A Multilayered Urdu Treebank

Tafseer Ahmed[1], Toqeer Ehsan[2], Almas Ashraf[3], Mutee u Rahman[4], Sarmad Hussain[2], Miriam Butt[5]

[2]*Centre for Language Engineering*
Al-Khawarizmi Institute of Computer Science, UET, Lahore
firstname.lastname@kics.edu.pk

[3]*NEDUET, Karachi*
almasashraf@neduet.edu.pk

[4]*Isra University, Hyderabad*
mutee.rahman@isra.edu.pk

[5]*University of Konstanz, Konstanz, Germany*
miriam.butt@uni-konstanz.de

Abstract

The paper presents the design and construction of a multilayered phrase structure treebank. The treebank consists of three layers for phrases, grammatical functions and semantic roles. A small phrase tagset (consisting of 12 tags) is used as the primary label of the phrase. Phrase label is followed by grammatical function (mainly inspired by lexical functional grammar). It is followed by the semantic role label using propbank roles. 1,300 sentences from CLE Urdu Digest Corpus are annotated using the treebank guideline.

1. Introduction

Treebank is an important linguistic resource for syntax analysis of languages. Creating a treebank involves choosing the theoretical model, creating the annotation guidelines and then annotating the corpus. The annotated corpus is used to create syntactic parsers and other syntax analysis tools.

Urdu is an Indo-Aryan language spoken mainly in Pakistan and India [1]. Urdu and Hindi share common grammar. However, there are differences in script, vocabulary and phonology.

The current work is part of a bigger project introducing intonation in Urdu Text to Speech System. One goal of the project is creation of phrase structure parser for the system. For this reason, a phrase structure Urdu Treebank is planned. The treebank design introduces three different layers of annotation to model phrase structure (constituents), their grammatical functions and semantic roles. This paper presents a description of the treebank creation task.

In subsequent sections, Section 2 presents the important treebanks and major works for Urdu/Hindi treebanks. Section 3 compares different treebank design options to create our treebank. Section 4 describes the design principles and a brief description of different layers of the treebank. Conclusion and Future work is mentioned in Section 5.

2. Literature Review

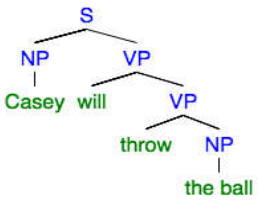
There are two major types of syntactic annotation: phrase structure and dependency structure. The phrase structure analysis of sentence was introduced by Chomsky [2]. The first major treebank, Penn Treebank has phrase structured annotation [3]. The Penn Treebank was inspiration for many treebanks for other languages. Penn Treebank (PTB) has around 25 phrase labels. Figure 1(a) shows a phrase structure of an English sentence annotated using PTB guidelines. Figure 1(b) has its representation in bracketed notation. The bracketed notation is in text format, so it can be processed by computer applications.

As different languages and different treebanks use different set of phrase labels in design, Han et. al. [4] introduced a common tagset after analyzing 25 different treebanks covering 21 languages. They introduced 9 universal phrase labels namely Noun Phrase (NP), Verb Phrase (VP), Adjectival Phrase (AJP), Adverbial Phrase (AVP), Prepositional Phrase (PP), Sentence (S), Conjunction Phrase (CONJP), Coordinated Phrase (COP) and others (X).

The other type of syntactic annotation is dependency structure. Its primary focus in not on the word order or constituency, but it deals with the syntactic relations between the words. A dependency structure along with corresponding phrase structure is presented in figure 1. The most important milestone is the introduction of universal dependencies [5]. There are more than 100 treebanks annotated using universal dependencies [6].

[1] The author was affiliated with *DHA Suffa University, Karachi* when this work was done.

a.



b. (S
(NP Casey)
(VP will (VP throw
(NP the ball))))

c.

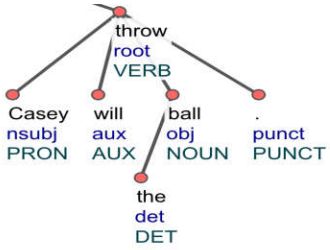


Figure 1 : (a) phrase structure, (b) its bracketed notation and (c) dependency structure for the English sentence *Casey will throw the ball*.

