Assas-Band, an Affix-Exception-List Based Urdu Stemmer

Qurat-ul-Ain Akram
Center for Research in Urdu Language Processing
NUCES, Pakistan
ainie.akram@nu.edu.pk

Asma Naseer
Center for Research in Urdu Language Processing
NUCES, Pakistan
asma.naseer@nu.edu.pk

Sarmad Hussain
Center for Research in Urdu Language Processing
NUCES, Pakistan
sarmad.hussain@nu.edu.pk

Abstract

Both Inflectional and derivational morphology lead to multiple surface forms of a word. Stemming reduces these forms back to its stem or root, and is a very useful tool for many applications. There has not been any work reported on Urdu stemming. The current work develops an Urdu stemmer or *Assas-Band* and improves the performance using more precise affix based exception lists, instead of the conventional lexical lookup employed for developing stemmers in other languages. Testing shows an accuracy of 91.2%. Further enhancements are also suggested.

1. Introduction

A stemmer extracts stem from various forms of words, for example words *actor, acted,* and *acting* all will reduce to stem *act*. Stemmers are very useful for a variety of applications which need to acquire root form instead of inflected or derived forms of words. This is especially true for Information Retrieval tasks, which search for the base forms, instead of inflected forms. The need of stemmers becomes even more pronounced for languages which are morphologically rich, and have a variety of inflected and derived forms.

Urdu is spoken by more than a 100 million people (accessed from http://www.ethnologue.com/show_ language.asp ?code =urd). It is the national language of Pakistan and a state language of India. It is an Indo-Aryan language, and is morphologically rich. Currently there is no stemmer for Urdu, however recent work has shown that it may have much utility for a variety of applications, much wider than some other languages. Due to the morphological richness of Urdu, its application to information retrieval tasks is quite apparent. However, there are also a few other areas of application, including automatic diacritization for text to speech systems, chunking, word sense disambiguation and statistical machine translation. In most of these cases, stemming addresses the sparseness of data caused by multiple surface forms which are caused mostly by inflections, though also applicable to some derivations.

Due to urgent need for some applications, an Urdu stemmer called *Assas-Band*[1], has been developed. The current work explains the details of *Assas-Band* and its enhancements using exceptions lists instead of lexical lookup methods, to improve its accuracy. Finally results are reported and discussed.

2. Literature Review

Urdu is rich in both inflectional and derivational morphology. Urdu verbs inflect to show agreement for number, gender, respect and case. In addition to these factors, verbs in Urdu also have different inflections for infinitive, past, non-past, habitual and imperative forms. All these forms (twenty in total) for a regular verb are duplicated for transitive and causative (di-transitive) forms, thus giving a total of more than sixty inflected variations. Urdu nouns also show agreement for number, gender and case. In addition, they show diminutive and vocative affixation. Moreover, the nouns show derivational changes into adjectives and nouns. Adjectives show similar agreement changes for number, gender and case. A comprehensive computational analysis of Urdu morphology is given by Hussain (2004).

Stemmers may be developed by using either rule-based or statistical approaches. Rule-based stemmers require prior morphological knowledge of the language, while statistical stemmers use corpus to calculate the occurrences of stems and affixes. Both rule-based and statistical stemmers have been developed for a variety of languages.

A rule-based stemmer is developed for English by Krovetz (1993) using machine-readable dictionaries. Along with a dictionary, rules for inflectional and derivational morphology are defined. Due to high dependency on dictionary the systems lacks consistency (Croft and Xu 1995). In Porter Stemmer (Porter 1980) the algorithm enforces some terminating conditions of a stem. Until any of the conditions is achieved, it keeps on removing endings of the word iteratively. Thabet has proposed a stemmer that performs stemming of classical Arabic

[1] In Urdu *Assas* means stem and *Assas-Band* means stemmer

Proceedings of the 7th Workshop on Asian Language Resources, ACL-IJCNLP 2009, pages 40–47,
Suntec, Singapore, 6-7 August 2009.

in Quran (Thabet 2004) using stop-word list. The main algorithm for prefix stemming creates lists of words from each *surah*. If words in the list do not exist in stop-word list then prefixes are removed. The accuracy of this algorithm is 99.6% for prefix stemming and 97% for postfix stemming. An interesting stemming approach is proposed by Paik and Parui (2008), which presents a general analysis of Indian languages. With respect to the occurrences of consonants and vowels, characters are divided into three categories. Different equivalence classes are made of all the words in the lexicon using the match of prefix of an already defined length. This technique is used for Bengali[2], Hindi and Marathi languages. A rule-based stemming algorithm is proposed for Persian language by Sharifloo and Shamsfard (2008), which uses bottom up approach for stemming. The algorithm identifies substring (core) of words which are derived from some stem and then reassembles these cores with the help of some rules. Morpheme clusters are used in rule matching procedure. An anti-rule procedure is also employed to enhance the accuracy. The algorithm gives 90.1 % accuracy.

