Adapting Predicate Frames for Urdu PropBanking

Riyaz Ahmad Bhat♣, Naman Jain♣, Dipti Misra Sharma♣, Ashwini Vaidya♠, Martha Palmer♠, James Babani♠ and Tafseer Ahmed♢
LTRC, IIIT-H, Hyderabad, India♣
University of Colorado, Boulder, CO 80309 USA♠
DHA Suffa University, Karachi, Pakistan♢
{riyaz.bhat, naman.jain}@research.iiit.ac.in, dipti@iiit.ac.in,
{vaidyaa, mpalmer, james.babani}@colorado.edu, tafseer@dsu.edu.pk

Abstract

Hindi and Urdu are two standardized registers of what has been called the Hindustani language, which belongs to the Indo-Aryan language family. Although, both the varieties share a common grammar, they differ significantly in their vocabulary to an extent where both become mutually incomprehensible (Masica, 1993). Hindi draws its vocabulary from Sanskrit while Urdu draws its vocabulary from Persian, Arabic and even Turkish. In this paper, we present our efforts to adopt frames of nominal and verbal predicates that Urdu shares with either Hindi or Arabic for Urdu PropBanking. We discuss the feasibility of porting such frames from either of the sources (Arabic or Hindi) and also present a simple and reasonably accurate method to automatically identify the origin of Urdu words which is a necessary step in the process of porting such frames.

1 Introduction

Hindi and Urdu, spoken primarily in northern India and Pakistan, are socially and even officially considered two different language varieties. However, such a division between the two is not established linguistically. They are two standardized registers of what has been called the Hindustani language, which belongs to the Indo-Aryan language family. Masica (1993) explains that, while they are different languages officially, they are not even different dialects or sub-dialects in a linguistic sense; rather, they are different literary styles based on the same linguistically defined sub-dialect. He further explains that at the colloquial level, Hindi and Urdu are nearly identical, both in terms of core vocabulary and grammar. However, at formal and literary levels, vocabulary differences begin to loom much larger (Hindi drawing its higher lexicon from Sanskrit and Urdu from Persian and Arabic) to the point where the two styles/languages become mutually unintelligible. In written form, not only the vocabulary but the way Urdu and Hindi are written makes one believe that they are two separate languages. They are written in separate orthographies, Hindi being written in Devanagari, and Urdu in a modified Persio-Arabic script. Given such (apparent) divergences between the two varieties, two parallel treebanks are being built under *The Hindi-Urdu treebanking Project* (Bhatt et al., 2009; Xia et al., 2009). Both the treebanks follow a multi-layered and multi-representational framework which features Dependency, PropBank and Phrase Structure annotations. Among the two treebanks the Hindi treebank is ahead of the Urdu treebank across all layers. In the case of PropBanking, the Hindi treebank has made considerable progress while Urdu PropBanking has just started.

The creation of predicate frames is the first step in PropBanking, which is followed by the actual annotation of verb instances in corpora. In this paper, we look at the possibility of porting related frames from Arabic and Hindi PropBanks for Urdu PropBanking. Given that Urdu shares its vocabulary with Arabic, Hindi and Persian, we look at verbal and nominal predicates that Urdu shares with these languages and try to port and adapt their frames from the respective PropBanks instead of creating them afresh. This implies that identification of the source of Urdu predicates becomes a necessary step in this process. Thus, in order to port the relevant frames, we need to first identify the source of Urdu predicates and then extract their frames from the related PropBanks. To state briefly, we present the following as contributions of this paper:

- Automatic identification of origin or source of Urdu vocabulary.

Language Technology for Closely Related Languages and Language Variants (LT4CloseLang), pages 47–55,
October 29, 2014, Doha, Qatar.

- Porting and adapting nominal and verbal predicate frames from the PropBanks of related languages.

The rest of the paper is organised as follows: In the next Section we discuss the Hindi-Urdu treebanking project with the focus on PropBanking. In Section 3, we discuss our efforts to automatically identify the source of Urdu vocabulary and in Section 4, we discuss the process of adapting and porting Arabic and Hindi frames for Urdu PropBanking. Finally we conclude with some future directions in Section 5.

