AutoParSe: An *Auto*matic *Par*adigm *Se*lector For Nouns in Konkani

Shilpa Desai
Department of Computer Science and Technology,
Goa University,
Goa - India
sndesai@gmail.com

Neenad Desai
N.D. Consultancy,
Cuncolim,
Goa - India
neenad@gmail.com

Jyoti Pawar
Department of Computer Science and Technology,
Goa University,
Goa - India
jyotidpawar@gmail.com

Pushpak Bhattacharyya
Department of Computer Science and Engineering,
IIT, Powai,
Mumbai - India
pb@cse.iitb.ac.in

Abstract

In this paper, we discuss a rule based method which automatically assigns paradigms to Konkani nouns using morphophonemic rules, stem formation rules and relevance score of the paradigms. The first contribution is computation of relevance score of a paradigm, which is computed using a corpus and paradigm differentiating measure assigned to inflectional suffixes in the paradigm. Relevance score helps assign multiple paradigms to the input word wherever appropriate. The other contribution is a method for computing paradigm differentiating measure for inflectional suffixes. We have proposed a pruning technique based on derivational suffixes to further improve the precision. The experimental study has been carried out using the Konkani WordNet and the Asmitai Corpus. The proposed method successfully assigned relevant paradigms to 10,068 nouns with F-Score of 0.93.

1 Introduction

A Morphological Analyzer for a language is a tool which is used to analyze a word into constituent morphemes. Morphological Analysis is required in NLP applications such as spell checkers and rule based machine translation. Various approaches have been used for morphology acquisition. Amongst all, finite state based approaches perform the best. Finite state approaches are based on paradigms[1]. These paradigms are defined by linguistic experts. Words in the language are then assigned to an appropriate paradigm. If this mapping of words to paradigms is done manually it requires linguistic experts and is time consuming. Based on the morphological richness of the language and level of expertise and availability of the linguists, it takes about 3 to 6 months to map words to paradigms.

Previous work which assign paradigms to words (Carlos et al., 2009) use POS tagged data to improve precision. Since we do not have a POS tagger tool for Konkani Language, our method attempts to improve precision of paradigm assignment by using morphophonemic rules and computing relevance score for paradigms.

The input to our system is as follows: a) raw text corpus, b) morphological rules and c) root word lexicon for Konkani. We output a noun paradigm repository, wherein a noun in the WordNet is assigned its inflectional paradigm. Our automatic paradigm selector method is implemented for Konkani nouns. We have defined our own paradigm structure for Konkani nouns which uses Finite State based sequencing of morphemes.

[1] Words which inflect similarly belong to the same paradigm. That is, these words follow the same stem formation rule and get attached to a common set of suffixes

In this paper we present a detailed study of Konkani noun morphology and a technique which can be used to assign paradigms to Konkani nouns. The rest of the paper is organized as follows: section 2 is devoted to related work. A description of the Konkani noun morphology is discussed in section 3. Section 4 presents our AutoParSe system design and description. In Section 5, we discuss experimental results and evaluation. Section 6 concludes the paper.

2 Related Work

Automatic mapping of word to a paradigm have been done earlier for other languages. Rule based systems which map words to paradigms have been attempted (Sanchez, 2012). These systems use POS information or some additional user input from native language speakers to map words to paradigms, instead of a corpus alone. A corpus assisted approach (Desai et al., 2012) has been used to map Konkani verbs to a single paradigm.

Lexicon acquisition methods exist for many languages that map words to morphological paradigms. (Attia et al., 2012; Clement et al., 2004; Carlos et al., 2009; Forsberg, 2006) Functional Morphology has been used to define morphology for languages like Swedish and Finnish, and tools based on Functional Morphology, namely Extract (Forsberg, 2006) which suggest new words for a lexicon and map them to paradigms, have been developed. Functional Morphology based tools use constraint grammars to map words correctly to paradigms. To be able to use a tool like Extract, the morphology of the language has to be fitted into the Functional Morphology definition.

N-gram based model of lexical categories (Linden, 2009) and inflection information to select a single paradigm in cases where more than one paradigm generates the same set of word forms are available.

