Context Based Morphological Analysis

Thesis submitted in partial fulfillment
of the requirements for the degree of

Master of Science (by Research)
in
Computer Science & Engineering

by

Malladi Deepak Kumar
200802022
deepak.malladi@research.iiit.ac.in

International Institute of Information Technology
Hyderabad - 500 032, INDIA
February 2016

International Institute of Information Technology
Hyderabad, India

CERTIFICATE

It is certified that the work contained in this thesis, titled "Context Based Morphological Analysis" by Malladi Deepak Kumar, has been carried out under my supervision and is not submitted elsewhere for a degree.

Date

Advisers: Prof. Dipti Misra Sharma

To my mother and father

Acknowledgments

I would like to express my deepest gratitude for my advisor Prof. Dipti Misra Sharma, without whom this thesis would not have been possible. I must sincerely thank her for constantly supporting me. I enjoyed the meetings I had with her early in the mornings which were so fresh and informative. Her deep linguistics insights helped me a lot in this thesis.

The person who actually gave me a direction for my thesis work is my supervisor Mr. Prashanth Mannem. His inputs in writing the research paper are invaluable. He expected nothing short of excellence and I really liked that quality of him. His constant guidance through out his stay at IIIT Hyderabad, had been really helpful in completing my thesis.

I had the honour of working with Prof. Rajeev Sangal for Dual Degree projects. He always encourages to come up with own ideas (believing in intuition) which in a way helped in my thesis. His work in the field of Natural Language Processing really inspired me and I also admire him as a great human being.

I would like to thank Prof. Radhika Mamidi for guiding me in an independent project which gave me the scope to explore more about Dialog Systems. Special thanks to Prof. Soma Paul who taught me the basic Linguistics courses which are the stepping stones for me in the field of NLP.

NLP classes taught by Dr. Sriram and Dr. Samar were fun. Working in LTRC would not have been same without the support of my seniors Prasanth Kolachina, Aswarth, Phani Gadde, Bharath Ram Ambati, Siva Reddy and Abhilash. HTB plays a major role in my work. I thank all the annotators for making things simpler for me.

Arjun Reddy, Puneeth, Jayendra Rakesh, Vinay, Sarvesh and Nitesh were wonderful as LTRC lab mates. Jackie deserves special mention for helping me with MT experiments. I had done the experiments related to Hindi Parsing Shared Task with Puneeth. Sai Krishna had helped me in performing experiments related to SMA++.

Puneeth and Rahul have been there for me through out the time I had spent in IIIT. UG2K8 batch was awesome and I cherish the friendships that I have formed with them. My juniors Vasanth, Trinath and my cricket friends made my stay even more memorable.

It would not be complete without acknowledging the support of my family. My father, Mr. Sripad, always supported me and most importantly he let me choose what I wanted to do. My mother, Mrs. Durga, and my brother accepted me for what I am and they had complete belief in me. My unconditional love to my grandparents for their constant guidance and care. Alekhya gave me the constant inspiration

during the writing part of thesis. A special round of mention to my friends and relatives who renewed my energies when needed. Finally, I thank the reviewers, Dr.Sriram and Dr.Soma, in helping shaping the thesis better.

Abstract

Morphological analysis is a fundamental task in most Natural Language Processing (NLP) applications, especially for morphologically rich languages such as Hindi. Morphological analysis for Hindi involves predicting lemma, POS, gender, number, person, case-marker, TAM and vibhakti. In general, morphological analyzers predicts all possible analyses for a given word. For most of the NLP tasks, instead of having multiple multiple analyses for a word, we need to disambiguate those multiple analyses and arrive at one analysis which best fits in the given sentential context.

The prime motivation for carrying out the research in this thesis comes from the initial set of Hindi parsing experiments carried out by us. The existing morphological analyzers for Hindi do not predict context based morphological analysis. Because of lack of automatic context based morphological information, parsing accuracy could not be improved. In this thesis, we try to predict a single analysis for a word in a given context. This thesis deals with predicting context based morph information for the attributes viz. lemma, gender, number, person, case-marker, TAM and vibhakti. The existing analyzers also perform poorly for Out-Of-Vocabulary (OOV) words. We also aim to address this issue and make an exhaustive evaluation of our predictions for OOV words.

For lemma prediction, we adopt a machine translation approach. For gender, number, person and case-marker prediction we perceive it as a classification problem. TAM and vibhakti are better predicted by rule based approach. For lemma, gender, number, person and case prediction, we achieved an overall accuracy and OOV accuracy of 84.25% and 63.06% respectively.

To present our case that the predicted morphological information helps in NLP applications, we carry out parsing experiments without and with the predicted morph information and report Labeled Attachment Score of 87.75% and 89.41% for both the experiments respectively.

Building machine translation models are time complex. Hence, in the later part of the thesis we conceive lemma prediction also as a classification problem. We also experiment with other features for gender, number, person and case prediction. We report overall and OOV accuracies of 85.87% and 65.96% respectively. We have seen an 1-2% improvement from our earlier set of experiment results. We extend our approach to other Indian languages viz. Urdu and Telugu for predicting context based morphological information.

Contents

Chapter Page

1 Introduction . . . 1
1.1 Motivation . . . 1
1.2 Problem Statement . . . 2
1.3 Contribution of this Thesis . . . 3
1.4 Thesis Organization . . . 3

2 Morphology . . . 6
2.1 Introduction . . . 6
2.1.1 Inflectional Morphology . . . 6
2.1.2 Derivational Morphology . . . 7
2.2 Morphological Analyzer . . . 7
2.3 Morphological Generator . . . 8
2.4 Context Based Morphological Analyzer . . . 10
2.5 Applications . . . 10

3 Literature Study on Existing Morphological Analyzers . . . 11
3.1 Morphological Analyzers for Hindi . . . 12
3.1.1 Paradigm based analyzer (PBA) . . . 12
3.1.2 Goyal and Lehal's Morph Analyzer . . . 14
3.1.3 Derivational Morphological Analyzer (DMA) . . . 14

4 Hindi Treebank (HTB) . . . 16
4.1 About Hindi Treebank . . . 16
4.2 HTB Data released for Hindi Shared Task on Parsing . . . 16
4.3 Data Statistics . . . 16

5 Hindi Parsing Shared Task . . . 18
5.1 Coling MTPIL shared task . . . 18
5.1.1 Data . . . 18
5.2 Experiments . . . 19
5.2.1 Malt Parser . . . 19
5.2.2 MST Parser . . . 19
5.2.3 Turbo Parser . . . 19
5.2.4 Voting . . . 20
5.2.5 Blending . . . 20

5.3 Results . . . 20
5.3.1 Feature set . . . 20
5.3.2 Evaluation Results . . . 20
5.4 Error Analysis & Conclusion . . . 21

6 Statistical Morphological Analyzer (SMA) . . . 23
6.1 Lemma prediction . . . 23
6.1.1 Lemma prediction perceived as a *Machine Translation Problem* . . . 24
6.1.1.1 Experiment-1 . . . 24
6.1.1.2 Experiment-2 . . . 28
6.1.1.3 Experiment-3 . . . 28
6.1.1.4 Experiment-4 . . . 28
6.1.1.5 Experiment-5 . . . 28
6.2 Gender, Number, Person and Case (GNPC) Prediction . . . 29
6.2.1 Gender . . . 30
6.2.2 Number . . . 30
6.2.3 Person . . . 31
6.2.4 Case Marker . . . 31
6.3 Vibhakti and TAM . . . 32
6.4 Evaluation Systems . . . 34
6.5 Experiments, Results & Error Analysis . . . 35
6.5.1 Lemma . . . 36
6.5.2 Gender, Number, Person and Case . . . 36
6.5.3 TAM and Vibhakti . . . 37
6.6 Effect on Parsing . . . 38
6.7 Extending the work to Telugu and Urdu . . . 40

7 Improvised Statistical Morphological Analyzer (SMA++) . . . 41
7.1 Improvised Statistical Morphological Analyzer (SMA++) . . . 41
7.1.1 Lemma perceived as *Classification Problem* . . . 41
7.1.1.1 Algorithm to generate lemma class label . . . 41
7.1.2 Gender, Number, Person and Case Marker . . . 42
7.1.3 Features . . . 42
7.1.3.1 Word Forms . . . 43
7.1.3.2 Length of the token . . . 43
7.1.3.3 Character Type . . . 43
7.1.3.4 POS . . . 43
7.1.3.5 Suffixes . . . 43
7.1.3.6 Previous Morph Tags . . . 43
7.1.3.7 Next Morph Tags . . . 44
7.1.4 Experiments & Results . . . 44

8 Conclusions and Future Work . . . 45
8.1 Conclusion and Future work . . . 45

9 Related Publications 46
9.1 Statistical Morphological Analyzer for Hindi 46
9.2 Context Based Statistical Morphological Analyzer and Its Effect On Hindi Dependency Parsing 46
9.3 Improvised and Adaptable Statistical Morph Analyzer (SMA++) 46
9.4 Ensembling Various Dependency Parsers: Adopting Turbo Parser for Indian Languages 47

Bibliography 48

List of Figures

Figure Page

2.1 Morphological Analyzer 7
2.2 Morphological Generator 9

3.1 PBA Input Ouput , Image Reference: [4] 14
3.2 DMA Algorithm , Image Reference: [25] 15

6.1 Word based translation, Image Reference: [27] 25
6.2 Phrase based translation, Image Reference: [27] 25
6.3 Character-Level Phrase-based Translation for Lemma Prediction 26

List of Tables

Table		Page
1.1	Multiple morphological analyses for the word `xeSa`	1
1.2	Examples: Morphological Analysis for Hindi word: *ladZake*	2
2.1	Types of Morphology and Differences	7
2.2	Examples: Morphological Analysis for Hindi words	8
2.3	Examples: Morphological Analysis for English words	8
2.4	Input and Output of Morphological Generator	9
2.5	Examples: Context Based Morphological Analysis	9
3.1	Word forms for the root word `ladakA`	13
3.2	Paradigm table for `ladakA` class	13
3.3	Multiple analyses given by the PBA for the words `xeSa` and `cAhie`	13
4.1	HTB statistics	17
5.1	UAS, LA, LAS denote the Unlabeled Attachment score, Labeled Accuracy Score and Labeled Attachment Score respectively. Accuracies on data annotated with gold standard POS tags	21
5.2	Accuracies on data annotated with automatic POS tags	21
5.3	Accuracies reported by all teams participated in Shared Task	22
6.1	Morph features and the values they take	23
6.2	Experiment-1: Sample parallel corpus for lemma prediction	27
6.3	Experiment-2: Sample parallel corpus for lemma prediction with POS as prefix on source side	27
6.4	Experiment-3: Sample parallel corpus for lemma prediction with POS as prefix on both source and target side	27
6.5	Experiment-4: Sample parallel corpus for lemma prediction with POS as suffix on source side	28
6.6	Experiment-5: Sample parallel corpus for lemma prediction with POS as suffix on both source and target side	28
6.7	Accuracies for Lemma Experiments	29
6.8	Gender value examples	30
6.9	Number value examples	31
6.10	Case value examples	31
6.11	Accuracies for Gender, Number, Person, Case Prediction	32
6.12	Sample TAM/Vibhakti Rules based on POS patterns	33

6.13 HTB statistics 34
6.14 Accuracies of SMA compared with F-PBA, O-PBA and baseline systems. 34
6.15 MALT Parser's accuracies on HTB test data. Unlabeled Attachment Score (UAS) is the percentage of words with correct heads. Labeled Accuracy (LA) is the percentage of words with correct dependency labels. Labeled Attachment Score (LAS) is the percentage of words with both correct heads and labels. 35
6.16 Joint Model for Gender, Number, Person, Case 37
6.17 Accuracy for OOV words of PBA 37
6.18 OOV accuracies for words (by POS tags) 38
6.19 Overall accuracies for words (by POS tags) 38
6.20 Evaluation of SMA in a challenging scenario: training data consists only of words analyzed by PBA and test data consists of remaining unanalyzed words. 39
6.21 Accuracy of SMA with auto POS tags 39
6.22 Telugu and Urdu Treebank Statistics 39
6.23 SMA for other Mor-FOW languages: Telugu and Urdu 39

7.1 Accuracies of SMA++ compared with SMA 44

Chapter 1

Introduction

Natural Language Processing (NLP) deals with developing methods which make it possible for computers to comprehend human language. The most fundamental task in NLP deals with understanding the unstructured natural language text in an automated fashion. The branch of NLP which deals with identification, analysis and description of structure of a given language is called *Morphology*.

	Lemma ↓	Gender ↓	Number ↓	Person ↓	Case ↓	TAM/Vibhakti ↓
xeSa (*country*)	xeSa	m	sg	3	d	0
	xeSa	m	pl	3	d	0
	xeSa	m	sg	3	o	0

Table 1.1 Multiple morphological analyses for the word xeSa

Morphological analysis is the task of analysing words in terms of lemma, affixes, parts of speech, gender, number, person, case-marker, TAM and vibhakti. Refer table-1.1 for sample morphological analyses for the Hindi word *xeSa*[1].