For Urdu (and Hindi), there was no freely available treebank at the start of this project. There were earlier works on computational grammar of Urdu (and Hindi). Urdu Pargram implements major parts of Urdu grammar using Lexical Functional Grammar (LFG) framework [7]. An important syntactic structure bank is Hindi Urdu Treebank (HUTB) [8] that has Urdu corpus annotated using Panini style dependencies. The dependency structure can be automatically converted to the phrase structure. An additional layer of dependencies based on LFG's f-structure is also proposed for HUTB [9].

Urdu.Kon.TB [10] is another Urdu treebank that uses a rich feature based pos tagset and a big phrase tagset. A parser was also developed using this treebank of 1400 sentences.

3. Comparison and Design Principles

The previous section presented some important treebanks generally for all languages and specifically for Urdu and Hindi. In this section, we compare the approaches used in these treebanks and find which approach is better for the design of our treebank.

The first question is whether the syntactic bank will have phrase structure or dependency structure representation. The dependency structures have become more popular due to the introduction of universal dependencies in language processing applications. However, our team have a bigger goal of creating a text to speech system and using syntactic information to predict prosody/intonation of the system.

We find that most of the work related to syntax-prosody interaction involves phrase structure models [11],[12] as phrase structure grammar is more commonly used by linguists. Hence, we decided to adopt the classical method of phrase structure treebank. After deciding to create phrase structure treebank, we analyzed the existing treebanks (described in previous section) on the basis of following criteria. This analysis recommends the design principles for our treebank.

1. How is the structure of the tree?
2. What is the granularity of the phrase label?
3. How additional information is encoded?

These criteria are discussed in the following subsections.

3.1 Structure of Tree

There are many ways to construct parse tree corresponding to a given sentence. There are linguistic theories such as X-bar theory that ask for strictly binary trees. HUTB has binary trees because they are inspired by X-bar style work.

Other theories and traditions prefer flatter trees having head and all its dependents on the same level. Penn Treebank and Urdu Pargram have flat structures having head and all the adjacent modifiers on the same level of tree.

The simplicity of annotation scheme to facilitate the annotator is one of our primary design policies. Hence, we prefer the flat structures as they are easy to annotate and many members of treebank community use it for its simplicity.

3.2 Granularity

The second issue is the granularity of the tags. Some schemes e.g. Penn Treebank use more phrase tags (27 for PTB). Multiple tags for the same/similar phrases are used to highlight the difference in structure and/or words used in the phrase. In PTB, most of the phrases have two versions, one for the general usage and the other for the phrases having wh-word. For example, "my books" is an NP and "whose books" or "how many books" are WHNP.

The phrase tagset of Urdu.Kon.TB, in this regard, is inspired by Penn Treebank. It has 26 main tags. Some of these tags have function subtags. The tags NPQ, ADJPQ, QWP and SQ etc, are the tags for question sentences/phrases. Similarly, there are four tags

corresponding to the verb complex i.e. VCmain, VP, VPI and VCP.

The other schemes e.g. Multilingual Tagset, HUTB and Urdu Pargram does not differentiate the phrases on the basis of their internal structure or usage. For this reason, PTB has 27 tags and Multilingual Tagset has 12 tags. So, following our design policy of choosing simpler annotation scheme, we prefer smaller tagset approach, and considered Multilingual Tagset as the starting point.

HUTB also uses a small tagset. We did not use some of its tags and the reasons are explained in the discussion of our tagset in section 4.

3.3 Additional Information

Penn Treebank introduced function tags that concatenate additional information e.g. grammatical role etc. to the phrase labels. The function part is attached with the main label by a hyphen. See the example.

```
1. (S (NP-SBJ He)
    (VP left
    (NP-TMP yesterday)))
```

HUTB uses -pred function tag for modeling small clause. So NP, AP, degP and NumP has -pred suffix e.g. NP-pred. Similarly, Urdu.Kon.TB uses function tags to encode case information of the phrase head.

Our treebank used the concept of function tag in a systematic way (as depicted in example 2 in section 4) to represent different layers of syntactic and semantic information.

4. Urdu Treebank Design

The basic design principles of Urdu Treebank were:

(a) a phrase structure bank, as it helps in syntax-intonation interface. However, a phrase to dependency convertor is part of the future work.
(b) smaller tagsets, if possible, to help annotators. The idea of smaller tagset is in line with the universal phrase labels [4] and propbank [13]
(c) a modular design, so different applications may retrieve the required annotation information from the treebank. The encoded semantic roles are not for immediate use. The parser will ignore this layer, however they can be used in the semantic parser in future.