Besides rule-based stemmers there are a number of statistical stemmers for different languages. Croft and Xu provide two methods for stemming i.e. Corpus-Specific Stemming and Query-Specific Stemming (Croft and Xu 1995). Corpus-Specific Stemming gathers unique words from the corpus, makes equivalence classes, and after some statistical calculations and reclassification makes a dictionary. Query-Based Stemming utilizes dictionary that is created by Corpus-Based Stemming. Thus the usual process of stemming is replaced with dictionary lookup. Kumar and Siddiqui (2008) propose an algorithm for Hindi stemmer which calculates n-grams of the word of length *l*. These n-grams are treated as postfixes. The algorithm calculates probabilities of stem and postfix. The combination of stem and postfix with highest probability is selected. The algorithm achieves 89.9% accuracy. Santosh et.al. (2007) presents three statistical techniques for stemming Telugu language. In the first technique the word is divided into prefix and postfix. Then scores are calculated on the basis of frequency of prefix, length of prefix, frequency of postfix, and length of postfix. The accuracy of this approach is 70.8%. The second technique is based on n-grams. Words are clustered using n-grams. Within the cluster a smallest word is declared as the stem of the word. The algorithm gives 65.4% accuracy. In the third approach a successive verity is calculated for each word's prefix. This approach increases accuracy to 74.5%.

Looking at various techniques, they can generally be divided into rule based or statistical methods. Rule based methods may require cyclical application of rules. Stem and/or affix look-ups are needed for the rules and may be enhanced by maintaining a lexicon. Statistical stemmers are dependent on corpus size, and their performance is influenced by morphological features of a language. Morphologically richer languages require deeper linguistic analysis for better stemming. Three different statistical approaches for stemming Telugu (Kumar and Murthy 2007) words reveal very low accuracy as the language is rich in morphology. On the other hand rule-based techniques when applied to morphologically rich languages reveal accuracy up to 99.6% (Thabet 2004). Like other South Asian languages, Urdu is also morphologically rich. Therefore, the current work uses a rule based approach with a variation from lexical look-up, to develop a stemmer for Urdu. The next sections discuss the details of development and testing results of this stemmer.

3. Corpus Collection

An important phase of developing *Assas-Band* is corpus collection. For this four different lexica and corpora[3]: C1 (Sajjad 2007), C2[4], C3 (Online Urdu Dictionary, available at www.crulp.org/oud) and C4 (Ijaz and Hussain 2007) are used for analysis and testing. Furthermore, prefix and postfix lists[5] are also used during the analysis. The summary of each of the resources is given in table 1.

Table 1: Corpora Words Statistics

Corpus	Total No. of Words	Unique Words
C1	63,298	10,604
C2	96,890	7,506
C3	149,486	149,477
C4	19,296,846	50,000

4. Methodology

The proposed technique uses some conventions for the Urdu stemmer *Assas-Band*. The stem returned by this system is the meaningful root e.g. the stem of لڑکیاں *larkiyan* (girls) is لڑکی *larki* (girl) and not the لڑک *larak* (boy/girl-hood; not a surface from). It also maintains distinction between the masculine and

[2] Also see Islam et al. (2007) for Bengali stemming

[3] Available from CRULP (www.crulp.org)

[4] Unpublished, internally developed by CRULP

[5] Internally developed at CRULP

feminine forms of the stem. *Assas-Band* gives the stem لڑکا *larka* (boy) for word لڑکوں *larkon* (boys) and stem لڑکی *larki* (girl) for لڑکیاں *larkiyan* (girls). The reason for maintaining the gender difference is its usability for other tasks in Urdu, e.g. machine translation, automatic diacritization etc. The word can easily be converted to underlying stem (e.g. لڑک *larak* (boy/girl-hood)), if needed.

Assas-Band is trained to work with Urdu words, though it can also process foreign words, e.g. Persian, Arabic and English words, to a limited extent. Proper nouns are considered stems, though only those are handled which appear in the corpora.