1.1 Motivation

Morphological Analysis plays a vital role in various NLP tasks, especially for morphologically rich languages (Ex: Indian languages), such as:

- Parsing
- Machine Translation
- Information Retrieval
- Annotating Corpora and Building Treebanks

[1] WX notation is used to represent Hindi and other Indian language alphabets. Refer [4]

- Search Engines
- Word Sense Disambiguation
- Spell Checkers

Existing Hindi Morphological Analyzers predict all the possible analysis for a given word. They do not consider the sentential context. In NLP tasks, such as parsing, having multiple analysis for a word does not help in improving the parsing accuracy. In sentential context, we need *context based morphological analysis*. We participated in Hindi Parsing Shared Task [44], and we encountered the similar situation. Lack of *context based morphological information* has affected the parsing accuracy. This was the most important factor that motivated us to come up with a method for '*Context Based Morphological Analysis*'

Also, the existing Hindi morphological analyzers provide default analysis for Out-Of-Vocabulary (OOV) words. We also aim to address this issue.

1.2 Problem Statement

Morphological analyzers takes a word in isolation and predict all the possible analyses for that word. For the Hindi word *ladZake*, morphological analyzers predict two possible analysis which the word can take in various contexts. Refer table-1.2.

Input Word	# Analysis ↓	Lemma ↓	Gender ↓	Number ↓	Person ↓	Case ↓	TAM/Vibhakti ↓
`ladZake` (*boy*)	Analysis-1	`ladZakA`	masculine	singular	third	oblique	0
	Analysis-2	`ladZake`	masculine	plural	third	direct	0

Table 1.2 Examples: Morphological Analysis for Hindi word: *ladZake*

Consider the two sentences:

- ladZake Kela rahe hE
- ladZake bETe hEM

In the first sentence, *ladZake* picks up the *Analysis-1* where as in the second sentence, it takes *Analysis-2*.

For NLP tasks such as parsing, to incorporate morph features, we need to disambiguate multiple analyses and pick the correct analysis for a word in a given sentential context. It helps in improving parsing accuracy [2]. With no proper morphological disambiguation module, often it leads to erroneous results. In this thesis, we pursue research on predicting context based morphological analysis.

Also the existing Hindi morphological analyzers are mostly rule based systems. Hence, for OOV words they either do not predict any analysis or predict default analysis. In this thesis, we do not generate any default analysis for OOV instead we propose an approach which predicts morphological information for OOV words.

The problem is formally stated as:

> For a word in a sentence, predict the appropriate morphological analysis in the given context. Also, for OOV words, predict morphological analysis instead of generating default analysis.

1.3 Contribution of this Thesis

There are two major contributions of this thesis:

- Context Based Morphological Analysis
- Handling OOV Words

Towards building Context Based Morphological Analyzer for Hindi, we used a data-driven approach to predict lemma, gender, number, person and case-marker. Simple heuristics are sufficient to predict TAM and vibhakti. We achieved an overall accuracy of 82.12% and for OOV words its 60.07% accurate.

The Hindi parsing accuracy without morph features, with morph features predicted by our system and with gold morph features are 87.75%, 89.41% and 89.82% respectively. Thus with the context based features we are able to improve parsing accuracy and its comparable to that of gold values.

1.4 Thesis Organization

Chapter-2

Chapter Title: Morphology

In this chapter, we introduce the readers to Morphology. We first brief about morphology and followed by a detailed description about Morphological Analyzer with examples. We then introduce the concept of 'Context Based Morphological Analyzer' which forms the basic plot of this thesis.

Chapter-3

Chapter Title: Literature Study on Existing Morphological Analyzers

This chapter reviews the current state of research on the topic relevant to our thesis i.e, Morphological Analysis. More emphasis is laid on the work specific to Hindi language.

Chapter-4

Chapter Title: Hindi Treebank

We describe about the Hindi Treebank Data which is used for the experiments in this thesis.

Chapter-5

Chapter Title: Hindi Parsing Shared Task

Chapter-5 briefly details about the initial set of experiments on Hindi parsing. The outcome of these experiments is the motivation for the problem statement of this thesis. Parsing accuracy can be improved if we have a context based morphological analyzer. We later, use the context based morphological information predicted by our model and show that it helps in improving parsing accuracy.

Chapter-6

Chapter Title: Statistical Morphological Analyzer

Chapter-6 focuses on the approach we have adopted for building a context based morphological analyzer. As said before, Morphological analysis deals with predicting lemma, gender, number, person, case-marker, TAM and vibhakti. For lemma prediction, we perceive it as a machine translation problem. For gender, number, person and case-marker prediction, we perceive it as a classification problem. TAM and vibhakti are better predicted by a rule-based approach. We also do an exhaustive comparison of our system with other existing systems. We re-run the parsing experiments with the morphological information predicted by our system and show the positive effect it has on parsing accuracy. We extend this approach to other languages and show that this approach can be adopted for other languages with minimal effort.

Chapter-7

Chapter Title: Improvised Statistical Morphological Analyzer

Chapter-7 is an extension of the work we did in Chapter-6. In Chapter-6, lemma model was built based on machine translation approach. Machine translation models are time complex, hence in this chapter, we try to perceive lemma prediction as a classification problem. Though its not a direct classification problem, with the help of edit distance operations, we modify it to a classification problem. Also, we experiment with other features and at the end we come up with a common feature set which helps in predicting lemma, gender, number, person and case-marker.

Chapter-8

Chapter Title: Conclusions and Future Work

We conclude our thesis in this chapter providing summary of the thesis and also put forward certain topics for future work.

Chapter-9

Chapter Title: Related Publications

We cite the papers published in various conferences which are part to this thesis work.

Chapter 2

Morphology

2.1 Introduction

Morphology is the study of internal structure of words which is an essential sub-field of Linguistics. It aims to describe the structures of words and patterns of word formation in a language [20]. In general, Morphology describe language structure in terms of *morphemes* and other linguistic units such as Parts-Of-Speech (POS), affixes, etc.

Morpheme is the smallest semantically meaningful unit in a language. There are two kinds of morphemes: *Bound Morpheme* and *Free Morpheme*

Bound Morpheme: It can not exist independently but adds meaning when associated with other words. Examples: '-tion','un-','-ing','-ible', etc.

Free Morpheme: It can exist independently and has meaning. Examples: 'play', 'forget', etc. It is also referred to as UnBound Morpheme.

The way morphemes combine to form new words determine the vocabulary of the language. There are two main branches of morphology: *Inflectional morphology* and *Derivational morphology*.

2.1.1 Inflectional Morphology

The process of bound grammatical morphemes combining with existing words to form new words is called inflectional morphology. In inflectional morphology, various inflectional forms of existing words results in new words. Inflectional morpheme inflect the form of words in order to express grammatical features, such as singular/plural or past/present tense. Thus *Play* and *Plays*, for example, are two different forms of the same word - *Play*. Inflectional morphology do not change the lexical category (part of speech) of the new derived word.

In Hindi, the word *ladZakA* inflects to form *ladZakA*, *ladZake* and *ladZakOM*. Refer table-2.1.

Inflectional Morphology	Derivational Morphology
Lexical category of the doesn't change	Lexical category of the word changes
Examples: Play(Verb) , Played(Verb) , Playing(Verb)	Play(Verb), Player(Noun)
Examples (Hindi): ladZakA, ladZake, ladZakOM	cuna, cunAva

Table 2.1 Types of Morphology and Differences

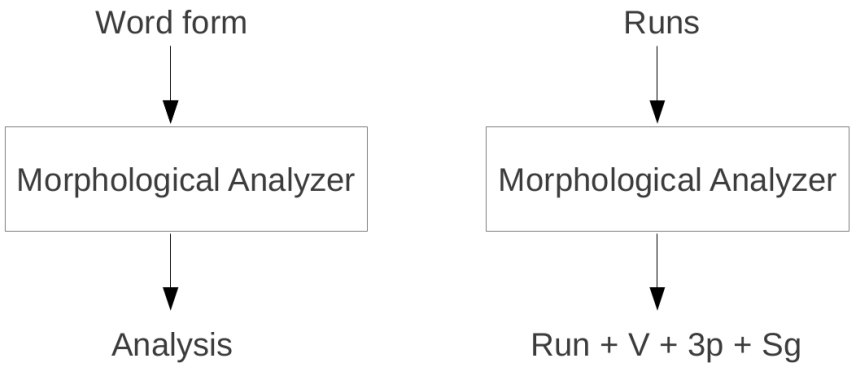

Figure 2.1 Morphological Analyzer

2.1.2 Derivational Morphology

Derivational morphology often involves the addition of a derivational affix. Such an affix usually applies to words of one lexical category and changes them into words of another category. For example, the English derivational suffix *-ly* changes adjectives into adverbs.

Derivational morphology produces a new word whereas inflectional morphology produces grammatical variants of the same word. Refer table-2.1

2.2 Morphological Analyzer

Morphological analysis is generally a prelude to further complex NLP tasks such as parsing, machine translation, semantic analysis, etc. These tasks needs an analysis of the words in the sentence in terms of lemma, affixes, parts of speech (POS), etc. The tool which does morphological analysis is called *morphological analyzer*.

Morphological analysis in English is relatively simple when compared to Indian languages. Hindi, being a morphologically rich language with a relatively free word order, need analysis of the words in

Input Word	Lemma ↓	Gender ↓	Number ↓	Person ↓	Case ↓	TAM/Vibhakti ↓
ladZakA (*boy*)	ladZakA	masculine	singular	third	direct	0
ladZake (*boy*)	ladZakA	masculine	singular	third	oblique	0
	ladZake	masculine	plural	third	direct	0
ladZakOM (*boy*)	ladZakA	masculine	plural	third	oblique	0
xeSa (*country*)	xeSa	masculine	singular	third	direct	0
	xeSa	masculine	plural	third	direct	0
	xeSa	masculine	singular	third	oblique	0
cAhie (*want*)	cAha	any	singular	second-honorific	_	ie
	cAha	any	plural	second-honorific	_	eM

Table 2.2 Examples: Morphological Analysis for Hindi words

Input Word	**Morphological Analysis**
run	run+Verb+Singular
runs	run+Verb+ThirdPerson+Singular
dog	dog+Noun+Singular
dogs	dog+Noun+Plural

Table 2.3 Examples: Morphological Analysis for English words

terms of lemma, affixes, POS, gender, number, person, case, vibhakti and TAM. *Hindi Morphological Analyzer* analyses a given word in terms of the above mentioned attributes.

Refer Table 2.2 to look at the sample morphological analyzer ouput for given Hindi words and Table 2.3 for English words.

2.3 Morphological Generator

For a given root word and its morphological feature values, the morphological generator generates the word form. Morphological generator is the inverse of morphological analyzer. Refer figure 2.2

Table 2.4 shows how the output of a morphological generator for sample inputs.

Morphological analysis is a non-deterministic process since more than one analysis is possible. Morphological generation is a deterministic process.

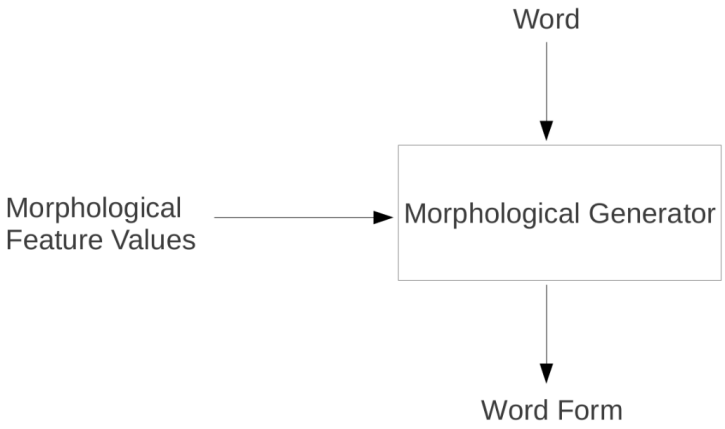

Figure 2.2 Morphological Generator

Word (Input)	Morphological Feature Values	Word Form (Output)
ladZakA	POS=n, Gender=m, Number=sg, case=d	ladZakA
ladZakA	POS=n, Gender=m, Number=sg, case=o	ladZake
ladZakA	POS=n, Gender=m, Number=pl, case=d	ladZake
ladZakA	POS=n, Gender=m, Number=pl, case=o	ladZakOM

Table 2.4 Input and Output of Morphological Generator

Word	Sentence	Analysis
ladZake	ladZake ne khaana khaaya	ladaZkA+m+sg+3+o
ladZake	ladZake bETe hEM	ladZakA+m+pl+3+d

Table 2.5 Examples: Context Based Morphological Analysis

2.4 Context Based Morphological Analyzer

As stated before, a word can have multiple analyses, but in a sentential context only one of them is relevant. For example, from table 2.2, *ladZake* has two analyses. But in table 2.5, in the sentential context the word *ladZake* picks only one analysis. We refer to the morphological analyzer which picks only one analysis in the given sentential context as context based morphological analyzer. It can also be viewed as a morphological analyzer with a morphological disambiguation model on top of the analyzer.

