Unity in Diversity: A unified parsing strategy for major Indian languages

Juhi Tandon and **Dipti Misra Sharma**
Kohli Center on Intelligent Systems (KCIS)
International Institute of Information Technology, Hyderabad (IIIT-H)
Gachibowli, Hyderabad, India
juhi.tandon@research.iiit.ac.in
dipti@iiit.ac.in

Abstract

This paper presents our work to apply non linear neural network for parsing five r esource p oor I ndian L anguages belonging to two major language families - Indo-Aryan and Dravidian. Bengali and Marathi are Indo-Aryan languages whereas Kannada, Telugu and Malayalam belong to the Dravidian family. While little work has been done previously on Bengali and Telugu linear transition-based parsing, we present one of the first parsers for Marathi, Kannada and Malayalam. All the Indian languages are free word order and range from being moderate to very rich in morphology. Therefore in this work we propose the usage of linguistically motivated morphological features (suffix and postposition) in the non linear framework, to capture the intricacies of both the language families. We also capture chunk and gender, number, person information elegantly in this model. We put forward ways to represent these features cost effectively using monolingual distributed embeddings. Instead of relying on expensive morphological analyzers to extract the information, these embeddings are used effectively to increase parsing accuracies for resource poor languages. Our experiments provide a comparison between the two language families on the importance of varying morphological features. Part of speech taggers and chunkers for all languages are also built in the process.

1 Introduction

Over the years there have been several successful attempts in building data driven dependency parsers using rich feature templates (Kübler et al., 2009) requiring a lot of feature engineering expertise. Though these indicative features brought enormously high parsing accuracies, they were computationally expensive to extract and also posed the problem of data sparsity. To address the problem of discrete representations of words, distributional representations became a critical component of NLP tasks such as POS tagging (Collobert et al., 2011), constituency parsing (Socher et al., 2013) and machine translation (Devlin et al., 2014). The distributed representations are shown to be more effective in non-linear architectures compared to the traditional linear classifier (Wang and Manning, 2013). Keeping in line with this trend, Chen and Manning (Chen and Manning, 2014) introduced a compact neural network based classifier for use in a greedy, transition-based dependency parser that learns using dense vector representations not only of words, but also of part-of-speech (POS) tags, dependency labels, etc. In our task of parsing Indian languages, a similar transition-based parser based on their model has been used. This model handles the problem of `sparsity, incompleteness and expensive feature computation` (Chen and Manning, 2014).

The last decade has seen quite a few attempts at parsing Indian languages Hindi, Telugu and Bengali (Bharati et al., 2008a; Nivre, 2009; Mannem, 2009; Kolachina et al., 2010; Ambati et al., 2010a). The research in this direction majorly focused on data driven transition-based parsing using MALT (Nivre et al., 2007), MST parser (McDonald et al., 2005) or constraint based method (Bharati et al., 2008b; Kesidi, 2013). Only recently Bhat et al. (2016a) have used neural network based non-linear parser to learn syntactic representations of Hindi and Urdu. Following their efforts, we present a similar parser for parsing five Indian Languages namely Bengali,

Proceedings of the Fourth International Conference on Dependency Linguistics (Depling 2017), pages 255-265,
Pisa, Italy, September 18-20 2017

Marathi, Telugu, Kannada, Malayalam. These languages belong to two major language families, Indo-Aryan and Dravidian. The Dravidian languages - Telugu, Kannada and Malayalam are highly agglutinative. The rich morphological nature of a language can prove challenging for a statistical parser as is noted by (Tsarfaty et al., 2010). For morphologically rich, free word order languages high performance can be achieved using vibhakti[1] and information related to tense, aspect, modality (TAM). Syntactic features related to case and TAM marking have been found to be very useful in previous works on dependency parsing of Hindi (Ambati et al., 2010b; Hohensee, 2012; Hohensee and Bender, 2012; Bhat et al., 2016b). We decided to experiment with these features for other Indian languages too as they follow more or less the same typology, all being free order and ranging from being moderate to very morphologically rich. We propose an efficient way to incorporate this information in the aforementioned neural network based parser. In our model, these features are included as suffix (last 4 characters) embeddings for all nodes. Lexical embeddings of case and TAM markers occurring in all the chunk are also included.

We also include chunk tags and gender, number, person information as features in our model. Taking cue from previous works where the addition of chunk tags[2] (Ambati et al., 2010a) and grammatical agreement (Bharati et al., 2008a; Bhat, 2017) has been proven to help Hindi and Urdu, our experiments test their effectiveness for other 5 languages in concern. Computationally, obtaining chunk tags can be done with ease. However, acquiring information related to gender, number, person for new sentences remains a challenge if we aim to parse resource poor languages for which sophisticated tools do not exist. We show that adding both these features definitely increases accuracy but we are able to gain major advantage by just using the lexical features, suffix features and POS tags which can be readily made available for low resource languages.

The rest of the paper is organised as follows. In Section 2 we talk about the data and the dependency scheme followed. Section 3 provides the rationale behind using each feature taking into account language diversity. Section 4 details about feature representations, models used and the experiments conducted. In Section 5 we observe the effects of inclusion of rich morpho-syntactic features on different languages and back the results with linguistic reasoning. In Section 6 we conclude and talk about future directions of research our work paves the way for.