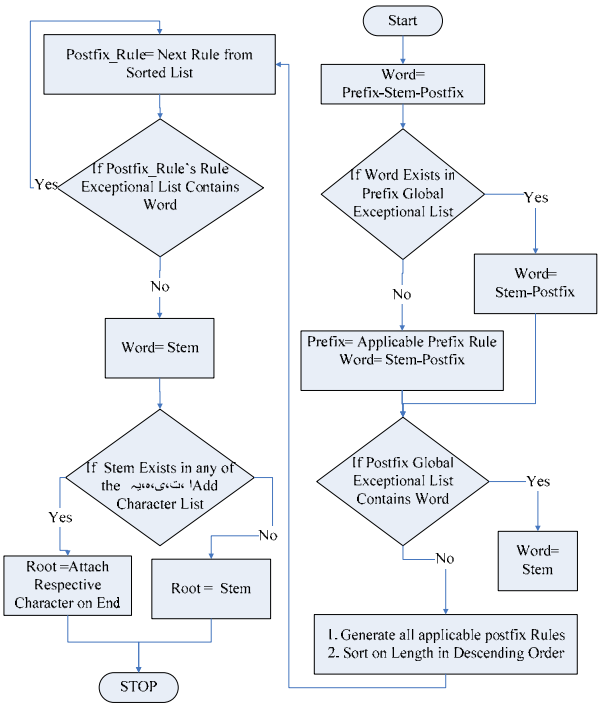

Figure 1: Flow Chart for the Stemming Process

An Urdu word is composed of a sequence of prefixes, stem and postfixes. A word can be divided into (*Prefix)-Stem-(Postfix)*. *Assas-Band* extracts *Stem* from the given word, and then converts it to surface form, as per requirement. The algorithm of the system is as follows. First the prefix (if it exists) is removed from the word. This returns the *Stem-(Postfix)* sequence. Then postfix (if it exists) is removed and *Stem* is extracted. The post-processing step (if required) is performed at the end to generate the surface form.

However, while applying affix rules for any word, the algorithm checks for exceptional cases and applies the affix stripping rules only if the exceptional cases are not found. This is different from other methods which first strip and then repair.

The algorithm for *Assas-Band* is given in Figure 1 and explained in more detail below.

Prefix Extraction: To remove the prefix from the word, first it is checked whether the input word exists in the Prefix-Global-Exceptional-List (PrGEL). If it exists in PrGEL, then it means that the word has an initial string of letters which matches a prefix but is part of the stem and thus should not be stripped. If the word does not exist in PrGEL, then prefix rules list is looked up. If an applicable prefix is found, starting from longest matching prefix to shorter prefix, appropriate rule is applied to separate *prefix* from *stem-postfix*. Both parts of the word are retained for further processing and output.

Postfix Extraction: This process separates the postfix from word and performs the post-processing step, if required, for generating the surface form.

First the remaining *Stem-(Postfix)* is looked up in a general Postfix-Global-Exceptional-List (PoGEL). If the word exists in the list, then it is marked as the stem. If the word does not exist in this list, it indicates that a possible postfix is attached. Postfix matching is then performed. The candidate postfix rules are sorted in descending order according to the postfix length. In addition, a Postfix-Rule-Exception-List (PoREL) is also maintained for each postfix. The first applicable postfix from the list is taken and it is checked if the word to be stemmed exists in PoREL. If the word does not exist in PoREL, then the current postfix rule is applied and the *Stem* and *Postfix* are extracted. If the word exists in the PoREL then the current postfix rule is not applied and the next postfix rule is considered. This process is repeated for all candidate postfix rules, until a rule is applied or the list is exhausted. In both cases the resultant word is marked as *Stem*.

A complete list of prefixes and postfixes are derived by analyzing various lexica and corpora (and using grammar books). In addition, complete rule exception list for each postfix (PoREL), complete general exception list for prefixes PrGEL and general exception list for postfixes PoGEL are developed using C1, C2, C3 and C4. PrGEL and PoGEL are also later extended to include all stems generated through this system.

After applying prefix and postfix rules, post processing is performed to create the surface form of the stem. The stem is looked up in the Add-Character-Lists (ACL). There are only five lists, maintained for each of the following letter(s): ا، ت، ہ، ی، ے (*yay-hay, choti-yah, gol-hay, tay, alif*), because only these can be possibly added. If the stem is listed, the corresponding letter(s) are appended at the end to

generate the surface form, else the stem is considered the surface form.