3 Konkani Noun Morphology

Noun forms are usually obtained by attaching prefix or suffix. In Konkani, prefixing is not very productive[2]. Suffixes could be derivational suffixes or inflectional suffixes. Mostly lexicons maintain the derivational forms of a word. Hence we do not need map derivational words to paradigms. Inflectional forms are not maintained in lexicons, hence there is a need to group all inflectional forms into paradigms. An inflectional paradigm for a noun in Konkani will be a set of suffixes which can be attached to a common stem.

3.1 Konkani Noun Inflectional Morphology

All nouns in Konkani are inflected for number and syntactic case. Number can be singular or plural. Konkani cases are nominative, accusative, dative, locative, instrumental, vocative and genitive. Besides nominatives, other cases show a suffix before the case marker which is referred to as oblique suffix. Thus Konkani cases are divided into two basic types, direct and oblique (Almeida, 1989). Cases which require oblique suffixes are called oblique case types.

Example 1: Consider the inflected form *ghodyakuch* (घोड्याकूच) of noun ghodo (घोडो). The inflectional segmentation can be given as: ghodyakuch = ghod+ya+k+uch (घोड्याकूच = घोड+्या+क+ूच)
where:
Noun root[3]: ghodo (घोडो means horse)
Noun stem: ghod (घोड)
Oblique singular suffix: ya (्या)
Case marker suffix: k (क)
Clitic: uch (ूच emphatic particle)

Table 1 shows the different case-marker suffixes a Konkani noun can take.

Case Marker	**Case Marker Suffix**		**Case type**
	Singular	**Plural**	
Nominative	*Unmarked*		Direct
Dative	-k (क)	-k (क)	Oblique
Accusative	-k (क)	-k (क)	Oblique
Locative1	-˜t (ंत)	-ni (नी)	Oblique
Locative2	-r/cer (र/चेर)	-r/cer (र/चेर)	Oblique
Instrumental	-n (न)	-ni (नी)	Oblique
Vocative	NULL ( )	-no (नो)	Oblique
Genitive	-co …(चो…)	-co …(चो…)	Oblique

Table 1: Case Marker suffixes in Konkani

Konkani leaves the nominative case unmarked. The genitive case has many case marker suffixes

[2] Attaching a prefix to a Konkani word to get inflected word forms is rare.

[3] Here we refer to the citation form of the noun in the dictionary or WordNet as the root.

which are not listed in Table 1. To obtain the inflections of a given Konkani noun which is in root form, we need to first obtain the corresponding stem and oblique form. We can than attach case marker suffixes to obtain noun forms. Clitics (Walawalikar et al., 2010) can be appended to noun forms to obtain variants of the noun forms.

4 AutoParSe Design and Description

In order to map Konkani nouns in root form to corresponding paradigm, we use a rule-based approach.

4.1 AutoParSe Design

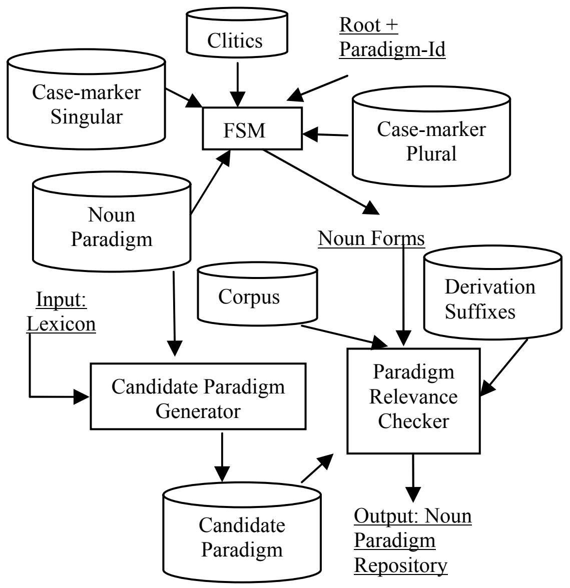

Figure 1: AutoParSe Design

AutoParSe, our noun paradigm selector has three modules listed below:

1. Finite State Machine (FSM).
2. Candidate Paradigm Generator (CPG)
3. Paradigm Relevance Checker (PRC)

Each of the modules is described in detail below.

4.2 FSM: Finite State Machine

FSM module is used to sequence morphemes in noun forms. It is called by the paradigm relevance checker module to obtain the inflectional word forms, for a given input root word and paradigm id. There will be an instance of the FSM for each noun paradigm. Figure 2 shows the general FSM for Konkani nouns. Sample input and output to FSM module is illustrated in Table 2

Sample Input	ghodo (घोडो) , P-19
Sample Output	**ghod**e, **ghod**yak, **ghod**yakch, **ghod**yan, **ghod**yacho … घोडे, घोडयाक, घोडयाकच, घोडयान, घोडयाचो ....