2 Data and Background

2.1 Dependency Treebanks

There have been several efforts towards developing robust data driven dependency parsing techniques in the last decade (Kübler et al., 2009). The efforts, in turn, initiated a parallel drive for building dependency annotated treebanks (Tsarfaty et al., 2013). Development of Hindi and Urdu multi-layered and multi-representational (Bhatt et al., 2009; Xia et al., 2009; Palmer et al., 2009) treebanks was a concerted effort in this direction. In line with these efforts, treebanks for Kannada, Malayalam, Telugu, Marathi and Bengali are being developed as a part of the Indian Languages - Treebanking Project. The process of treebank annotation for various languages took place at different institutes[3]. These treebanks are manually annotated and span over various domains, like that of newswire articles, conversational data, agriculture, entertainment, tourism and education, thus making our models trained on them robust. The treebanks are annotated systematically with part of speech (POS) tags, morphological features (such as root, lexical category, gender, number, person, case, vibhakti, TAM (tense, aspect and modality) label in case of verbs, or postposition in case of nouns), chunking information and syntactico-semantic dependency relations. There has been a shift from the Anncorra POS tags (Bharati et al., 2006) that were initially used for Indian Languages to the new common tagset for all Indian languages which we would refer to as the Bureau of Indian Standards (BIS) tagset (Choudhary and Jha, 2011). This new POS tagging scheme is finer than the previous scheme. The dependency relations are marked following the Computational Paninian Grammar (Bharati et al., 1995; Begum

[1] vibhakti is a generic term for postposition and suffix that represent case marking

[2] a chunk is a set of adjacent words which are in dependency relation with each other, and are connected to the rest of the words by a single incoming arc to the chunk

[3] The organizations involved in this project are Jadavpur University-Kolkata (Bengali), MIT-Manipal (Kannada), C-DIT,Trivandrum (Malayalam), IIT-Bombay (Marathi), IIIT-Hyderabad (Hindi)

	Types	Tokens	Chunks	Sentences	Avg. tokens / per sentence
Kannada	36778	188040	143400	16551	11.36
Malayalam	20107	65996	54818	5824	11.33
Telugu BIS	4079	11338	8203	2173	5.21
Telugu Ann.	4582	13477	8363	2322	5.80
Bengali	18172	87321	69458	8209	10.64
Marathi	24792	94844	69214	7983	11.88

Table 1: Treebank statistics for the 5 languages used in the experiments

et al., 2008). Partial corpus of all the languages containing 25,000 tokens has been released publicly in ICON 2017 [4], the rest is still being annotated with multi layered information and sanity-checked. The Telugu treebank data corresponding to BIS tagset is still being built so we used the data from ICON10 parsing contest (Husain et al., 2010). It was cleaned and appended with some more sentences. We automatically converted this data from Anncorra tagset to BIS tagset against some word lists and rules. Since 149 sentences are lost in automatic conversion we report results on both the datasets. The statistics of the treebank data in this work can be found in the Table 1. Previous work has been done to convert the Hindi Treebank to Universal Dependencies (UD) (Tandon et al., 2016). These new treebanks which are built on the same underlying principle, could also be converted to UD by the same process as a future work.

2.2 Computational Paninian Grammar

Computational Paninian Grammar (CPG) formalism lies at the heart of Indian language treebanking. Dependency Structure–the first layer in these treebanks–involves syntactico-semantic dependency analysis based on this framework(Bharati et al., 1995; Begum et al., 2008). The grammar treats a sentence as a series of modified-modifier relations where one of the elements (usually a verb) is the primary modified. This brings it close to a dependency analysis model as propounded in Tesnière's Dependency Grammar (Tesnière, 1959). The syntactico-semantic relations between lexical items provided by the Pāṇinian grammatical model can be split into two types.

1. **Kāraka**: These are semantically related to a verb as the direct participants in the action denoted by a verb root. The grammatical model has six 'kārakas', namely **'kartā'** (the doer), **'karma'** (the locus of action's result), **'karaṇa'** (instrument), **'sampradāna'** (recipient), **'apādāna'** (source), and **'adhikaraṇa'** (location). These relations provide crucial information about the main action stated in a sentence.

2. **Non-kāraka**: These relations include reason, purpose, possession, adjectival or adverbial modifications etc.

Both the **Kāraka** and **Non-kāraka** relations in the scheme are given in Table 2. The * in the gloss name signifies that the relation can be more granular in function and branches to different types. [5]

Relation	Meaning
k1	Agent / Subject / Doer
k2*	Theme / Patient / Goal
k3	Instrument
k4*	Recipient / Experiencer
k5	Source
k7*	Spatio-temporal
rt	Purpose
rh	Cause
ras	Associative
k*u	Comparative
k*s	(Predicative) Noun / Adjective Complements
r6	Genitives
relc	Modification by Relative Clause
rs	Noun Complements (Appositive)
adv	Verb modifier
adj	Noun modifier

Table 2: Some major dependency relations belonging to Computational Paninian Grammar

3 Getting the best Features

We first describe the rationale behind choosing each feature, why it is important for each language and report a series of experiments by adding them one by one to observe their effects. It is a known fact that language specific features play a crucial role in robust dependency parsing, but their generation may require expensive tools.

3.1 Part of Speech Tags

POS tags are very important for dependency parsing, as a purely lexical parser may lead to sparseness but adding POS tags provides a coarser grammatical category. This generalization of words

[4] (http://kcis.iiit.ac.in/LT)

[5] The complete set of dependency relation types can be found in (Bharati et al., 2009)

help as words belonging to the same part-of-speech are expected to have the same syntactic behavior. McDonald et al. (2011) have shown in their delexicalised parser that most of the information is captured in POS tags and just using them as features provides high unlabeled attachment score (UAS). However, for labeled dependency parsing, especially for semantic-oriented dependencies like Paninian dependencies these non-lexical features are not predictive enough.

3.2 Word

It is an indispensable unit for labeled dependency parsing. It is important for resolving ambiguous relationships for dependency parsing. But lexical units are sparse and difficult to learn given a limited training data set. This sparsity is observed more in morphologically rich languages.