Though the algorithm is straight forward, to the lists have been developed manually after repeated analysis, which has been a very difficult task, as explained in next section. Some sample words in these lists are given in the Appendices A and B.

5. Analysis Phase

The analysis has been divided into two phases. First phase involved the extraction of prefixes and postfixes. The second phase dealt with the development of Prefix and Postfix Global Exceptional Lists (PrGEL, PoGEL), Postfix Rule Exceptional Lists (PoREL) and Add Character Lists (ACL). These are discussed here.

5.1. Extraction of Affixes

C1 and C2 are used for the extraction of affixes. These corpora are POS tagged. The analysis is performed on 11,000 high frequency words. The details of these corpora are given in Table 1. By looking at each word, prefixes and postfixes are extracted. Words may only have a prefix e.g. بدصورت *bud-surat* (ugly), only a postfix, e.g. تصورات *tasawr-aat* (imaginations), or both prefix and postfix, e.g. بداخلاقی *bud-ikhlaq-i* (bad manners). After analysis, 40 prefixes and 300 postfixes are extracted. This list is merged with an earlier list of available postfixes and prefixes[6]. A total of 174 prefixes and 712 postfixes are identified. They are listed in Appendix C. In this phase, the post-processing rules are also extracted separately.

5.2. Extraction of Exception and Word Lists

The following lists are used to improve the accuracy of *Assas-Band*.

1. Prefix and Postfix Global Exceptional Lists (PrGEL, PoGEL)
2. Postfix Rule Exceptional List (PoREL) for each postfix
3. Add Character List (ACL) for each letter/sequence

The second phase of analysis is performed to generate these lists. This analysis is based on C3.

Development of PrGEL: The PrGEL contains all those words from which a prefix cannot be extracted. The list contains words with first few letters which match a prefix but do not contain this prefix, e.g. باندھے *bandh-ay* (tied). This word exists in PrGEL to ensure that the prefix با *ba* (with) is not removed to give invalid stem ندھے *ndhay*. This single list is maintained globally for all prefixes.

Development of PoGEL: There are also many words which do not contain any postfix but their final few letters may match with one. If they are not identified and prevented from postfix removal process, they may result in erroneous invalid stems. For example, ہاتھی *hathi* (elephant) may be truncated to ہاتھ hath (hand), which is incorrect removal of the postfix ی (letter *choti-yay)*. All such words are kept in the PoGEL, and considered as a stem. This single list is maintained globally for all the postfixes.

Rule Exceptional Lists: Candidate postfixes are applied in descending order of length. For example, for the word بستیاں *bastiyan* (towns), the following postfixes can be applied: تیاں *tiyan*, یاں *yan*, اں *aan* and ں *noon-gunna*.

First, if the maximal length postfix matches, it is stripped. However, there are cases, when there is a match, but the current postfix should not be detached (a shorter postfix needs to be detached). In this case a postfix specific list is needed to list the exceptions to ensure preventing misapplication of the longer postfix. For this situation PoREL is maintained for each postfix separately. So for بستیاں *bastiyan* (towns), first the maximum length postfix تیاں *tiyan* is matched. However, this creates the stem بس *bas* (bus) which is incorrect. Thus, بستیاں *bastiyan* (towns) is stored in the PrREL of تیاں *tiyan*. Due to this, this postfix is not extracted and the next longest postfix rule is applied. Even in this case nonsense stem بست *bast* is generated. Thus, بستیاں *bastiyan* (towns) is also stored in the PrREL of postfix یاں *yan*. Next the postfix اں *an* is applied. This yields بستی *basti* (town), which is correct. This checking and PrREL development process is manually repeated for all the words in the corpus.

Add Character Lists: During second phase the ACLs (already developed in the first phase) are updated against each of the five possible letter sequences, i.e. ا،ت،ی،ہ،یہ, to generate correct surface forms. For example, when postfix گی *gi* is removed from زندگی *zindagi* (life), it creates the stem زند *zind,* which is not a surface form. The letter ہ *hay* has to be appended at the end to produce the correct surface form زندہ *zinda* (alive). So زند *zind* is stored in the ACL of letter ہ. In the same way the lists are developed and maintained for the five letters separately. After applying a particular postfix rule on

[6] Internally developed at CRULP

the word, the result is checked in each ACL. If the string is found in any of the lists then respective character is attached at the end.

