Developing a Basic Lexical Resource for Urdu Using Hindi WordNet

Tafseer Ahmed, Annette Hautli
Department of Linguistics, University of Konstanz
{tafseer.khan, annette.hautli}@uni-konstanz.de

Abstract

This paper reports on a first experiment with developing a lexical knowledge resource for Urdu on the basis of Hindi WordNet. Due to the structural similarity of Urdu and Hindi, we can focus on overcoming the differences in the scriptual systems of the two languages by using transliterators. Various natural language processing tools, among them a computational semantics based on the Urdu ParGram grammar, can use the resulting basic lexical knowledge base for Urdu.

1. Introduction

In the development of sophisticated natural language processing systems, it is understood that a rich lexical knowledge base is at the heart of any intelligent system that attempts to go beyond the syntactic analysis of sentences. A lexical resource can shed light on the meaning of a sentence by providing information on the lexical semantics of the words in the sentence.
However, these lexical resources face a serious drawback: their development is time-consuming, costly and requires trained linguists that are aware of the lexical variation of a language. The task becomes even harder when only few other resources for the language exist and the possibilities for an automatic acquisition of data are rather restricted.
With research being mainly focused on European languages like English and German (Bender 2009), this resource sparseness is partly a problem for Urdu. Up to date, no lexical knowledge base for Urdu exists. We attempt to provide a basic lexical resource that can be used by various natural language processing tools and contributes to closing the resource gap for Urdu.
One of these tools is the Urdu ParGram grammar (Bögel et al. 2007, 2009), based within the formalism of Lexical Functional Grammar (LFG) (Bresnan 2000, Dalrymple 2001). In addition to providing a deep syntactic analysis of the language, a semantic representation provides an even more abstract analysis of the sentence, using the syntactic analysis. In order to develop the semantic representation further, we require the availability of a lexical knowledge base for Urdu words.
In this paper, we report on a first approach to develop an Urdu WordNet on the basis of the already existing Hindi WordNet (Bhattacharyya 2010), using carefully designed transliterators. This provides us with a basic lexical knowledge base for Urdu nouns, verbs, adjectives and adverbs, which can serve as the starting point for extensive further refinement and completion.

2. Concepts

2.1. Hindi WordNet

Inspired in methodology and architecture by the English WordNet (Fellbaum 1998), Hindi WordNet (Bhattacharyya 2010) provides lexical information on Hindi words. The "net" character stems from the methodology by which words are grouped in the resource: nouns, verbs, adjectives and adverbs have separate semantic nets where they are grouped according to their meaning similarity. It is a resource that encodes more abstract information on the words of a language. For instance, the verb *rO* 'cry' shares a synset with the verb *ANsU bahA* 'tear emission', showing that they are semantically related. These synsets are further specified by a short description on the relation of the words contained in the synset and an examplary use of a word of the synset (gloss).
In the knowledge base, the synsets are put in semantic relations to one another via relations such as *IS-A* (hypernymy/hyponymy) and *PART-OF* (meronymy/holonymy). In the case of the nouns *gHar* 'home/house' and *makAn* 'house', the synset is called *Physical Place* with the hypernym of *Place* (more general synset) and no hyponym synset (no more specific concept). In the case of the nouns *ghar* 'house' and *kamrA* 'room', the relation is more of a meronymy or holonymy type, with *kamrA* being the meronym of *gHar*, i.e. 'room' being a part of 'house'. In total, Hindi WordNet has 16 of these semantic relations.
This results in a tree structure whereby traversing the tree from top to bottom, synsets get more specific. See Figure 1 for a schematic view of the knowledge base

and Figure 2 for the actual user interface of Hindi WordNet.

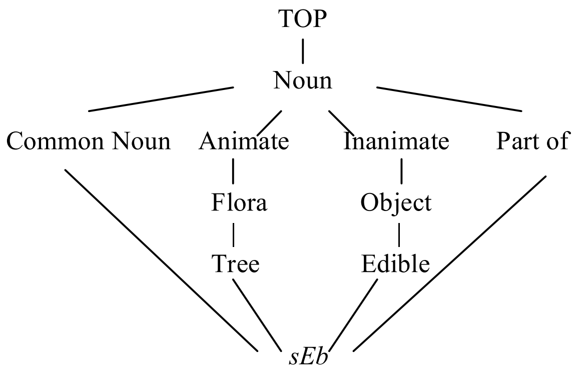
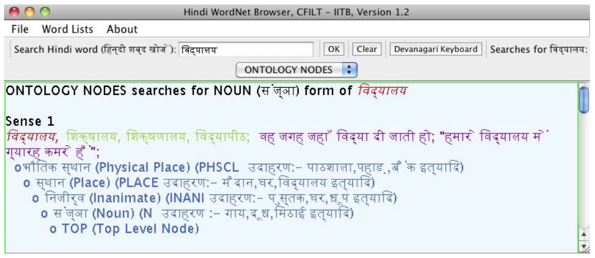