3.3 Vibhakti (Suffix and Postpositions)

In a relatively fixed word order language like English the position of a word or phrase relative to the verbal head, gives cues for grammatical relations. On the other hand free word order and morphologically rich languages change the morphological form of the dependent word, the head word, or both in order to represent grammatical relations. This information about grammatical relations thus remains available irrespective of the position of words. The morphemes (suffixes) in Dravidian languages explicitly represent grammatical and semantic relations in a sentence. This is in contrast to Indo-Aryan languages where case marking can also be expressed lexically as postpositions to establish relations between nominals and verbal predicates, the degree of which depends on their varying morphological richness. Hindi and Urdu are relatively sparse in morphology when compared to Bengali, which in turn is less rich than Marathi. These units called vibhakti that exhibit case marking are important surface cues that help identify various dependency relations. Also are important the units that mark Tense, Aspect, Modality (TAM) of a verb. There exists a direct mapping between many TAM labels and the nominal case markers because TAMs control the case markers of some nominals. Different languages tend to encode syntactically relevant information in different ways. It has been shown in previous works for Hindi(Ambati et al., 2010b) that the integration of morphological and syntactic information boosts the accuracy for treebanks that are syntacto-semantic in nature. We experiment to see the extent to which it helps the other Indian languages.

3.4 Chunk Tag

Previous work on Hindi (Ambati et al., 2010a) has shown that considerable improvement in parsing could be achieved using the local morphosyntactic features like chunk tags. In analytical languages, where information about finiteness or non finiteness of verbs is not captured in the chunk head alone but is also indicated by postpositions and auxiliaries following the head, the different chunk level tags[6]can help the parser identify different syntactic behavior of these verbs. For example a finite verb can become the root of the sentence, whereas a non-finite or infinitival verb cannot. Ambati et al. (2010a) used a coarser POS tag scheme so the improvement observed on addition of chunk was major. But in the new tagset that we are using, the finiteness information for verbs is marked at the POS level too. Therefore we experiment to see how far the chunk information helps us in this setting.

3.5 Gender, Number, Person

We want to capture the agreement between verb and its arguments in all languages by the addition of other morphological features such as gender, number and person (GNP) for each node. The verb agrees in GNP with the highest available karaka `k1` usually. But agreement rules can be complex, it may sometimes take default feature or agree with karaka `k2` in some cases. The problem worsens when there is a complex verb. Similar problems with agreement features have also been noted by (Goldberg and Elhadad, 2009). So we experiment to see if the parser can learn selective agreement pattern for different languages.

Kannada and Malayalam have a three gender system - gender marking is based on semantics. Human males and females are masculine and feminine gender respectively, whereas all things and animals are neuter gender. Telugu also has a three-gender system but human females are grouped with neuter nouns in singular, and human males in plural. The verb in Malayalam is not marked for number, gender person. Similarly in Bengali, the verb changes according to the person information

[6] finite, non-finite, infinitival and gerundial (Bharati et al., 2006)

only, it exhibits no grammatical gender phenomena at all. Marathi also has a three gender system - masculine, feminine and neuter.

4 Experimental Setup

In our experiments, we focus on establishing dependency relations between the chunk heads which we henceforth denote as inter-chunk parsing. The relations between the tokens of a chunk (intra-chunk dependencies) are not considered for experimentation as they can easily be predicted automatically using a finite set of rules (Kosaraju et al., 2012). Moreover we also observed the high learnability of intra-chunk relations from an initial experiment. We found the accuracies of intra-chunk dependencies to be more than 99.00% for both Labeled Attachment and Unlabeled Attachment. The treebanks available to us are in the SSF format (Bharati et al., 2007). We use in house built tool to convert from SSF to CoNLL format. This tool uses head and vibhakti computation tools as its dependencies. The head computation tool finds the head of a chunk based on certain rules written using POS tag information of nodes. The vibhakti computation module is again a simple, rule based tool that uses POS tag information to decide whether a lexical unit qualifies as a postposition or not. It then augments the head of the chunk with its postpositional features in the SSF format. Our parser uses data in the converted CoNLL format.

We use the arc-eager parsing model for parsing sentences containing projective arcs only, discarding the non-projective sentences. The data set is split in the ratio of 80-10-10 for training, testing and tuning the parsing model. Baseline for parsing is set using a delexicalised model having only POS tags as features . We explore with different feature sets by adding features like words, suffix, chunk tags and GNP information one by one. These features are represented as described below. In order to parse in more realistic settings, we also show parsing results using predicted POS and chunk tags obtained from the models discussed below. We report auto accuracy of the parsing model on the same training, development and testing sets that are used for parsing with gold tags.

4.1 Parsing Model

We have used a non-linear neural network greedy transition-based parser, similar in structure to (Chen and Manning, 2014). A few new features have been introduced in the input layer of the model as described below. Our parsing model is based on transition-based dependency parsing paradigm (Nivre, 2008). Particularly, we use an arc-eager transition system (Nivre, 2003). The arc-eager system defines a set of configurations for a sentence w_1 ,...,w_n where each configuration C = (S, B, A) consists of a stack S, a buffer B, and a set of dependency arcs A. For each sentence, the parser starts with an initial configuration where S = [ROOT], B = [w_1 ,...,w_n] and A = ϕ and terminates with a configuration C if the buffer is empty and the stack contains the ROOT. The parse trees derived from transition sequences are given by A. To derive the parse tree, the arc-eager system defines four types of transitions (t): 1) Shift, 2) Left-Arc, 3) Right-Arc, and 4) Reduce. We use a non-linear neural network to predict the transitions for the parser configurations. The neural network model is the standard feed-forward neural network with a single layer of hidden units. We use 200 hidden units and RelU activation function. The output layer uses softmax function for probabilistic multi-class classification. The model is trained by minimizing cross entropy loss with an l2-regularization over the entire training data. We also use mini-batch Adagrad for optimization (Duchi et al., 2011) and apply dropout (Hinton et al., 2012). The parameters like number of iterations, learning rate, embedding size were tuned on the development set.

