A Proposition Bank of Urdu

Maaz Anwar Nomani[♣], Riyaz Ahmad Bhat[♣], Dipti Misra Sharma[♣], Ashwini Vaidya[♡], Martha Palmer[♠], and Tafseer Ahmed[◇]
FC Kohli Center on Intelligent Systems (KCIS), IIIT-H, Hyderabad, India[♣]
Indian Institute of Technology, New Delhi, India[♡]
University of Colorado, Boulder, CO 80309 USA[♠]
DHA Suffa University, Karachi, Pakistan[◇]
{*maaz.anwar*, *riyaz.bhat*}*@research.iiit.ac.in*, *dipti@iiit.ac.in*,
ashwini.vaidya@gmail.com, *martha.palmer@colorado.edu* *tafseer@dsu.edu.pk*

Abstract

This paper describes our efforts for the development of a *Proposition Bank for Urdu*, an Indo-Aryan language. Our primary goal is the labeling of syntactic nodes in the existing Urdu dependency Treebank with specific argument labels. In essence, it involves annotation of predicate argument structures of both simple and complex predicates in the Treebank corpus.
In this paper, we describe the overall process of building the PropBank of Urdu. We discuss various statistics pertaining to the Urdu PropBank and the issues which the annotators encountered while developing the PropBank. We also discuss how these challenges were addressed to successfully expand the PropBank corpus. While reporting the Inter-annotator agreement between the two annotators, we show that the annotators share similar understanding of the annotation guidelines and of the linguistic phenomena present in the language.

Keywords: PropBank, Semantic Role Labelling, Treebanking

1. Introduction

The last decade has seen the development of several corpora with semantic role annotation in addition to syntactic annotation because they offer rich data for empirical research, building language understanding systems and other natural language processing (NLP) applications.
Proposition Bank (from now on, PropBank) (Kingsbury and Palmer, 2003) is a corpus in which the arguments of each verb predicate (simple or complex) are marked with their semantic roles. PropBank for English establishes a layer of semantic representation for Penn Treebank which is already annotated with Phrase structure trees. Capturing the semantics through predicate-argument structure involves quite a few challenges peculiar to each predicate type because the syntactic notions in which the verb's arguments and adjuncts are realized can vary based on the senses.
Our report on PropBanking an Indian language, *Urdu*, spoken in major parts of India and Pakistan, describes the annotation process of labeling the predicate-argument structure on top of an Urdu Dependency Treebank. The need for such a resource arises from the fact that while the Urdu Treebank does provide syntactico-semantic labels for some adjuncts like *location* and *time*, it fails to categorize the arguments which a verb can take depending upon subject and object pertaining to the verb. Adding a semantic layer to the Treebank will help in resolving and addressing this requirement.
Urdu is spoken primarily in Northern and Southern India. The language's Indo-Aryan word base has been enriched by borrowing from Persian and Arabic. Urdu is written right-to-left in an extension of the Persian alphabet, which is itself an extension of Arabic writing. Urdu is associated with the *Nastaliq* style of Persian calligraphy, whereas Arabic is generally written in the *Naskh* or *Ruq'ah* styles. There are between 60 and 70 million native speakers of Urdu in the world.
This paper is arranged as follows: Section 2 discusses the Literature Survey related to PropBanking. In Section 3, we briefly discuss the CPG Framework and the Urdu Dependency Treebank on which Urdu PropBank (henceforth, UPB) is being built. In Section 4, we describe the approach taken to build the Urdu PropBank corpus and its development process. Section 5 illustrates the challenges pertaining to Urdu Propbank annotation. In section 6, we report the Inter-annotator Agreement between the annotators. In section 7, we present several experiments which we did on the Urdu Propbank and their results. We conclude the paper in Section 8 and give future directions in section 9.

2. Related Work and comparison with other PropBanks

PropBanks are being built for several *European* and *Asian* languages. The first PropBank, English PropBank was built on top of the phrase structure trees in Penn Treebank and adds a semantic layer to its syntactic structures (Palmer et al., 2005). Among European languages, work on PropBank has been reported in French, German, Dutch and Portuguese languages. Merlo and Van Der Plas (2009) showed the cross-lingual validity of PropBank by applying the annotation scheme developed for English on a large portion of French sentences. Van Der Plas et al. (2010) used English FrameNet to build a corpus of German PropBank manually. Monachesi et al. (2007) used PropBank semantic labels for semi-automatic annotation of a corpus of Dutch. Duran and Aluísio (2012) talk about the annotation of a Brazilian Portuguese Treebank with semantic role labels based on Propbank guidelines.
As far as Asian languages are concerned, PropBanks exist for Chinese, Korean and Arabic. A PropBank for Korean

Ashwini Vaidya was at University of Colorado, Boulder, USA when the work on Urdu PropBank was done.