Instead of manually doing all the work, the process is automated using an online Urdu dictionary (OUD) (available at www.crulp.org/oud) using the following algorithm.

1. *Take a word from corpus.*
2. *Generate all applicable rules.*
3. *Sort all rules in descending order according to the maximum length of each.*
4. *Extract upper- most rule from the rules list.*
5. *Apply extracted rule on the word. Check remaining word's existence in the dictionary.*
   a. *If remaining word exists in the dictionary, store that original word in the respective rule's Stem List and stop the loop.*
   b. *Otherwise store original word in the Rule Exceptional List of the respective rule and go to Step 4 for the next rule.*
6. *Repeat steps 4 and 5 until*
   a. *Stop condition (5a) occurs, or*
   b. *All the generated rules have been traversed.*
7. *If termination of the loop is due to step 6b, then the word is stored in the Global Exceptional List which is universal for all the rules.*
8. *Repeat step 1-7 for all the words in the corpus.*

The above algorithm is first run for prefixes. Once a complete manual check is performed on the results, the same algorithm is applied for the postfixes.

6. Manual Corrections

Manual inspection is needed to fix the errors generated by the automated system. The stem list is manually scanned to identify real-word errors, i.e. the stemming is incorrect but results in a valid word. For example when ری *ri* postfix is applied to the word ٹوکری *tokri* (basket), the word ٹوک *tok* (stop) is obtained which exists in the dictionary but is incorrect stemming. The inspection is also needed to ensure that the distinction between the masculine and feminine forms of a word is maintained. As discussed the gender distinction is kept to ensure better use in other applications.

Postfix Rule Exceptional List is scanned manually to check for any missing entries (in case the lexicon contains incomplete information about a word) or spurious entries (in case a word is not in the lexicon). Similarly, the process is also useful in identifying additional missing prefixes and postfixes. For example, the word آنسوؤں *aansuon* (tears) is found in the Exceptional List during manual analysis, because the postfix ؤں *on* was not initially identified. Thus, the algorithm applied the postfix ں *n*, leaving the incorrect stem آنسوؤ *aansuo*. This was (obviously) not found in OUD dictionary, so it was placed in PoGEL. By manually scanning each of the words in this list, new postfix was found, which created the correct stem آنسو *aansu* (tear). ACL is also updated by this manual analysis.

7. Testing

The test results are given in this section.

Testing Phase 1: The corpora C1 and C2 are used which have combined 11,339 unique words. The following table summarizes the testing results.

Table 2: Initial Testing Results

Testing Results	Values
Total Number of tested words	11339
Accurately Stemmed	7241
Incorrect Stemming	4098
Accuracy Rate	64%
Inaccurate Add Character	278
Inaccurate Prefix Stripping	754
Inaccurate Postfix Stripping	1006
Errors due to Foreign Words	2107
Number of Times Prefix Rules Applied	1656
Correct	942
Incorrect	714
Number of Times Postfix Rules Applied	5990
Correct	4984
Incorrect	1006
Number of Times Character Added	819
Correct	541
Incorrect	278

The accuracy of 64% is achieved. Some of the stems created are not in the lists and are erroneous. They are created by invalid prefix/postfix removal. Analysis showed that some prefixes and postfixes contributed to this error rate because they were derived from foreign words transliterated in Urdu. For example ز *z* postfix is correctly applied to the English

word لیڈیز *ladiez* (ladies) yielding the stem لیڈی *ladie* (lady). But this ز *z* postfix rule when applied to Urdu words increases the error rate. Similarly Arabic prefix ال *al (the),* which applies to Arabic words correctly e.g. القرآن *al-Quran* (the Quran), wrongly applies to Urdu words.

Another reason for error in stemming is ineffective post-processing due to insufficient words in the lists. There are also some other sources of errors which are not directly associated with stemming but are common for Urdu corpora. Errors are caused by spelling errors, including space character related errors (Naseem and Hussain 2007). There are also encoding normalization issues, which need to be corrected before string matching. This is caused by the variation in keyboards.

Testing Phase II: On the basis of previous result analysis, prefix and postfix rules which are applicable to only foreign words are removed from the rule lists. Such rules create errors in Urdu word stemming, while trying to cater non-essential task of stemming transliterated foreign words. The foreign words found in C1 and C2 are stored in global lists i.e. PrGEL and PoGEL to ensure that they are not processed.