From each parser configuration, we extract features related to the top four nodes in the stack, top four nodes in the buffer and leftmost and rightmost children of the top two nodes in the stack and the leftmost child of the top node in the buffer.

4.2 Part of Speech Tagging and Chunking Model

We trained POS taggers and Chunkers for all the five languages using a similar neural network architecture like parsing, discussed above. Second order structural features in the form of lexical and non-lexical units were used. The input layer consisted of the current word, words in the context size of 2 surrounding the current word and the last four characters of all these words. Intra-word information is extremely useful when dealing with morphologically rich languages as word internal features contribute more context than word external features while predicting POS and chunk tags.

Using POS tags as feature has obvious benefits for chunking. At least chunk tags can be deterministically predicted if the POS tags are known. But a chunking model using auto POS tags gives less accuracy than a sans POS model. For example in Kannada, using gold POS tags in chunker gave an accuracy of 99.46%, sans POS model gave 95.25% but model having auto POS tags reduced it to 95%. So we stuck to using only lexical and suffix features while chunking.

4.3 Representation of Lexical Units

In our non-linear parsing model, we use distributed representation of lexical features. Using distributed representation, units of words are projected to a low dimensional continuous vector space. Unlike sparse representation in linear models, these word embeddings allow words that are closer in the embedding space to share the model parameters, thus providing an efficient solution to the problem of data sparsity. Moreover since word embeddings are assumed to capture semantic and syntactic aspects of a word, they can also improve the correlation between words and dependency labels. The same representations are also used in the POS tagger.

The monolingual corpora of all the languages are used to learn their respective word embeddings. The data is collected from various sources such as Wikipedia dump[7], ILCI - health, tourism agriculture and entertainment data (Jha, 2010), raw corpus from EMILLE / CIIL (Xiao et al., 2004), LCC (Goldhahn et al., 2012), part of Opensubtitles corpus (Tiedemann, 2009), to train rich domain independent word-embeddings so that our parsing model is not biased. We use the Skip-gram model with negative sampling implemented in the open-source `word2vec` toolkit (Mikolov et al., 2013) to learn word representations. The context window size was kept to 1, as shorter context captures more syntactic relatedness compared to longer contexts that capture semantic and topical similarity. The word embedding size was experimented with and embeddings of dimension 64 gave the best results.

4.4 Representation of POS, Chunk and GNP Tags

POS tags are small in number, but show semantic similarity like words. We use distributed representations for POS tags also by projecting them to a continuous low dimensional vector space. Similar settings as the above word embedding mode were used, while keeping the embeddings' dimension to be 20. The model for each language was trained on ILCI POS tagged data and treebank data that we were already using. The words were replaced by their corresponding tags to form a sequence. To represent chunk tags and GNP information, we use randomly initialized embeddings in the range of -0.25 to +0.25. The dimension of input vectors are taken to be 5.

In a real time setting, GNP information cannot be learnt from unlabeled monolingual data but require the presence of a morphological analyzer. It is an expensive tool to build. Due to the unavailability of a decently accurate tool for these resource poor languages, we have used gold tags in all our experiments just to observe their influence on parsing.

4.5 Representation of Vibhakti (Suffix and Postpositions)

Morphologically rich languages like Dravidian Languages, are highly agglutinative. The same root words inflect to have many word forms with different suffixes and prefixes. These morphemes denote the grammatical relation between a word and its arguments and may also represent TAM. This poses a problem to efficiently learn word embeddings for them. Most word embedding models consider word as a basic independent entity without considering its internal structure and shape. No explicit relationship among morphologically related words are captured too. While some work has been done to learn character based embeddings using deep neural networks for specific tasks like POS tagging, learning language models, learning word similarity etc, they are a different end to end architecture in themselves and cannot be used in integration with our parsing model. Therefore we thought it might be a good idea to treat suffixes - the last 4 characters of a word as separate units and learn embedding for them using `word2vec` to capture the linguistic regularity. This provides a potential solution for estimating rare and complex words rather than representing them in a crude way using only one or a few vectors. Instead of using the last few characters we could have used the case and TAM information present in the treebank in the form of

[7] `https://dumps.wikimedia.org`

linguistic morphemes for each word, but due to the absence of a decent or no morphological analyzer for these languages, these features would not have been available for real time parsing of development and test set. Moreover since there are more than one morpheme in a word, methods to jointly learn word and character embeddings and composing them to yield a single representation (Bojanowski et al., 2016), need to be explored for these languages.

For Indo-Aryan langauges the degree of case and TAM marking being a part of word morphology varies according to the morphological richness of the language. This information can also be expressed lexically as postpositions or as auxiliaries in contrast to the Dravidian languages. Since we experiment on inter-chunk parsing and establish relations between heads of chunks, this information is lost. So we compose a vector by averaging the representations (that are looked up from the `word2vec` embedding model described above) of these postpositions and auxiliaries present in a chunk, and use it as a feature.