2.5 Applications

Morphological Analysis is a fundamental operation in many NLP applications. [1] and [2] have shown that morphological analysis improves parsing accuracy. In machine translation, morphology helps in predicting better target language sentences ([32], [48], [10]). Morphological analysis also finds its use in Semantic Analysis, Named Entity Recognizer, Word Sense Disambiguation, Spell Checkers, Annotation of corpora, etc.

Having multiple analyses for a word do not address the problem instead we need a single analysis, in other words, we need a context based morphological analyzer. In this thesis, we shall discuss more about *Context Based Morphological Analyzer*.

Chapter 3

Literature Study on Existing Morphological Analyzers

Different approaches have been adopted for developing morphological analyzers for various languages. Languages vary considerably in morphological complexity. English, for example, has a simple morphology compared with languages such as Arabic and Hebrew. European languages involve more complex morphologies than does English [46]. For example, Slovenian is similar to English, concatenative in its nature, but with a more complex morphology [40]. The affixation process in English is simpler compared to other languages such as Arabic.

Verbs are commonly marked with the indications of the time at which the situation described by the sentence occurred, or the state of completion of the situation. The former is called *tense* marking and the latter *aspect* marking. Capturing this information also comes as part of morphological analysis. Some languages mark verbs for both tense and aspect, others mark for one or the other. Languages such as Hindi and Latin, mark for both [46]. Mandarin has no tense marking but has a rich aspectual marking system [33]. Hence predicting morph tags for languages such as Hindi is complex.

Earlier, *rule* based approaches were employed to build the morphological analyzers. Later, efforts were put in developing *statistical* based morphological analyzers.

Rule based morphological analyzer often uses suffix lists to obtain the correct suffix partition. We then select the combination which has maximum number of suffixes or the suffix which has more priority, etc (depends on the kind of rules that are designed for the language). Each suffix is assigned an appropriate grammatical feature which decides the morphological analysis.

Several rule based morphological analyzers have been built for diverse languages. The rules are often encoded as finite state transducers (FST) [26, 29]. Morphological analyzers for different Indian languages were developed based on the paradigm model by Akshar Bharati et al. [4]. For Indian languages such as Sanskrit, Hindi, Telugu, Tamil and Malayalam numerous efforts (especially rule based approach) were put in by various researchers [6, 9, 24, 49].

In rule based morphological analyzer, the accuracy depends on the number of correct entries in the suffix list. That is if all possible suffixes are present in suffix lists, the correct splitting of the words happens.

Statistical based morphological analyzers can be either *supervised* or *unsupervised*. One common approach to unsupervised morphological analyzer is to identify morpheme boundaries and then use the segmented words to extract a list of morphemes. Zellig Harris [21] algorithm is based on the number of different letters which follow a given sequence of letters. The increase of this number indicates a morpheme boundary. For instance, after the English sequence `direc`, we only only find the letter `t` in our corpus. After `direct`, we find four letters: `i`, `l`, `o` and `e` (`directly`, `director`, `directed` and `direction`). This increase indicates a boundary between the root (`direct`) and the suffixes (`-ion`, `-ly`, `-or` and `-ed`). The main problem with Harris's algorithm is that it predicts lot of incorrect morpheme boundaries. The algorithm proposed by Harris was further extended by Hafer and Weiss [18].

Goldsmith has done extensive research on unsupervised learning of morphology and is based on minimum description length which attempts to identify morphemes and analysis for each word [15]. [41] presents an unsupervised learning approach to building a Arabic stemmer which is based on statistical machine translation and parallel corpora as its resource.

Morfette [13] is a data driven supervised system which performs lemmatization and predicts other morph information from morphologically annotated corpora. Morfette has two models built using Maximum Entropy classifier, one to predict the lemma and another to predict morphological tags. Morfette was evaluated on three morphologically rich languages viz. Romanian, Spanish and Polish.

Morphological disambiguation is the task of selecting morphological tag sequence for a given word in a given context. Yuret et al. [50] learns morphological disambiguation rules for Turkish. Habash et al. [17] does morphological disambiguation for Arabic. [38] uses context based Arabic morphological analysis for machine translation. Smith et al. [45] does morphological disambiguation with Conditional Random Fields. They evaluated their model for Korean and Arabic languages. All of these approaches do not provide analyses for a word, instead they are limited to performing morphological disambiguation.

3.1 Morphological Analyzers for Hindi

Previous efforts on Hindi morphological analysis primarily concentrated on building rule based systems that give all the possible analyses for a word form. There has been no research, to the best of our knowledge, to predict context based morphological analysis or to do morphological disambiguation for Hindi.

3.1.1 Paradigm based analyzer (PBA)

Paradigm based analyzer built by Akshar Bharati uses paradigms for analysis. A linguistic paradigm is the complete set of related word forms associated with a given lexeme. In paradigm based analysis, words are grouped into a set of paradigms based on the inflections they take. Each paradigm has a set of

	Case	
Number	direct	oblique
Singular	ladakA	ladake
Plural	ladake	ladakOM

Table 3.1 Word forms for the root word ladakA

	Case	
Number	direct	oblique
Singular	(0,ϕ)	(1,ϵ)
Plural	(1,ϵ)	(1,OM)

Table 3.2 Paradigm table for ladakA class

add-delete rules to account for its inflections. Words belonging to a paradigm take the same inflectional forms.

Table 3.1 represents the various word forms that the word ladakA takes and table 3.2 represents the add-delete rules of the paradigm class ladakA. Each entry in the table 3.2 shows the number of characters to be deleted from the root and the string to be suffixed.

Given a word, PBA identifies the lemma, coarse POS tag, gender, number, person, case marker, vibhakti (Vibhakti is a Sanskrit grammatical term that encompasses post-positionals and case endings for nouns, as well as inflection and auxiliaries for verbs [39]) and TAM (tense, aspect, modality). Being a rule-based system, the PBA takes a word as input and gives all the possible analyses as output. Table 3.3 presents an example of the analyses given by the PBA.

As with traditional morphological analyzers, PBA does not take the context into consideration and outputs all the possible analyses. Also for *Out Of Vocabulary* (OOV) words it does not give any analysis (or by default gives same analyses for all OOV words).

	L ↓	G ↓	N ↓	P ↓	C ↓	T/V ↓
xeSa (*country*)	xeSa	m	sg	3	d	0
	xeSa	m	pl	3	d	0
	xeSa	m	sg	3	o	0
cAhie (*want*)	cAha	any	sg	2h	_	ie
	cAha	any	pl	2h	_	eM
L-lemma, G-gender, N-number, P-person C-case, T/V-TAM or Vibhakti						

Table 3.3 Multiple analyses given by the PBA for the words xeSa and cAhie

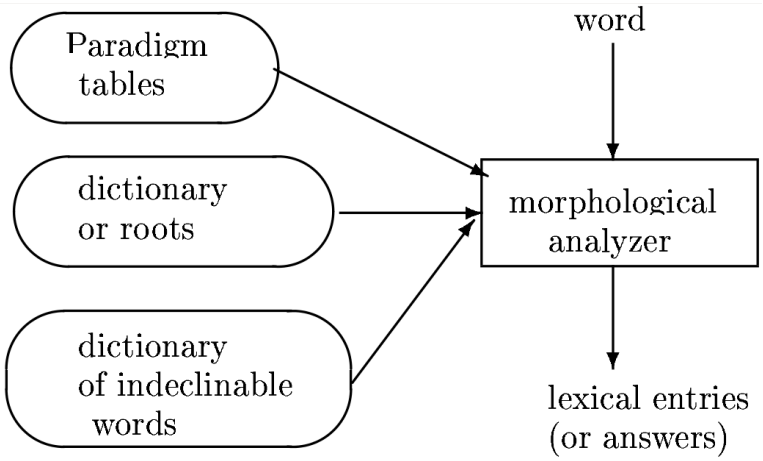

Figure 3.1 PBA Input Ouput , Image Reference: [4]

3.1.2 Goyal and Lehal's Morph Analyzer

Goyal and Lehal's analyzer is a re-implementation of the PBA with few extensions apart from the way they have stored the data. They have not done any comparative evaluation and showed how their system is better than the PBA.

It has similar drawbacks to that of the PBA and this isn't any better approach than the PBA. Also this is a GUI based tool and cannot be used directly for real application purposes.

3.1.3 Derivational Morphological Analyzer (DMA)

Nikhil et al. [25] built a derivational morphological anlayzer for Hindi by introducing a layer over the PBA. It identifies 22 derivational suffixes which helps in providing derivational analysis for the word whose suffix matches with one of these 22 suffixes. Check figure 3.2 for the algorithm employed by [25].

This particular morphological analyzer has an added advantage over the PBA as it identifies certain types of derivational analysis there by improving the coverage when compared to the PBA.

DMA is also a rule-based analyzer which doesn't cover the entire vocabulary. Analysis for the OOV words is still a problem. Also apart from these 22 derivational suffixes there can be other derivational suffixes too.

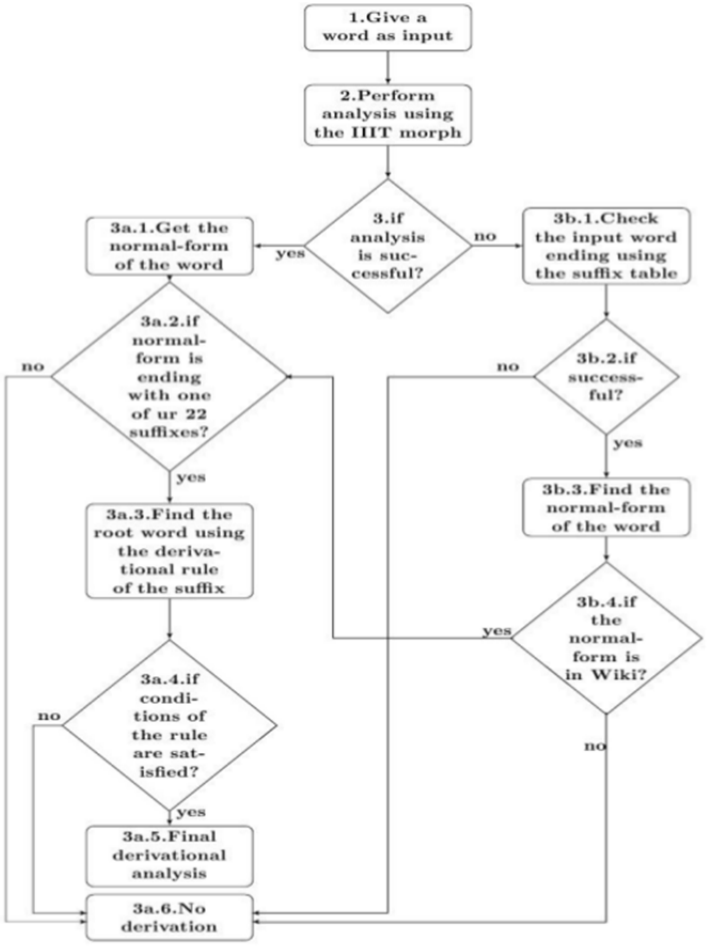

Figure 3.2 DMA Algorithm , Image Reference: [25]

Chapter 4

Hindi Treebank (HTB)

4.1 About Hindi Treebank

The goal of the Hindi Treebank (HTB) project is to build a multi-representational and multi-layered treebank for Hindi. HTB is annotated with morpho-syntactic information. Morpho-syntacto information comprises of morphological information, POS and chunk information. Morphological information constitutes of lemma, suffix, gender, number, person, case-marker, TAM and vibhakti. HTB is also annotated with dependency information. Dependency annotation is based on Paninian Grammar Framework [5, 8]. As part of future work, each verb in HTB will be annotated with predicate-argument structure (PropBank). Apart from having dependency representation, HTB will also have phrase structure representation.

This thesis work mainly deals with the morphological information.

4.2 HTB Data released for Hindi Shared Task on Parsing

MT and Parsing in Indian Languages (MTPIL), is a workshop co-located with COLING-2012 [44]. MTPIL hosted a dependency parsing shared task for Hindi. For the shared task, a part of HTB was released in two different settings. One containing gold standard morphological analyses, part-of-speech tags, chunks and dependency relations labeled in the computational paninian framework whereas the other one contains only automatic POS tags without any other information.

The data is released in both SSF [7] and CoNLL format.