language is being built which treats each verb and adjective in the Korean Treebank as a semantic predicate and annotation is done to mark the arguments and adjuncts of the predicate (Palmer et al., 2006). For Arabic, a Semitic language, Palmer et al. (2010) presents a revised Arabic proposition bank (APB) in which predicates are identified with their relevant arguments and adjuncts in Arabic texts and an automatic process was put in place to map existing annotation to the new trees. Xue (2008) shows the use of diathesis alternation patterns for forming sense distinctions for Chinese verbs as a crucial step in marking the predicate-structure of Chinese verbs.
For *Indian languages*, a PropBank for Hindi is being built. Unlike PropBanks in other European or Asian languages, the Hindi PropBank is annotated on top of Dependency structure, the Hindi Dependency Treebank (Vaidya et al., 2011). Dependency structure is particularly suited for flexible word order languages such as Hindi.
Urdu is another free word order language for which a Propbanking effort is going on and it is built on top of Urdu Dependency Treebank (Bhat and Sharma, 2012).

3. CPG Formalism

The CPG formalism or Computational Paninian Grammar (Begum et al., 2008) is influenced by the grammatical framework of Panini, the fifth century B.C. grammarian of Sanskrit. It is a dependency grammar in which the syntactic structures consist of a set of paired, asymmetric relations between words of a sentence.
A *dependency relation* is defined between a dependent, a syntactically subordinate word and a head word on which it depends. In CPG, the verb is treated as the primary modified or as the root of the dependency tree and the elements modifying the verb event are defined by it. Karaka relations describe the manner in which arguments occur in the activity described by the verb in a sentence. There are six basic karakas given by Panini, namely (i) karta, (ii) karma 'theme', (iii) karana 'instrument', (iv) sampradaan 'recipient', (v) apaadaan 'source', and (vi) adhikarana 'location'. Apart from karaka relations, dependency relations also exist between nouns, between nouns and their modifiers and between verbs and their modifiers. An exhaustive tag-set containing all the different kinds of dependency relations has been defined in the annotation scheme based on the CPG formalism (Begum et al., 2008).

3.1. Urdu Dependency Treebank

The Urdu Dependency Treebank (UDT) (Bhat and Sharma, 2012) is built following the CPG framework. It is comprised of morphological, part-of-speech, chunking information and dependency relations. The sentences are represented in the Shakti Standard Format (Bharati et al., 2007). The Dependency tagset consists of about 43 labels. PropBank is mainly concerned with those labels depicting dependencies in the context of verb predicates. The Dependency Treebanks for Hindi and Urdu are developed following a generic pipeline. It involves building multi-layered and multi-representational Treebanks for Hindi and Urdu. The steps in the process of building the Urdu Treebank under this pipeline consists of (i) Tokenization, (ii) Morph-Analysis, (iii) POS-tagging, (iv) Chunking, and (v) Dependency annotation (Bhatt et al., 2009).
Annotation process commences with the tokenization of raw text. The tokens thus obtained are annotated with morphological and POS information. After morph-analysis and POS-tagging, words are grouped into chunks. All the above processing steps have been automated by high accuracy tools (rule-based or statistical) thus speeding up the manual process. The last process in this pipeline so far is the manual dependency annotation. The inter-chunk dependencies are marked leaving the dependencies between words in a chunk unspecified for the intra-chunk dependencies.
PropBanking is the next step in this generic pipeline which is aimed at establishing another layer of semantics on the Urdu Treebank. As part of the overall effort, a PropBank for Hindi is also built thus adding a semantic layer to the Hindi Treebank. The Urdu Dependency Treebank is developed following this Treebanking pipeline for the newspaper articles using a team of linguistics annotators. The tool used for the annotation is Sanchay (Singh and Ambati, 2010). All the annotations are represented in Shakti Standard Format (SSF). So far, $\sim$7,000 sentences (around 200K words) have been annotated with dependency structure. Each sentence contains an average of 29 words and an average of 13.7 chunks of average length 2.0.

4. Urdu PropBank Development Process

In this section, we describe the development of Urdu PropBank. In general, PropBank annotation is a 2-step process. The initial step involves the creation of frame files. The subsequent step in this process is the actual annotation of predicate-argument structure using the frame files. In case of Urdu, instead of creating frame files afresh, the frame files from Hindi and Arabic were ported (see Bhat et al. (2014) for more details).