[2] This paper presents the design of the treebank and pilot annotation of 1300 sentences. The further work is mentioned, but that is not in the scope or not a contribution of this paper.

The Urdu Treebank consists of three layers: phrase labels, grammatical function and semantic role. The text is annotated in the form of XML representation. In this paper, we show the equivalent bracketed notation that is widely comprehensible. The labels of each bracketed phrase encode all the three layers of the representation. The labels of each layer are separated by a hyphen. Following is the template of annotation scheme.

```
2. (PhraseLabel-GrammaticalFunction-
   Semanti cRole-ChunkId
         word1/pos1   word2/pos2   ….
         wordn/posn)
```

The chunkId part is explained in 4.2.9. An example from English using our representation scheme is following

```
3. (S (NP-SUBJ-Agent Casey)
      (VP will
      (VP throw
      (NP-OBJ-Theme the ball)     )))
```

Following section discusses the details of the corpus and the layers of annotation.

4.1 POS Tagged Corpus

We used CLE POS tagged Urdu Digest Corpus [14] for syntactic annotation. The corpus consists of sentences having unique ids. The corpus was manually edited to deal the common segmentation problems of Urdu text The token are separated by space and multiwords have Zero Width Non Joiner (ZWNJ) character between its components. The corpus was tagged by using CLE POS tagset [15].

The tasks of annotation was divided in three steps. The first step is of pilot annotation for testing and revising the annotation guidelines. In this step, 200 sentences were annotated. Annotation scheme and guidelines are revised according to the feedback of the annotators. In second step 1100 more sentences were annotated. In the third step, the whole of the remaining corpus (around 6,000 sentences) will be annotated[2].

4.2 Phrase Labels

The first layer of treebank consists of phrase labels. We are inspired by the small tagset introduced by Han et. Al [4]. At the design phase, a list of 10 phrase labels are identified. During the pilot annotation phase two

more phrase labels are added to the list. The description of phrase labels are given below.

4.2.1 S and SBAR. The phrase label S is used for main/matrix/independent sentences and clauses. SBAR is used for subordinate clause/sentence. Penn Treebank has SBAR, SINV and SQ for different types (and word order) of clauses, however we do not use these labels that are designed for English syntax. The main reason for SBAR is that the POS tagset differentiate between coordinating and subordinating conjunctions. So we want to keep this distinction in all the layers (if possible). Some examples of S and SBAR are following.

```
4. (S vuh[he] chAhtA[want] he[is]
      (SBAR kah[that]
      (S sEb[apple] kHAyE[eat])))
      'He wants to eat apple.'

5. (NP laRkA[boy] (SBAR jo[who]
      (S sEb[apple] kHA[eat]
      rahA[progressive] he[is]) ) )
      'the boy who wants to eat apple'
```

4.2.2 VC (Verb Complex). The phrase label VC is used for verbs, auxiliaries, light verbs and particles/adverbs of the verbs. The object is not part of verbal complex as we followed the analysis used in Urdu Pargram [7].

```
6. (S (NP vuh[he]) (NP kitAb[book])
      (VC parH[read] hi[intensifier]
      nahIN[not]   rahI[progressive]
      hE[is]) )
      'She is not reading the book.'
```

Urdu has Noun+LightVerb and Adjective+LightVerb complex predicates [16] e.g. *Yad* 'memory.noun' *kar* 'do.verb' for 'memorize' and *sAf* 'clean.adj' *kar* 'do.verb' for *clean*. In our annotation scheme, the noun or adjective is not the part of VC as these act syntactically as noun or adjective phrases.

```
7. (S (NP vuh[he]) (NP sabaq[noun]) (NP
   yAd[memory]      (VC      kar[do]
   rahA[progressive] tHA[was]) )
      'He was memorizing the lesson.'
```

4.2.3 Noun Phrase (NP). Noun Phrase has noun and its modifiers, specifiers and intensifiers. The CLE POS tagset considers the adverbials like *andar* 'inside' and *Aj* 'today' as a noun because these are syntactically similar to nouns. We use the same argument to label the following as a noun phrase. Following are some examples of NP.

```
8. (S (NP vuh[he) bHI[too] )
(NP ye[this] acHcHI[good]kitAb[book])
(VC parHtA[read] hE[is] ) )
      'He also reads this good book.'

9. (S (NP tum[you] (NP
kal[yesterday]) (NP andar[inside])
(VC AyE[come]  tHE[was]) )
      'You came inside yesterday.'
```

4.2.4 AdjP, QP, DMP and ValaP. Adjective Phrase (AdjP), Quantifier Phrase (QP), Demonstrative Phrase (DemP) and Vala Phrase (ValaP) are usually (not always) embedded inside the noun phrase (NP).