2 A multi-layered, multi-representational treebank

Compared to other existing treebanks, Hindi/Urdu Treebanks (HTB/UTB) are unusual in that they are multi-layered. They contain three layers of annotation: dependency structure (DS) for annotation of modified-modifier relations, PropBank-style annotation (PropBank) for predicate-argument structure, and an independently motivated phrase-structure (PS). Each layer has its own framework, annotation scheme, and detailed annotation guidelines. Due to lack of space and relevance to our work, we only look at PropBanking with reference to Hindi PropBank, here.

2.1 PropBank Annotation

The first PropBank, the English PropBank (Kingsbury and Palmer, 2002), originated as a one-million word subset of the Wall Street Journal (WSJ) portion of Penn Treebank II (an English phrase structure treebank). The verbs in the PropBank are annotated with predicate-argument structures and provide semantic role labels for each syntactic argument of a verb. Although these were deliberately chosen to be generic and theory-neutral (e.g., ARG0, ARG1), they are intended to consistently annotate the same semantic role across syntactic variations. For example, in both the sentences *John broke the window* and *The window broke*, *'the window'* is annotated as ARG1 and as bearing the role of '$Patient$'. This reflects the fact that this argument bears the same semantic role in both the cases, even though it is realized as the structural subject in one sentence and as the object in the other. This is the primary difference between PropBank's approach to semantic role labels and the Paninian approach to karaka labels, which it otherwise resembles closely. PropBank's ARG0 and ARG1 can be thought of as similar to Dowty's prototypical '$Agent$' and '$Patient$' (Dowty, 1991). PropBank provides, for each sense of each annotated verb, its "roleset", i.e., the possible arguments of the predicate, their labels and all possible syntactic realizations. The primary goal of PropBank is to supply consistent, simple, general purpose labeling of semantic roles for a large quantity of coherent text that can provide training data for supervised machine learning algorithms, in the same way that the Penn Treebank supported the training of statistical syntactic parsers.

2.1.1 Hindi PropBank

The Hindi PropBank project has differed significantly from other PropBank projects in that the semantic role labels are annotated on dependency trees rather than on phrase structure trees. However, it is similar in that semantic roles are defined on a verb-by-verb basis and the description at the verb-specific level is fine-grained; e.g., a verb like 'hit' will have '$hitter$' and '$hittee$'. These verb-specific roles are then grouped into broader categories using numbered arguments (ARG). Each verb can also have a set of modifiers not specific to the verb (ARGM). In Table 1, PropBank-style semantic roles are listed for the simple verb de 'to give'. In the table, the numbered arguments correspond to the giver, thing given and recipient. Frame file definitions are created manually and include role information as well as a unique roleset ID (e.g. de.01 in Table 1), which is assigned to every sense of a verb. In addition, for Hindi the frame file also includes the transitive and causative forms of the verb (if any). Thus, the frame file for de 'give' will include $dilvaa$ 'cause to give'.

de.01	*to give*
`Arg0`	the giver
`Arg1`	thing given
`Arg2`	recipient

Table 1: A Frame File

The annotation process for the PropBank takes place in two stages: the creation of frame files for individual verb types, and the annotation of predicate argument structures for each verb instance. The annotation for each predicate in the corpus is carried out based on its frame file definitions.

The PropBank makes use of two annotation tools viz. Jubilee (Choi et al., 2010b) and Cornerstone (Choi et al., 2010a) for PropBank instance annotation and PropBank frame file creation respectively. For annotation of the Hindi and Urdu PropBank, the Jubilee annotation tool had to be modified to display dependency trees and also to provide additional labels for the annotation of empty arguments.

3 Identifying the source of Urdu Vocabulary

Predicting the source of a word is similar to language identification where the task is to identify the language a given document is written in. However, language identification at word level is more challenging than a typical document level language identification problem. The number of features available at document level is much higher than at word level. The available features for word level identification are word morphology, syllable structure and phonemic (letter) inventory of the language(s).