Figure 1. Schematic View of *sEb* in Hindi WordNet



Search Hindi word (हिन्दी शब्द खोजें): विद्यालय

Searches for विद्यालय:

ONTOLOGY NODES searches for NOUN (संज्ञा) form of विद्यालय

Sense 1
विद्यालय, शिक्षालय, शिक्षणालय, विद्यापीठ; वह जगह जहाँ विद्या दी जाती हो; "हमारे विद्यालय में ग्यारह कमरे हैं";
o भौतिक स्थान (Physical Place) (PHSCL उदाहरण:– पाठशाला,पहाड़,बैंक इत्यादि)
 o स्थान (Place) (PLACE उदाहरण:– मैदान,घर,विद्यालय इत्यादि)
  o निर्जीव (Inanimate) (INANI उदाहरण:– पुस्तक,घर,धूप इत्यादि)
   o संज्ञा (Noun) (N उदाहरण :– गाय,दूध,मिठाई इत्यादि)
    o TOP (Top Level Node)

Figure 2. User Interface of Hindi WordNet

The release that we use in this paper is Hindi Wordnet 1.0. It contains 28.687 synsets and a total number of 63.800 lexical items (Hindi WordNet Documentation, 2010).

2.2. Hindi and Urdu

2.2.1 Two Different Scripts, One Structure

Urdu is structurally almost identical to Hindi. One major difference is that the vocabulary of Urdu is more influenced by Persian and Arabic, whereas Hindi bears more Sanskrit influence. Another difference lies within the writing systems of the two languages. Where Urdu is written in modified Perso-Arabic script, Hindi is written in Devanagari, a phonetic-based script that was originally used for Sanskrit. An example for *kitAb* 'book' with Urdu script (left) and Devanagari (right) is given in Figure 3.

كتاب   किताब

Figure 3. Urdu and Hindi script for *kitAb*

Although there is a major difference with regard to the script, in general, an Urdu speaker is able to understand everyday Hindi and vice versa. This provides the justification for basing our first attempt to create a lexical resource for Urdu on Hindi WordNet. To overcome the scriptural differences, we employ carefully built transliterators that are discussed in section 2.2.2.
Using this methodology, we cannot create a complete lexical resource, but we can nevertheless provide a starting point for further efforts and make use of already existing lexical semantic information on Indo-Aryan languages.

2.2.2 Bi-directional Transliterators

The careful construction of two bi-directional transliterators allows us to build a first basic Urdu lexical resource with the help of Hindi WordNet. The transliterators are finite-state transducers using XFST (Beesley and Karttunen, 2003), which traverse each character of a word and produce a transducer with an upper side (Urdu script or Devanagari) and the corresponding Roman character on the lower side. Multiple Roman words for one Urdu word originate from the underspecification of short vowels in the Arabic/Urdu script. For the solution of this issue, see section 3.3. For the transducer output of *kitAb* 'book', see below.

upper side:	كتاب
lower side:	kattAb
	katAb
	kittAb
	kitAb
	kuttAb
	kutAb

The use of XFST transducers has one important advantage. A carefully designed Urdu to Roman transducer can also be used in reverse as a Roman to Urdu transliterator, i.e. it can generate Urdu script from Roman script. The same holds for the Hindi to Roman transliterator. Putting these transliterators in a sequence generates an Urdu to Hindi transliterator, which can also be used in reverse.
The Roman script that both transliterators either use or generate is based on the Roman transliteration scheme as presented in Malik et al. (2010). The scheme covers all special characters of Urdu and Hindi.
Earlier work on transliteration from Urdu to Hindi has been done by Malik (2006). However, Jawaid and Ahmed (2009) pointed out some difficulties with this approach. One problem is the fact that in some cases,

multiple acceptable spellings for one Hindi word exist. In particular, these words contain borrowed characters like *za*, *fa*, *xa* and *qa*. Concerning the orthography, Hindi words can have either the borrowed characters or their native Hindi equivalent with a similar sound. For instance, the word *zamIn* 'earth, ground', borrowed from Persian, can also be written as *jamIn*, as the non-native sound *za* can be replaced by the equivalent and similar native Hindi sound *ja*.

There are also other issues related to the spelling. For example, Urdu has the character *aen* that is represented as *a2* in the transliteration scheme used here (Malik et al. 2010). This character is written in Urdu words, but is usually not pronounced in the middle or at the end of the word. In contrast, it is never written in the Hindi spelling of these words. Therefore, the transliterators need to be designed in a way that an Urdu word with a middle or final *aen* can be matched with an equivalent word of Hindi that does not have it.