Table 2: Sample input and Output for FSM Module

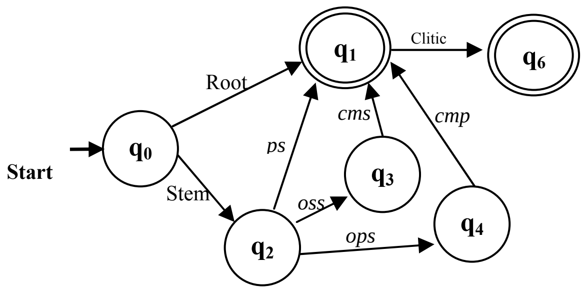

Figure 2: FSM for Konkani Noun Morphology

Input: Root noun and paradigm id
Output: Inflected noun forms with analysis
Resources used: The FSM module uses four resources namely

- Clitics : List of clitic suffixes for the language
- Case-marker Singular (CMS): List of case marker suffixes[4] which gets attached to singular noun oblique form with analysis.
- Case-marker Plural (CMP) : List of case marker suffixes which gets attached to plural noun oblique form with analysis
- Noun Paradigm: This resource is made of 5-tuple which consists of
  - Paradigm id: Unique identifier for paradigm
  - Stem Rule (SR)[5]: Rule to obtain stem from root.
  - Plural Suffix: Ordered pair (*ps*, *pdm*) where *ps* is the plural suffix and *pdm*[6] is corresponding paradigm differentiating measure assigned.
  - Oblique singular suffix: Ordered pair (*oss*,*pdm*) where *oss* is the oblique singular suffix and *pdm* is corresponding paradigm differentiating measure assigned.
  - Oblique plural suffix: Ordered pair (*ops*,*pdm*) where *ops* is the oblique plural suffix and *pdm* is corresponding paradigm differentiating measure assigned.

[4] CMS and CMP suffixes are listed in Table 1
[5] SR rule is discussed in subsection 4.2.1
[6] Computation of *pdm* is discussed in subsection 4.4.2

Constraint on noun paradigm design: The noun paradigms are designed in such a way that the tuple (SR, *ps, oss, ops*) is unique for a paradigm in Konkani.

Algorithm:

```
For input noun_root and paradigm id,
    NF = Empty Set
    Get corresponding noun paradigm
    stem = SR(noun_root)
    plural = stem+ps
    NF = NF U {plural}
    For each cms suffix in CMS
        nf = stem+oss+cms
        NF = NF U {nf}
    End For
    For each cmp suffix in CMP
        nf = stem+ops+cmp
        NF = NF U {nf}
    End For
    For each c suffix in Clitics
        nf = nf+c
        NF = NF U {nf}
    End For
End For
Output NF
```

4.2.1 Stem Rule (SR)

Most of the time a stem is obtained from the root by simply dropping the end letters as in case of example 1. Let us consider another example.

Example 2: Consider the word form *sorpak* (सोरपाक)
The inflectional segmentation can be given as:
sorpak = sorp+a+k (सोरपाक = सोरप+ा+क)
where
Noun root: sorop (सोरोप means *snake*)
Noun stem: sorp (सोरप)
Oblique suffix: a (ा)
Case marker suffix: k (क)

Here the stem is not obtained directly by dropping end characters, but also follows some morphophonemic rules. The following Table 3 lists some of the morphophonemic rules to be followed to obtain a stem. Note that the morphophonemic rule stated above will be applied to those root words which do not end in a vowel.

Type	Morphophonemic Rule	Example (root → stem)
Vowel Shortening	If syllable count equals two and penultimate character is i:/u: replace by i/u	**ki:r** → kir (कीर→किर)
Vowel Deletion	If syllable count greater than two and penultimate character is i:/u:/o delete penultimate character	**hũdi:r** → **hũdr** (हुंदीर→हूंदर)

Table 3: Morphophonemic Rules in Konkani

For noun roots which end in a vowel, the stem rule is of the form αβγ.
α → Delete (refers to the operation to be performed)
β → END (refers to position)
γ → ending vowel (refers to the ending vowel to be deleted to get stem)
The Delete operation converts root to stem by removing the end vowel. For noun roots which do not end in a vowel, the stem rule is the morphophonemic rule.