In the case of Urdu, the problem is even more complex as the borrowed words don't necessarily carry the inflections of their source language and don't retain their identity as such (they undergo phonetic changes as well). For example, $khabar$ 'news' which is an Arabic word declines as per the morphological paradigm of feminine nominals in Hindi and Urdu as shown in Table (2). However, despite such challenges, if we look at the character histogram in Figure (1), we can still identify the source of a sufficiently large portion of Urdu vocabulary just by using letter-based heuristics. For example neither Arabic nor Persian has aspirated consonants like $b^{ɦ}$, p^{h} *Aspirated Bilabial Plosives*; $tʃ^{h}$, $dʒ^{ɦ}$ *Aspirated Alveolar Fricatives*; $ɖ^{ɦ}$ *Aspirated Retroflex Plosive*; $g^{ɦ}$, k^{h} *Aspirated Velar Plosives* etc. while Hindi does. Similarly, the following sounds occur only in Arabic and Persian: ʒ *Fricative Postalveolar*; θ, ð *Fricative Dental*; ħ *Fricative Pharyngeal*; χ *Fricative Uvular* etc. Using these heuristics we could identify 2,682 types as Indic, and 3,968 as either Persian or Arabic out of 12,223 unique types in the Urdu treebank (Bhat and Sharma, 2012).

	Singular	**Plural**
Direct	*khabar*	*khabarain*
Oblique	*khabar*	*khabaron*

Table 2: Morphological Paradigm of $khabar$

This explains the efficiency of n-gram based approaches to either document level or word level language identification tasks as reported in the recent literature on the problem (Dunning, 1994; Elfardy and Diab, 2012; King and Abney, 2013; Nguyen and Dogruoz, 2014; Lui et al., 2014).

In order to predict the source of an Urdu word, we frame two classification tasks: (1) binary classification into Indic and Persio-Arabic and, (2) triclass classification into Arabic, Indic and Persian. Both the problems are modeled using smoothed n-gram based language models.

3.1 N-gram Language Models

Given a word w to classify into one of k classes c_1, c_2, ... , c_k, we will choose the class with the maximum conditional probability:

$$\begin{aligned} \mathbf{c}^* &= \arg\max_{c_i} \; p(c_i|w) \\ &= \arg\max_{c_i} \; p(w|c_i) * p(c_i) \end{aligned} \tag{1}$$

The prior distribution $p(c)$ of a class is estimated from the respective training sets shown in Table (3). Each training set is used to train a separate letter-based language model to estimate the probability of word w. The language model $p(w)$ is implemented as an n-gram model using the IRSTLM-Toolkit (Federico et al., 2008) with Kneser-Ney smoothing. The language model is defined as:

$$p(w) = \prod_{i=1}^{n} \; p(l_i|l_{i-k}^{i-1}) \tag{2}$$

where, l is a letter and k is a parameter indicating the amount of context used (e.g., $k = 4$ means 5-gram model).

3.2 Etymological Data

In order to prepare training and testing data marked with etymological information for our classification experiments, we used the *Online*