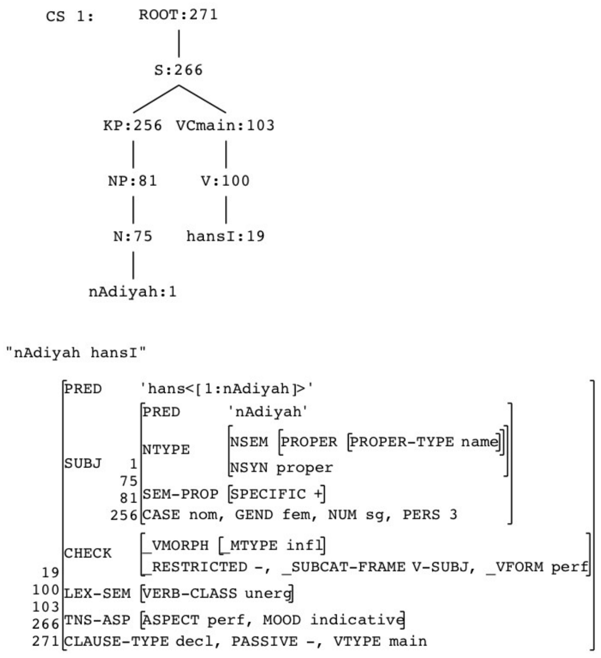
Another problem of Urdu to Hindi transliteration is the existence of different graphemes for the same sound character in Urdu script. For example, Urdu has three characters for the sound *sa*. Those characters are represented as *s*, *s1* and *s2* in the transliteration scheme used in this paper. Hindi only has one *sa* character. Therefore, if a Hindi word cotains an *s* sound, then the Hindi to Roman transliterator generates all words with the variants for *s*, i.e. *s*, *s1* and *s2*, again supporting the correct matching of Hindi script words with Urdu script words. The same principle works for different Hindi graphemes that correspond to one Urdu grapheme (e.g. *za*/*ja* used in *zamIn*/*jamIn* discussed above).

Combining the Urdu to Roman transliterator with the Roman to Hindi transliterator mostly overcomes the issues that are mentioned above[1].

2.3. The Urdu ParGram Grammar

The Urdu ParGram grammar (Butt and King 2007, Bögel et al. 2007, 2009), developed at the University of Konstanz, is part of a worldwide grammar development project that attempts to build large-scale, robust and parallel grammars (ParGram) for various languages, among them English, German, French, Norwegian, Turkish, Indonesian and Urdu, all based on common linguistic principles and a shared technology (Butt et al. 2002).

The common linguistic principles are based within the syntax theory of Lexical Functional Grammar (LFG) (Bresnan 2000, Dalrymple 2001) with its two-fold syntactic analysis. On the one hand, a tree structure (c-structure) analyzes the surface word order and constituency relations, whereas on the other hand, the f-structure (functional structure) abstracts away from the surface arrangement of words and assigns grammatical relations to the arguments of a sentence. See Figure 3 for c- and f-structure of the sentence *nAdiyah hansI* (Nadya laughed) as an example.



```
"nAdiyah hansI"
    [PRED      'hans<[1:nAdiyah]>'                                        ]
    [          [PRED      'nAdiyah'                             ]         ]
    [          [          [NSEM [PROPER [PROPER-TYPE name]]]    ]         ]
    [          [NTYPE     [NSYN proper                     ]    ]         ]
    [SUBJ    1 [                                                ]         ]
    [       75 [                                                ]         ]
    [       81 [SEM-PROP [SPECIFIC +]                           ]         ]
    [      256 [CASE nom, GEND fem, NUM sg, PERS 3              ]         ]
    [          [_VMORPH [_MTYPE infl]                                    ]]
    [CHECK     [_RESTRICTED -, _SUBCAT-FRAME V-SUBJ, _VFORM perf         ]]
  19[                                                                     ]
 100[LEX-SEM [VERB-CLASS unerg]                                           ]
 103[                                                                     ]
 266[TNS-ASP [ASPECT perf, MOOD indicative]                               ]
 271[CLAUSE-TYPE decl, PASSIVE -, VTYPE main                              ]
```

Figure 3. C- and f-structure for *nAdiyah hansI*

The shared technology is provided by XLE, developed at Palo Alto Research Center (PARC), an efficient, rule-based grammar development platform, with cutting-edge algorithms for parsing and generating Lexical Functional Grammars (Crouch et al. 2009).