4.3 Data Statistics

Refer table 4.1 for word counts of training, development and test sections of HTB data released for MTPIL shared task. Target of HTB is to annotate 400k words. In total, around 300K word data was released which is manually annotated and validated. There are around 15000 sentences in this data.

Data	#Sentences	#Words
Training	12,041	268,096
Development	1,233	26,416
Test	1,828	39,775
Total	**15,102**	**334,287**

Table 4.1 HTB statistics

Average sentence length is 22.14 words/sentence. For our experiments in this thesis, we shall use this data which forms the crux of our work.

Chapter 5

Hindi Parsing Shared Task

This section deals with the initial work which forms the basis/motivation for the main contribution of this thesis. The experiments and results mentioned in this section, provided the direction for this thesis.

5.1 Coling MTPIL shared task

Machine Translation and Parsing for Indian Languages (MTPIL)[44] workshop was hosted with the 24th International Conference on Computational Linguistics (COLING-2012). The MTPIL workshop also hosted a dependency parsing shared task for Hindi. As part of the shared task, part of the Hindi Treebank(HTB) containing gold standard morphological analyses, part-of-speech tags, chunks and dependency relations labeled in the computational Paninian framework was also released.

Below we discuss about our experiments on Hindi dependency parsing in brief. For this shared task on Hindi dependency parsing, we explored a new parser: Turbo parser [37]. We have also explored MST parser and Malt parser with previous best configuration for Hindi. We tried the combination of various parsing systems using two different methods i.e., simple voting [51] and blending method [42].

5.1.1 Data

The training and development datasets were released by the organizers as part of the shared task in two different settings. One being the manually annotated data with POS tags, Chunks and other information such as gender, number, person etc. whereas the other one contains only automatic POS tags without any other information. Training set contains 12,041 sentences (2,68,093 words) and development dataset contains 1233 sentences (26,416 words). The test dataset contains 1828 sentences (39,775 words). The size of the data set is significantly higher when compared to the previous shared tasks on Hindi Dependency Parsing [22, 23].

We participated in MTPIL shared task [30] and while performing the parsing experiments as part of shared task we were motivated to build a context based morphological analyzer which forms the main

contribution of this thesis. In further sections, we shall discuss in brief about the experiments we have performed as part of the Hindi parsing shared task and the outcome of it.

5.2 Experiments

We experiment with various data-driven parsers for Hindi language. We tried different types of features and selected the best feature set by tuning it on the development dataset.

5.2.1 Malt Parser

Malt parser is a transition based dependency parser.

Malt parser provides two learning algorithms LIBSVM and LIBLINEAR. It also gives various options for parsing algorithms and we have experimented on nivre-eager, nivre-standard and stack- proj parsing algorithms. Finally, we chose nivre-standard parsing algorithm and LIBSVM learning algorithm options by tuning it on the development data set. A 5 fold cross validation has been done for selecting the best template, best algorithm and best classifier for both sets of data, we experimented on training and development sets to get the best templates, algorithms and classifiers.

5.2.2 MST Parser

MST parser is a graph based dependency parser.

MSTParser is a non-projective dependency parser that searches for maximum spanning trees over directed graphs. Models of dependency structure are based on large-margin discriminative training methods. Projective parsing is also supported. The main features of this parser are:

- First and second order projective and non projective parsing.
- Perceptron and k-best MIRA training.

The setting that performed best for MST parser is second order non-projective with beam width (k-best parses) of 5 and default iterations of 10. The tuning of MST parser in second order non- projective is hard since it is computationally intensive.

5.2.3 Turbo Parser

To the best of our knowledge, the process of adopting Turbo parser for Indian languages has not been explored previously.

Turbo parser provides three settings for training: basic, standard and full. We experimented with all three models for both data settings and consistently the Turbo parser in full mode performs the better and it trains a second-order non-projective parser. For the two different data settings, we used the same

feature set used in [37]. For the gold-standard POS data setting, we added chunk based features as previously mentioned. We also experimented with the clause based features but we didnt get much performance gain using them.

5.2.4 Voting

In case of voting, once the outputs from all the parsing systems are obtained, each dependency relation that has the maximum number of votes from the various systems are included in the output. In case of a tie, the dependency relation predicted by the high accurate parser is picked in the final output.

5.2.5 Blending

The one drawback that is inherent in the voting method is that the dependency tree resulted for each sentence may not be fully connected as we are including each dependency relation at a time rather including the whole dependency tree. In order to mitigate this drawback, the concept of blending has been introduced by [42] and the software has been released as MaltBlender[19]. In this approach, a graph is built for the dependency relations obtained from the various outputs and then it selects a maximum spanning tree out of it. The tool also provides various options for selecting the weights for different parsers and these weights are determined by tuning on the development dataset.

In this work, we used Malt, MST and Turbo parsers for the parser combination using the above mentioned two approaches viz. Voting and Blending.

5.3 Results

5.3.1 Feature set

The number of features a dependency parser uses are typically huge. Selection of features has a lot of impact on both the run-time complexity and also on the performance of a parser. We tried various uni-gram and bi-gram features related to words, lemmas, POS tags, Coarse POS tags, vibhakti (post-positional marker), TAM (tense, aspect and modality) and other available morphological information. In addition to these features, we also used chunk features like chunk head, chunk distance and chunk boundary information. Finally we also experimented with some clause-based features like head/child of a clause, clausal boundary information. Turbo parser [37] uses the concept of supported and unsupported features to mitigate the effect of having large number of features to some extent.

5.3.2 Evaluation Results

Table 5.1 lists the accuracy with gold POS tags setting of the data using various parsers. It also lists the accuracy obtained when the parsers are combined using simple voting method and an intelligent

blending method. It can be observed from the table that Turbo parser performs well in all the evaluation metrics in terms of using a single parser whereas voting method performs well overall. The results submitted for this data setting using Turbo parser was ranked second in terms of LAS, LS but first in UAS. If we take the voting results, then it will be ranked first in terms of all the evaluation metrics when compared to the other systems submitted for this shared task.

Table 5.2 gives the accuracy for the test data with automatic POS tags. The results on this data setting submitted using Turbo parser is significantly higher among all the methods we tried and all the other systems that submitted test results. The voting and blending systems didn't get the increase in accuracy than Turbo Parser because of the lower accuracies from Malt and MST. It can be inferred from the results that the voting and blending systems benefit if the accuracies produced by the parsers are comparable to each other. On the other hand if such a difference is very high then it will hurt their performance.

With Gold Part-Of-Speech Tags			
Method	*UAS*	*LA*	*LAS*
Malt	93.32%	90.56%	88.86%
MST	94.88%	88.13%	86.45%
Turbo	96.37%	92.14%	90.83%
Voting	**96.50**%	**92.90**%	**91.49**%
Blending	96.34%	92.83%	91.49%

Table 5.1 UAS, LA, LAS denote the Unlabeled Attachment score, Labeled Accuracy Score and Labeled Attachment Score respectively. Accuracies on data annotated with gold standard POS tags

With Automatic Part-Of-Speech Tags			
Method	*UAS*	*LA*	*LAS*
Malt	81.23%	76.76%	73.69%
MST	91.02%	84.76%	82.55%
Turbo	**93.99**%	**90.04**%	**87.84**%
Voting	93.24%	89.01%	86.62%
Blending	93.47%	89.15%	87.05 %

Table 5.2 Accuracies on data annotated with automatic POS tags

5.4 Error Analysis & Conclusion

It has been observed that the accuracy of predicting NULL or Empty categories is very low. [31] addressed this issue to better predict NULL or Empty categories there by Hindi parsing accuracy can be improved.

Table 5.3 lists the accuracies reported by the teams participated in the Shared Task. We can notice that the accuracies reported over gold data is higher than that of the accuracies reported over auto data. Though we report parsing accuracy of 96% for UAS and 92% for LA, in reality we cannot achieve these

TeamID	Auto			Gold		
	LAS	UAS	LA	LAS	UAS	LA
cuni	81.48%	89.96%	84.09%	89.48%	95.19%	91.27%
DFKI	82.44%	90.91%	85.22%	85.31%	92.88%	87.29%
iiitCL	83.91%	91.70%	86.77%	**90.99%**	95.87%	**92.58%**
iitkgp	-	-	-	89.36%	95.08%	91.19%
isi_cvpr_nlp	32.34%	38.25%	32.93%	86.51%	91.80%	88.47%
JNTUH	80.77%	88.91%	83.03%	90.66%	95.18%	92.28%
ltrciiit	**87.84%**	**93.99%**	**90.04%**	90.83%	**96.37%**	92.14%

Table 5.3 Accuracies reported by all teams participated in Shared Task

unless we have gold data. In real applications, we never have the access to gold annotated data (which includes context based morphological inforamtion, chunk information, etc). We need automatic context based morph information and chunk information in real applications.

[1], [2] have documented the effect of morphological features on parsing. The above parsing experiments also confirms the same that context based morphological information increases the parsing accuracy. These experiments gave me the motivation to build a `context based morphological analyzer`. Also with the existing morphological analyzers not giving analyses for `OOV words`, we choose to build one such framework which addresses these two issues.

Chapter 6

Statistical Morphological Analyzer (SMA)

As discussed in previous sections, there is a need for context based morphological analyzers. The existing morph analyzers for Hindi do not take the sentential context into consideration and also perform poorly on OOV words. With the availability of an annotated treebank, we set out to build a high-coverage automatic Hindi morph analyzer by learning each of the morphological attributes separately from the Hindi Treebank. During this process, it was realized that vibhakti and TAM can be better predicted using heuristics on fine-grained POS tags than by training on the HTB.

In the rest of the section, we discuss the methods to predict each of the seven morphological attributes. Table 6.1 lists the values that each of the morph attributes take in HTB. The HTB consists of 15,102 sentences (334,287 words) annotated with morphological features, POS tags, chunks and dependency relations. In this work, we only use morph and POS information.

6.1 Lemma prediction

The PBA uses a large vocabulary along with paradigm tables consisting of add-delete rules to find the lemma of a given word. All possible add-delete rules are applied on a given word form and the resulting lemma is checked against the vocabulary to find if it is right or not. If no such lemma exists (for OOV words), it returns the word itself as the lemma.

While the gender, number and person of a word form varies according to the context (due to syntactic agreement with head words), there are very few cases where a word form can have more than one lemma

MorphFeature	Values
Gender	`masculine`, `feminine`, `any`, `none`
Number	`singular`, `plural`, `any`, `none`
Person	`1`, `1h`, `2`, `2h`, `3`, `3h`, `any`, `none`
CaseMarker	`direct`, `oblique`, `any`, `none`

Table 6.1 Morph features and the values they take

in a context. This makes lemma simpler to predict among the morphological features, provided there is access to a dictionary of all the word forms along with their lemmas. Unfortunately, such a large lemma dictionary doesn't exist.

6.1.1 Lemma prediction perceived as a *Machine Translation Problem*

In this work, we perceive lemma prediction from a machine translation perspective.

Machine translation is a complex NLP task and even humans require special training to translate between languages. Earlier the machine translation systems were built by analyzing the language and then framing a set of rules. These rules are encoded into a computer program. Rule based systems have its own limitations. As the language is rich and complex, rules alone could never capture the entire semantics of the language. Instead of these rule based approaches, efforts were put in to develop a model that discovers the rules of translation automatically from a large corpus of translated text (parallel corpus). This approach is commonly know as 'Statistical Machine Translation' (SMT) [27]. To perform SMT, *Moses* offers several translation models such as word-based translation models, phrase-based translation models, tree-based translation models, etc.

Word-based translation: In word-based translation, the fundamental unit of translation is a word. Figure 6.1 demonstrates sample example of how word-based translation model works.

Phrase-based translation: Phrase-based translation models translate phrases as atomic units. It has advantages over word-based translation. This is a standard model used by Google Translate and others. In figure 6.2, the source sentence is segmented into phrases (not necessarily linguistically motivated), which are translated one-to-one into phrases in target sentence and possibly reordered.

We perform a series of experiments related to lemma prediction with little variations from one to another but the underlying concept remains the same in all of these experiments.

6.1.1.1 Experiment-1

Given an input word form, we need to predict the lemma. We achieve this by building a machine translation model and then using this translation model we run it on the given word and get the lemma.

We use the HTB data released for MTPIL Shared Task to build the translation model. The characters in the input word form (token) treated as the source sentence and those in the lemma as the target sentence. The strings on source and target side are split into *sequences of characters* separated by space (*character-level translation*), as shown in Table 6.2.

Also, refer to figure 6.3 to know about character-level phrase-based translation. It contains examples to understand word-lemma prediction using Machine Translation.