4.1. Porting and Adapting Predicate frames from Hindi and Arabic PropBanks

The first step in building a PropBank is to make the predicate frames for simple and complex predicates (described later) available. Instead of creating predicate frames for each and every verb present in Urdu, which would indeed be very demanding and time-consuming task, (Bhat et al., 2014) explored the feasibility of porting predicate frames from already built Arabic and Hindi PropBank for use in Urdu PropBanking. This instigation was prompted from the fact that Hindi and Urdu are linguistically similar languages and belongs to a larger Indo-Aryan family. (Masica, 1993) illustrates that at colloquial level (core vocabulary and grammar), the two languages are almost similar. However, as we proceed to a higher literary level, the differences in vocabulary and lexical structure between the two languages increases profoundly. This is due to the fact that Hindi derives its higher vocabulary from Sanskrit whereas Urdu borrows it from Persian and Arabian.
Given the fact that Urdu shares much of its vocabulary with Hindi, Arabic and Persian, Bhat et al. (2014) examined the verbal predicates that Urdu shares with these languages. The verb predicates which Urdu shared with these

languages were ported and then frames were adapted from Hindi and Arabic Propbank.
Bhat et al. (2014) proposed an approach for automatic identification of the source of Urdu lexicon, thereby porting and adapting the verbal predicate frames from the PropBanks of respective languages.
There are two types of predicates in Urdu, viz. *complex predicates and simple predicates*. Complex predicates are those in which there is a nominal element present along with the verb, for example

(1)	*(Imran)*	*(snaan*	*karta*	*hai).*
	Imran	bath	do	be-PRS .

Imran takes bath.

In this sentence, *snaan karta* is a complex predicate because there is a nominal element *snaan* present along with the verb *kar*.
Simple predicates are those which only have a light verb present in them. For example

(2)	*(Shoeb*	*ne)*	*(Saba*	*ko)*	*(kitaab)*	*(di).*
	Shoeb	ERG	Saba	DAT	kitaab	give.

Shoeb gave book to Saba.

Here, *di* is the simple predicate present in this sentence.

4.2. Annotating the Urdu PropBank

Following the porting and adaptation of predicate frames from Hindi and Arabic PropBanks, the annotation process for marking the predicate argument structures for each verb instance commenced. The Jubilee annotation tool (Choi et al., 2010) was used for the annotation of the Urdu PropBank. Some changes were made in the tool to take into account the dependency trees of Urdu sentences and to make them available for annotators as a reference.
The UPB currently consists of 20 labels including both numbered arguments and modifiers (Table 1).

Label	Description
ARG0	Doer, Experiencer
ARG1	Patient, Theme, Undergoer
ARG2	Beneficiary
ARG3	Instrument
ARG2-ATR	Attribute or Quality
ARG2-LOC	Physical Location
ARG2-GOL	Goal
ARG2-SOU	Source
ARGM-PRX	noun-verb construction
ARGM-ADV	Adverb
ARGM-DIR	Direction
ARGM-EXT	Extent or Comparison
ARGM-MNR	Manner
ARGM-PRP	Purpose
ARGM-DIS	Discourse
ARGM-LOC	Abstract location
ARGM-MNS	Means
ARGM-NEG	Negation
ARGM-TMP	Time
ARGM-CAU	Cause or Reason

Table 1: **The Urdu PropBank labels and their description.**

Similar to Hindi PropBank (Vaidya et al., 2011), the UPB labels make some distinctions that are not made in other languages such as English. For example, ARG2 is subdivided into labels with function tags, in order to avoid it from being semantically overloaded (Loper et al., 2007). ARG-A marks the arguments of morphological causatives in Urdu. We also use a label to represent complex predicate constructions: 'ARGM-PRX'.

5. Annotation Challenges

This section deals with the annotation challenges of building the UPB. We discuss some seemingly simple examples present in the resource nevertheless difficult to annotate because of their structure and context. We also discuss challenging cases where high interannotator disagreement is observed between the annotators.

5.1. Challenges and redundancies

The challenges which the two annotators encountered are described below. We also discuss the approach taken by us to resolve these challenges and other redundancies.

- The major issue was to differentiate between 'ARG0' and 'ARG1' labels. As dependency trees are provided with the sentences in Jubilee (tool for UPB annotation) (Choi et al., 2010), the node had 'k1' karaka label for agent, doer, possessor etc. according to the Dependency annotation guidelines. The annotators were influenced by the dependency trees and the labels ('k1' and 'k2') on each node.

- The argument labels for the core arguments are not repeated within the same sentence in PropBank. This observation represents the fact that each possible core argument manifests itself only once in a sentence, i.e. multiple arguments cannot be given the same core argument for the same verb in the same sentence.

- *hai* 'be' and *ho* 'become' verbs are the most ambiguous of all the simple predicates. The frame files of these verbs do not typically contain ARG0 but there are sentences in the UDT in which the verb's arguments tend themselves to an ARG0 label as exemplified below:

(3)	*(Yousuf)*	*(Shaheen*	*ka)*	*(pata)*	*(lagane*
	Yousuf	Shaheen	GEN	address	to
	mein)	*(masroof)*	*(hai).*		
	in	busy	be-PRS .		