Only recently, ParGram has extended its parallelism efforts beyond the syntactic analysis to a parallelism of semantic representations. The technology for doing so is also provided by PARC with an extension of the XLE parser, namely the XFR rewrite system (Crouch and King, 2006). Based on information coming from the LFG syntactic analysis, the XFR rewrite system allows, among other things, for a further abstraction from the surface structure by matching open-class words to concepts as well as assigning thematic roles to the arguments of a sentence. The assignment cannot be done manually but has to be done by accessing a lexical knowledge base that contains this information.

Having a semantic representation with conceptual information on the words allows for reasoning, which

[1] We have just presented an overview of the transliterators. The complete description about their design is beyond the scope of this paper.

is done in the Abstract Knowledge Representation (Bobrow et al, 2007).

3. The Urdu WordNet pipeline

Our aim is to create a basic lexical knowledge base for Urdu words. Due to structural similarities between Urdu and Hindi, we can justify to base our Urdu WordNet on the already existing lexical resource of Hindi WordNet.
The pipeline to generate a basic Urdu WordNet can be used in two different ways, with the basic architecture of both of these applications being the same. On the one hand, the WordNet output can be provided at runtime for a given word or a list of words. On the other hand, one can also use the pipeline to create a database by processing a larger list of Urdu words. This database output can then be post-processed to make it compatible with other applications, such as the semantic representation of the Urdu ParGram grammar. A schematic architecture of the Urdu WordNet pipeline that makes use of Hindi WordNet is shown in Figure 4.

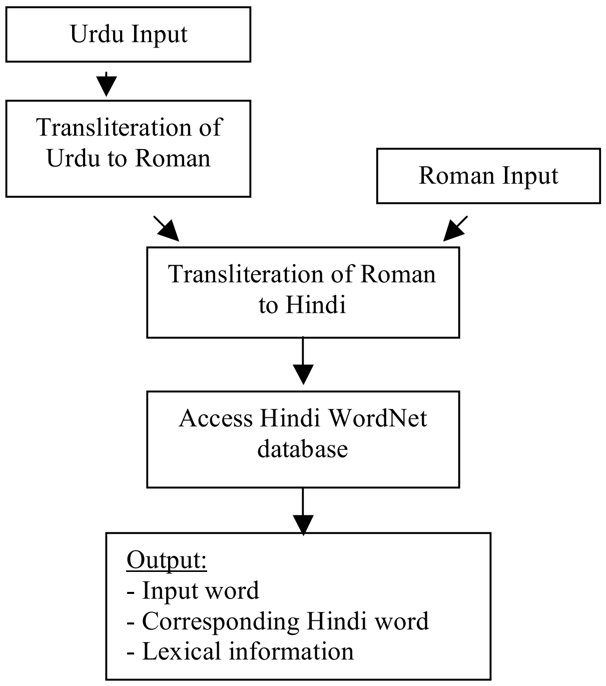

Urdu Input
Transliteration of Urdu to Roman
Roman Input
Transliteration of Roman to Hindi
Access Hindi WordNet database
Output:
- Input word
- Corresponding Hindi word
- Lexical information

Figure 4. Pipeline model for Urdu WordNet

3.1. Input

The Urdu WordNet pipeline accepts input in Urdu or Roman script. The reason for Urdu script input is obvious in that it allows Urdu speakers to gain information on the lexical semantics of Urdu words. It is also useful for other natural language processing applications that parse Urdu script.
Beside the Urdu script, we also allow for Roman script input. This is of particular interest to non-native speakers of Urdu who are unfamiliar with the Arabic script. In addition, the syntactic output of the Urdu ParGram grammar contains the words in Roman and as this output is used for the semantic representation, which in turn looks up the words in the lexical resource, the Roman script version also has to be included.

3.3. Urdu to Roman transliterator

The design principles of the Urdu to Roman transliterator have been introduced in section 2.4. The transliterator converts Urdu input to the equivalent Roman word, following the guidelines of the transliteration scheme presented in Malik et al. (2010). As Urdu text is generally underspecified for short vowels, the transliterator generates all possible combinations of the unwritten short vowels, in order not to loose any information.
As our goal lies in transliterating Urdu words to Hindi words that can then be looked up in Hindi WordNet, all possible forms (Roman transliterated words) of the Urdu to Roman transliterator are passed on to the Roman to Hindi transliterator.

3.4. Roman to Hindi transliterator

The Roman to Hindi transliterator converts Roman words following the given transliteration scheme into Devanagari script. Naturally, the conversion results in the loss of some information, as the same Urdu sounds that have different Urdu characters in the Roman transliteration are now mapped onto a single Hindi character. This results in some potential limitations that are discussed in section 6.
In some cases, the output of the Urdu to Roman transliterator, which serves as the input to this transliterator, is ambiguous (as can be seen for *kitAb*). We get the words in Hindi for all Roman input words and then match them with a list of valid Hindi words, filtering out the wrong Roman words.