5 Results

The results of experimenting with the features described in Section 4 for all the 5 languages are presented in the Table 4. The metrics used for evaluation are Unlabeled and labeled attachment score (UAS and LAS) and label accuracy (LA). The performance corresponding to the highest performing feature set has been highlighted. The tags in our treebanks are syntactico-semantic and it has been observed with other treebanks that learning such tags is difficult (Nivre et al., 2007a). Despite that we achieve decent LAS for all 5 languages. We also experiment with a coarser scheme of POS tags for Telugu to see the effect of the granularity of POS tag on dependency parsing. Since some 149 sentences are lost in automatic conversion from coarser to finer treebank representation for Telugu, we cannot directly compare their parsing performance but can still get an idea that the coarser scheme is better in predicting LAS. This was not so intuitive as the richer information encoded in finer POS tagset should have helped the parser disambiguate dependency relations. We leave the label wise dependency relation analysis, taking into account the granularity of the POS tags for future work. Our delexicalised parser using only POS tags (f1) achieves good results for unlabeled parsing for all languages and serves as a good baseline. However it gives poor results for LA and in turn for LAS as was expected, lowest LAS and UAS being for Marathi. On addition of suffix features to POS tags (f2) LAS shows a substantial increase for all languages, for an example +21.37% for Kannada gold test set. Though the highest increase is for Marathi as its baseline is very poor and even the partial lexical information gives the parser a major boost. The lowest increase of +9.1% is in Telugu gold test set. Different Dravidian languages show different levels of sophistication in case marking encoded in their suffixes. While in Kannada adding full lexical information (f3) to the baseline delexicalised parser does not increase accuracy a lot in comparison to f2, in Malayalam f2 that is partial lexical information (suffix and POS tags) perform better than f3.

We see that addition of suffix embeddings to word and POS tags (f4), acts as a complementary feature and shows substantial increase for Kannada and Malayalam, whereas quite less for Telugu. Bengali parser however does not show much increase as it is an Indo-Aryan language. Marathi shows a considerable increase despite being an Indo-Aryan language as it is morphologically richer and behaves likes pseudo Dravidian. Geographically it is also the southernmost Indo-Aryan language and shows syntactic convergence with the neighboring Dravidian language family. Similarly adding postposition information (f5) benefits Bengali parser considerably as compared to other Dravidian languages and Marathi.

It is noticed that adding chunk tag information (f6) helps across all languages, specially in LA as was conjectured. However the increase is slightly more for Indo-Aryan compared to Dravidian as in the latter the the average number of words in a chunk is less owing to the agglutinative nature of the languages. The head word and its morphemes encode most of the information for finite or non finiteness of verbs and case markers and is available to us in inter-chunk parsing. While in Bengali and Marathi the information marked by verb auxiliaries and postpositions supporting the head word in a chunk are lost, so the additional chunk information helps to disambiguate between the root and non root verb in complex constructions.

Next we see the effect of GNP information (f7) on parsing accuracies. There is an increase in all languages except Malayalam. It is reason-able as there is no agreement between Malayalam verbs and their arguments. However it increases for Malayalam in the auto development and auto test set. It could be due to inconsistencies within the data. GNP marking is also very noisy for Marathi data, may be it could be looked into for validation. We could not report results for Bengali for this feature as the data is not marked for morphological information.

We have also reported the performance of our POS tagger and chunker for all 5 languages in Table 3. With very simple features it gives better or comparable results for all languages compared to Bengali (Ghosh, 2013; Alam et al., 2016), Malayalam (V V and Sharma, 2016). Our results on Telugu and Bengali parsing or POS tagging cannot be compared directly to the previous works as we used a different dataset with a finer POS tagging scheme. Numerically it is still better than their results, it could be owed to the increase in size of the dataset, the architecture of our neural network models and dense representation of features.

Thus we show empirically that the presented feature set is useful for a range of morphologically rich languages across different language families, however some features are more important to certain languages than others.

6 Conclusion and Future work

We have presented our work to adapt an existing neural network parser to suit the particularities for 5 Indian languages Kannada, Malayalam, Telugu, Bengali and Marathi belonging to two major language families Dravidian and Indo-Aryan. We proposed a unified strategy for all languages for the inclusion of rich-morphosyntactic cues in the existing parsing framework. The cost effective representation of the linguistically motivated features such as suffix, postposition, chunk and GNP aim to capture the linguistic intricacies of all languages. A detailed discussion of the rationale behind each feature and their effect on parsing accuracy was presented. Our results provided the comparison that suffix information is more useful for parsing Dravidian languages while postposition is for Indo-Aryan languages, with the exception of Marathi. We showed the performance of our parser in real time settings by using auto POS and chunk tag. In turn we also built POS taggers and chunkers for these resource poor languages. Through our work we aimed to open avenues for further research in dependency parsing for these underrepresented languages. As a future work we propose to build cross-lingual parsers for these languages by exploiting the topological and genetic similarities among them. Since Indian languages are morphologically very rich, ways of learning character-aware POS tagging and dependency parsing models could also be explored.

		Kan	Mal	Tel Bis	Tel Ann.	Ben	Mar
chunk	D	95.23	96.59	93.73	91.89	94.26	94.42
	T	95.25	96.74	91.28	93.17	94.25	94.93
pos	D	92.85	93.06	83.76	90.29	89.74	91.49
	T	92.31	92.78	83.31	88.81	89.34	91.83

Table 3: Accuracy of Chunker and POS Model for Kannada (Kan), Malayalam (Mal), Bengali (Ben), Marathi (Mar), Telugu (Tel) Bis and Anncorra (Ann.) tagset. D=Development Set, T=Test Set.

Acknowledgement

We would like to thank the reviewers for their valuable and insightful comments that helped to improve the quality of this paper. We also extend our thanks to Irshad Ahmad Bhat for making his code from previous works available and giving relevant inputs.

References

Firoj Alam, Shammur Absar Chowdhury, and Sheak Rashed Haider Noori. 2016. Bidirectional lstmscrfs networks for bangla pos tagging. In *Computer and Information Technology (ICCIT), 2016 19th International Conference on*, pages 377–382. IEEE.

Bharat Ram Ambati, Samar Husain, Sambhav Jain, Dipti Misra Sharma, and Rajeev Sangal. 2010a. Two methods to incorporate local morphosyntactic features in hindi dependency parsing. In *Proceedings of the NAACL HLT 2010 First Workshop on Statistical Parsing of Morphologically-Rich Languages*, pages 22–30. Association for Computational Linguistics.