For our experiments we use phrase-based translation. The phrase based model [28] in Moses is trained on the parallel data created from the training part of HTB (as mentioned above, we employ the technique 'character-level translation'). The translation model accounts for the changes in the affixes (sequence of characters) from word form to lemma whereas the language model accounts for which

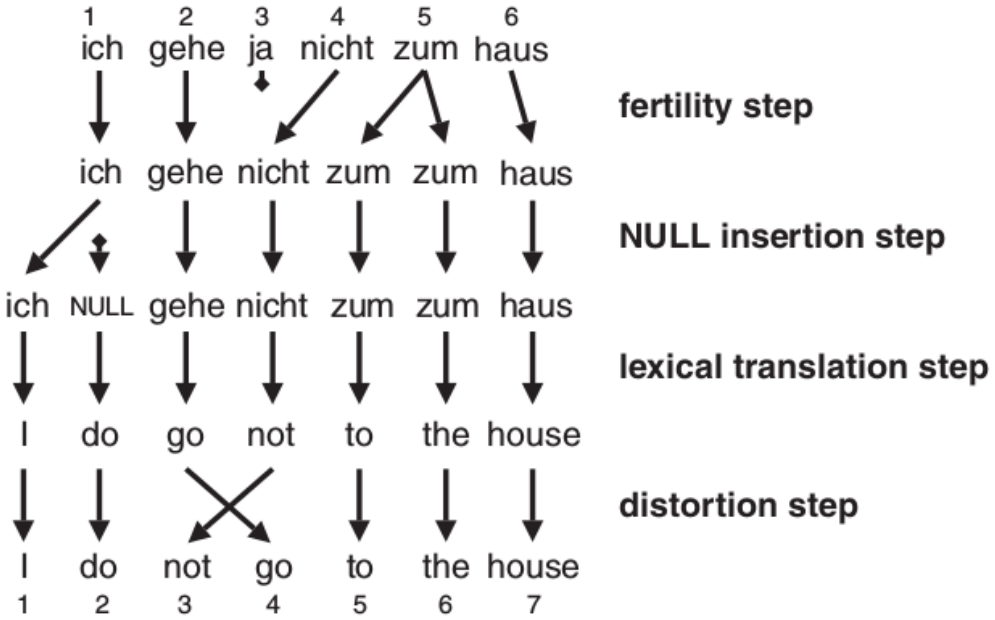

Figure 6.1 Word based translation, Image Reference: [27]

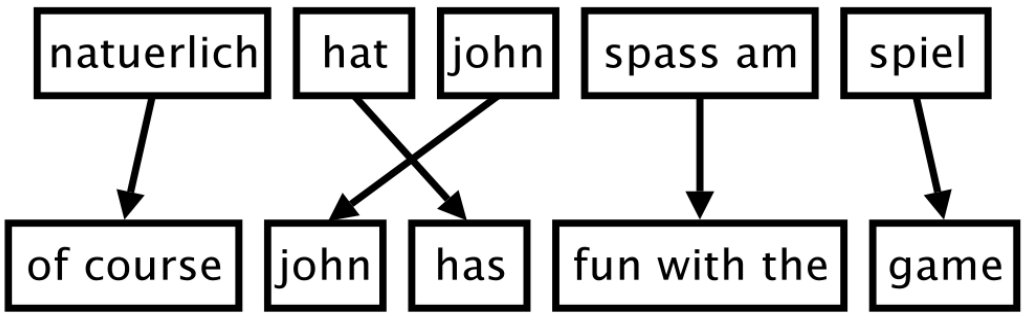

Figure 6.2 Phrase based translation, Image Reference: [27]

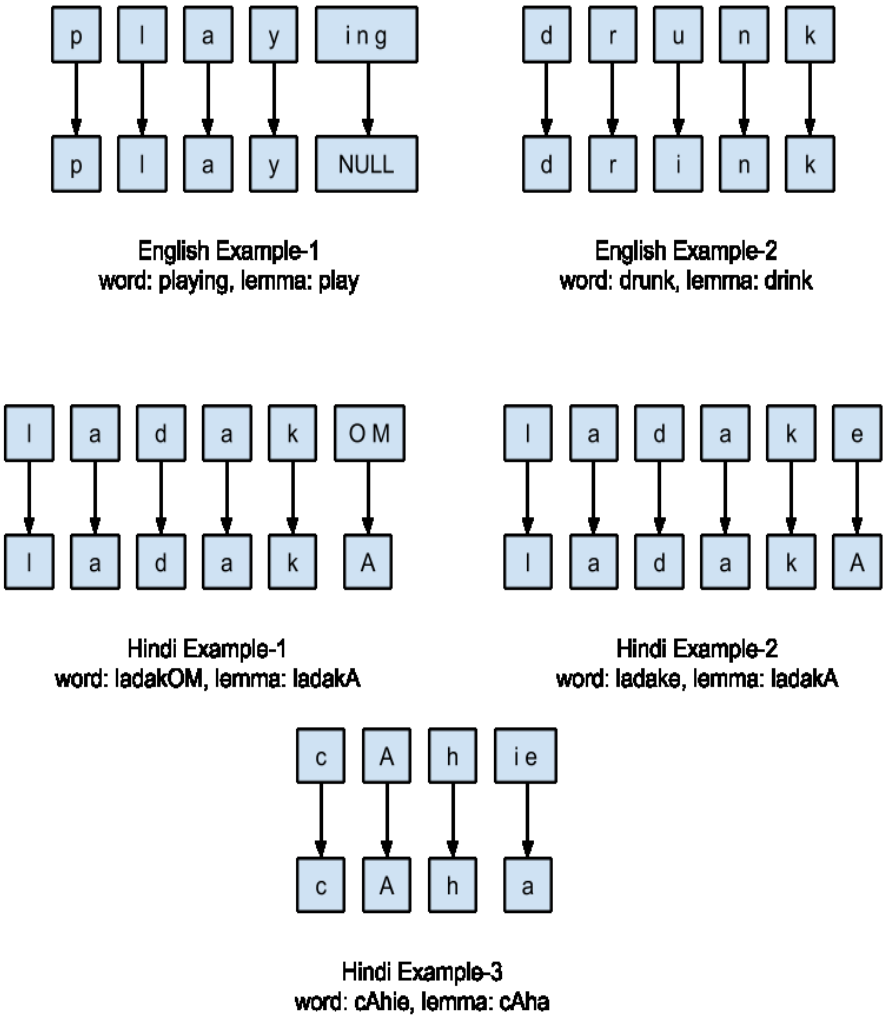

Figure 6.3 Character-Level Phrase-based Translation for Lemma Prediction

Experiment-1		
source	target	gloss
k i y A	k a r a	*do*
l a d a k e	l a d a k A	*boy*
l a d a k I	l a d a k I	*girl*
l a d a k I y A M	l a d a k I	*girl*

Table 6.2 Experiment-1: Sample parallel corpus for lemma prediction

source	target	gloss
VM k i y A	k a r a	*do*
NN l a d a k e	l a d a k A	*boy*
NN l a d a k I	l a d a k I	*girl*
NN l a d a k I y A M	l a d a k I	*girl*

Table 6.3 Experiment-2: Sample parallel corpus for lemma prediction with POS as prefix on source side

affixes go with which stems. In this perspective, the standard MT experiment of switching source and target to attain better accuracy would not apply since it is unreasonable to predict the word form from the lemma without taking the context into account.

Apart from the above mentioned approach, we apply a heuristic on top of SMA, wherein proper nouns (NNP) take the word form itself as the lemma.

In Experiment-1, we did not include POS as a feature in building the translation model instead it was used as a simple rule on top of output derived from the model. We perform a series of experiments similar to Experiment-1 in which we include POS tag in the machine translation training data in different formats.

source	target	gloss
DEM i s a	DEM y a h a	*do*
PSP k e	PSP k A	*boy*
VM k a r a n e	VM k a r a	*girl*
NN l a d a k I y A M	NN l a d a k I	*girl*

Table 6.4 Experiment-3: Sample parallel corpus for lemma prediction with POS as prefix on both source and target side

source	target	gloss
i s a DEM	y a h a	*do*
k e PSP	k A	*boy*
k a r a n e VM	k a r a	*girl*
l a d a k I y A M NN	l a d a k I	*girl*

Table 6.5 Experiment-4: Sample parallel corpus for lemma prediction with POS as suffix on source side

source	target	gloss
i s a DEM	y a h a DEM	*do*
k e PSP	k A PSP	*boy*
k a r a n e VM	k a r a VM	*girl*
l a d a k I y A M NN	l a d a k I NN	*girl*

Table 6.6 Experiment-5: Sample parallel corpus for lemma prediction with POS as suffix on both source and target side

6.1.1.2 Experiment-2

We perform the character-level translation similar to the one employed in Experiment-1 but in addition to that we include POS tag as prefix only on the source side. Refer table 6.3

6.1.1.3 Experiment-3

In Experiment-2, we included POS tag as prefix only the on the source side. In this experiment, we include POS tag as prefix on both source and target side. Refer table 6.4

6.1.1.4 Experiment-4

Instead of including POS tag as prefix, we include it as suffix on the source side. Refer table 6.5 for the sample training data.

6.1.1.5 Experiment-5

In experiment-4, we included POS tag as suffix on the source side. In this experiment, we include POS tag as suffix on both source and target side. Refer table 6.6

We have used only automated POS tags (not gold POS tags) for the above experiments.

Experiment Type	Overall Accuracy(%)	OOV Accuracy(%)
Experiment-1	94.14	89.51
Experiment-2	96.37	88.46
Experiment-3	96.74	92.02
Experiment-4	96.57	93.07
Experiment-5	96.57	93.07

Table 6.7 Accuracies for Lemma Experiments

Experiment Configuration

For all the above experiments, we use phrase based model [28] in Moses to build the training model. The configuration related to Moses is mentioned below.

- **Translation Model**: Phrase-based model
- **Language Model Order**: 6. Maximum order of language models used in translation is 6 ie. word combinations upto 6 words are included while calculating language models.
- **Phrase Pair Maximum Length**: 8. Phrase pairs generated at the end of training can have a maximum word count of 8 on source or target side.
- **Tuning**: MIRA

Results

Translation models are built for each of the above experiment setup mentioned using the training part of the HTB (released for MTPIL Shared Task). Table 6.7 reports the accuracies of the different experiments performed for lemma prediction on the *test* part of HTB. Out-Of-Vocabulary (OOV) words are the ones which are present in *test* part of HTB and are not present in *train* and *development* parts of HTB. We shall compare the lemma accuracy with other existing systems in *Experiments and Results* section.

Other experiments were tried out such as having the previous words and next words as the features, but they didn't help in improving the lemma accuracy.

6.2 Gender, Number, Person and Case (GNPC) Prediction

In this thesis, we treat GNPC prediction as a classification problem. We use a liblinear classifier [14] to build linear SVM classification models for GNPC prediction.

Gender	Word	Gloss
masculine	cAvala, paMKA	*rice*, *fan*
feminine	rela, xAla	*train*, *pulse*
any	jA	*go*
none	karIba	*near*

Table 6.8 Gender value examples

Though knowing the syntactic head of a word helps in enforcing agreement (and thereby accurately predicting the correct GNP), parsing is usually a higher level task and is not performed before morphological analysis. Hence, certain cases of GNP prediction are similar in nature to the standard chicken and egg problem.

6.2.1 Gender

Hindi has a complex gender system when compared to English. Gender in Hindi is a grammatical feature whereas in English gender is reflected only in pronoun. In Hindi, verbs agree with nouns and it takes noun's gender. The values that gender can take for a word in a given context are *masculine*(*m*) and *feminine*(*f*). Some words can take either *masculine* and *feminine*, instead of representing it with two values, we represent it as *any*. In Hindi morphological analyzers, gender does not apply to certain values. For such cases we represent gender value as *none*. Table 6.8 gives example for each gender value taken from HTB.

In Hindi, Nouns inherently carry gender information. Pronouns (of genitive form), adjectives and verbs inflect according to the gender of the noun they refer to.

6.2.2 Number

Every noun belongs to a unique number class. Noun modifiers and verbs have different forms for each number class and inflect accordingly to match the grammatical number of the nouns to which they refer.

Number takes the values *singular* (*sg*) or *plural* (*pl*). Hindi morphological analyzer also takes the values *any* and *none*. If the *number* is not applicable then we shall assign the word with *none* as the morph value. Table 6.9 lists examples for each of the values from HTB. In it, ladZake takes the grammatical number *sg* (in *direct* case) or *pl* (in *oblique* case) depending on the context in which it occurs. It may be noted that since PBA does not consider the word's context, it outputs both the values and leaves the disambiguation to the subsequent stages.

Number	Word	Gloss
singular	ladZake	*boy-Sg-Oblique*
plural	ladZake	*boy-Pl-Direct*
any	banA	*make*
none	karIba	*near*

Table 6.9 Number value examples

Case	Word	Gloss
direct	ladZake	*boy-Pl*
oblique	ladZake	*boy-sg*
any	bAraha	*twelve (cardinals)*
none	kaha	*say*

Table 6.10 Case value examples

6.2.3 Person

Apart from *first*, *second* and *third* persons, Hindi also has the honorific forms, resulting in *1h*, *2h* and *3h*. Postpositions do not have person information, hence *none* is also a possible value. Apart from the above mentioned grammatical person values, *any* is also a feasible value.