4.3 Candidate Paradigm Generator (CPG)

This module uses linguistic knowledge of the Konkani language to choose candidate paradigms. All paradigms which have compatible SR to the input root will be picked. In Konkani, a compatible paradigm will be based on the ending vowel of root word. Sample input and output to CPG module is illustrated in Table 4

Sample Input	man (मान)
Sample Output	P-1, P-2, P-6

Table 4: Sample input and output for CPG Module

Input: Word from lexicon
Output: List of candidate paradigm ids
Resources used: Noun Paradigm
Algorithm:

```
For paradigm id in Noun Paradigm,
    CP = Empty Set
    Get corresponding SR
    If SR compatible with root noun
    CP = CP U {paradigm id}
End For
Output CP
```

4.4 Paradigm Relevance Checker

This module computes relevance score for input paradigms and outputs those paradigms with relevance score greater than a set threshold. It makes use of paradigm differentiating measure and suffix evidence (Carlos et al.) to compute the relevance score for a paradigm. We first discuss a few terms which are used to compute relevance score of a paradigm.

4.4.1 Common Paradigm Group (CPG)

The paradigms in Candidate Paradigms (CP) are compatible with SR rule. We can further prune CP to get CPG. We prune paradigms in CP to whose stem either *ps*, *oss* or *ops* suffix cannot be attached. Such an attachment is not possible in cases where stem ends in vowel and any one of *ps*, *oss* or *ops* begins with a vowel.

4.4.2 Paradigm Differentiating Measure

We define suffix paradigm differentiating measure *pdm* for a suffix *s*, as the number of times a suffix *s* occurs in Common Paradigm Group (CPG). A *pdm* of one indicates that the presence of this suffix is a deciding factor to assign the corresponding paradigm. Table 5 illustrates the computation of *pdm* for a suffix.

CPG	P-1, P-2,P-5,P-6		
Suffix	***ps***	***oss***	***ops***
P-1	-a	-o	-yã
P-2	-i:	-ya	-yã
P-5	-a	-a	-yã
P-6	-e	-a	-ã
	pdm(-a) = 2 *pdm*(-i:) = 1 *pdm*(-e) = 1	*pdm*(-o) = 1 *pdm*(-ya) = 1 *pdm*(-a) = 2	*pdm*(-yã) = 3 *pdm*(-ã) = 1

Table 5: Paradigm differentiating measure (*pdm*) computation

From Table 5, we observe that the *oss* suffix ***–o*** is more useful to disambiguate the paradigm as compared to *oss* suffix ***–a***. That is, if many words with *oss* suffix ***–o*** are obtained in the corpus, we give a high relevance score to paradigm P-1.

4.4.3 Suffix Evidence Value (SEV)

Suffix evidence value for a paradigm is the cardinality of intersection of two sets, namely word forms in the corpus with word forms the given paradigm generates. Table 6 illustrates the computation of suffix evidence value for a given paradigm.

Corpus	**ghodo**, **ghode**, ghodyacho, ghodekar, ghodegiri, **ghodyan**
P-1 Generated	**ghodo**, **ghode**, ghodyak, **ghodyan**, ghodyat, ghodyar, ghodyacer
SES for P-1	**ghodo**, **ghode**, **ghodyan**
SEV for P-1	**3**

Table 6: Suffix Evidence value computation

4.4.4 Suffix Evidence Set (SES)

Suffix evidence set for a paradigm is the intersection of two sets, namely word forms in the corpus with word forms the given paradigm generates.

4.4.5 Derivational cum Oblique Suffix (DOS)

Derivational cum oblique suffix set is the intersection of derivational suffix set for nouns and oblique suffix set for nouns. Table 7 illustrates an example of DOS set.