Bharat Ram Ambati, Samar Husain, Joakim Nivre, and Rajeev Sangal. 2010b. On the role of morphosyntactic features in hindi dependency parsing. In *Proceedings of the NAACL HLT 2010 First Workshop on Statistical Parsing of Morphologically-Rich Languages*, pages 94–102. Association for Computational Linguistics.

Rafiya Begum, Samar Husain, Arun Dhwaj, Dipti Misra Sharma, Lakshmi Bai, and Rajeev Sangal. 2008. Dependency annotation scheme

Feat.	Gold						Auto					
	Development			Test			Development			Test		
	LAS	UAS	LA	LAS	UAS	LA	LAS	UAS	LA	LAS	UAS	LA
Kannada												
f1	54.95	79.04	56.92	55.62	80.61	57.28	52.94	77.4	55.31	53.6	78.9	55.67
f2	75.82	90.8	78.58	76.99	92.29	79.18	73.15	88.71	76.76	73.89	89.88	76.99
f3	76.01	90.06	78.63	77.16	91.32	79.41	73.36	88.02	76.87	74.27	89.2	77.26
f4	79.46	91.54	82.37	80.74	92.94	83.03	76.63	89.36	80.42	77.53	90.76	80.71
f5	79.5	91.76	82.46	80.89	92.99	83.39	76.88	89.68	80.73	77.67	90.74	81.21
f6	**79.61**	**91.48**	**82.64**	80.68	93.07	83.27	76.92	89.95	80.79	77.59	90.98	81.02
f7	79.52	91.62	82.5	**80.99**	**93.26**	**83.47**	**77.01**	**89.83**	**80.82**	**78.07**	**91.03**	**81.45**
Malayalam												
f1	50.36	77.35	54.66	49.08	76.17	53.79	48.43	75.63	53.12	47.29	74.42	52.11
f2	65.15	84.29	70.57	66.69	85.94	71.34	62.06	82.29	68.21	64.18	83.8	69.66
f3	61.95	82.93	66.43	61.52	82.92	66.08	59.46	81.19	64.24	59.67	81.58	64.44
f4	67.88	85.41	72.88	68.5	85.99	73.47	64.66	83.35	70.5	65.91	84.37	71.12
f5	68.17	84.58	73.39	**70.76**	**86.29**	**75.64**	65.21	82.68	71.2	**68.12**	**84.58**	**73.36**
f6	**68.94**	**85.31**	**73.83**	70.02	86.5	74.79	65.28	83.03	70.88	67.44	84.65	72.81
f7	68.38	84.76	73.59	69.89	86.09	74.96	**65.78**	**83.32**	**71.42**	67.57	84.94	73.3
Telugu (Anncorra)												
f1	57.37	87.84	58.85	54.53	85.16	56.54	55.28	86.24	57.13	52.89	83.51	55.24
f2	69.04	92.63	70.02	66.67	93.17	67.49	68.3	91.65	69.16	65.72	92.46	67.02
f3	74.2	94.1	75.43	70.67	92.93	71.85	72.73	93.37	74.32	69.85	92.11	71.26
f4	74.69	93.98	75.92	**73.14**	**94.11**	**74.32**	74.03	93.73	75.68	**71.61**	**93.29**	**73.14**
f5	74.82	93.61	76.29	72.44	94.11	73.5	74.2	93.73	75.92	70.44	92.58	72.08
f6	**75.43**	**94.84**	**76.78**	71.73	93.05	72.91	**74.45**	**93.37**	**76.17**	70.55	93.05	71.97
f7	75.31	94.59	76.29	72.79	93.76	73.97	72.97	92.75	74.57	70.91	92.82	72.2
Telugu (BIS)												
f1	54.73	90.03	55.63	56.26	87.61	57.65	53.45	89.0	55.12	55.37	87.23	57.14
f2	66.11	93.09	67.65	65.36	92.04	66.88	63.55	91.82	65.86	65.23	91.66	66.75
f3	69.95	93.48	71.61	69.15	91.91	70.54	69.31	93.09	70.97	67.64	91.28	69.28
f4	70.72	93.09	72.76	69.28	91.4	71.3	70.72	92.97	72.89	68.72	91.15	70.54
f5	72.25	93.99	73.66	69.28	91.66	71.3	71.1	93.73	72.63	69.15	91.66	71.18
f6	**73.53**	**94.63**	**74.81**	71.55	93.17	72.95	**72.38**	**93.99**	**73.91**	**69.91**	**92.16**	**71.93**
f7	72.89	94.88	74.17	**71.93**	**92.92**	**73.58**	71.61	93.86	73.53	68.35	90.39	70.67
Marathi												
f1	34.81	59.92	39.29	34.06	59.11	38.5	34.83	60.24	39.1	33.45	58.63	38.24
f2	64.25	83.52	68.79	62.57	81.15	67.38	63.96	83.38	68.44	61.98	80.74	66.94
f3	66.27	84.6	69.67	65.22	83.66	69.2	66.08	84.58	69.45	65.11	83.41	69.14
f4	70.33	86.99	74.1	68.12	84.33	72.69	70.39	87.04	74.04	68.07	84.48	72.61
f5	70.47	87.32	74.26	68.42	85.18	72.44	70.25	87.19	74.15	68.0	84.92	72.07
f6	71.01	87.72	74.75	69.56	86.28	73.42	70.46	87.5	74.18	68.45	86.06	72.3
f7	**71.56**	**88.05**	**74.95**	**69.75**	**86.41**	**73.52**	**70.97**	**87.69**	**74.47**	**69.01**	**85.98**	**72.82**
Bengali												
f1	52.71	78.08	55.22	52.52	78.7	54.93	49.33	74.65	52.82	48.34	74.89	51.63
f2	68.19	85.37	70.82	67.6	84.86	70.68	64.42	82.1	68.26	63.61	81.75	67.38
f3	71.54	85.43	74.51	70.45	85.23	73.29	68.76	83.09	72.48	66.96	82.22	70.8
f4	72.86	85.81	76.07	71.66	86.26	74.55	69.55	83.22	73.55	68.5	83.62	72.69
f5	75.82	87.6	79.05	74.66	87.27	78.22	73.26	85.72	77.21	**72.65**	**85.86**	**76.55**
f6	**76.43**	**88.41**	**79.67**	**75.64**	**88.41**	**78.63**	**73.28**	**86.08**	**77.29**	72.24	85.99	75.92
f7	-	-	-	-	-	-	-	-	-	-	-	-