[1] http://www.langsci.ucl.ac.uk/ipa/IPA_chart_%28C%292005.pdf

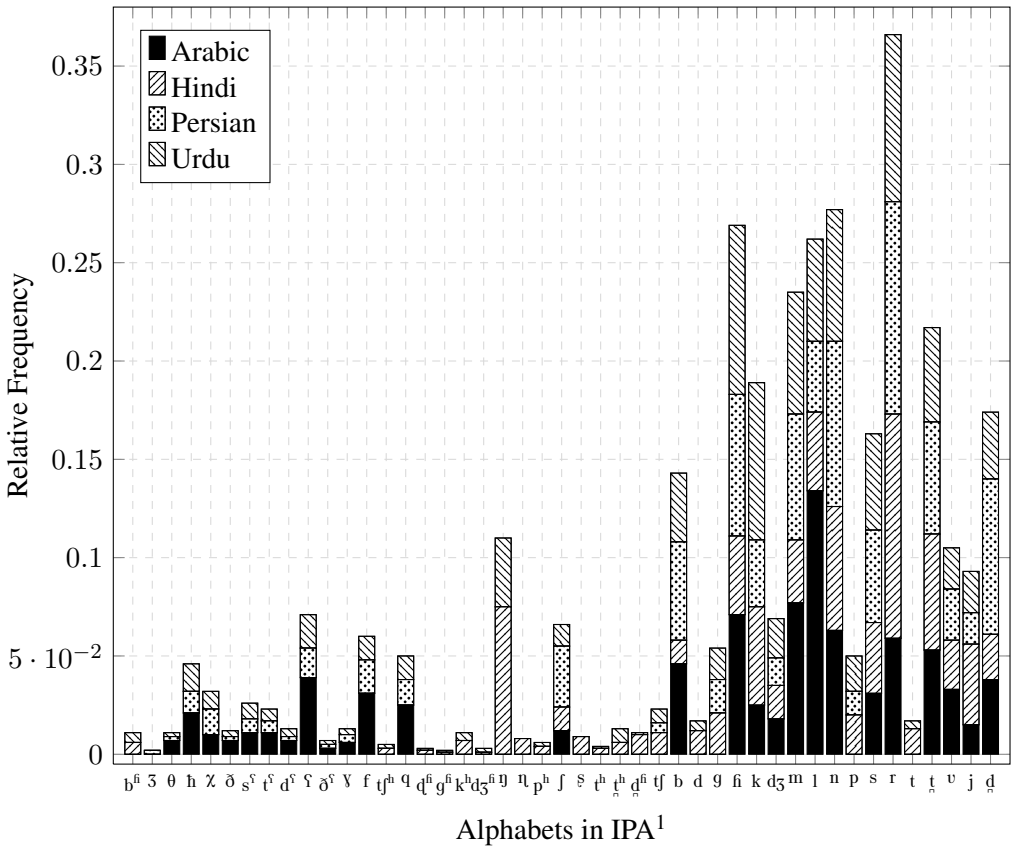

Figure 1: *Relative Distribution of Arabic, Hindi, Persian and Urdu Alphabets (Consonants only)*

Urdu Dictionary[2] (henceforth OUD). OUD has been prepared under the supervision of the e-government Directorate of Pakistan[3]. Apart from basic definition and meaning, it provides etymological information for more than 120K Urdu words. Since the dictionary is freely[4] available and requires no expertise for extraction of word etymology which is usually the case with manual annotation, we could mark the etymological information on a reasonably sized word list in a limited time frame. The statistics are provided in Table (3). We use **Indic** as a cover term for all the words that are either from Sanskrit, Prakrit, Hindi or local languages.

Language	Data Size	Average Token Length
Arabic	6,524	6.8
Indic	3,002	5.5
Persian	4,613	6.5

Table 3: Statistics of Etymological Data

[2]http://182.180.102.251:8081/oud/default.aspx
[3]www.e-government.gov.pk
[4]We are not aware of an offline version of OUD.

3.3 Experiments

We carried out a number of experiments in order to explore the effect of data size and the order of n-gram models on the classification performance. By varying the size of training data, we wanted to identify the lower bound on the training size with respect to the classification performance. We varied the training size per training iteration by 1% for n-grams in the order 1-5 for both the classification problems. For each n-gram order 100 experiments were carried out, i.e overall 800 experiments for binary and tri-class classification. The impact of training size on the classification performance is shown in Figures (2) and (3) for binary and tri-class classification respectively. As expected, at every iteration the additional data points introduced into the training data increased the performance of the model. With a mere 3% of the training data, we could reach a reasonable accuracy of 0.85 in terms of F-score for binary classification and for tri-class classification we reached the same accuracy with 6% of the data.

Similarly, we tried different order n-gram models to quantify the effect of character context on

the classification performance. As with the increase in data size, increasing the n-gram order profoundly improved the results. In both the classification tasks, unigram based models converge faster than the higher order n-gram based models. The obvious reason for it is the small, finite set of characters that a language operates with ($\sim$ 37 in Arabic, $\sim$ 39 in Persian and $\sim$ 48 in Hindi). A small set of words (unique in our case) is probably enough to capture at least a single instance of each character. As no new n-gram is introduced with subsequent additions of new tokens in the training data, the accuracy stabilizes. However, the accuracy with higher order n-grams kept on increasing with an increase in the data size, though it was marginal after 5-grams. The abrupt increase after 8,000 training instances is probably due to the addition of an unknown bigram sequence(s) to the training data. In particular, the Recall of Persio-Arabic increased by 2.2%.