Yousuf is busy in finding Shaheen whereabouts.

(4)	*(Tariq)*	*(Hyderabad*	*mein)*	*(800*	*voton*
	Tariq	Hyderabad	in	800	votes
	se)	*(aage)*	*(hai).*		
	ABL	front	be-PRS .		

Tariq is leading by 800 votes in Hyderabad.

Here, 'Yousuf' and 'Tariq' are not involved in any volitional act, but for the predicate 'hai', these two get the 'ARG0' label whereas 'Shaheen ka' and '800 voton se' gets the 'ARG1' label respectively.

This annotation challenge also includes the annotation of 'Arg0' label for possessor subjects as in the following complex predicate example:

(5) *(Har) (zarra) (shaoor) (rakhta hai).*
Every particle honor keep be-PRS .
Each particle possess conscience.

In this example, the 'zarra' argument lacks agentivity but is given an 'ARG0' label based on its possession of a conscious.

5.2. Resolving the Annotation challenges

Following are the approaches taken to handle the annotation challenges in UPB:

- The issue of 'ARG0' and 'ARG1' was resolved by labeling those arguments of a verb in a sentence as 'ARG0' which display agentivity or in which one argument was clearly 'ARG1' so other argument tends to get the label 'ARG0' because 'ARG1' is already used in this sentence as shown in example 4. Among the intransitive verbs, the unergative type verbs get an 'Arg0' argument. We typically find 'Arg0' in those intransitive verbs that take an animate subject (i.e. a subject that has voluntary control over the action expressed by the verb). For example, the verbs nAca 'dance' and xORa 'run' are examples of intransitive verbs that require an agentive subject, typically animate subjects, because the actions expressed by these verbs involve a participant that has voluntary control over the action described by the verb.
- The frame files of *hai* 'be' and *ho* 'become' were deemed similar and argument labeling was done based on the labels specified by these frame files.

There are cases in which a nominal can take multiple types of PropBank labels based upon its usage in a given sentence. Such ambiguous cases usually lead to disagreement between the annotators. Consider the following 3 sentences with predicate 'hai':

(6) *(Sohail) (apko) (Azhar ke) (talluq se)*
Sohail you Azhar via ABL know
(jaanta) (hun).
be-PRS.
Sohail knows you via Azhar.

(7) *(Azhar) (yahan) (apni) (naukri ke) (talluq*
Azhar here his job related to
se) (aaya) (hai).
come be-PRS .
Azhar has come here because of his job.

(8) *(Azhar) (Hindustan ke talluq se) (kuch*
Azhar Hindustan about some talks do
baat) (karna chahta hai).
wish be-PRS .
Azhar wants to say something about India.

In these set of examples, 'talluq se' takes 3 different kinds of variations and meanings based upon its usage in the sentence and therefore it tends to get 3 different UPB labels given the predicate 'hai'. In the first example, 'talluq se' is 'ARGM-CAU' because it is the 'reason' of knowing someone. In the second case, 'talluq se' is 'ARGM-PRP' because it is 'purpose' of 'Azhar' coming here. Finally, in the third instance, 'Hindustan ke talluq se' takes 'ARGM-LOC' because it denotes the 'about' sense in the meaning of the sentence.
Such types of ambiguous constructions are present in UPB which may lead to disagreement between the annotators. One may find it difficult to annotate them just by looking at the frame files of the predicate in question. They are annotated by considering the context, the overall meaning of the sentence and the tree structure of the sentence in addition to taking help from the frame file.

6. Inter-Annotator Agreement

To establish the credibility of predicate-argument structure annotations in UPB, we measured the Inter-annotator agreement between the two annotators using a data set of 44,000 words double-annotated (double PropBanked) by two annotators, without either annotator knowing other's judgment of marking semantic role labels. A healthy agreement on the data set will ensure that the decisions taken by the annotators during building UPB and thereafter the annotations on arguments of verb predicates are consistent and reliable.
We measured the Inter-annotator agreement using Fleiss' kappa (Randolph, 2005) which is the frequently used agreement coefficient for annotation tasks on categorical data. Kappa was introduced by (Carletta, 1996) and since then many linguistics resources have been evaluated by the coefficient. The kappa statistics in this section show the agreement between the annotators on a given data-set and the compatibility and consistency with which the annotations have been performed on predicate-argument structures by the two annotators. These measures also demonstrate the conformity in their understanding of the PropBank annotation guidelines.
The Fleiss' kappa is calculated as:

$$k = \frac{Pr(a) - Pr(e)}{1 - Pr(e)} \tag{9}$$

The factor $1 - Pr(e)$ estimates the degree of agreement that is attainable above chance, and $Pr(a) - Pr(e)$ evaluates the degree of agreement actually achieved above chance.