Derivation Suffix Set (DSS)	{ari, er, ist, vot, **i**, kar, pon, al, est, adik, vot, ik, **e**, si, li, ul}
oss	{a, **i**, **e**, ya, ye, va}
ops	{ã, ĩ, ĕ, yã, vã}
Oblique suffix set O = *oss* U *ops*	{ a, **i**, **e**, ya, ye, va, ã, ĩ, ĕ, yã, vã }
DOS = O ∩ DSS	{ **i**, **e**}

Table 7: DOS computation

Sample input and output to Paradigm Relevance Checker module is illustrated in Table 8

Sample Input	{raja, P-1, P-2,P-8,P-9}
	{ghodo, P-4, P-6}
	{man, P-1,P-2,P-3, P-7}
	{soso, P-15}
	{goru, P-17,P-18}
	………

Sample Output	{raja = P-8}
	{ghodo = P-6}
	{man = P-1,P-3}
	{soso = P-15}
	{goru, P-17}

Table 8: Sample input and output for Paradigm Relevance Checker module

Note that in case of the word *man*, paradigm relevance checker outputs two paradigms for one word which is correct. If a word has more than one sense as a noun and the senses inflect differently two paradigms should be given to that word. In Konkani, the word *man* (मान) has two senses namely *respect* and *neck*.

Computation of relevance score: The relevance score for a paradigm is computed with respect to the CPG for the input root word. For each paradigm in a CPG, we compute paradigm differentiating measure for *ps*, *oss* and *ops* and compute SEV, using SES to check only for those having suffix with *pdm* one. We refer to the SEV computed in this manner as the relevance score for that paradigm.

Methodology: To assign a paradigm to a given input noun, we first get candidate paradigms CP. We prune the CP list to obtain CPG applicable to the input noun. We compute relevance score for each paradigm in CPG.

All paradigms in CPG with relevance score greater than a threshold are assigned to the input noun. We have set the threshold to two[7].

An important step of the relevance checker to increase precision, in case more than one paradigm is assigned to a noun, is to prune the derived word inflectional paradigm that is incorrectly assigned. To do this pruning, we prune the paradigm assigned to a noun if its corresponding SES can be generated by the derived form of the noun. We use DOS to get the derived form of the noun. We check if the derived form is present in the lexicon. We re-compute SES for the derived form and match it to the SES generated earlier. If SES matches, then the assigned paradigm has to be pruned.

Since there are oblique suffixes which can also act as derivational suffixes, we observed that a wrong paradigm was being assigned to the input noun in some cases where the derived word inflections present in the corpus resulted in a high SEV for the wrong paradigm. Example; for the noun *naukar* which means *servant*, two paradigms were being assigned. The other paradigm was in fact corresponding to the derived word *naukari* which means *job*. We use DOS to prune such erroneous paradigm assignments and improve the precision of the relevance checker.

[7] If many paradigms find relevance score of more than two in the corpus, it indicates that the inflections are commonly used in the language for that word.

The details for the paradigm relevance checker module are as follows:
Input: List of candidate paradigm CP for input root noun.
Output: Paradigm id corresponding to input root.
Resources used: Corpus, Word Forms

Algorithm:

```
FP = Empty set
For para_id in CP,
    Obtain CPG
End For
For para_id in CPG,
    Compute Suffix Evidence
     If Suffix Evidence = 0
        Delete paradigm id from CPG
     End If
End For
For para_id in CPG,
    If only one paradigm in CPG
        Found_Paradigm_id = para_id
    Else
        count = 0
        For wf in suffix evidence set
            If wf has suffix pdm 1
                and MBS > threshold
                count++
            End If
            If count > 2
                FP = FP U {para_id}
            End If
        End For
    End If
End For
Prune derivational paradigms assigned.
Output FP
```

5 Experimental Results and Evaluation

The goal of our experiment was to build a high accuracy paradigm selector for nouns and to be able to identify nouns from other part of speech categories. We selected 2000 words randomly from the input lexicon which were assigned paradigms manually, to obtain a gold standard data for comparison. These consisted of nouns for which paradigms were assigned, and other parts of speech for which paradigms were not assigned. The implementation of AutoParSe is done in Java using NetBeans IDE on Windows.

5.1 Results for AutoParSe

AutoParSe tool selects a paradigm for an input noun and generates Noun Paradigm Repository. Data sets used and the results obtained for Konkani nouns are as follows:

5.1.1 Data Sets

The following resources were used as input:

- Konkani WordNet, developed as part of the Indradhanush WordNet Project funded by DIT, New Delhi, India was used as an input resource to get the word lexicon for Konkani.
- Asmitai corpus consisting of approximately 268,000 unique word forms was used for selecting the appropriate paradigm.
- Noun Paradigm List was manually prepared with help from linguists and reference from Konkani grammar book and Konkani linguistic dissertation studies. (Almeida, 1989; Sardesai, 1986; Borkar, 1992)
- Case marker lists and clitic list were manually prepared.