One of the goals of annotation guideline is to make speed of annotation faster, if possible, without compromising on the quality of representation/modeling. Hence, it is decided that if the phrase consists of a single word (e.g. an adjective only) inside the noun phrase then the annotator will not enclose the word with the phrase brackets and phrase label. In example 8, the adjective acHcHI is not enclosed by AdjP. However, if the adjective has modifier or intensifier then AdjP will be created. For example:

```
10. (NP
      (AdjP buhat [very] acHCHI[good]
      sI[particle]) kitAb[book] )
        'very good book'
```

The similar guideline applies for QP, DemP and ValaP used inside the NP. If any of these phrases appear at clause level i.e. directly inside S (or SBAR) then we always put the bracket even around the single word. See the following example.

```
11. (S (NP kitAb[book])
       (ADJP acHcHI[good]) hE[is])
        'Book is good.'
```

The labels DemP and ValaP were not part of the set of phrase labels listed in the design phase. However, the pilot annotation provides the cases for which these labels are required. Like other pos categories, demonstrative can also have particles like intensifiers and focus particles. Hence we use the general rule that if the category word has some other word attached to it as a modifier or particle then the whole sequence is enclosed in the phrase label. See the following example:

```
12. (NP
     (DMP   kOI[any]   bHI[intensifier])
            bAt[matter.noun] )
        'any matter'
```

The ValaP phrase is used in the constructions having the pos vAlA (roughly translated as 'one'). See the examples:

13. **(NP (ValaP (NP** tasvIr vAlI) kitAb[book]))
'books with/having pictures'

It must be noted that we introduced ValaP instead of VP, DMP instead of DP and VC (verbal complex) instead of VP as the later ones have their formal definition and usage in different syntactic theories and nomenclatures. Hence, we used longer or different names for the new labels introduced in our design.

4.2.5 Pre-and-Postpositional Phrases. Urdu has postpositions (that follow noun phrase). There are some borrowed positions from Arabic and Persian [17] that are rarely used in Urdu. Hence, the phrase labels PP (postpositional phrase) and PrP (prepositional phrase) are used in the treebank guideline. The examples are:

14. (**PP** tum[you] nE[*ergative*])
'You'

15. (**PP** gHar[home] tak[till])
'till home'

16. (**PrP** sivAE (**NP** mErE))
'except me'

It must be noted that neither the pos tagset nor the phrase labels distinguish between case markers and postpositions as distinguished in Urdu Pargram. This is done for the sake of simplicity (at phrase layer) and similar syntax. The functional difference between *nE* and *tak* is modelled through the grammatical function layer.

As described earlier, the adverbial nouns like *andar* 'inside' and *Upar* 'above' etc. are the head of the noun phrase as in the following example:

17. (**NP**(**PP** gHar[house] kE[of]) andar[inside]))
'inside the house'

4.2.6 Adverbial Phrase. The adverbial phrase has adverbs as the head word. For example:

18. (**S** vuh[he] **(ADVP** bA_AsAnI[easy]) (**VC** AyA[came]))
'He came easily.'

In Urdu, adverbial function is usually expressed by a prepositional phrase or noun phrase. For example, the following sentence has a PP. However, both (18) and (19) will the same grammatical function in the second layer of annotation.

19. **(S** vuh[he] **(PP (NP (**AsAnI[easy] sE[with])) (**VC** AyA[came]))
'He came easily.'

4.2.7 X. The phrase label X is used for fragments that cannot have a phrase label from the above list.

4.2.8 Conjunction. The conjunction is modelled by enclosing the components into a parent phrase label. For example,

20. (**NP (NP** sEb[apple]) yA **(NP** Am[mango]))
'apple and mango'

We do not introduce any phrase label e.g. conjunction phrase for enclosing the conjuncted components.