3.5. Accessing Hindi WordNet

Hindi WordNet as an open-source tool provides an API (Application Programming Interface) that allows other programs to access its database, making use of the linguistic information enocded in it.

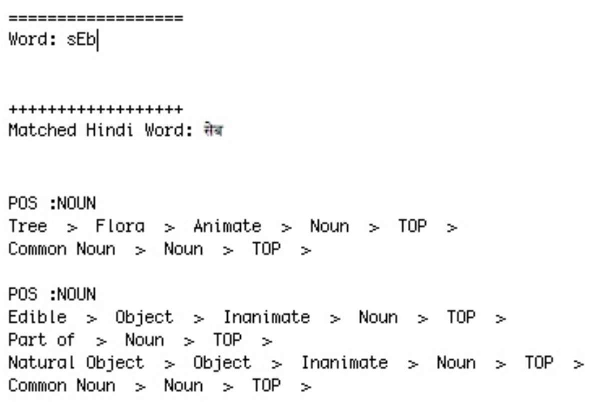
As the pipeline in Figure 4 shows, the input word or the list of input words are eventually transliterated into Hindi. They are then looked up in Hindi WordNet by making use of the API. This step also serves as the disambiguation step between different versions of Hindi words for one Urdu input, which arises due to the underspecification of short vowels and the multiple scriptual variants in Roman for one Urdu word.
Looking up a word in Hindi WordNet results in information in Roman and Devanagri script. The English labels for the linguistic terms like ‘Noun’ as part of speech and the information on the ontological location, e.g. the synset name *Natural Object*, are in Roman script. The information in Devanagri consists of the Hindi word itself, its synonyms and the gloss.
Due to our focus on using Urdu WordNet for the semantic representation of the Urdu ParGram grammar, we restrict ourselves to extracting information on the ontological location, in particular the hypernym relation, and the part of speech, without considering the gloss with synset explanation and exemplary sentence. Hence, the application reduces the information from Hindi WordNet to the components that can be used for the semantic representation.
Adding this information will turn the basic lexical resource into a full-fledged knowledge base. This step will require extensive manual labour because it requires translation apart from transliteration, going far beyond what can be done by the approach presented in this paper.

3.6. Run-time Application of the Urdu WordNet pipeline

The run-time output of the Urdu WordNet pipeline contains the word in Urdu or Roman script, depending on the input mode, and the Hindi word together with its part of speech (POS) and the information relevant for the semantic representation. In cases where a word has more than one part of speech or can be found in multiple synsets, all variants are provided in the output. Figure 5 shows the output for the word *sEb 'apple'*, a word that is contained in a number of synsets, depending on the context that *sEb* is used in.

3.7. Generating a Database with the Urdu WordNet pipeline

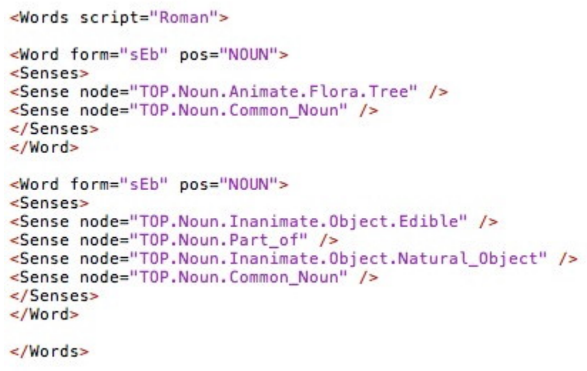
Using the pipeline that is shown in Figure 4, it is possible to generate an Urdu WordNet database, which is, however, by no means a complete lexical resource for Urdu. To generate the database, a list of Urdu words (in either Urdu or Roman) is passed through the transliterator on to Hindi WordNet. In order to make



```
==================
Word: sEb


++++++++++++++++++
Matched Hindi Word: सेब


POS :NOUN
Tree  >  Flora  >  Animate  >  Noun  >  TOP  >
Common Noun  >  Noun  >  TOP  >

POS :NOUN
Edible  >  Object  >  Inanimate  >  Noun  >  TOP  >
Part of  >  Noun  >  TOP  >
Natural Object  >  Object  >  Inanimate  >  Noun  >  TOP  >
Common Noun  >  Noun  >  TOP  >
```

Figure 5. Output of Urdu WordNet for *sEb* ‘apple’

the database accessible to other systems and provide a platform independent tool, it is formatted in XML (Extensible Markup Language).
This enables any kind of application to make use of the Urdu WordNet database. It also allows for manual post-editing and the addition of Urdu words that are not present in Hindi WordNet. See Figure 6 for the XML output for *sEb*.



```
<Words script="Roman">

<Word form="sEb" pos="NOUN">
<Senses>
<Sense node="TOP.Noun.Animate.Flora.Tree" />
<Sense node="TOP.Noun.Common_Noun" />
</Senses>
</Word>

<Word form="sEb" pos="NOUN">
<Senses>
<Sense node="TOP.Noun.Inanimate.Object.Edible" />
<Sense node="TOP.Noun.Part_of" />
<Sense node="TOP.Noun.Inanimate.Object.Natural_Object" />
<Sense node="TOP.Noun.Common_Noun" />
</Senses>
</Word>

</Words>
```