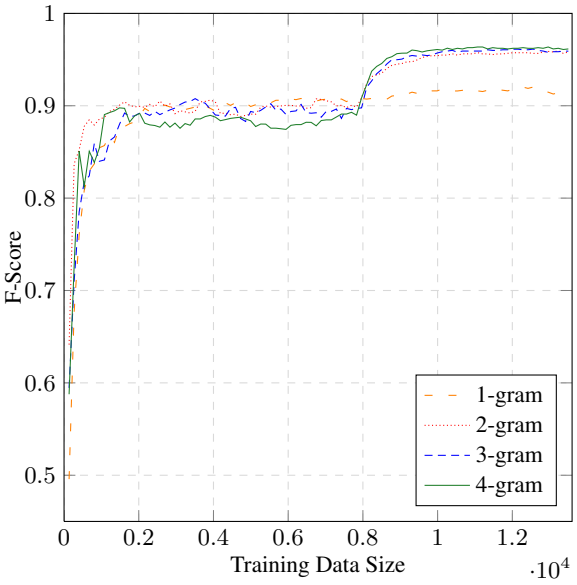

Figure 2: *Learning Curves for Binary Classification of Urdu Vocabulary*

3.4 Results

We performed 10-fold cross validation over all the instances of the etymological data for both the binary and tri-class classification tasks. We split the data into training and testing sets with a ratio of 80:20 using the stratified sampling. Stratified sampling distributes the samples of each class in training and testing sets with the same percentage as in the complete set. For all the 10-folds, the order of

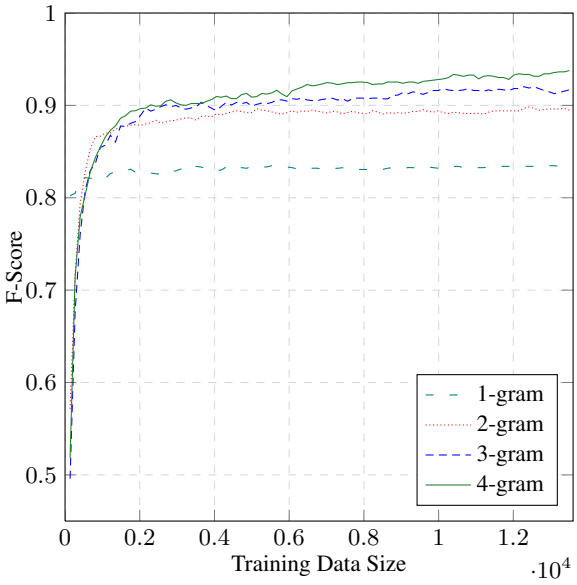

Figure 3: *Learning Curves for Tri-class Classification of Urdu Vocabulary*

n-gram was varied again from 1-5. Tables (4) and (5) show the consolidated results for these tasks with a frequency based baseline to evaluate the classification performance. In both the tasks, we achieved highest accuracy with language models trained with 5-gram letter sequence context. The best results in terms of F-score are 0.96 and 0.93 for binary and tri-class classification respectively.

Type	**Precision** (P)	**Recall** (R)	**F1-Score** (F)
Baseline	0.40	0.50	0.40
1-gram	0.89	0.89	0.89
2-gram	0.95	0.95	0.95
3-gram	0.96	0.96	0.96
4-gram	0.96	0.96	0.96
5-gram	0.96	0.96	0.96

Table 4: Results of 10-fold Cross Validation on Binary Classification

Although, we have achieved quite reasonable accuracies in both the tasks, a closer look at the confusion matrices shown in Tables (6) and (7) show that we can still improve the accuracies by balancing the size of data across classes. In binary classification our model is more biased towards Persio-Arabic as the data is highly imbalanced. Our binary classifier misclassifies 0.86% of Indic tokens as Persio-Arabic since the prior probability of the latter is much higher than that of the former. While in the case of tri-class classification, using

Type	Precision (P)	Recall (R)	F1-Score (F)
Baseline	0.15	0.33	0.21
1-gram	0.83	0.83	0.83
2-gram	0.89	0.89	0.89
3-gram	0.91	0.91	0.91
4-gram	0.93	0.93	0.93
5-gram	0.93	0.93	0.93

Table 5: Results of 10-fold Cross Validation on Tri-Class Classification

higher order n-gram models can resolve the prominent confusion between Arabic and Persian. Since both Arabic and Persian share almost the same phonetic inventory, working with lower order n-gram models doesn't seem ideal.