Tokens	Pr (a)	Pr(e)	Kappa
44,000	0.81	0.089	0.88

Table 2: **Kappa statistics for Inter-Annotator Experiment.**

We consider the agreement between the annotators (shown as kappa value in Table 2) to be reliable based on the interpretation matrix proposed by Landis and Koch (1977) (see Table 3). There is almost perfect agreement between the annotators which implies their analogous understanding of the annotation guidelines and of the linguistic phenomenon present in the Urdu language.

Kappa Statistic	Strength of agreement
0.00	Poor
0.0-0.20	Slight
0.21-0.40	Fair
0.41-0.60	Moderate
0.61-0.80	Substantial
0.81-1.00	Almost perfect

Table 3: **Coefficients for the agreement-rate based on (Landis and Koch, 1977).**

7. Inter-annotator Agreement Experiments

PropBank labels are quite similar to the dependency labels in their orientation and structure. Vaidya et al. (2011) has shown a very interesting correlation between dependency and predicate argument structures. In sections 7.1 and 7.3, we performed a series of experiments to analyze both of the structures together.

7.1. Influence of Dependency Annotations on PropBanking Annotations

We performed an experiment to see the influence on annotators of Dependency labels when marking Propbank labels. We performed the following 2 experiments:

- Annotating PropBank without seeing the Dependency Tree.
- Annotating PropBank while seeing the Dependency Tree.

The idea of conducting such an experiment was to see the impact of Dependency labels on the annotators while marking the PropBank labels. It was conducted on 7,720 tokens and 300 predicates from the Urdu PropBank. Since PropBank labels share a close proximity and resemblance with Dependency labels as far as the syntacto-semantic notion is concerned, we wanted to exploit the interference of Dependency tags in the marking of predicate-argument structure. As shown by Vaidya et al. (2011) there is a significant correlation between the Hindi Propbank labels and Dependency labels and since Hindi and Urdu are structurally similar languages, we expected that such a mapping would exist for the Urdu Propbank labels as well.

As can be seen from Table 4, there is a substantial agreement of 95 percent between the annotators on marking Arg0 given a k1 dependency tag. Such high agreements can be noted for *Arg2-k4*, *Arg2-LOC-k7p*, *ArgM-LOC-k7*, *ArgM-TMP-k7t*, *ArgM-PRP-rt* etc. These stats confirmed our intuition that there is a significant mapping between the Urdu PropBank labels and the Urdu Dependency labels and that the influence of Dependency labels is profound while marking Propbank labels. There are certain exceptions to this intuition though, as can be seen by the Arg1 label. Annotators are influenced by ‘k1’ and ‘k2’ while giving an Arg1 tag to an argument of a predicate. This complies with the fact that marking ‘Arg1’ is a difficult task as can be understood from the following example:

Dep. labels	***Agreement***	**Prop. labels**
k1	95	***ARG0***
k1	80.4	***ARG1***
k2	95.6	***ARG1***
k2	66.7	***ARG2***
k4	100	***ARG2***
k2	57.1	***ARG2-SOU***
k5	41.1	***ARG2-SOU***
ras-k1	83.3	***ARG2-LOC***
k5	22.2	***ARG2-LOC***
k7p	78.2	***ARG2-LOC***
k7	40	***ARG2-LOC***
k7	87.2	***ARGM-LOC***
k7p	11.2	***ARGM-LOC***
k7t	83.4	***ARGM-TMP***
vmod	71.4	***ARGM-TMP***
rh	71.4	***ARGM-CAU***
rt	98.2	***ARGM-PRP***
vmod	100	***MNR***
vmod	76.9	***ARGM-ADV***
k7a	63.6	***ARGM-ADV***
adv	73.17	***ARGM-ADV***
ras-k1	85.7	***ARGM-ADV***

Table 4: **Agreement (in percentage) among the annotators on PropBank labels while seeing the Dependency labels.**

(10)

ahmed	*so*	*raha*	*hai*	.
Ahmed	sleep	live	be-PRS	.

Ahmed is sleeping.

(11)

ahmed	*daudh*	*rahaa*	*hai*	.
Ahmed	run	live	be-PRS	

Ahmed is running.

In sentence 8, Ahmed is in a state of ‘sleep’ as shown by complex predicate ‘so raha hai’, while in sentence 9, Ahmed is performing the action of ‘running’.

Tokens	**Pr (a)**	**Pr(e)**	**Kappa**
7,720	0.81	0.079	0.739

Table 5: Kappa statistics for Inter-Annotator Experiment without seeing Dependency labels.

Tokens	**Pr (a)**	**Pr(e)**	**Kappa**
7,720	0.85	0.085	0.801

Table 6: Kappa statistics for Inter-Annotator Experiment with seeing Dependency labels.