To obtain the input word lexicon we had to undertake some pre-processing tasks which are covered in the following subsection.

Data Pre-processing

We extracted the words for our input lexicon from the synset in a WordNet (Bhattacharyya, 2010). We performed two pre-processing tasks discussed below.

Multiword removals: Words in the synset represented by multiword expression such as lhan_dongor (ल्हान_दोंगर means *hillock*) were pruned from the input. Such multiword expressions will inflect the same as its component last word namely dongor (दोंगर).

Unique word forms: Since in a WordNet, a word can appear in more than one synsets, we removed the duplicates and retained only unique word forms in our input lexicon.

5.1.2 Experimental Results

Total number of unique root words, which form the input lexicon, used for the study was 18301. Following results were obtained

- Total number of nouns identified by the program, for which noun paradigms were mapped: 10068
- Total number of words for which noun paradigms were not mapped by the program: 8233[8]

Comparison with gold standard data:
The output generated by the program was filtered to obtain those words which are present in our gold standard for comparison. The comparison is as follows:

- Total number of words in gold standard manually assigned to some noun paradigm: 1109
- Total number of words in gold standard unassigned to any noun paradigm: 891
- Total number of words AutoParSe assigns to some noun paradigm: 1101
- Total number of words AutoParSe does not assign any noun paradigm: 899
- Total number of words assigned to same noun paradigm in gold standard and AutoParSe (true positives): 1066
- Total number of words unassigned to any noun paradigm in gold standard and AutoParSe (true negatives): 774
- Total number of words unassigned to any noun paradigm in gold standard, but incorrectly assigned to a noun paradigm by AutoParSe (false positives): 26
- Total number of words assigned to noun paradigm by both gold standard and AutoParSe which mismatch (false positives): 9
- Total number of words assigned to noun paradigm by gold standard, but is incorrectly unassigned by AutoParSe (false negatives): 125

  Precision = 1066 / (1066+26+9) = 0.968

  Recall = 1066 / (1066+125) = 0.895

  F-Score = 0.93

[8] Along with nouns, the input lexicon also contained verbs, adjectives and adverbs, which were correctly not mapped.

5.1.3 Analysis of Results

We analyzed the results from two perspectives namely

1. Precision of paradigm assignment
2. Recall of paradigm assignment

Precision of Paradigm Assignment

We observed a high precision of words assignment to noun paradigms. However we observed certain places where the assignment faltered. The cases are as below:

- Some verbs were incorrectly assigned to noun paradigms. This happens because a verb gerund in Konkani acts as a nominal and all suffixes that apply to nouns also apply to verbs. This cannot actually be termed as error but needs to be handled separately.
- In some rare cases, the inflected form of the noun obtained is not actually an inflection, but another word present in the corpus, which happens to have a word ending which matches the noun suffixes; for the word *kat (कात* means *skin)*, a particular paradigm generates an inflected form *katar (कातार* means *Qatar country)* found in the corpus and hence adds a wrong paradigm to the word. Correct inflection for *kat* would be *katir (कातीर* means *on the skin)*

We prune derivational word inflections paradigms incorrectly assigned. This pruning increases precision. Table 9 illustrates the effect of pruning on precision.

Method	Precision
AutoParSe without pruning	0.932
AutoParSe with pruning	0.968

Table 9: Effect of pruning on precision

Recall of Paradigm Assignment

When we analyzed the results to check if we could further improve the recall, we realized that it was not absolutely necessary to do so. We looked into the factors which caused the recall to reduce and found that it was unassigned nouns. We refer to those nouns for which paradigms were not assigned as unassigned nouns. We studied these unassigned nouns to find possible reasons why they remained unassigned. We observed that the nouns which remained unassigned were not present in the corpus used. Table 10 shows the classification of the unassigned nouns. We can clearly see that these words are not natural to Konkani language. A natural question arises as how they appear in the input if they are not natural to the language. The answer lies in the resource used, namely, Konkani WordNet, to create the lexicon.
In a WordNet, a word is a part of a synset[9] which follows the principal of minimality, coverage and replacebility (Bhattacharyya, 2010). As a result, to comply with the principal of coverage, rarely used synonyms of the word are also present in the WordNet.