Class	Indic	Persio-Arabic
Indic	235	60
Persio-Arabic	15	1,057

Table 6: Confusion Matrix of Binary Classification

Class	Arabic	Indic	Persian
Arabic	605	5	26
Indic	11	268	18
Persian	22	9	415

Table 7: Confusion Matrix of Tri-class Classification

4 Adapting Frames from Arabic and Hindi PropBanks

As discussed in Section 2.1.1, the creation of predicate frames precedes the actual annotation of verb instances in a given corpus. In this section, we describe our approach towards the first stage of Urdu PropBanking by adapting related predicate frames from Arabic and Hindi PropBanks (Palmer et al., 2008; Vaidya et al., 2011). Since a PropBank is not available for Persian, we could only adapt those predicate frames which are shared with Arabic and Hindi.

Although, Urdu shares or borrows most of its literary vocabulary from Arabic and Persian, it retains its simple verb (as opposed to compound or complex verbs) inventory from Indo-Aryan ancestry. Verbs from Arabic and Persian are borrowed less frequently, although there are examples such as '$khariid$' buy, '$farma$' say etc.[5] This overlap in the verb inventory between Hindi and Urdu might explain the fact that they share the same grammar.

The fact that Urdu shares its lexicon with these languages, prompted us towards exploring the possibility of using their resources for Urdu PropBanking. We are in the process of adapting frames for those Urdu predicates that are shared with either Arabic or Hindi.

Urdu frame file creation must be carried out for both simple verbs and complex predicates. Since Urdu differs very little in simple verb inventory from Hindi, this simplifies the development process as the frames could be ported easily. However, this is not the case with nominal predicates. In Urdu, many nominal predicates are borrowed from Arabic or Persian as shown in Table (8). Given that a PropBank for Persian is not available, the task of creating the frames for nominal predicates in Urdu would have been fairly daunting in the paucity of the Arabic PropBank, as well.

	Simple Verbs		Nominal Predicates	
Language	**Total**	**Unique**	**Total**	**Unique**
Arabic	12	1	6,780	765
Hindi	7,332	441	1,203	258
Persian	69	3	2,276	352
Total	7,413	445	10,259	1,375

Table 8: Urdu Treebank Predicate Statistics

4.1 Simple Verbs

The simple verb inventory of Urdu and Hindi is almost similar, so the main task was to locate and extract the relevant frames from Hindi frame files. Fortunately, with the exception of $farmaa$ 'say', all the other simple verbs which Urdu borrows from Persian or Arabic (cf. Table (8)) were also borrowed by Hindi. Therefore, the Hindi simple verb frame files sufficed for porting frames for Urdu simple verbs.

There were no significant differences found between the Urdu and Hindi rolesets, which describe either semantic variants of the same verb or its causative forms. Further, in order to name the frame files with their corresponding Urdu lemmas, we used Konstanz's Urdu transliteration scheme

[5] Borrowed verbs often do not function as simple verbs rather they are used like nominals in complex predicate constructions such as **mehsoos** in '*mehsoos karnaa*' *to feel*.

(Malik et al., 2010) to convert a given lemma into its romanized form. Since the Hindi frame files use the WX transliteration scheme[6], which is not appropriate for Urdu due to lack of coverage for Persio-Arabic phonemes or sounds like **ڈ** *'pharyngealized voiced alveolar stop'*. The frame files also contain example sentences for each predicate, in order to make the PropBank annotation task easier. While adapting the frame files from Hindi to Urdu, simply transliterating such examples for Urdu predicates was not always an option, because sentences consisting of words with Sanskrit origin may not be understood by Urdu speakers. Hence, all the examples in the ported frames have been replaced with Urdu sentences by an Urdu expert.