In Tables 5 and 6, we show the numbers for Inter-annotator agreement for this experiment. It can be seen by comparing the tables that there is an increase of 0.062 percentage in the agreement among the annotators when Dependency labels were introduced in the annotation process. The Dependency labels were not provided to the annotators in the first experiment and they marked the predicate-argument structure based on the frame files of that particular predicate only.

7.2. Agreement between Postpositions/Case Markers/Vibhaktis and PropBank labels

While in English, the major constituents of a sentence (subject, object, etc.) can usually be identified by their position in the sentence, Urdu is a relatively free word-order language. Constituents can be moved around in the sentence without impacting the core meaning. For example, the following sentence pair conveys the same meaning, but with a difference of emphasis:

(12)

Shahid	*ne*	*France*	*mein*	*tiranga*	*lehraya.*
Shahid	ERG	France	in	flag	hoist-PERF.3MSg

'Shahid hoisted a flag in France.'

(13)

France	*mein*	*tiranga*	*Shahid*	*ne*	*lehraya.*
France	in	flag	Shahid	ERG	hoist-PERF.3MSg

'In France, Shahid hoisted the flag.'

The identity of *Shahid* as the doer and *France* as the Location in both sentences comes from the case markers *ne* (ergative) and *mein* (locative). Therefore, even though Urdu is pre-dominantly SOV in its word-order, correct case-marking is a crucial part of extracting the arguments in a predicate-argument structure. The description of case markers is given in Table 7.

Urdu Case marker	**Meaning**
ne	Ergative
mein	Locative
par	Locative
ko	Dative/Accusative
se	Instrumental/Ablative
ki	Genetive

Table 7: **Urdu Case markers and their description.**

Case markers are decided by semantic relations and tense-aspect information in suffixes. For example, if a clause has an object, and has a perfective form, the doer usually requires the case marker *ne* as in (14) and (16):

(14)

maryam	*ne*	*party*	*mein*	*gaanaa*	*gaayaa* .
Maryam	ERG	party	in	song	recite-PERF.3MSg

'Maryam sang a song in the party.'

As can be seen from Table 8, the annotators have maximum agreement over 'ARG0' label given a *ne* vibhakti. In Urdu, the case clitic *ne* is often indicative of an agent, and as 'ARG0' is mostly associated with agentivity, this can sometimes provide a clue about the identity of Arg0. However, note that an Arg0 argument does not necessarily get 'ne-marking'. For example.

(15)

choonki	*(Iraq*	*ko)*	**[ARG0]**	*america*	*se*	*apne*	*rishte*	*acche*	*rakhne*	*hain,*	*islie*	*wah*	*yeh*	*qadam*	*nahi*	*uthaaega.*
since	Iraq	DAT		America	with	self	relations	good	keep	be-PRS,	therefore	it	this	step	not	lift-PRS.3MSg.

'Since Iraq has to keep good relations with America, it will not take this step.'

(16)

anwar	*ne*	*shaziya*	*se*	*agra*	*chalne*	*ko*	*kahaa.*
Anwar	ERG	Shazia	with	Agra	go-INF	DAT	say-PERF.3MSg.

'Anwar asked Shazia to go to Agra.'

Prop. labels	ne	mein	par	se	ko	ki	ke
Arg0	65						07
Arg1				10	39		
Arg2				09	15		
Arg2-ATR				01		01	
Arg3/ArgM-MNS				02			01
Arg0-GOL					01		
Arg2-GOL					01		
Arg2-SOU				27		04	01
ArgM-CAU			01	09		07	01
ArgM-LOC		52	32				06
Arg2-LOC		09	03				11
ArgM-TMP		04		01	06		12
ArgM-PRP				01		01	36
ArgM-ADV			06				28
ArgM-MNR							02
ArgM-DIS	01	01					01
ArgM-EXT						01	01
ArgM-DIR						02	

Table 8: **Agreement among the annotators on PropBank labels given a Vibhakti/Case-Marker.**

Even though case marking can provide a clue to the use of a PB label for an argument in some tense and aspect combinations etc., PropBanking does not entirely depend on it, For PropBanking, we analyze the verbs event and its participants irrespective of the aspect and transitivity of the main verb or the use of a light verb that changes the case on the noun.

7.3. Agreement between Dependency labels and PropBank labels

The experiments performed in this and subsequent subsections were carried on 44,000 tokens of UPB. The Dependency labels which we have considered here for agreement between PB labels are of two types: (1) karaka, and (2) Relations other than karakas. The karaka relations given by Panini are described in Table 9.