6.2.4 Case Marker

Case markers in Hindi (*direct* and *oblique*) are attributed to nouns and pronouns. Table 6.10 lists few examples.

Words which inflect for gender, number, person and case primarily undergo affixation at the end.

Features for GNP & Case Marker

The following features were tried out in building the models for gender, number, person and case prediction:

- Word level features
  - Word
  - Last 2 characters
  - Last 3 characters
  - Last 4 characters

Morph Type	Overall Accuracy(%)	OOV Accuracy(%)
Gender	96.19	82.65
Number	95.37	90.44
Person	96.38	94.85
Case	95.32	88.52

Table 6.11 Accuracies for Gender, Number, Person, Case Prediction

  - Character N-grams of the word
  - Lemma
  - Word Length

- Sentence level features
  - Lexical category[1]
  - Next word
  - Previous word

Combinations of these features have been tried out to build the SVM models for GNP and case. For each of these tasks, feature tuning was done separately. In [35], a linear SVM classification [14] is used to build statistical models for GNP and case but we found that with RBF kernel (non-linear SVM)[2] we achieve better accuracies. Furthermore, the parameters (C, γ) of the RBF kernel are learned using grid search technique.

Results

Classification models are built on the training part of HTB (released for MTPIL Shared Task). Refer table 6.11 for the accuracies of Gender, Number, Person and Case reported by test part of HTB run on the classification models. We shall compare these values with the accuracies of other existing systems in *Experiments and Results* section.

6.3 Vibhakti and TAM

Vibhakti and TAM are helpful in identifying the *karaka*[3] dependency labels in HTB. While nouns and pronouns take vibhakti, verbs inflect for TAM. Both TAM and vibhakti occur immediately after the words in their respective word classes.

[1]POS is considered as a sentence level feature since tagging models use the word ngrams to predict the POS category
[2]LIBSVM tool is used to build non-linear SVM models for our experiments [11].
[3]karakas are syntactico-semantic relations which are employed in Paninian framework [3, 5]

Word/POS Tag Sequence	TAM/Vibhakti
$word_1$/NN	0 or _
$word_1$/NN $word_2$/PSP	0_lemma($word_2$)
$word_1$/NNP	_
$word_1$/NNP $word_2$/PSP	0_lemma($word_2$)
$word_1$/NNC	0 or _
$word_1$/NNPC	0 or _
$word_1$/CC	_
$word_1$/JJ	_
$word_1$/PRP	lemma($word_1$)
$word_1$/PSP	_
$word_1$/SYM	_
$word_1$/VAUX	lemma($word_1$)
$word_1$/VM	suffix($word_1$)
$word_1$/VM $word_2$/VAUX	suffix($word_1$)_lemma($word_2$)+suffix($word_2$)

Table 6.12 Sample TAM/Vibhakti Rules based on POS patterns

Instead of building statistical models for vibhakti and TAM prediction, we built a system that uses heuristics on POS tag sequences to predict the correct value. The POS tags of words following nouns, pronouns and verbs give an indication as to what the vibhakti/TAM are. Words with PSP (postposition) and NST (noun with spatial and temporal properties) tags are generally considered as the vibhakti for the preceding nouns and pronouns. A postposition in HTB is annotated as PSP only if it is written separately (usane/PRP vs usa/PRP ne/PSP). For cases where the postposition is not written separately we rely on the treebank data to get the suffix. Similarly, words with VAUX tag form the TAM for the immediately preceding verb.

The PBA takes individual words as input and hence does not output the entire vibhakti or TAM of the word in the sentence. It only identifies these values for those words which have the information within the word form (e.g. usakA *he+Oblique*, kiyA *do+PAST*).

The POS patterns mentioned in table 6.12 are the most frequent ones (occurrence >5000) observed in the HTB training data. The corresponding TAM/Vibhakti for the POS pattern is mentioned in the table 6.12. We apply these rules on the test data and report the accuracies.

In the sentence,

rAma/NNP kA/PSP kiwAba/NN cori/NN ho/VM sakawA/VAUX hE/VAUX,

PBA identifies rAma's vibhakti as *0* and ho's TAM as *0*. Whereas in HTB, vibhakti and TAM of rAma and ho are annotated as *0_kA* and *0_saka+wA_hE* respectively. This is because, TAM and vibhakti are features generally identified at chunk level and not at word level. Where as PBA, functions only at word level and so it needs to go through chunking stage to predict correct TAM and vibhakti values. Our approach determines this information precisely.

Data	#Sentences	#Words
Training	12,041	268,096
Development	1,233	26,416
Test	1,828	39,775

Table 6.13 HTB statistics

Further sections deals with results and comparing our system with the existing systems.

Analysis	Test Data - Overall(%)					Test Data - OOV of SMA(%)				
	Baseline	F-PBA	O-PBA	Morfette	SMA	Baseline	F-PBA	O-PBA	Morfette	SMA
L	71.12	83.10	86.69	94.14	95.84	78.10	82.08	82.48	90.30	89.51
G	37.43	72.98	79.59	95.05	96.19	60.22	43.07	44.06	72.03	82.65
N	52.87	72.22	80.50	94.09	95.37	69.60	44.53	47.56	84.89	90.44
P	45.59	74.33	84.13	94.88	96.38	78.30	52.51	53.89	84.76	94.85
C	29.31	58.24	81.20	93.91	95.32	43.60	31.40	47.36	80.21	88.52
V/T	65.40	53.05	59.65	NA	97.04	58.31	33.58	34.56	NA	96.04
L+C	16.46	48.84	72.06	88.56	91.39	32.52	28.50	44.66	72.89	79.09
L+V/T	54.78	44.57	51.71	NA	93.06	53.56	31.73	32.72	NA	86.41
G+N+P	23.05	61.10	73.81	88.36	91.11	47.49	35.75	39.58	62.33	76.52
G+N+P+C	9.72	45.73	70.87	84.43	87.78	21.04	20.91	35.95	55.74	69.99
L+G+N+P	20.27	53.29	66.28	83.44	87.51	44.72	34.63	38.46	57.85	69.13
L+G+N+P+C	8.57	38.25	63.41	79.73	**84.25**	19.33	19.92	34.89	51.52	**63.06**
L+G+N+P+C+V/T	1.25	32.53	42.80	NA	**82.12**	4.02	14.51	18.67	NA	**60.07**

L-lemma, G-gender, N-number, P-person, C-case, V/T-Vibhakti/TAM

Table 6.14 Accuracies of SMA compared with F-PBA, O-PBA and baseline systems.

6.4 Evaluation Systems

In the previous sections we have built statistical models for Lemma, Gender, Number, Person, Case prediction. Hence we named our analyzer as *Statistical Morphological Analyzer* (SMA) [36].

SMA is compared with a baseline system, Morfette [13] and two versions of the PBA wherever relevant. The *baseline* system takes the word form itself as the lemma and selects the most frequent value for the rest of the attributes.

Since PBA is a rule based analyzer which gives more than one analysis for words, we use two versions of it for comparison. The first system is the oracle PBA (referred further as O-PBA) which uses an oracle to pick the *best* analysis from the list of all analyses given by the PBA. The second version of the PBA (F-PBA) picks the *first* analysis from the output as the correct analysis.

Metric	Exp-1[a]	Exp-2[b]	Exp-3[c]
LAS	87.75	89.41	89.82
UAS	94.41	94.50	94.81
LA	89.89	91.67	91.96

Table 6.15 MALT Parser's accuracies on HTB test data. Unlabeled Attachment Score (UAS) is the percentage of words with correct heads. Labeled Accuracy (LA) is the percentage of words with correct dependency labels. Labeled Attachment Score (LAS) is the percentage of words with both correct heads and labels.

[a]Exp-1: Without morph features
[b]Exp-2: With morph features predicted by SMA
[c]Exp-3: With gold morph features (as annotated in HTB)

Morfette can predict lemma, gender, number, person and case attributes but it cannot predict TAM and Vibhakti as they do not have a definite set of predefined values unlike other morphological attributes.

6.5 Experiments, Results & Error Analysis

SMA approach to Hindi morphological analysis is based on handling each of the seven attributes (*lemma, gender, number, person, case, vibhakti* and *TAM*) separately. However, evaluation is performed on individual attributes as well as on the combined output.

SMA builds models for lemma, gender, number, person and case prediction trained on the training data of the HTB. All the models are tuned on development data and evaluated on test data of the HTB.

Table 6.14 presents the accuracies of five systems (baseline, F-PBA, O-PBA, Morfette and SMA) in predicting the morphological attributes of all the words in the HTB's test data and also for OOV words of SMA (i.e. words that occur in the test section but not in training section of HTB)[4]. The accuracies are the percentages of words in the data with the correct analysis. It may be noted that SMA performs significantly better than the best analyses of PBA and the baseline system in all the experiments conducted. As far as Morfette is concerned, it performs on par with SMA in terms of overall accuracy but for OOV words, except for lemma prediction, SMA outperforms Morfette by significant margin.

Table 6.19 lists the accuracies of lemma, gender, number, person and case for the most frequently occurring POS tags. Table 6.18 reports the same for OOV words. The number of OOV words in postposition and pronoun categories is quite less and hence have not been included in the table.

Hindi derivational morph analyzer [25] and the morph analyzer developed by Punjab University [16] do not add much to PBA accuracy since they are developed with PBA as the base. Out of 334,287 words in HTB, the derivational morph analyzer identified only 9,580 derivational variants. For the remaining words, it gives similar analysis as PBA.

[4]OOV words for SMA need not be *out of vocabulary* for PBA's dictionaries. Table 6.14 lists accuracies for OOV words of SMA. We also report accuracies for OOV words of PBA in the later part of the paper (Table 6.17).

6.5.1 Lemma

The evaluation metric for lemma's model is *accuracy*, which is the percentage of predicted lemmas that are correct. The phrase based translation system used to predict lemmas achieved an accuracy of 95.84% compared to O-PBA's 86.69%. For OOV words, the PBA outputs the word itself as the lemma whereas the translation-based lemma model is robust enough to give the analysis.

The translation-based lemma model and O-PBA report accuracies of 89.51% and 82.48% respectively for OOV words of SMA. In terms of both overall and OOV accuracies, translation-based model outperforms PBA. Though SMA performs better than Morfette in terms of overall accuracy, but for OOV accuracy Morfette narrowly outperforms SMA.

The postposition accuracy is significantly worse than the overall accuracy. This is because the confusion is high among postpositions in HTB. For example, out of 14,818 occurrences of `ke`, it takes the lemma `kA` in 7,763 instances and `ke` in 7,022 cases. The reason for this is, `ke` when it occurs in adverbial context (not necessary that the POS tag is adverb), the lemma remains `ke`. The accuracies for verbs are low (when compared to Nouns, Adjectives) as well mainly because verbs in Hindi take more inflections than the rest. The accuracy for verbs is even lower for OOV words (69.23% in Table 6.18).

6.5.2 Gender, Number, Person and Case

The accuracies of gender, number, person and case hover around 95% separately but the combined (G+N+P) accuracy drops to 91.11%. This figure is important if one wants to enforce agreement in parsing.

The OOV accuracy for person is close to overall accuracy as most of the OOV words belong to the 3rd person category. It is not the same case for gender and number. Gender particularly suffers a significant drop of 14% for OOV words confirming the theory that gender prediction is a difficult problem without knowing the semantics of the word.

The number and person accuracies for verbs are consistently low for OOV words as well as for seen words. This could be because SMA doesn't handle long distance agreement during GNP prediction.

Until now, we reported accuracies for OOV words of SMA. Table 6.17 lists accuracies for OOV words of the PBA (i.e. words which are not analyzed by the PBA) in the test section of HTB. SMA clearly outperforms baseline system and also performs better than F-PBA and O-PBA as they do not give any analyses.

In a nutshell, we have evaluated SMA for OOV words of the PBA as well as for OOV words of SMA. In both the cases, SMA performed better than other systems. We evaluated SMA in a challenging scenario wherein *training* data consists of the words from the HTB which are analyzed by the PBA and *test* data consists of the remaining unanalyzed words by the PBA. Thereby, the entire test data contains only *out of vocabulary* instances for both SMA and PBA. Table 6.20 presents the results of this new evaluation. The results are almost similar with that of OOV results shown in Table 6.14 except for *Person*. The reason behind that could be, in the training data there are only 0.1% instances of *3h* class

Analysis	Accuracy	OOV Accuracy
Gender	95.74	80.08
Number	95.29	89.71
Person	96.12	94.06
Case	95.16	88.32
G+N+P	90.92	74.14
G+N+P+C	87.72	68.47

Table 6.16 Joint Model for Gender, Number, Person, Case

Analysis	Baseline	SMA
Lemma	65.40	95.96
Gender	57.09	95.93
Number	76.79	95.17
Person	65.76	96.42
Case	46.39	95.17

Table 6.17 Accuracy for OOV words of PBA

but in test data their presence is quite significant (approximately 10%). The training instances for *3h* class were not sufficient for the model to learn and hence very few of these instances were identified correctly. This explains the drop in *Person* accuracy for this experiment scenario.