In general we find that the Urdu verbs are quite similar to Hindi verbs, and this simplified our task of adapting the frames for simple verbs. The nouns, however, show more variation. Since a large proportion (up to 50%) of Urdu predicates are expressed using verb-noun complex predicates, nominal predicates play a crucial role in our annotation process and must be accounted for.

4.2 Complex Predicates

In the Urdu treebank, there are 17,672 predicates, of which more than half have been identified as noun-verb complex predicates (NVC) at the dependency level. Typically, a noun-verb complex predicate *chorii* 'theft' *karnaa* 'to do' has two components: a noun *chorii* and a light verb *karnaa* giving us the meaning 'steal'. The verbal component in NVCs has reduced predicating power (although it is inflected for person, number, and gender agreement as well as tense, aspect and mood) and its nominal complement is considered the true predicate. In our annotation of NVCs, we follow a procedure common to all PropBanks, where we create frame files for the nominal or the 'true' predicate (Hwang et al., 2010). An example of a frame file for a noun such as *chorii* is described in Table (9).

The creation of a frame file for the set of true predicates that occur in an NVC is important from the point of view of linguistic annotation. Given the large number of NVCs, a semi-automatic method has been proposed for creating Hindi nominal frame files, which saves the manual effort required for creating frames for nearly

[6]http://en.wikipedia.org/wiki/WX_notation

Frame file for *chorii*-n(oun)	
chorii.01: theft-n	light verb: *kar*'do; to steal'
Arg0	person who steals
Arg1	thing stolen
chorii.02 : theft-n	light verb: *ho* 'be/become; to get stolen'
Arg1	thing stolen

Table 9: Frame file for predicate noun *chorii* 'theft' with two frequently occurring light verbs *ho* and *kar*. If other light verbs are found to occur, they are added as additional rolesets as chorii.03, chorii.04 and so on.

3,015 unique Hindi noun and light verb combinations (Vaidya et al., 2013).

For Urdu, the process of nominal frame file creation is preceded by the identification of the etymological origin for each nominal. If that nominal has an Indic or Arabic origin, relevant frames from Arabic or Hindi PropBanks were adapted for Urdu. On the other hand, if the Urdu nominal originates from Persian, then frame creation will be done either manually or using other available Persian language resources, in the future.

In Table (8), there are around 258 nominal predicates that are common in Hindi and Urdu, so we directly ported their frames from Hindi PropBank with minor changes as was done for simple verb frames. Out of 765 nominal predicates shared with Arabic, 308 nominal predicate frames have been ported to Urdu. 98 of these nominal predicate frames were already present in the Arabic PropBank and were ported as such. However, for the remaining 667 unique predicates, frames are being created manually by Arabic PropBanking experts and will be ported to Urdu once they become available.

Porting of Arabic frames to Urdu is not that trivial. We observed that while Urdu borrows vocabulary from Arabic it does not borrow all the senses for some words. In such cases, the rolesets that are irrelevant to Urdu have to be discarded manually. The example sentences for all the frames ported from Arabic PropBank have to be sourced from either the web or manually created by an Urdu expert, as was the case with Hindi simple verbs.

5 Conclusion

In this paper we have exploited the overlap between the lexicon of Urdu, Arabic and Hindi for the creation of predicate frames for Urdu Prop-

Banking. We presented a simple and accurate classifier for the identification of source or origin of Urdu vocabulary which is a necessary step in the overall process of extraction of predicate frames from the related PropBanks. In the case of simple verbs that occur in the Urdu treebank, we have extracted all the frames from the Hindi PropBank and adapted them for Urdu PropBanking. Similarly for complex predicates, frames for Urdu treebank nominal predicates are extracted from Hindi as well as from Arabic PropBanks. Since a PropBank is not available for Persian, the creation of frames for shared predicates with Persian is a prospect for future work. We plan to create these frames either manually or semi-automatically, using the available Persian Dependency treebanks (Rasooli et al., 2011; Rasooli et al., 2013).