Dependency Labels	Description
k1	karta - doer/agent/subject
k2	karma - object/patient
k3	karana - instrument
k4	sampradana - recipient
k5	apadana - source
k7	location elsewhere
k7p	location in space
k7t	time
adv	adverbs
rh	reason
rd	direction

Table 9: **Major karaka relations or Dependency labels of UDT**

There can also be other labels when an action is being carried out. These labels may not have any direct role in the action though. Reason (rh) and purpose (rt) are two examples of such labels. 'Karakas' are the roles of various direct participants in an action. An action in a sentence is normally denoted by a verb. Hence, a verb becomes the primary modified (root node of a dependency tree) in a sentence.
Table 10 shows that the two annotators have good agreement on marking certain PropBank labels like *ARGM-MNR*, *ARGM-PRP*, *ARGM-CAU*, *ARGM-TMP*, *ARGM-LOC*, *ARG2-GOL* and *ARG3* when there similar counterpart dependency labels viz. *vmod*, *rt*, *rh*, *k7t*, *k7*, *k2p* and *k3* respectively were present in the dependency tree of the sentence. This agreement also highlights the mapping between the Urdu Dependency labels and the Urdu PropBank labels.

	k1	k2	k3	k4	k5	k2p	k7p	k7	k7t	rh	rt	rd	vmod
Arg0	**59**												
Arg1	5	**41**											
Arg3			**2**										
Arg2		1		**22**									
Arg2-SOU	2	5			**13**							9	1
Arg2-GOL						**1**							
Arg2-LOC	3	6					**15**	1					
ArgM-LOC							2	**110**	1				3
ArgM-TMP									**28**				
ArgM-CAU										**25**			
ArgM-PRP											**72**		
ArgM-MNR													**2**
ArgM-DIS	1							1					**2**
ArgM-ADV	2	2		4				8					**20**
ArgM-EXT								**3**					1

Table 10: **Agreement (in count) among the annotators on PropBank labels given a Dependency label.**

7.4. Conjoined scrutiny of Case-Markers, Dependency labels and PropBank labels

In this section, Table 11 presents a combined investigation on the effect and role of *Case-Markers (post-positions)* and *Dependency labels* on marking *PropBank labels*.

Prop. labels	*ne*	*mein*	*par*	*se*	*ko*	*ki*	*ke*	**Dep. labels**
Arg0	81				05			***k1***
Arg1					07			***k1***
Arg1				19	59			***k2***
Arg2-SOU				09				***k2***
Arg2-LOC							07	***k2***
Arg2				40	54			***k4***
Arg0-GOL								***k4***
Arg3				66				***k3***
ArgM-MNS							33	***k3***
Arg2-SOU				92			07	***k5***
ArgM-LOC		60	25				05	***k7***
ArgM-ADV			02				02	***k7***
Arg2-LOC		52	21				10	***k7p***
ArgM-LOC		10						***k7p***
ArgM-TMP		25			25		45	***k7t***
ArgM-ADV							100	***k7a***
ArgM-ADV			42					***adv***
ArgM-LOC			14					***adv***
ArgM-EXT						14		***adv***
Arg2-LOC							28	***adv***
ArgM-CAU				50		43	07	***rh***
Arg2-SOU				62		37		***rd***
ArgM-PRP							94	***rt***

Table 11: **Agreement (in percentage) among the annotators on PropBank labels given Vibhakti/Case-Markers and Dependency labels.**

We show that case-markers and Dependency labels play a significant role in the annotation of predicate-argument structure. Though the annotators have not primarily used these cues to mark the semantic labels and have relied extensively on frame-files of the verbs, still these two types of information provide ineffable traces of similarity between Dependency labels and PropBank labels.
For example as discussed above, the '*ne*' case marker provides information about agentivity. This is reflected in Table 11 as maximum agreement between the annotators is captured for *ne-k1-Arg0* instances. Dependency label *k2* is more distributed as far as case-markers are concerned and finds itself spread across in *ko-k2-Arg1*, *se-k2-Arg1* and *ke-k2-Arg2-LOC* instances.

8. Conclusion

In this work, we have discussed an ongoing effort of building a *Proposition Bank for Urdu*. The present size of this Propbank is around 180,000 tokens which is double-propbanked by the two annotators for simple predicates. Another 100,000 tokens have been annotated for complex predicates of Urdu.
We discussed the Propbank development process, Propbank labels and agreement among the annotators of marking Propbank labels given dependency labels. We also discussed the evaluation of Propbank by measuring the inter-annotator agreement between the two annotators using the Kappa statistics.
The agreement calculated here is considered to be reliable and substantial ensuring that the predicate-argument structure annotation in the Urdu Propbank is consistent.

9. Future Directions

We aim at increasing the size of the Urdu PropBank. It requires annotating all the simple predicates and complex predicates present in the Urdu Treebank. Future directions also include plans to automate the process of labeling semantic roles of a predicate by building a statistical Semantic Role Labeler for Urdu.