4.2.9 Discontinuous Phrases. We find examples of discontinuous phrases during the pilot annotation phase. The discontinuous NP in Urdu was earlier discussed in [17]. Consider the following example.

21. (**S** (**NP** vuh[he]) (**VC#1** rO[cry] (ADVP kiyoN[why]) (**VC#1** rahA hay))
'Why is he crying ?'

In this example, the VC is not contiguous. We assign the same chunk id to all the components of discontinuous phrases.

4.3 Grammatical Function

The second layer of treebank is of grammatical function. As depicted in (2), the grammatical function follows the phrase label separated by a hyphen. The set of grammatical functions is inspired primarily by lexical functional grammar. Following is a brief introduction of grammatical functions.

4.3.1 Subject and Object. The syntactic subject and object have the corresponding grammatical functions. See the following example.

22. (**S** (**NP-SUBJ** laRkI[girl])
(**NP-OBJ** kitAb[book[)
(**VC** paRHtI[read] hE[is]))
'The girl reads book.'

Universal Dependencies have three different labels for subject. nsubj (nominal subject), csubj (clausal subject) and npaassubj (nominal subject of passive construction). However, we do not follow this scheme because the information about nominal (noun phrase) vs clause is already represented through phrase label.

4.3.2 Oblique (OBL). The oblique grammatical function (OBL) is used with those compulsory arguments that are not the syntactic subject or object e.g. the source/goal of the motion verbs, non-canonical second argument [18] and genitive marked argument in N+V complex predicate.

```
23. (S (NP-SUBJ vuh[she])
   (NP-OBL gHar[home])
   (VC ponhcHI[reached]) )
       'She came home.'

24. (S (NP-SUBJ vuh[she])
       (PP-OBL (NP sANp[snake])
       sE[from])
       (VC dartI[fear] hE[is]) )
       'She fears snake.'
```

4.3.3 Adjunct (ADJ). The non-mandatory arguments are marked as ADJ (adjunct). Any adverbial function, whether syntactically realized as NP, PP or ADVP are marked as having ADJ grammatical function. For example, both ADVP and PP in examples (18) and (19) in 4.2.6 (Adverbial Phrase) are marked as having ADJ.

4.3.4 COMP. The dependent clauses have COMP grammatical function. We do not differentiate between COMP and XCOMP for the sake of simplicity. For example:

```
25. (S (NP-SUBJ vuh[he])
   (VC chAhtA[want] he[is])
   (SBAR kah[that] (S-COMP (NP-SUBJ
   sEb[apple]) (VC kHAyE[eat]))) )
       'He wants that he eats apple.'
```

4.3.5 Predicate Link (PDL). The grammatical function PDL (Predicate Link) is used in the copular constructions. For example:

```
26. (S (NP-SUBJ laRkI)girl] )
       (ADJP-PDL aqalmand[wise])
       (VC hE[is]))
       'The girl is wise.'

27. (S (NP-SUBJ vuh[he])
       (NP-PDL sadar[president])
       (vC banA[made])
       'He became president.'
```

4.3.6 INTJ. This grammatical function was introduced as the result of pilot annotation. It occurs with NPs having addressees. For example:

```
28. (S (NP-INJ bETI[daughter])
       (NP-SUBJ tum[you]) (NP-ADJ
       kab[when])
       (VC AI[come]) )
       'Daughter, when did you came?'
```

4.3.7 POF (Part of Function). Part of function marks the noun or adjective part of the complex predicate. In 4.2.2, we mentioned that these noun/adjective are not phrasal part of the VC (Verb Complex). However these are functional related with the verb, hence we introduced a functionaltag to encode this relation. The example (7), described in 4.2.2, with the grammatical function layer becomes:

```
29. (S (NP-SUBJ vuh[he])
(NP-OBJ sabaq[noun])
(NP-POF yAd[memory) (VC kar[do]
rahA[progressive] tHA[was]))
       'He was memorizing the lesson.'
```

4.3.8 Other grammatical functions. For the annotation guideline, we introduced only the sentence/clause level grammatical functions. The other types of grammatical function (e.g. modifiers/specifiers of the noun) are not part of the scheme. Our assumption is that there is one to one correspondence between such phrase labels and grammatical functions i.e. the grammatical function ADJ should follow the phrase label ADJP used inside NP and the grammatical function SPEC (as used in LFG framework) should attach with DMP etc.