Table 3: Test Results after Removing Foreign Prefixes and Postfixes Rules

Testing Results	Values
Total Number of tested words	10418
Accurately Stemmed	9476
Incorrect Stemming	942
Accuracy Rate	90.96%
Inaccurate Add Character	35
Inaccurate Prefix Stripping	473
Inaccurate Postfix Stripping	469
Errors due to Foreign Words	0
Number of Times Prefix Rules Applied	660
Correct	187
Incorrect	473
Number of Times Postfix Rules Applied	3445
Correct	2976
Incorrect	469
Number of Times Character Added	626
Correct	591
Incorrect	35

As errors from C1 and C2 have been manually fixed, testing is again performed by using 10,418 high frequency Urdu words from C4 (Ijaz and Hussain 2007). The summary of testing results is in Table 3.

Table 3 shows that removing foreign language affixes improves the results significantly. The prefix error rate is higher than the postfix error rate. In addition, the ACL has to be more comprehensive. There are also some errors because some words require both prefix and postfix to be extracted, but during stemming, if the prefix is wrongly applied and a faulty stem is generated, then the postfix is also incorrectly applied.

Testing Phase III: After analyzing test results of the second phase, amendments are made in the algorithm. Following post-processing, the stem generated is verified in PoGEL. If it does not exist, it is assumed that wrong rule is applied and thus it is skipped and the next rule is applied. This is repeated until the resulting stem is found in PoGEL. By implementing this methodology, the accuracy is enhanced from 90.96% to 91.18% for C4 corpus based word list as shown in Table 4.

Table 4: Test Results after Enhancing Algorithm

Testing Results	Values
Total Number of tested words	10418
Accurately Stemmed	9499
Incorrect Stemming	919
Accuracy Rate	91.18%
Inaccurate Add Character	35
Inaccurate Prefix Stripping	473
Inaccurate Postfix Stripping	446
Errors due to Foreign Words	0
Number of Times Prefix Rules Applied	660
Correct	187
Incorrect	473
Number of Times Postfix Rules Applied	3445
Correct	2999
Incorrect	446
Number of Times Character Added	626
Correct	591
Incorrect	35

The methodology does not affect prefix removal and the process of adding characters. The improvement made by this methodology is only in the accuracy of

postfixes because this modification is only performed on the second phase i.e. extraction of postfixes.

8. Conclusion

The current paper presents work performed to develop an Urdu stemmer. It first removes the prefix, then the postfix and then adds letter(s) to generate the surface form of the stem. In the first two steps it uses exception lists if a prefix and/or postfix can be applied. A successful lookup bypasses the stripping process. This is different from lexical or stem look up in other work which triggers the stripping process. The current stemming accuracy can be further improved by making the lists more comprehensive. ACL should also be maintained against each postfix for more accuracy. The developed system is currently being used for various other applications for Urdu language processing, including automatic diacritization.

Acknowledgements

The work has been partially supported by PAN Localization Project grant by International Development Research Center (IDRC) of Canada.

References

Croft, W. B. and Xu, J. 1995. Corpus-Specific Stemming using Word Form Co-occurrences. In Fourth Annual Symposium on Document Analysis and Information Retrieval.

Krovetz, R. 1993. View Morphology as an Inference Process. In the Proceedings of 5th International Conference on Research and Development in Information Retrieval.

Porter, M. 1980. An Algorithm for Suffix Stripping. *Program,* 14(3): 130-137.

Thabet, N. 2004. Stemming the Qur'an. In the Proceedings of the Workshop on Computational Approaches to Arabic Script-based Languages.

Hussain, Sara. 2004. *Finite-State Morphological Analyzer for Urdu*. Unpublished MS thesis, Center for Research in Urdu Language Processing, National University of Computer and Emerging Sciences, Pakistan.

Sajjad, H. 2007. *Statistical Part-of-Speech for Urdu*. Unpublished MS Thesis, Center for Research in Urdu Language Processing, National University of Computer and Emerging Sciences, Pakistan.

Ijaz, M and Hussain, S. 2007. Corpus Based Urdu Lexicon Development. In the Proceedings of Conference on Language Technology (CLT07), Pakistan.

Naseem, T., Hussain, S. 2007. Spelling Error Trends in Urdu. In the Proceedings of Conference on Language Technology (CLT07), Pakistan.