Table 4: Parsing accuracies of our neural network based parser for all 5 languages. Auto development and test set contain predicted POS and chunk tags. Gloss of the features are f1 = POS only, f2 = f1+ suffix, f3 = POS + word, f4 = f3 + suffix , f5 = f4+ PSP, f6 = f5 + chunk, f7 = f6 + GNP

for indian languages. In *IJCNLP*, pages 721–726. Citeseer.

A. Bharati, V. Chaitanya, R. Sangal, and KV Ramakrishnamacharyulu. 1995. *Natural Language Processing: A Paninian Perspective*. Prentice-Hall of India.

Akshar Bharati, Rajeev Sangal, Dipti Misra Sharma, and Lakshmi Bai. 2006. Anncorra: Annotating corpora guidelines for pos and chunk annotation for indian languages. *LTRC-TR31*.

Akshar Bharati, Rajeev Sangal, and Dipti M Sharma. 2007. Ssf: Shakti standard format guide.

Akshar Bharati, Samar Husain, Bharat Ambati, Sambhav Jain, Dipti Sharma, and Rajeev Sangal. 2008a. Two semantic features make all the difference in parsing accuracy. *Proc. of ICON*, 8.

Akshar Bharati, Samar Husain, Dipti Misra Sharma, and Rajeev Sangal. 2008b. A two-stage constraint based dependency parser for free word order languages. In *Proceedings of the COLIPS International Conference on Asian Language Processing 2008 (IALP)*.

Akshar Bharati, DM Sharma S Husain, L Bai, R Begam, and R Sangal. 2009. Anncorra: Treebanks for indian languages, guidelines for annotating hindi treebank (version–2.0).

Riyaz Ahmad Bhat, Irshad Ahmad Bhat, Naman Jain, and Dipti Misra Sharma. 2016a. A house united: Bridging the script and lexical barrier between hindi and urdu. In *International Conference on Computational Linguistics (COLING 2016)*.

Riyaz Ahmad Bhat, Irshad Ahmad Bhat, and Dipti Misra Sharma. 2016b. Improving transition-based dependency parsing of hindi and urdu by modeling syntactically relevant phenomena. *ACM Transactions on Asian and Low-Resource Language Information Processing (TALIP)*.

Riyaz Ahmad Bhat. 2017. Exploiting linguistic knowledge to address representation and sparsity issues in dependency parsing of indian languages.

Rajesh Bhatt, Bhuvana Narasimhan, Martha Palmer, Owen Rambow, Dipti Misra Sharma, and Fei Xia. 2009. A multi-representational and multi-layered treebank for hindi/urdu. In *Proceedings of the Third Linguistic Annotation Workshop*, pages 186–189. Association for Computational Linguistics.

Piotr Bojanowski, Edouard Grave, Armand Joulin, and Tomas Mikolov. 2016. Enriching word vectors with subword information. *arXiv preprint arXiv:1607.04606*.

Danqi Chen and Christopher D Manning. 2014. A fast and accurate dependency parser using neural networks. In *EMNLP*, pages 740–750.

Narayan Choudhary and Girish Nath Jha. 2011. Creating multilingual parallel corpora in indian languages. In *Language and Technology Conference*, pages 527–537. Springer.

Ronan Collobert, Jason Weston, Léon Bottou, Michael Karlen, Koray Kavukcuoglu, and Pavel Kuksa. 2011. Natural language processing (almost) from scratch. *Journal of Machine Learning Research*, 12(Aug):2493–2537.

Jacob Devlin, Rabih Zbib, Zhongqiang Huang, Thomas Lamar, Richard M Schwartz, and John Makhoul. 2014. Fast and robust neural network joint models for statistical machine translation. In *ACL (1)*, pages 1370–1380. Citeseer.

John Duchi, Elad Hazan, and Yoram Singer. 2011. Adaptive subgradient methods for online learning and stochastic optimization. *Journal of Machine Learning Research*, 12(Jul):2121–2159.

Arup Ratan Ghosh. 2013. *Memory Based Learner for Bengali POS Tagging*. Ph.D. thesis, JADAVPUR UNIVERSITY.

Yoav Goldberg and Michael Elhadad. 2009. Hebrew dependency parsing: Initial results. In *Proceedings of the 11th International Conference on Parsing Technologies*, pages 129–133. Association for Computational Linguistics.

Dirk Goldhahn, Thomas Eckart, and Uwe Quasthoff. 2012. Building large monolingual dictionaries at the leipzig corpora collection: From 100 to 200 languages. In *LREC*, pages 759–765.

Geoffrey E Hinton, Nitish Srivastava, Alex Krizhevsky, Ilya Sutskever, and Ruslan R Salakhutdinov. 2012. Improving neural networks by preventing co-adaptation of feature detectors. *arXiv preprint arXiv:1207.0580*.

Matt Hohensee and Emily M Bender. 2012. Getting more from morphology in multilingual dependency parsing. In *Proceedings of the 2012 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 315–326. Association for Computational Linguistics.

Matthew Hohensee. 2012. *It's only morpho-logical: Modeling agreement in cross-linguistic dependency parsing*. Ph.D. thesis.