4.4 Semantic Role

Semantic Role is the third layer of treebank. We used the Propbank roles, as these are (a) specially designed to have a small set of roles and (b) an Urdu corpus has already been tagged using these roles [19] and Urdu specific roles e.g. for dative subjects, causer and intermediate agent were already introduced. For example:

```
30. (S (NP-SUBJ-ARG0_GOAL Ali ko[dtv])
   (NP-OBJ-ARG1 THanD[cold])
   (VC lagi[hit])
        'Ali felt cold.'
31. (S (NP-SUBJ-ARGA Ali nE[ergative])
   (NP-OBL-ARG0_MNS Ahmed sE[from] )
   (NP-OBJ-ARG1 sEb[apple] )
   (VC katvayA[cut.caus])
       'Ali caused Ahmed to cut
      apple.'
```

5. Conclusion and Future Work

In this paper, we describe the design of a multilayer annotation scheme of Urdu corpus and then annotation of 1,300 sentence using this annotation scheme. The immediate purpose of this treebank is to create parse trees for the Text to Speech System.

We used small sets of tags to annotate the phrase, grammatical functions and semantic roles. Most importantly, we introduced demonstrative phrase, Interjection grammatical function and modeling of discontinuous phrases.

As further work, more sentences are annotated and probabilistic parser is created. However, the creation of the parser is not in the scope or contribution of this paper.

9. References

[1] J. E. Grimes and B. F. Grimes (eds.), *Ethnologue. Volume 1: Languages of the World; Volume 2: Maps and Indexes*. 14th edition, SIL International, Dallas, 2000.

[2] N. Chomsky, *Syntactic Structures*, Mouton, The Hague, 1957.

[3] M. P. Marcus, B. Santorini, and M. A. Marcinkiewicz, "Building a large annotated corpus of English: the Penn Treebank", *Computational Linguistics, 19(2)*, 1993.

[4] A. Han, et al., "A Universal Phrase Tagset for Multilingual Treebanks", *Chinese Computational Linguistics and Natural Language Processing Based on Naturally Annotated Big Data,* Springer International Publishing, 2014.

[5] Marie-Catherine De Marneffe, et al, **"**Universal Stanford Dependencies: A cross-linguistic typology*"* in *Proceedings of LREC 2014*, 2014.

[6] J. Nivre, et al., "Enhancing Universal Dependency Treebanks: A Case Study", In *Proceedings of the Second Workshop on Universal Dependencies (UDW 2018)*, 2018.

[7] M. Butt, et. al., "The Parallel Grammar project", In *Proceedings of COLING 2002. Workshop on Grammar Engineering and Evaluation*, 2002.

[8] R. Bhatt, et al., "A multi-representational and multi-layered treebank for Hindi/Urdu", In *Proceedings of the Third Linguistic Annotation Worksho*p, 2009.

[9] A. Hautli, et. al. "Adding an Annotation Layer to the Hindi/Urdu Treebank", *Linguistic Issues in Language Technology, 7(3)*, Stanford: CSLI Publications, 2012.

[10] Q. Abbas, *Building Computational Resources : The URDU.KON-TB Treebank and the Urdu Parser,* KOPS, Konstanz, 2014.

[11] M. Steedman, "Information Structure and the Syntax–Phonology Interface", *Linguistic Inquiry, 31*, 2000.

[12] D. Büring, "Syntax, information structure, and prosody", *The Cambridge Handbook of Generative Syntax*, Cambridge University Press, 2013.

[13] P. Kingsbury and M. Palmer, "From TreeBank to PropBank", In *Proceedings of LREC 2002*, 2002.

[14] S. Urooj, et al. "CLE Urdu Digest Corpus". In *Proceedings of Conference on Language and Technology 2012 (CLT12),* 2012.

[15] T. Ahmed, et al., The CLE Urdu POS Tagset, In *Proceedings of LREC 2014*, 2014.

[16] M. Butt, et. al., "Complex predicates via restriction", In *Proceedings of the LFG03 Conference*, 2003.

[17] G. Raza, *Subcategorization Acquisition and Classes of Predication in Urdu*, KOPS, Konstanz, 2011.

[18] T. Ahmed, *Spatial Expression and Case in South Asian Languages*, KOPS, Konstanz, 2009.

[19] M. Anwar, et. Al., A Proposition Bank of Urdu, In *Proceedings of LREC 2016*, 2016.