10. Acknowledgements

We would like to thank Sameena Yasmeen and Anjum Parveen for annotating the data set used for the experiments of inter-annotator agreement. The work reported in this article is supported by the NSF grant (Award Number: CNS 0751202; CFDA Number: 47.070). Any opinions, findings, and conclusions or recommendations expressed in this material are those of the author(s) and do not necessarily reflect the views of the National Science Foundation.

11. Bibliographical References

Begum, R., Husain, S., Dhwaj, A., Sharma, D. M., Bai, L., and Sangal, R. (2008). Dependency annotation scheme for indian languages. In *IJCNLP*, pages 721–726. Citeseer.

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

Bhat, R. A. and Sharma, D. M. (2012). A dependency treebank of urdu and its evaluation. In *Proceedings of the Sixth Linguistic Annotation Workshop*, pages 157–165. Association for Computational Linguistics.

Bhat, R. A., Jain, N., Sharma, D. M., Vaidya, A., Palmer, M., Babani, J., Ahmed, T., and LTRC, I.-H. (2014). Adapting predicate frames for urdu propbanking. *LT4CloseLang 2014*, page 47.

Bhatt, R., Narasimhan, B., Palmer, M., Rambow, O., Sharma, D. M., and Xia, F. (2009). A multi-representational and multi-layered treebank for hindi/urdu. In *Proceedings of the Third Linguistic Annotation Workshop*, pages 186–189. Association for Computational Linguistics.

Carletta, J. (1996). Assessing agreement on classification tasks: the kappa statistic. *Computational linguistics*, 22(2):249–254.

Choi, J. D., Bonial, C., and Palmer, M. (2010). Propbank instance annotation guidelines using a dedicated editor, jubilee. In *LREC*.

Duran, M. S. and Aluísio, S. M. (2012). Propbank-br: a brazilian treebank annotated with semantic role labels. In *LREC*, pages 1862–1867.

Kingsbury, P. and Palmer, M. (2003). Propbank: the next level of treebank. In *Proceedings of Treebanks and lexical Theories*, volume 3. Citeseer.

Landis, J. R. and Koch, G. G. (1977). The measurement of observer agreement for categorical data. *biometrics*, pages 159–174.

Loper, E., Yi, S.-T., and Palmer, M. (2007). Combining lexical resources: mapping between propbank and verbnet. In *Proceedings of the 7th International Workshop on Computational Linguistics, Tilburg, the Netherlands*.

Masica, C. P. (1993). *The Indo-Aryan Languages*. Cambridge University Press.

Merlo, P. and Van Der Plas, L. (2009). Abstraction and generalisation in semantic role labels: Propbank, verbnet or both? In *Proceedings of the Joint Conference of the 47th Annual Meeting of the ACL and the 4th International Joint Conference on Natural Language Processing of the AFNLP: Volume 1-Volume 1*, pages 288–296. Association for Computational Linguistics.

Monachesi, P., Stevens, G., and Trapman, J. (2007). Adding semantic role annotation to a corpus of written dutch. In *Proceedings of the Linguistic Annotation Workshop*, pages 77–84. Association for Computational Linguistics.

Palmer, M., Gildea, D., and Kingsbury, P. (2005). The proposition bank: An annotated corpus of semantic roles. *Computational linguistics*, 31(1):71–106.

Palmer, M., Ryu, S., Choi, J., Yoon, S., and Jeon, Y. (2006). Korean propbank. *LDC Catalog No.: LDC2006T03 ISBN*, pages 1–58563.

Randolph, J. J. (2005). Free-marginal multirater kappa (multirater k [free]): An alternative to fleiss' fixed-marginal multirater kappa. *Online Submission*.

Singh, A. K. and Ambati, B. R. (2010). An integrated digital tool for accessing language resources. In *LREC*.

Vaidya, A., Choi, J. D., Palmer, M., and Narasimhan, B. (2011). Analysis of the hindi proposition bank using dependency structure. In *Proceedings of the 5th Linguistic Annotation Workshop*, pages 21–29. Association for Computational Linguistics.

van der Plas, L., Samardžić, T., and Merlo, P. (2010). Cross-lingual validity of propbank in the manual annotation of french. In *Proceedings of the Fourth Linguistic Annotation Workshop*, pages 113–117. Association for Computational Linguistics.

Xue, N. (2008). Labeling chinese predicates with semantic roles. *Computational linguistics*, 34(2):225–255.

Zaghouani, W., Diab, M., Mansouri, A., Pradhan, S., and Palmer, M. (2010). The revised arabic propbank. In *Proceedings of the Fourth Linguistic Annotation Workshop*, pages 222–226. Association for Computational Linguistics.