It may be noted that, we have used gold POS tags for all our experiments related to GNP and case prediction. There are numerous efforts on building POS taggers for Hindi. The ILMT pos tagger[5] is 96.5% accurate on the test data of the HTB. Table 6.21 reports the accuracies of gender, number, person and case using the automatic POS tags predicted by the ILMT tagger. The results are similar to that of the experiments conducted with gold POS tags.

[35] have build separate models for gender, number, person and case. Table 6.16 reports the results of *Joint Model* for these morph attributes. In terms of accuracy, Joint Model is as efficient as individual models.

6.5.3 TAM and Vibhakti

The proposed heuristics for Vibhakti and TAM prediction gave accuracy of 97.04% on test data set of HTB. On the entire HTB data, SMA achieved accuracy of 98.88%. O-PBA gave accuracy of 59.65% for TAM and Vibhakti prediction on test part of HTB. The reason behind low performance of O-PBA is that it identifies the TAM and vibhakti values for each word separately and doesn't consider the neighbouring words in the sentence.

[5] http://ilmt.iiit.ac.in/

Analysis	Noun	Verb	Adjective
Lemma	92.18	69.23	88.35
Gender	80.49	86.15	92.23
Number	92.35	76.92	87.38
Person	96.64	75.38	100.00
Case	88.81	98.46	70.87

Table 6.18 OOV accuracies for words (by POS tags)

Analysis	N	V	PSP	JJ	PRP
Lemma	98.50	94.28	89.41	97.99	98.78
Gender	93.30	95.34	98.93	98.42	94.24
Number	96.26	89.67	96.45	96.26	88.98
Person	98.58	85.28	99.45	99.57	90.94
Case	94.67	98.95	93.26	83.76	95.90
N:Noun, V:Verb, PSP:postposition, JJ:adjective, PRP:pronoun					

Table 6.19 Overall accuracies for words (by POS tags)

6.6 Effect on Parsing

The effect of morphological features on parsing is well documented [1]. Previous works used gold morphological analysis to prove their point. In this work, we also evaluated the effect of *automatic* morph features (predicted by SMA) on dependency parsing. MALT parser was trained on gold-standard POS tagged HTB data with and with out morph features. Table 6.15 lists the evaluation scores for these settings. While the unlabeled attachment score (UAS) does not show significant improvement, the labeled attachment score (LAS) and label accuracy (LA) have increased significantly. [1] also reported similar results with *gold-standard* morph features. Lemma, case, vibhakti and TAM features contribute to the increase in label accuracy because of the karaka labels in Paninian annotation scheme [3].

Table 6.15 also lists the performance of MALT parser with gold morph features (as annotated in HTB). It may be noted that, predicted morph features had similar effect on hindi dependency parsing as of gold features which is desirable making SMA usable for real scenario applications.

Our motivation for this work began with experiments on Hindi Parsing and there arose a need for context based morphological analyzer. Finally, we have built such analyzer and it helps in improving the parsing accuracy.

Analysis	Baseline	SMA
Gender	57.09	73.09
Number	76.79	85.71
Person	65.76	77.93
Case	33.62	89.05

Table 6.20 Evaluation of SMA in a challenging scenario: training data consists only of words analyzed by PBA and test data consists of remaining unanalyzed words.

Analysis	Overall	OOV
Gender	95.68	80.41
Number	94.97	90.30
Person	96.09	96.17
Case	94.61	88.19

Table 6.21 Accuracy of SMA with auto POS tags

Language	#Sentences	#Words
Urdu	5230	68588
Telugu	1600	6321

Table 6.22 Telugu and Urdu Treebank Statistics

Analysis	Telugu		Urdu	
	Overall	OOV	Overall	OOV
Gender	96.49	89.85	89.14	88.18
Number	90.65	75.13	91.62	91.35
Person	94.82	85.79	93.37	95.53
Case	96.49	89.34	85.49	79.01

Table 6.23 SMA for other Mor-FOW languages: Telugu and Urdu

6.7 Extending the work to Telugu and Urdu

In previous sections, we have reported the results of SMA for Hindi language. In this section, we look at how SMA performs in prediciting GNP and case for other morphologically rich Indian languages: Telugu and Urdu. At this stage, we have not done any language-dependent engineering effort in improving the results rather we want to see how well the system works for other languages using the minimalistic feature set employed for Hindi morphological analysis.

Telugu Treebank was released for ICON 2010 Shared Task[23] and a modified version of that data is used for our experiments. Urdu Treebank which is still under development at IIIT Hyderabad is used for experiments related to Urdu morph analysis. Refer table 6.22 for treebank statistics.

Table 6.23 shows the evaluation results for Telugu and Urdu. Though Telugu Treebank and Urdu Treebank are relatively small when compared to Hindi Treebank, we achieved good results. Lemma accuracies predicted by SMA were low. Various factors resulting in poor accuracy could be tokenisation handling and size of corpus.

[47] has built modified version of SMA and tested it for some Indian languages viz. Urdu, Telugu and Tamil. The results of modified SMA for those languages outperformed Morfette. Hence, we can conclude that SMA can be extended for other languages.

Chapter 7

Improvised Statistical Morphological Analyzer (SMA++)

7.1 Improvised Statistical Morphological Analyzer (SMA++)

SMA [36, 34] outperforms the existing analyzers. For lemma prediction in SMA, we adopted a machine translation model. It is computationally expensive to train and build a machine translation model (which is in the order of hours). In this section we look at the alternate approach for lemma prediction which is not only computationally inexpensive but also outperforms the lemma prediction of SMA in terms of accuracy. Also we try various other features for GNPC prediction and report their accuracies. We refer to this improvised SMA as *SMA++* [43].

7.1.1 Lemma perceived as *Classification Problem*

In SMA, we addressed lemma prediction with help of Machine Translation, now we perceive it as a classification problem. We generate class labels with the edit-distance operations required to convert reversed word tokens to the corresponding reversed lemmas. This idea was inspired by Chrupala (2006)[12].

7.1.1.1 Algorithm to generate lemma class label

Step-1. Token and lemma are reversed.

Step-2. Calculate the *Shortest Edit Distance* operations required to convert reversed token to reversed lemma.

Step-3. The list of edit-distance[1] operations from *Step-2* form the class label.

Consider the token-lemma pairs <*playing*,*play*>, <*realising*,*realise*>. Lemma for the tokens *playing* and *realising* are *play* and *realise* respectively. We explain the algorithm with the above examples.

Step-1:

Reversed strings are <*gniyalp*,*yalp*>, <*gnisilaer*,*esilaer*>.

Step-2:

In edit-distance operations, we denote the operations *delete* by **D**, *insert* by **I**, *replace* by **R**. <D,i> denotes *delete* character at position *i*. <R,c,i> denotes *replace* character *c* at position *i*. <I,c,i> denotes *insert* character *c* at position *i*. List of edit-distance operations for the reversed token-lemma pair <*playing*,*play*>: [<D,1>,<D,2>,<D,3>]. List of edit-distance operations for the reversed token-lemma pair <*realising*,*realise*>: [<D,0>,<D,1>,<R,2,e>].

Step-3:

[<D,1>,<D,2>,<D,3>] is the class label for the token-lemma pair <*playing*,*play*>.

[<D,0>,<D,1>,<R,2,e>] is the class label for the token-lemma pair <*realising*,*realise*>.

The class labels denotes the operations to be performed on the token to get the lemma.

Here, reversing the strings prior to computation of edit-distance operations is an important step as we get the same class labels for pairs such as <*crying*,*cry*>, <*playing*,*play*>, <*talking*,*talk*>. Ideally, they belong to same class and we are able to retain the information. This way lemma prediction is reduced to a classification problem. In brief, *we need to predict the class label for a token and the class label denotes the operations to apply on the token which gives the lemma*.

In the training part of HTB, there are 367 different class labels as against to 268,096 unique word tokens. So the percentage of class labels for lemma prediction is 0.14% which is quite a small number.

7.1.2 Gender, Number, Person and Case Marker

In SMA, we had to employ two different approaches for lemma and GNPC prediction. In SMA++, we have reduced lemma prediction to a classification problem and since GNPC already being a classification problem, we can adopt a single approach for lemma and GNPC prediction. We shall explore other features in SMA++ with more emphasis on having a common feature set to predict lemma and GNPC.

7.1.3 Features

The feature set was chosen suiting the linguistic characteristics of Morphologically Rich Languages. We explain each feature with the help of below sentence,

” `ladZake`/NN `ne`/PSP `kAnA`/NN `kAyA`/VM ”

7.1.3.1 Word Forms

We consider the word forms of previous word, current word and next word as features.

In the example sentence, for the word `ladZake`, the features are previous word - *NULL* (as there is no previous word in this example), current word - *ladZake* and next word - *ne*.

7.1.3.2 Length of the token

Most of the stop words or prepositions have similar small lengths. They possess similar morphology. Hence, length of the token is included as a feature as it helps to group tokens of such type.

In the example sentence, the length of the token for *ladZake*, *ne*, *kAnA*, *kAyA* are 7, 2, 4 and 4 respectively.

7.1.3.3 Character Type

For example, all the words which have only numbers show specific morphology. To groups such type of tokens, we use `Character Type` as a feature. It helps to capture the various character types which are present in the given word. If the word token has number, we denote it as `number`, if it has only characters we denote it as `character`.

In the example sentence, the feature value is '*character*' for all the words.

7.1.3.4 POS

Parts-of-Speech (POS) is the fundamental feature of any NLP task. Verbs display specific type of morphology as compared to Nouns. To capture such information, we include POS in the feature set.

7.1.3.5 Suffixes

Suffixes are important indicators of the morphology of a word especially in Morphologically Rich Languages. Morphemes (which do not exist independently and occur only with a root word) carry the gender, number, person and case marker information of a word. Suffixes with a maximum string length of 7 are considered as part of the feature set.

For the word ladZake: '*-e*', '*-ke*', '*-ake*', '*-Zake*', '*-dZake*', '*-adZake*', '*-ladZake*' are the suffix features varying from length 1 to 7.

7.1.3.6 Previous Morph Tags

Agreement is an important characteristic of Morphologically Rich Language. We consider predicted morph tag of 3 previous tokens as part of the feature set to capture this characteristic.

Analysis	Test Data - Overall(%)		Test Data - OOV(%)	
	SMA	SMA++	SMA	SMA++
L	95.84	98.43	89.51	93.07
G	96.19	96.21	82.65	83.11
N	95.37	95.47	90.44	92.81
P	96.38	96.28	74.85	96.17
C	95.32	95.43	88.52	89.45
L+C	91.39	94.01	79.09	82.92
G+N+P	91.11	90.36	76.52	77.24
G+N+P+C	87.78	88.51	69.99	72.36
L+G+N+P	87.51	89.26	69.13	72.82
L+G+N+P+C	84.25	85.87	63.06	65.96

L-lemma, G-gender, N-number, P-person, C-case

Table 7.1 Accuracies of SMA++ compared with SMA

In the example sentence, for the word kAyA, the previous three gender morph tags are *masculine* (for *kAnA*), *none* (for *ne*), *masculine* (for *ladake*). Similarly for the other morph attributes we populate the feature set accordingly.

7.1.3.7 Next Morph Tags

Earlier, we captured the agreement with the previous tokens. Agreement with next neighbouring token is also an important feature. All the possible morph tags, the next neighbouring token associates itself in the training corpus are considered as features for the present token.

7.1.4 Experiments & Results

The same datasets used in building SMA are used to build SMA++. With the training part of HTB (released for MTPIL) we build LIBLINEAR classification models to predict lemma, gender, number, person and case marker. On the test part of HTB, we run these classification models.

Table 7.1 reports the accuracies of SMA++ compared with SMA. SMA++ performed better than SMA in all metrics especially for OOV words. The main differentiating factor between SMA and SMA++ is the lemma prediction model. SMA++ lemma prediction model performs better than that of SMA. SMA performs marginally low for lemma prediction of OOV words when compared to Morfette. But SMA++ outperforms both SMA and Morfette. Thus we have built a superior lemma prediction model. Overall, SMA++ outperforms SMA and Morfette.

SMA++ can be extended to other language similar to how we did for SMA. Saikrishna et al. [47] has evaluated SMA++ for other languages viz. Urdu, Telugu and Tamil.