Unassigned Types	Examples
Named Entities	*Angola* (अंगोला, name of a country)
Foreign words or borrowed words	*Accordion* (अकार्डियन) *Infrastructure* (इंफ्रास्ट्रक्चर) *Attack* (अटॅक)
Coined words	*Aksharganit* (अक्षरगणीत means *algebra*)
Rare usage words	*Akrutya* (अकृत्य means *an action not to be performed*)

Table 10: Unassigned Nouns

The fall in recall is due such rare words, which would have to be manually assigned paradigms.

6 Conclusion

Selecting paradigms by giving priority to morphophonemic rules of the language helps improve the precision of paradigm selection. Overlap in inflectional and derivational suffixes in Konkani tend to reduce the precision and need to be handled appropriately.

By assigning *pdm* to paradigm differentiating suffixes, we get a new way to correctly map multiple paradigms to a noun root, which reflects the different senses in which the noun could occur. To the best of our belief, most automatic paradigm

[9] A synset stands for synonymous set, consists of a group of synonymous words which can be used interchangeably. Synset are used to represent a concept in the language.

selectors tend to produce a single mapping which does not capture multiple senses of words. Sometimes, linguists assigning paradigms to words could miss a word sense which is better captured by our automatic paradigm selector. Multiple paradigms also suggest multiple acceptable inflectional form uses for a word. For example, *nhayr* and *nhayer* (न्हंयर and न्हंयेर both mean *on the river*) inflectional forms used are both present in the corpus and should be captured by a good system. A corpus is bound to have spelling dialect variations of word form. AutoParSe is flexible enough to accommodate such inflectional form variations which a linguist may disagree with and stick to standard inflectional rules of the language. Thus, our AutoParSe method will be able cater better to a true real life corpus morphological analysis requirement.

References

Almeida Matthew, S.J. 1989. A Description of Konkani. Thomas Stephens Konknni Kendr, Miramar Panaji, Goa.

Attia Mohammed, Samih Younes, Khaled Shaalan Josef van Genabith. 2012. The Floating Arabic Dictionary: An Automatic Method for Updating a Lexical Database through the Detection and Lemmatization of Unknown Words. In Proceedings of COLING 2012, Mumbai.

Bhattacharyya Puskpak, IndoWordNet, Lexical Resources Engineering Conference 2010 (LREC 2010), Malta, May, 2010.

Borkar Suresh Jayvant, Konkani Vyakran, Konkani Bhasha Mandal, 1992.

Carlos Cohan Sujay, Manojit Choudhury and Sandipan Dandapat. 2009. Large-Coverage Root Lexicon Extraction for Hindi. In Proceedings of the 12th Conference of the European Chapter of the ACL, Athens, Greece.

Clement Lionel, Sagot Benoit and Lang Bernard. 2004. Morphology based automatic acquisition of large coverage lexica. In Proceedings of LREC 2004, Lisbon, Portugal.

Desai Shilpa, Pawar Jyoti, Bhattacharya Pushpak. 2012. Automated Paradigm Selection for FSA based Konkani Verb Morphological Analyzer. In Proceedings of COLING 2012: Demonstration Papers, Mumbai.

Forsberg Markus, Hammarstrom Harald and Ranta Aarne. 2006. Morphological Lexicon Extraction from Raw Text Data. In Proceedings of the 5th International Conference on Advances in Natural Language Processing, FinTAL, Finland.

Linden, K. and Tuovila, J. 2009. Corpus-based Paradigm Selection for Morphological Entries. In Proceedings of NODALIDA 2009, Odense, Denmark.

Sardesai Madhavi, Some Aspects of Konkani Grammar, M. Phil Thesis(1986).

V. M. Sanchez-Cartagena, M. Espla-Gomis, F. Sanchez-Martnez, J. A. and Perez-Ortiz 2012. Choosing the correct paradigm for unknown words in rule-based machine translation systems. In Proceedings of the Third International Workshop on Free/Open-Source Rule-Based Machine Translation, Gothenburg, Sweden.

Walawalikar Shantaram, Desai Shilpa, Karmali Ramdas, Naik Sushant, Ghanekar Damodar, D'Souza Chandralekha and Pawar Jyoti. 2010. Experiences In Building The Konkani WordNet Using The Expansion Approach. In Proceedings of the 5th Global WordNet Conference, Mumbai, Narosa Publishing House, India.