Acknowledgments

We would like to thank Himani Chaudhry for her valuable comments that helped to improve the quality of this paper.

The work reported in this paper is supported by the NSF grant (Award Number: CNS 0751202; CFDA Number: 47.070)[7].

References

Riyaz Ahmad Bhat and Dipti Misra Sharma. 2012. A dependency treebank of urdu and its evaluation. In *Proceedings of the Sixth Linguistic Annotation Workshop*, pages 157–165. Association for Computational Linguistics.

Rajesh Bhatt, Bhuvana Narasimhan, Martha Palmer, Owen Rambow, Dipti Misra Sharma, and Fei Xia. 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.

Jinho D Choi, Claire Bonial, and Martha Palmer. 2010a. Propbank frameset annotation guidelines using a dedicated editor, cornerstone. In *LREC*.

Jinho D Choi, Claire Bonial, and Martha Palmer. 2010b. Propbank instance annotation guidelines using a dedicated editor, jubilee. In *LREC*.

David Dowty. 1991. Thematic proto-roles and argument selection. *Language*, 67(3):547–619.

[7]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.

Ted Dunning. 1994. *Statistical identification of language*. Computing Research Laboratory, New Mexico State University.

Heba Elfardy and Mona T Diab. 2012. Token level identification of linguistic code switching. In *COLING (Posters)*, pages 287–296.

Marcello Federico, Nicola Bertoldi, and Mauro Cettolo. 2008. Irstlm: an open source toolkit for handling large scale language models. In *Interspeech*, pages 1618–1621.

Jena D Hwang, Archna Bhatia, Clare Bonial, Aous Mansouri, Ashwini Vaidya, Nianwen Xue, and Martha Palmer. 2010. Propbank annotation of multilingual light verb constructions. In *Proceedings of the Fourth Linguistic Annotation Workshop*, pages 82–90. Association for Computational Linguistics.

Ben King and Steven P Abney. 2013. Labeling the languages of words in mixed-language documents using weakly supervised methods. In *HLT-NAACL*, pages 1110–1119.

Paul Kingsbury and Martha Palmer. 2002. From treebank to propbank. In *LREC*. Citeseer.

Marco Lui, Jey Han Lau, and Timothy Baldwin. 2014. Automatic detection and language identification of multilingual documents. *Transactions of the Association for Computational Linguistics*, 2:27–40.

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 *LREC*.

Colin P Masica. 1993. *The Indo-Aryan Languages*. Cambridge University Press.

Dong Nguyen and A Seza Dogruoz. 2014. Word level language identification in online multilingual communication. In *Proceedings of the 2013 Conference on Empirical Methods in Natural Language Processing*.

Martha Palmer, Olga Babko-Malaya, Ann Bies, Mona T Diab, Mohamed Maamouri, Aous Mansouri, and Wajdi Zaghouani. 2008. A pilot arabic propbank. In *LREC*.

Mohammad Sadegh Rasooli, Amirsaeid Moloodi, Manouchehr Kouhestani, and Behrouz Minaei-Bidgoli. 2011. A syntactic valency lexicon for persian verbs: The first steps towards persian dependency treebank. In *5th Language & Technology Conference (LTC): Human Language Technologies as a Challenge for Computer Science and Linguistics*, pages 227–231.

Mohammad Sadegh Rasooli, Manouchehr Kouhestani, and Amirsaeid Moloodi. 2013. Development of a persian syntactic dependency treebank. In *Proceedings of the 2013 Conference of the North American Chapter of the Association for Computational*

Linguistics: Human Language Technologies, pages 306–314.

Ashwini Vaidya, Jinho D Choi, Martha Palmer, and Bhuvana Narasimhan. 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.

Ashwini Vaidya, Martha Palmer, and Bhuvana Narasimhan. 2013. Semantic roles for nominal predicates: Building a lexical resource. *NAACL HLT 2013*, 13:126.

Fei Xia, Owen Rambow, Rajesh Bhatt, Martha Palmer, and Dipti Misra Sharma. 2009. Towards a multi-representational treebank. In *The 7th International Workshop on Treebanks and Linguistic Theories. Groningen, Netherlands*.