Kumar, M. S. and Murthy, K. N. 2007. Corpus Based Statistical Approach for Stemming Telugu. Creation of Lexical Resources for Indian Language Computing and Processing (LRIL), C-DAC, Mumbai, India.

Paik, J. H. and Parui, S. K. 2008. A Simple Stemmer for Inflectional Languages. Forum for Information Retrieval Evaluation,

Islam, M. Z., Uddin, M. N. and Khan, M. 2007. A Light Weight Stemmer for Bengali and Its Use in Spelling Checker. In the Proceedings of 1st Intl. Conf. on Digital Comm. and Computer, Amman, Jordan.

Sharifloo, A. A. and Shamsfard, M. 2008. A Bottom up Approach to Persian Stemming. In the Proceedings of the Third International Joint Conference on Natural Language Processing. Hyderabad, India.

Kumar, A. and Siddiqui, T. 2008. An Unsupervised Hindi Stemmer with Heuristics Improvements. In Proceedings of the Second Workshop on Analytics for Noisy Unstructured Text Data.

Appendix A

A.1 Postfix Rule Exceptional List Samples

Postfix	Some Exceptional Words
بات	ترکیبات
انے	سرسرانے,گڑانے,روشانے,تلملانے
ئے	آیئے,قرینے,مہینے,خزانے,شامیانے,ترانے,تھانے
ہیں	افواہیں,نگاہیں,کراہیں,جاپناہیں,چراگاہیں,سراہیں

A.2 Postfix Global Exception List Samples

راوی	مشہور	مسلح	پیروی	بشیر
ایوانِ	ذہن	جذب	فائرنگ	سکون
آفتاب	حاوی	چونکہ	سیبل	راستہ

A.3 Prefix Global Exception List Samples

مہنگی	نالیوں	نکالتے	یکایک
منائے	ناشپاتی	نکھارنے	یکساں
منائی	نایاب	نکالیں	یکسانیت
منگوا	نادار	نکالی	ہمدانی

Appendix B

Add Character List Samples

Add ت	Add ا
قیاد + ت = قیادت	انتہ + ا = انتہا
کاش + ت = کاشت	ایذ + ا = ایذا
سِس + ت = سِست	ایس + ا = ایسا
شدّ + ت = شدّت	کت + ا = کتا
Add ہ	**Add ی**
آزمود + ہ = آزمودہ	ترق + ی = ترقی
امریک + ہ = امریکہ	
افتاد + ہ = افتادہ	
آزمود + ہ = آزمودہ	

Appendix C

C.1 List of Sample Prefixes

ما	باد	تو	صاحب	مس
نا	غم	اشک	فور	بالا
پا	گلو	ناز	ان	شپس
برائے	شہ	تنگ	سوڈو	پس
بازی	نیل	بن	بال	پارہ
انڈر	صد	برائے	قبل	زود
نو	مابعد	روبہ	پاک	ثرائی
ادا	بد	آن	آبی	طالع
ایکس	دم	پر	خرد	آرام
روئے	ابو	غیر	من	مافوق
گراں	ام	تہ	آتش	آہن
دل	ڈی	مے	باطنی	زبر

C.2 List of Sample Postfixes

سوزی	انی	وائزرز	گاہی	انوں	آہنگیوں
نمائی	آرائی	ویز	دست	ہندی	نگیں
نفسی	رنگی	یز	ارتے	پروریاں	پروریوں
انگیزی	فروشی	وائز	انے	ٹیگیاں	بردایوں
نامی	سرائی	گرافیز	نامے	نوازیوں	راں
وتی	گردانی	ز	ئے	لیواؤں	سازیاں
تھانی	رسانی	سوز	ے	خیزیوں	آرائیاں
دلی	پروری	آمےز	چے	واں	شکنوں
پوشی	آمیزی	گرافز	اتے	گاہئیں	بوسیوں
بیانی	ائی	ازمز	واے	نوازیاں	ریزیاں
برداری	نشینی	اندوز	نے	بیانیاں	گریوں
اتی	ستانی	ریز	خانے	فشانیاں	کناں
خوری	آزاری	آموز	پنے	اندوزوں	ورزیاں
نگاہی	گردی	کیسز	کدے	بریوں	سراؤں
چاری	وتی	نواز	پلے	نویسوں	کاریوں
سنجی	بندی	راز	وے	گوئیوں	خوانیوں
فشانی	آفرینی	پرداز	ائے	تراشیاں	رانیوں