Chapter 8

Conclusions and Future Work

8.1 Conclusion and Future work

Existing Hindi Morphological Analyzers predicts all the possible analysis for a word. They do not consider sentential context. Also, for Out-Of-Vocabulary (OOV) words, existing analyzers just predict the default analysis. In this thesis, we built a Statistical Morphological Analyzer (SMA) which predicts context based analysis and also predicts analysis for OOV words. SMA is a robust state-of-the-art statistical morphological analyzer which outperforms previous analyzers for Hindi by a considerable margin. SMA achieved an accuracy of 63.06% for lemma, gender, number, person and case whereas PBA and Morfette are 34.89% and 51.52% accurate respectively. With the predicted morphological attributes by SMA, we achieve a labeled attachment score of 89.41 while without these morphological attributes the parsing accuracy drops to 87.75. SMA++ is an improvised version of SMA and its 65.96% accurate.

The agreement phenomenon in Hindi provides challenges in predicting gender, number and person of words in their sentential context. These can be better predicted if dependency relations are given as input. However, the standard natural language analysis pipeline forbids using parse information during morphological analysis. This provides an opportunity to explore *joint modelling of morphological analysis and syntactic parsing* for Hindi. In SMA++, for words with new set of add-delete operations, we pick the most relevant add-delete operations from our existing list but can not predict the new set of operations. This makes for an interesting case to predict completely new set of add-delete operations for a word. We plan to experiment them as part of our future work.

Chapter 9

Related Publications

9.1 Statistical Morphological Analyzer for Hindi

Authors: *Deepak Kumar Malladi*, Prashanth Mannem

Conference: Proceedings of 6th International Joint Conference on Natural Language Processing. 2013

Relevance to this thesis: This paper briefs about the research discussed in Chapter-6.

9.2 Context Based Statistical Morphological Analyzer and Its Effect On Hindi Dependency Parsing

Authors: *Deepak Kumar Malladi*, Prashanth Mannem

Conference: Fourth Workshop on Statistical Parsing of Morphologically Rich Languages, at EMNLP 2013 in Seattle, USA.

Relevance to this thesis: This paper talks in detail about the work discussed in Chapter-6.

9.3 Improvised and Adaptable Statistical Morph Analyzer (SMA++)

Authors: Saikrishna Srirampur, *Deepak Kumar Malladi* and Radhika Mamidi

Conference: 2nd Workshop on Indian Language Data: Resources and Evaluation (WILDRE) at LREC 2014 in Reykjavik, Iceland.

Relevance to this thesis: Research discussed in Chapter-7 is published in this paper.

9.4 Ensembling Various Dependency Parsers: Adopting Turbo Parser for Indian Languages

Authors: Puneeth Kukkadapu, *Deepak Kumar Malladi*, Aswarth Dara

Conference: MTPIL Workshop at 24th International Conference on Computational Linguistics. 2012

Relevance to this thesis: Chapter-5 gives an overview about the parsing experiments mentioned in this paper.

Bibliography

[1] B. R. Ambati, S. Husain, S. Jain, D. M. Sharma, and R. Sangal. 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, 2010.

[2] B. R. Ambati, S. Husain, J. Nivre, and R. Sangal. 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, 2010.

[3] R. Begum, S. Husain, A. Dhwaj, D. M. Sharma, L. Bai, and R. Sangal. Dependency annotation scheme for indian languages. In *Proceedings of IJCNLP*, 2008.

[4] A. Bharati, V. Chaitanya, R. Sangal, and K. Ramakrishnamacharyulu. *Natural language processing: A Paninian perspective*. Prentice-Hall of India New Delhi, 1995.

[5] A. Bharati, D. S. S. Husain, L. Bai, R. Begam, and R. Sangal. Anncorra: Treebanks for indian languages, guidelines for annotating hindi treebank (version–2.0), 2009.

[6] A. Bharati, A. P. Kulkarni, and V. Sheeba. Building a wide coverage sanskrit morphological analyser: A practical approach. In *The First National Symposium on Modelling and Shallow Parsing of Indian Languages, IIT-Bombay*, 2006.

[7] A. Bharati, R. Sangal, and D. M. Sharma. Ssf: Shakti standard format guide. *Language Technologies Research Centre, International Institute of Information Technology, Hyderabad, India*, pages 1–25, 2007.

[8] R. Bhatt, B. Narasimhan, M. Palmer, O. Rambow, D. M. Sharma, and F. Xia. A multi-representational and multi-layered treebank for hindi/urdu. In *Proceedings of the Third Linguistic Annotation Workshop*, ACL-IJCNLP '09, pages 186–189, Stroudsburg, PA, USA, 2009. Association for Computational Linguistics.

[9] T. Bögel, M. Butt, A. Hautli, and S. Sulger. Developing a finite-state morphological analyzer for urdu and hindi. *Finite State Methods and Natural Language Processing*, page 86, 2007.

[10] B. Cartoni. Lexical morphology in machine translation: A feasibility study. In *Proceedings of the 12th Conference of the European Chapter of the Association for Computational Linguistics*, pages 130–138. Association for Computational Linguistics, 2009.

[11] C.-C. Chang and C.-J. Lin. LIBSVM: A library for support vector machines. *ACM Transactions on Intelligent Systems and Technology*, 2:27:1–27:27, 2011.

[12] G. Chrupała. Simple data-driven contextsensitive lemmatization. *Procesamiento del Lenguaje Natural*, 37:121–127, 2006.

[13] G. Chrupała, G. Dinu, and J. Van Genabith. Learning morphology with morfette. 2008.

[14] R.-E. Fan, K.-W. Chang, C.-J. Hsieh, X.-R. Wang, and C.-J. Lin. LIBLINEAR: A library for large linear classification. *Journal of Machine Learning Research*, 9:1871–1874, 2008.

[15] J. Goldsmith. Unsupervised learning of the morphology of a natural language. *Computational linguistics*, 27(2):153–198, 2001.

[16] V. Goyal and G. S. Lehal. Hindi morphological analyzer and generator. In *Emerging Trends in Engineering and Technology, 2008. ICETET'08. First International Conference on*, pages 1156–1159. IEEE, 2008.

[17] N. Habash and O. Rambow. Arabic tokenization, part-of-speech tagging and morphological disambiguation in one fell swoop. In *Proceedings of the 43rd Annual Meeting on Association for Computational Linguistics*, pages 573–580. Association for Computational Linguistics, 2005.

[18] M. A. Hafer and S. F. Weiss. Word segmentation by letter successor varieties. *Information storage and retrieval*, 10(11):371–385, 1974.

[19] J. Hall, J. Nilsson, J. Nivre, G. Eryigit, B. Megyesi, M. Nilsson, and M. Saers. Single malt or blended? a study in multilingual parser optimization. In *Proceedings of the CoNLL Shared Task Session of EMNLP-CoNLL*, pages 933–939, 2007.

[20] Z. Hamawand. *Morphology in English: word formation in cognitive grammar*. Bloomsbury Publishing, 2011.

[21] Z. S. Harris. *Morpheme boundaries within words: Report on a computer test*. Springer, 1970.

[22] S. Husain. Dependency parsers for indian languages. *Proceedings of ICON09 NLP Tools Contest: Indian Language Dependency Parsing*, 2009.

[23] S. Husain, P. Mannem, B. R. Ambati, and P. Gadde. 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, 2010.

[24] S. M. Idicula and P. S. David. A morphological processor for malayalam language. *South Asia Research*, 27(2):173–186, 2007.

[25] N. Kanuparthi, A. Inumella, and D. M. Sharma. Hindi derivational morphological analyzer. In *Proceedings of the Twelfth Meeting of the Special Interest Group on Computational Morphology and Phonology*, pages 10–16. Association for Computational Linguistics, 2012.

[26] R. M. Kaplan and M. Kay. Phonological rules and finite-state transducers. In *Linguistic Society of America Meeting Handbook, Fifty-Sixth Annual Meeting*, pages 27–30, 1981.

[27] P. Koehn. *Statistical machine translation*. Cambridge University Press, 2009.

[28] P. Koehn, H. Hoang, A. Birch, C. Callison-Burch, M. Federico, N. Bertoldi, B. Cowan, W. Shen, C. Moran, R. Zens, et al. Moses: Open source toolkit for statistical machine translation. In *Proceedings of the 45th*

Annual Meeting of the ACL on Interactive Poster and Demonstration Sessions, pages 177–180. Association for Computational Linguistics, 2007.

[29] K. Koskenniemi. A general computational model for word-form recognition and production. In *Proceedings of the 10th international conference on Computational Linguistics*, pages 178–181. Association for Computational Linguistics, 1984.

[30] P. Kukkadapu, D. K. Malladi, and A. Dara. Ensembling various dependency parsers: Adopting turbo parser for indian languages. In *24th International Conference on Computational Linguistics*, page 179, 2012.

[31] P. Kukkadapu and P. Mannem. A statistical approach to prediction of empty categories in hindi dependency treebank. In *Fourth Workshop on Statistical Parsing of Morphologically Rich Languages*, page 91, 2013.

[32] Y.-S. Lee. Morphological analysis for statistical machine translation. In *Proceedings of HLT-NAACL 2004: Short Papers*, pages 57–60. Association for Computational Linguistics, 2004.

[33] C. Li and S. Thompson. A functional reference grammar of mandarin chinese, 1981.

[34] D. K. Malladi and P. Mannem. Context based statistical morphological analyzer and its effect on hindi dependency parsing. In *Fourth Workshop on Statistical Parsing of Morphologically Rich Languages*, volume 12, page 119, 2013.

[35] D. K. Malladi and P. Mannem. Statistical morphological analyzer for hindi. In *Proceedings of 6th International Joint Conference on Natural Language Processing*, 2013.

[36] D. K. Malladi and P. Mannem. Statistical morphological analyzer for hindi. In *Proceedings of 6th International Joint Conference on Natural Language Processing*, 2013.

[37] A. F. Martins, N. A. Smith, E. P. Xing, P. M. Aguiar, and M. A. Figueiredo. Turbo parsers: Dependency parsing by approximate variational inference. In *Proceedings of the 2010 Conference on Empirical Methods in Natural Language Processing*, pages 34–44. Association for Computational Linguistics, 2010.

[38] T. Nguyen and S. Vogel. Context-based arabic morphological analysis for machine translation. In *Proceedings of the Twelfth Conference on Computational Natural Language Learning*, pages 135–142. Association for Computational Linguistics, 2008.

[39] M. Pedersen, D. Eades, S. K. Amin, and L. Prakash. Relative clauses in hindi and arabic: A paninian dependency grammar analysis. *COLING 2004 Recent Advances in Dependency Grammar*, pages 9–16, 2004.

[40] M. Popovič and P. Willett. The effectiveness of stemming for natural-language access to slovene textual data. *Journal of the American Society for Information Science*, 43(5):384–390, 1992.

[41] M. Rogati, S. McCarley, and Y. Yang. Unsupervised learning of arabic stemming using a parallel corpus. In *Proceedings of the 41st Annual Meeting on Association for Computational Linguistics-Volume 1*, pages 391–398. Association for Computational Linguistics, 2003.

[42] K. Sagae and A. Lavie. Parser combination by reparsing. In *Proceedings of the Human Language Technology Conference of the NAACL, Companion Volume: Short Papers*, pages 129–132. Association for Computational Linguistics, 2006.

[43] D. K. M. Saikrishna Srirampur and R. Mamidi. Improvised and adaptable statistical morph analyzer (sma++). In *2nd Workshop on Indian Language Data: Resources and Evaluation (WILDRE) at LREC 2014 in Reykjavik, Iceland.*, 2014.

[44] D. M. Sharma, P. Mannem, J. VanGenabith, S. L. Devi, R. Mamidi, and R. Parthasarathi, editors. *Proceedings of the Workshop on Machine Translation and Parsing in Indian Languages*. Mumbai, India, December 2012.

[45] N. A. Smith, D. A. Smith, and R. W. Tromble. Context-based morphological disambiguation with random fields. In *Proceedings of the conference on Human Language Technology and Empirical Methods in Natural Language Processing*, pages 475–482. Association for Computational Linguistics, 2005.

[46] R. W. Sproat. *Morphology and computation*. MIT press, 1992.

[47] S. Srirampur, R. Chandibhamar, and R. Mamidi. Statistical morph analyzer (sma++) for indian languages. *COLING 2014*, page 103, 2014.

[48] K. Toutanova, H. Suzuki, and A. Ruopp. Applying morphology generation models to machine translation. In *ACL*, pages 514–522. Citeseer, 2008.

[49] G. Uma Maheshwar Rao. Morphological analyzer for telugu.(electronic form). *Hyderabad: University of Hyderabad*, 1999.

[50] D. Yuret and F. Türe. Learning morphological disambiguation rules for turkish. In *Proceedings of the main conference on Human Language Technology Conference of the North American Chapter of the Association of Computational Linguistics*, pages 328–334. Association for Computational Linguistics, 2006.

[51] D. Zeman. Maximum spanning malt: Hiring worlds leading dependency parsers to plant indian trees. *ICON09 NLP Tools Contest: Indian Language Dependency Parsing. Hyderabad, India*, 2009.