Figure 6. XML output of Urdu WordNet for *sEb* ‘apple’

Both pipelines, the run-time application and the generation of the database, can be used with any version of Hindi WordNet that is released, given that the API is kept the same. This makes our system very flexible and handy for future usage.

4. Using Urdu WordNet for an Urdu Semantic Representation

Based on the syntactic information with c- and f-structure, the Urdu ParGram Grammar allows for an additional module that converts the syntactic information into a semantic representation, using XFR rewrite rules (Crouch and King, 2006). The XFR rewrite system was originally developed for the English ParGram grammar, one of the most sophisticated LFG grammars which is employed in the search engine Powerset, part of Microsoft's Bing search. Building on the semantic representation, there is also an additional level called the Abstract Knowledge Representation (Bobrow et al. 2007), where XFR rules are used for reasoning. This is not currently used for the Urdu ParGram grammar, but will be employed in the future.

The XFR semantic rules by no means describe a theory of the syntax-semantics interface, but they can, from a computational standpoint, robustly produce semantic representations based on syntactic information. On the one hand, the XFR semantics gives a flat representation of the sentence's predicate argument structure and the semantic contexts in which those predications hold. On the other hand, XFR semantic rules replace words with concepts and grammatical functions, i.e. SUBJ and OBJ, with thematic roles by using external lexical resources. This results in an even more abstract level of sentence analysis [2].

The XFR semantic representation for Urdu is still in an infant state, at the moment simply rewriting the f-structure information into a structure that is more targeted for a semantic interpretation of the sentence. In the future, when Urdu WordNet is integrated into the XFR semantic representation, we will have lexical information available in the semantic representation. As an example, consider the XFR rewrite rule, which manually replaces the SUBJ of the sentence *nAdiyah hansI* with the thematic role of Agent.

```
PRED(%1,hans), SUBJ}(%1,%subj), -OBJ(%1,%obj)
==>
word(%1, hans, verb), role(Agent, %1, %subj).
```

XFR rules are based on the output of f-structures. The rule given above applies, whenever the predicate of a sentence is *hans* and there is a SUBJ in the sentence, but no OBJ. It assigns the part of speech 'verb' to *hans* and the thematic role of Agent to the subject, dismissing the information on grammatical function. These rules cannot be hard-coded into the system, but have to be dealt with in an automatic way.

Considering the f-structure of *nAdiyah hansI* in Figure 3, a first semantic representation for the sentence can be seen in Figure 7. The context head is the predicate of the sentence, *hans* 'laugh', and SUBJ of the sentence has been converted to the Agent. In the future, the concepts of the words in the sentence will be included, e.g. the information that *hans* is an grouped under *Expression*.

```
context_head(t,hans:9),
in_context(t,cardinality(nAdiyah:1,sg)),
in_context(t,proper_name(nAdiyah:1,name,nAdiyah)),
in_context(t,role('Agent',hans:9),nAdiyah:1),
original_fsattr('SUBJ',hans:9,nAdiyah:1),
original_fsattr(gender,nAdiyah:1,fem),
original_fsattr(subcat,hans:9,'V-SUBJ'),
```

Figure 7. Semantic representation for *nAdiyah hansI*

By turning words into more abstract concepts and performing other semantic conversions as well, we provide the ground for reasoning in the Abstract Knowledge Representation.

One application area for reasoning is marking sentences like *#sEb hansA* as semantically ill-formed or potentially idiomatic by employing selectional restrictions that restrict the context in which a word can occur. In the case of *sEb hansA*, with the concepts *Tree/Edible* for *sEb* and *Expression* for *hansA*, the occurrence of the verb would be restricted to subjects that have the hypernym *Agentive*, which is not the case for *sEb*. The XFR semantics produces a representation nevertheless, reasoning would then judge the sentence as semantically being ill-formed.

The Urdu ParGram grammar is only one example as to how a lexical resource for Urdu, although still restricted in completeness, can be used by a natural language processing tool, with many more applications to follow.

5. Evaluation

The evaluation in this paper is two-fold: at first, we test the transliterators, followed by a preliminary evaluation of our basic Urdu WordNet pipeline.