Samar Husain, Prashanth Mannem, Bharat Ambati, and Phani Gadde. 2010. The icon-2010 tools contest on indian language dependency parsing. *Proceedings of ICON-2010 Tools Contest on Indian Language Dependency Parsing, ICON*, 10:1–8.

Girish Nath Jha. 2010. The tdil program and the indian langauge corpora intitiative (ilci). In *LREC*.

Sruthilaya Reddy Kesidi. 2013. *CONSTRAINT-BASED HYBRID DEPENDENCY PARSER FOR TELUGU*. Ph.D. thesis, International Institute of Information Technology Hyderabad, India.

Sudheer Kolachina, Prasanth Kolachina, Manish Agarwal, and Samar Husain. 2010. Experiments with malt parser for parsing indian languages. *Proc of ICON-2010 tools contest on Indian language dependency parsing. Kharagpur, India*.

Prudhvi Kosaraju, Samar Husain, Bharat Ram Ambati, Dipti Misra Sharma, and Rajeev Sangal. 2012. Intra-chunk dependency annotation: expanding hindi inter-chunk annotated treebank. In *Proceedings of the Sixth Linguistic Annotation Workshop*, pages 49–56. Association for Computational Linguistics.

Sandra Kübler, Ryan McDonald, and Joakim Nivre. 2009. Dependency parsing. *Synthesis Lectures on Human Language Technologies*, 1(1):1–127.

Prashanth Mannem. 2009. Bidirectional dependency parser for hindi, telugu and bangla. *Proceedings of ICON09 NLP Tools Contest: Indian Language Dependency Parsing, India*.

Ryan McDonald, Koby Crammer, and Fernando Pereira. 2005. Online large-margin training of dependency parsers. In *Proceedings of the 43rd annual meeting on association for computational linguistics*, pages 91–98. Association for Computational Linguistics.

Ryan McDonald, Slav Petrov, and Keith Hall. 2011. Multi-source transfer of delexicalized dependency parsers. In *Proceedings of the conference on empirical methods in natural language processing*, pages 62–72. Association for Computational Linguistics.

Tomas Mikolov, Kai Chen, Greg Corrado, and Jeffrey Dean. 2013. Efficient estimation of word representations in vector space. *arXiv preprint arXiv:1301.3781*.

Joakim Nivre, Johan Hall, Jens Nilsson, Atanas Chanev, Gülsen Eryigit, Sandra Kübler, Svetoslav Marinov, and Erwin Marsi. 2007. Maltparser: A language-independent system for data-driven dependency parsing. *Natural Language Engineering*, 13(02):95–135.

Joakim Nivre. 2003. An efficient algorithm for projective dependency parsing. In *Proceedings of the 8th International Workshop on Parsing Technologies (IWPT*. Citeseer.

Joakim Nivre. 2008. Algorithms for deterministic incremental dependency parsing. *Computational Linguistics*, 34(4):513–553.

Joakim Nivre. 2009. Parsing indian languages with maltparser. *Proceedings of the ICON09 NLP Tools Contest: Indian Language Dependency Parsing*, pages 12–18.

Martha Palmer, Rajesh Bhatt, Bhuvana Narasimhan, Owen Rambow, Dipti Misra Sharma, and Fei Xia. 2009. Hindi syntax: Annotating dependency, lexical predicate-argument structure, and phrase structure. In *The 7th International Conference on Natural Language Processing*, pages 14–17.

Richard Socher, John Bauer, Christopher D Manning, and Andrew Y Ng. 2013. Parsing with compositional vector grammars. In *ACL (1)*, pages 455–465.

Juhi Tandon, Himani Chaudhary, Riyaz Ahmad Bhat, and Dipti Misra Sharma. 2016. Conversion from pāninian kārakas to universal dependencies for hindi dependency treebank. *LAW X*, page 141.

Lucien Tesnière. 1959. *Eléments de syntaxe structurale*. Librairie C. Klincksieck.

Jörg Tiedemann. 2009. News from OPUS - A collection of multilingual parallel corpora with tools and interfaces. In N. Nicolov, K. Bontcheva, G. Angelova, and R. Mitkov, editors, *Recent Advances in Natural Language Processing*, volume V, pages 237–248. John Benjamins, Amsterdam/Philadelphia, Borovets, Bulgaria.

Reut Tsarfaty, Djamé Seddah, Yoav Goldberg, Sandra Kübler, Marie Candito, Jennifer Foster, Yannick Versley, Ines Rehbein, and Lamia Tounsi. 2010. Statistical parsing of morphologically rich languages (spmrl): what, how and whither. In *Proceedings of the NAACL HLT 2010 First Workshop on Statistical Parsing of Morphologically-Rich Languages*, pages 1–12. Association for Computational Linguistics.

Reut Tsarfaty, Djamé Seddah, Sandra Kübler, and Joakim Nivre. 2013. Parsing morphologically rich languages: Introduction to the special issue. *Computational Linguistics*, 39(1):15–22.

Devadath V V and Dipti Misra Sharma. 2016. Significance of an accurate sandhi-splitter in shallow parsing of dravidian languages. In *Proceedings of the ACL 2016 Student Research Workshop*, pages 37–42, Berlin, Germany, August. Association for Computational Linguistics.

Mengqiu Wang and Christopher D Manning. 2013. Effect of non-linear deep architecture in sequence labeling. In *IJCNLP*, pages 1285–1291.

Fei Xia, Owen Rambow, Rajesh Bhatt, Martha Palmer, and Dipti Misra Sharma. 2009. Towards a multi-representational treebank. In *The 7th International Workshop on Treebanks and Linguistic Theories. Groningen, Netherlands*, pages 159–170.

RZ Xiao, AM McEnery, JP Baker, and Andrew Hardie. 2004. Developing asian language corpora: standards and practice. In *The 4th Workshop on Asian Language Resources*.