Nathan D Ratliff, J Andrew Bagnell, and Martin A Zinkevich. 2007. (online) subgradient methods for structured prediction. In *Proceedings of the Eleventh International Conference on Artificial Intelligence and Statistics (AIStats)*, pages 380–387, San Juan, Puerto Rico. JMLR.org.

Vikas Reddy, Amrith Krishna, Vishnu Sharma, Prateek Gupta, Vineeth M R, and Pawan Goyal. 2018. Building a Word Segmenter for Sanskrit Overnight. In *Proceedings of the Eleventh International Conference on Language Resources and Evaluation (LREC 2018)*, Miyazaki, Japan. European Language Resources Association (ELRA).

Carsten Rother, Vladimir Kolmogorov, Victor Lempitsky, and Martin Szummer. 2007. Optimizing binary mrfs via extended roof duality. In *2007 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR 2007)*, pages 1–8, Minneapolis, Minnesota, USA. IEEE Computer Society.

Peter Scharf, Anuja Ajotikar, Sampada Savardekar, and Pawan Goyal. 2015. Distinctive features of poetic syntax preliminary results. *Sanskrit syntax*, pages 305–324.

M. Schuster and K. Nakajima. 2012. Japanese and korean voice search. In *2012 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 5149–5152.

Wolfgang Seeker and Özlem Çetinoğlu. 2015. A graph-based lattice dependency parser for joint morphological segmentation and syntactic analysis. *Transactions of the Association for Computational Linguistics*, 3:359–373.

Stuart M Shieber and Xiaopeng Tao. 2003. Comma restoration using constituency information. In *Proceedings of the 2003 Conference of the North American Chapter of the Association for Computational Linguistics on Human Language Technology-Volume 1*, pages 142–148. Association for Computational Linguistics.

Noah A. Smith, David A. Smith, and Roy W. Tromble. 2005. Context-based morphological disambiguation with random fields. In *Proceedings of Human Language Technology Conference and Conference on Empirical Methods in Natural Language Processing*, pages 475–482, Vancouver, British Columbia, Canada. Association for Computational Linguistics.

Richard Socher, Christopher D Manning, and Andrew Y Ng. 2010. Learning continuous phrase representations and syntactic parsing with recursive neural networks. In *Proceedings of the NIPS-2010 Deep Learning and Unsupervised Feature Learning Workshop*, pages 1–9.

Weiwei Sun. 2010. Word-based and character-based word segmentation models: Comparison and combination. In *Coling 2010: Posters*, pages 1211–1219, Beijing, China. Coling 2010 Organizing Committee.

Hiromitsu Takahashi. 1980. An approximate solution for steiner problem in graphs. *Math. Japonica*, 24(6):573–577.

Ben Taskar, Vassil Chatalbashev, Daphne Koller, and Carlos Guestrin. 2005. Learning structured prediction models: A large margin approach. In *Proceedings of the 22nd International Conference on Machine Learning*, pages 896–903, New York, NY, USA. ACM.

Ben Taskar, Carlos Guestrin, and Daphne Koller. 2003. Max-margin markov networks. In *Proceedings of the 16th International Conference on Neural Information Processing Systems*, NIPS'03, pages 25–32, Cambridge, MA, USA. MIT Press.

Etsuji Tomita, Akira Tanaka, and Haruhisa Takahashi. 2006. The worst-case time complexity for generating all maximal cliques and computational experiments. *Theoretical Computer Science*, 363(1):28–42.

J. Wolf and W. Woods. 1977. The hwim speech understanding system. In *ICASSP '77. IEEE International Conference on Acoustics, Speech, and Signal Processing*, volume 2, pages 784–787.

Jie Yang, Yue Zhang, and Fei Dong. 2017. Neural word segmentation with rich pretraining. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 839–849, Vancouver, Canada. Association for Computational Linguistics.