Although our transliterators are working robustly, the matching of Urdu words into either Roman or Arabic script with Hindi words in Devanagri can be wrong in some cases. We have not performed a large-scale evaluation, however using corpora that are also used for the Urdu ParGram grammar show that the chance of a mismatch is rather low. Still, potential examples can be cited. For instance, Urdu has two similar sounding words *Saa2ar* (verse) and *Ser* (lion). There is a single entry (with multiple senses) for both of these

[2] Due to an immediate need for lexical resources, we developed the basic Urdu WordNet pipeline presented in this paper, bringing together existing resources such as transliterators and Hindi WordNet. The interface between lexical resource and semantic representation has already been developed for the English ParGram grammar that uses the lexical information contained in the English WordNet (Fellbaum, 1998) and VerbNet (Kipper et al. 2008).

words in Hindi WordNet. In Urdu WordNet, one needs to manually distinguish between the two senses and associate each with the corresponding Urdu word. These cases, in comparison to the words that can be correctly looked up in Hindi WordNet, are rather rare, nevertheless they have to be accounted for.

Evaluating lexical resources in general and the basic Urdu WordNet pipeline in particular is complicated and time-consuming because it can hardly be done automatically. Nevertheless, we conducted two preliminary tests to get an idea about the coverage of the Urdu WordNet. For that, we used a list of Urdu simple verbs in Urdu compiled by Humayoun (2006), containing 781 verbs. However, the list does not include complex predicates that are more frequently used in Urdu. When this list is given as input to the Urdu WordNet pipeline, 575 verbs are found with matching entries in Hindi WordNet. We have not calculated the exact precision of these matched entries, however a manual check of the output suggests that most of the matched Hindi words and corresponding lexical resource entries are correct.

Another test evaluates a smaller number of words. The Urdu WordNet pipeline is tested on the predicates obtained from the children's story *piyAsA kavvA* (The Thirsty Crow). As the lexical resource will be used by the XFR semantic rules that have f-structures as input, we only concentrate on the main contentful predicates (corresponding to the grammatical functions) in the sentences. This results in a list of 27 unique root forms (in Roman script). These root forms are input to the Urdu WordNet pipeline.

The transliterator successfully transliterates all words into Hindi. When these 27 words are matched with Hindi WordNet, we find that 23 have corresponding entries in Hindi WordNet. The ontological information extracted for these matched entries match the intuition of an Urdu native speaker.

The unmatched words are *kavvA* 'crow', *piyAsA* 'thirsty', *daraxt* 'tree' and *dikHAI dE* 'become visible'. One of these, *daraxt* 'tree', is a loanword from Persian. Hindi speakers prefer the word *pER* 'tree'. The other three are native Urdu/Hindi words. Even such a small-scale experiment shows that we have to develop the Urdu WordNet further by entering lexical items that simply are not present in Hindi, using the architecture proposed in this paper as a starting point.

These preliminary evaluations point to some of the areas that have to dealt with in the future, in particular the inclusion of loan words that cannot be located by using a Hindi lexical resource.

6. Discussion and Future Work

This paper presents a first experiment with building a basic lexical knowledge base for Urdu, making use of existing resources in Indo-Aryan languages, namely transliterators and Hindi WordNet. We are well aware that the basic Urdu WordNet pipeline as it is presented here is by no means a full-fledged lexical resource for Urdu, but it leads towards an application that can be used by natural language processing applications like the one described above and also the average user who wants to know more about Urdu. A strength of our system lies in the availability of two scriptual versions, namely Roman and Urdu. This makes it usable for non-Urdu speakers who are not familiar with the Urdu script as well as computational applications, which work with the Roman script.

At the moment, the basic Urdu WordNet pipeline is still at an infant stage as the architecture fully relies on what Hindi WordNet provides and what the transliterators generate as output. Nevertheless it is worth experimenting with the similarity of Hindi and Urdu and see how tools can benefit from this fact.

Extending the Urdu WordNet pipeline involves further work in different areas, one being the implementation of complex predicates that are used more frequently than simple verbs in Urdu. Another aspect will be to focus on including Arabic and Persian loan words that do not exist in Hindi. This will not only be interesting for the resource itself, but also provide new insights from a theoretical point of view as to how the two languages are interwoven. By including the loanwords, Urdu WordNet will seriously contribute to the overall goal of developing an Indo WordNet.

Another area for future work is the inclusion of the gloss, i.e. the description of the synset and the example sentence. As this goes beyond transliteration and towards translation, the problem has not been tackled yet. This extension will be one of the main parts of our future work and we hope to benefit from extensions of Hindi WordNet with refined concepts and the removal of some inconsistencies.

Another strength is that, as long as the API of Hindi WordNet is not changed majorly, the Urdu WordNet pipeline can be run on any of its releases and achieve a refined Urdu WordNet.

7. Conclusion

In this paper, we reported on a first approach to developing a basic lexical resource for Urdu by extracting information contained in an existing Hindi WordNet. This is justified by the similarity in vocabulary between Hindi and Urdu. To overcome the scriptual difference between the two languages, two transliterators are employed. Extracting information from Hindi WordNet can be done for single words as well as a large list of words to create a database. The extracted information consists of the input word in Urdu or Roman script, the equivalent Hindi word, its part of speech and its ontological location. The gloss with the example sentence and the synset description is left out.

Despite the existence of Urdu specific words that cannot be found in Hindi, we can still use this basic Urdu resource for computational approaches such as the semantic representation of the Urdu ParGram grammar.

Evaluating the transliterators and the Urdu WordNet pipeline generates promising results, and although some issues, especially concerning the transliterators remain, both components work robustly.

The approach taken in this paper follows the general aim of creating parallel lexical resources for languages, in particular languages that are as similar as Urdu and Hindi. Therefore, our Urdu WordNet pipeline can serve as a starting point for the further development of a lexical resource for Urdu and can contribute to the availability of more tools for Indo-Aryan languages.

References

Kenneth Beesley and Lauri Karttunen. 2003. *Finite State Morphology*. CSLI Publications.

Bender, Emily M. 2009. Linguistically Naïve != Language Independent: Why NLP Needs Linguistic Typology. In *Proceedings of the EACL 2009 Workshop on the Interaction between Linguistics and Computational Linguistics: Virtuous, Vicious or Vacuous?* pp.26-32.

Bhattacharyya, Pushpak. 2010. IndoWordNet. In *Proceedings of LREC2010.*

Daniel G. Bobrow, Bob Cheslow, Cleo Condoravdi, Lauri Karttunen, Tracy Holloway King, Rowan Nairn, Valeria de Paiva, Charlotte Price, Annie Zaenen. 2007. PARC's Bridge and Question Answering System. *Proceedings of the Grammar Engineering Across Frameworks (GEAF07) Workshop*, pp. 46-66, CSLI Publications.

Tina Bögel, Miriam Butt, Annette Hautli, and Sebastian Sulger. 2007. Developing a Finite-State Morphological Anlayzer for Urdu and Hindi: Some Issues. In *Proceedings of FSMNLP07, Potsdam, Germany.*

Tina Bögel, Miriam Butt, Annette Hautli, and Sebastian Sulger. 2009. Urdu and the modular architecture of ParGram. In *Proceedings of CLT 2009.*

Joan Bresnan. 2000. *Lexical-Functional Syntax*. Blackwell, Oxford.

Miriam Butt and Tracy Holloway King (2007). Urdu in a parallel grammar development environment. In *Proceedings of LREC' 07.*

Miriam Butt, Helge Dyvik, Tracy Holloway King, Hiroshi Masuichi and Christian Rohrer. 2002. The Parallel Grammar Project. In *Proceedings of the COLING- 2002 Workshop on Grammar Engineering and Evaluation*, pages 1–7.

Dick Crouch and Tracy Holloway King. 2006. Semantics via f-structure rewriting. In *LFG06 Proceedings*. CSLI Online Publications.

Dick Crouch, Mary Dalrymple, Ron Kaplan, Tracy Holloway King, John Maxwell and Paula Newman. 2009. XLE documentation. http://www2.parc.com/isl/groups/nltt/xle/doc/.

Mary Dalrymple. 2001. *Lexical Functional Grammar*. Syntax and Semantics, Vol. 34. Academic Press.

Christine Fellbaum (eds). 1998. WordNet: An Electronic Lexical Database. The MIT Press.

Muhammad Humayoun. 2006. Urdu Morphology, Orthography and Lexicon Extraction. MSc Thesis, Department of Computing Science, Chalmers University of Technology.

Bushra Jawaid and Tafseer Ahmed. 2009. Hindi to Urdu Conversion: Beyond Simple Transliteration. In *Proceedings of CLT 2009.*

Karin Kipper, Anna Korhonen, Neville Ryant, Martha Palmer. 2008. A Large-scale Classification of English Verbs, Language Resources and Evaluation Journal, 42(1), pp. 21-40, Springer Netherland.

Abbas Malik. 2006. Hindi Urdu machine transliteration system. MSc Thesis, University of Paris 7.

Muhammad Kamran Malik, Tafseer Ahmed, Sebastian Sulger, Tina Bögel, Atif Gulzar, Ghulam Raza, Sarmad Hussain, and Miriam Butt. 2010. Transliterating Urdu for a Broad-coverage Urdu/Hindi LFG Grammar. In *Proceedings of LREC' 10.*

S. Rahman and Sarmad Hussain. 2003. Development of character based Urdu Nastaleeq font. *Asian Media and Communication Bulletin*, 33(2).